Robust algebraic Schur complement preconditioners based on low rank corrections

Laura Grigori, Frédéric Nataf, Soleiman Yousef

Préconditionneurs algébriques basés sur des corrections de rang faible rapport de recherche Inria

Laura Grigori, Frédéric Nataf, Soleiman Yousef

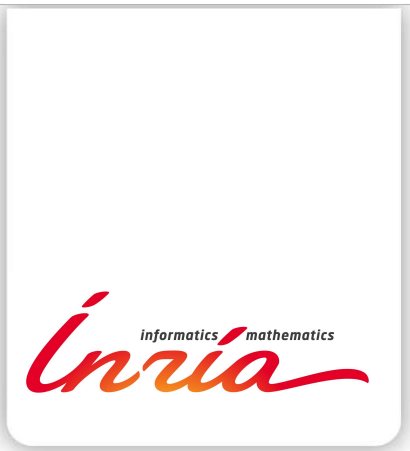
RESEARCH REPORT

N° 8557

June 2014

Project-Teams
ALPINES

ISSN 0249-6399 ISRN INRIA/RR--8557--FR+ENG



Préconditionneurs algébriques basés sur des corrections de rang faible rapport de recherche Inria

Laura Grigori*, Frédéric Nataf, Soleiman Yousef *

Équipes-Projets
ALPINES

Rapport de recherche n° 8557 — June 2014 — 18 pages

Résumé : Dans cet article, nous présentons LORASC, un préconditionneur pour la résolution des systèmes linéaires d'équations de très grande taille. Ce préconditionneur algébrique est basé sur des corrections de rang faible qui permettent de garantir sa robustesse.

Mots-clés : algèbre linéaire, méthodes itératives, préconditionneurs, corrections de rang faible

* INRIA

RESEARCH CENTRE
PARIS – ROCQUENCOURT

Domaine de Voluceau, - Rocquencourt
B.P. 105 - 78153 Le Chesnay Cedex

Robust algebraic Schur complement preconditioners based on low rank corrections

Abstract: In this paper we introduce LORASC, a robust algebraic preconditioner for solving sparse linear systems of equations involving symmetric and positive definite matrices. The graph of the input matrix is partitioned by using k-way partitioning with vertex separators into N disjoint domains and a separator formed by the vertices connecting the N domains. The obtained permuted matrix has a block arrow structure. The preconditioner relies on the Cholesky factorization of the first N diagonal blocks and on approximating the Schur complement corresponding to the separator block. The approximation of the Schur complement involves the factorization of the last diagonal block and a low rank correction obtained by solving a generalized eigenvalue problem or a randomized algorithm. The preconditioner can be build and applied in parallel. Numerical results on a set of matrices arising from the discretization by the finite element method of linear elasticity models illustrate the robusteness and the efficiency of our preconditioner.

Key-words: linear algebra, iterative methods, preconditioners, low rank corrections

1 Introduction

This paper focuses on algebraic robust preconditioners for solving iteratively linear systems of equations of the form $Ax = b$ arising from the discretization of partial differential equations on unstructured two-dimensional (2D) and three-dimensional (3D) grids. We consider that A is a symmetric positive definite (SPD) matrix and that the linear system $Ax = b$ is solved by using a Krylov subspace iterative method as CG [1]. The convergence rate of CG depends on the condition number of the matrix A and on the distribution of its eigenvalues [2]. The convergence rate of Krylov subspace methods is often accelerated by seeking a preconditioner M such that the preconditioned matrix $M^{-1}A$ has better spectral properties and the linear system $M^{-1}Ax = M^{-1}b$ converges faster.

There are many preconditioners proposed in the litterature, that could be either algebraic or application specific. Among algebraic preconditioners, several widely used preconditioners are incomplete LU factorizations, sparse approximate inverses (SPAI), algebraic domain decomposition methods as additive Schwarz. Such preconditioners can be applied to different classes of problems. One level methods have limited scalability when executed in parallel on large number of processors and/or they have a slow convergence or even stagnation for very difficult problems. This slow convergence is often due to the presence of a few very small eigenvalues in the spectrum of the preconditioned matrix. Several solutions exist to address this problem, the usage of a coarse space in domain decomposition methods, the usage of deflation through projection techniques, see e.g. [3, 4, 5] or [6] for an algebraic framework that connects the two approaches as well as multigrid methods. Another solution is provided by direction preserving or filtering preconditioners in which the preconditioner is identical with the input matrix on a set of vectors T, that is $MT = AT$. This idea has been used for block tridiagonal systems, e.g. [7, 8, 9, 10], general block factorizations [11, 12], by using semiseparable approximations [13], or in the context of multigrid methods, e.g. [14]. If the set T is formed by eigenvectors corresponding to the smallest eigenvalues of A, then in the preconditioned matrix $M^{-1}A$ these eigenvalues are shifted to 1. However, unlike deflation techniques, there are few general theoretical results in direction preserving factorizations that exploit this filtering property. For example in [11, 12], even if in practice the preconditioner filters well the small eigenvalues, there is no lower bound on the eigenvalues of the preconditioned matrix $M^{-1}A$.

Application specific preconditioners, that is preconditioners that require some information from the underlying PDE, can lead to faster convergence and can be more scalable on parallel computers than algebraic preconditioners. Such examples are domain decomposition methods [15, 16, 17, 18] or in the multigrid framework [19] in which the usage of a coarse space based on solving generalized eigenvalues problems, allows for a control of the condition number of the preconditioned matrix.

In this paper we introduce LORASC, a robust algebraic preconditioner of the form $M = (L + \tilde{D})\tilde{D}^{-1}(\tilde{D} + L^T)$ that can be efficiently build and applied in parallel. The graph of the input matrix A is first partitioned by using k-way partitioning with vertex separators into N disjoint domains and a separator formed by the vertices connecting the N domains. The permuted matrix based on this partitioning has a block arrow structure, as presented in equation (1), in which the first N diagonal blocks correspond to the disjoint domains, while the last diagonal block $A_{\Gamma\Gamma}$ corresponds to the separator.

$$A = \begin{pmatrix} A_{11} & & & A_{1\Gamma} \\ & \ddots & & \vdots \\ & & A_{NN} & A_{N\Gamma} \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & A_{\Gamma\Gamma} \end{pmatrix}. \tag{1}$$

LORASC is algebraic in the sense that no information from the underlying PDE is required, neither for the construction of the preconditioner, nor for the graph partitioning method used for parallelism. It is robust in the sense that the spectral condition number (defined as the ratio of the largest to the smallest eigenvalue) of the preconditioned matrix $\kappa(M^{-1}A)$ is bounded by a user defined value τ. Here both M and A are SPD matrices. The preconditioner relies on the Cholesky factorization of the first

N diagonal blocks and on approximating the Schur complement $S = A_{\Gamma\Gamma} - \sum_{j=1}^{N} A_{\Gamma j} A_{jj}^{-1} A_{j\Gamma}$ which would be computed in a direct factorization of A (however prohibitive for 3D large problems). The approximation of the Schur complement involves $A_{\Gamma\Gamma}$ and a low rank matrix obtained by solving a generalized eigenvalues problem of the form

$$Su = \lambda A_{\Gamma\Gamma} u, \quad S = A_{\Gamma\Gamma} - \sum_{j=1}^{N} A_{\Gamma j} A_{jj}^{-1} A_{j\Gamma}. \tag{2}$$

Note a main difference with direction preserving preconditioners, in which for approximating S, we would compute the smallest eigenvalues and associated eigenvectors T of S and then we would compute an approximation $\tilde{S}$ satisfying $\tilde{S}T = ST$. In LORASC, we approximate first the inverse of S by $A_{\Gamma\Gamma}^{-1}$, and then compute the smallest eigenvalues and associated eigenvectors of $A_{\Gamma\Gamma}^{-1}S$. The approximation $\tilde{S}^{-1}$ is obtained by correcting the first approximation by a low rank matrix using a technique inspired from Wielandt's deflation (see [20]). This approximation provides a bounded condition number of $\tilde{S}^{-1}S$ and also of the overall preconditioned matrix $M^{-1}A$. It involves a global generalized eigenvalues problem, which allows not to increase the size of the low rank approximation linearly with the number of domains/processors. We also propose in this paper a different approach to approximate the smallest eigenvalues and associated eigenvectors of $A_{\Gamma\Gamma}^{-1}S$. It is based on transforming the problem into a problem to find the largest eigenvalues and associated eigenvectors of a new matrix, and then approximate them using the randomized algorithms for low-rank matrix factorizations proposed in [21].

We do not present in this paper a parallel implementation of LORASC, this is the object of future work. However, both the construction and the application of the preconditioner are suitable for parallelism. In particular, even if the generalized eigenvalues problem in equation (2) involves all the domains, it relies on a summation and hence it can be solved by an iterative method in parallel. In addition, each term j in the summation involves only the vertices which connect the domain j to the separator Γ.

The paper is organized as follows, we introduce in Section 2 some notations and identify the Schur complement preconditioner. In Section 3 we consider the LORASC preconditioner, discuss its general properities and its application. We propose also different approach to obtain the LORASC preconditioner. Finally, in Section 4 we apply the LORASC preconditioner on a set of matrices arising from the discretization by the finite element methodes of linear elasticity models. Illustrative results conclude the robusteness and efficiency of LORASC preconditioner.

2 Preliminaries

In this section we introduce some notations and discuss the Schur complement preconditioner.

2.1 Direct methods of factorization

Given a matrix A of size $n \times n$, we refer to its spectrum $\Lambda(A)$ as

$$\Lambda(A) = \{\lambda_1, \lambda_2, ..., \lambda_n\},$$

where $\lambda_1 = \lambda_{min}(A)$ is its smallest eigenvalue and $\lambda_n = \lambda_{max}(A)$ is its largest eigenvalue. We denote the spectral condition number of a matrix A as $\kappa(A) := \lambda_{max}(A)/\lambda_{min}(A)$.

In the following we consider that the input matrix A has been reordered by using k-way graph partitioning with vertex separators. The obtained matrix A has a bordered block diagonal form, also referred

to as block arrow matrix,

$$A = \begin{pmatrix} A_{11} & & & A_{1\Gamma} \\ & \ddots & & \vdots \\ & & A_{NN} & A_{N\Gamma} \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & A_{\Gamma\Gamma} \end{pmatrix}, \tag{3}$$

where each diagonal block corresponds to a domain, while the last diagonal block $A_{\Gamma\Gamma}$ corresponds to the separator, the frontier between domains.

The matrix A can be factored as

$$A = \begin{pmatrix} A_{11} & & & \\ & \ddots & & \\ & & A_{NN} & \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & S \end{pmatrix} \begin{pmatrix} A_{11}^{-1} & & & \\ & \ddots & & \\ & & A_{NN}^{-1} & \\ & & & S^{-1} \end{pmatrix} \begin{pmatrix} A_{11} & & & A_{1\Gamma} \\ & \ddots & & \vdots \\ & & A_{NN} & A_{N\Gamma} \\ & & & S \end{pmatrix}, \tag{4}$$

where S is the Schur complement computed as

$$S = A_{\Gamma\Gamma} - \sum_{j=1}^{N} A_{\Gamma j} A_{jj}^{-1} A_{j\Gamma}. \tag{5}$$

Consequently, the factorization of A can be written as

$$A = (L + D)D^{-1}(D + L^T), \tag{6}$$

with $D = \text{Block-Diag}(A_{11}, A_{22}, ..., A_{NN}, S)$ and

$$L = \begin{pmatrix} 0 & & & \\ & \ddots & & \\ & & 0 & \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & 0 \end{pmatrix}, \tag{7}$$

Since for problems arising from the discretization of PDEs on large 3D grids, the Schur complement S becomes fairly dense, direct methods of factorization are prohibitive in terms of memory and computation costs. In our preconditioner we approximate S by a much sparser matrix $\tilde{S}$.

2.2 LORASC preconditioner

Consider a symmetric positive definite matrix A of size $n \times n$, which has a bordered block diagonal structure as in Equation (3). We refer in the following to the Schur complement preconditioner as M, and the preconditioned linear system that we solve is

$$M^{-1}Ax = M^{-1}b \tag{8}$$

The preconditioner M is defined by the following approximate factorization

$$\begin{aligned} M &= (L + \tilde{D})\tilde{D}^{-1}(\tilde{D} + L^T) \\ &= \begin{pmatrix} A_{11} & & & \\ & \ddots & & \\ & & A_{NN} & \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & \tilde{S} \end{pmatrix} \begin{pmatrix} A_{11}^{-1} & & & \\ & \ddots & & \\ & & A_{NN}^{-1} & \\ & & & \tilde{S}^{-1} \end{pmatrix} \begin{pmatrix} A_{11} & & & A_{1\Gamma} \\ & \ddots & & \vdots \\ & & A_{NN} & A_{N\Gamma} \\ & & & \tilde{S} \end{pmatrix}, \end{aligned} \tag{9}$$

where $\tilde{D} = \text{Block-Diag}(A_{11}, A_{22}, ..., A_{NN}, \tilde{S})$, L is defined as in Equation (7), and $\tilde{S}$ is an approximation of the Schur complement S from Equation (5).

It can be easily shown that the eigenvalue distribution of $M^{-1}A$ is equivalent to the eigenvalue distribution of $\tilde{S}^{-1}S$.

Lemma 2.1 *Let A a symmetric positive definite matrix of size $n \times n$, which can be factored as in Equation* (4)*. Let M the Schur complement preconditioner defined in Section 2.2. Then there is an equivalence between the spectrum of the matrix $M^{-1}A$ and $\tilde{S}^{-1}S$.*

Proof. The factorization (4) can be written as

$$A = \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ A_{\Gamma 1}A_{11}^{-1} & \cdots & A_{\Gamma N}A_{NN}^{-1} & I \end{pmatrix} \begin{pmatrix} A_{11} & & & \\ & \ddots & & \\ & & A_{NN} & \\ & & & S \end{pmatrix} \begin{pmatrix} I & & & A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix}.$$

Similarly, one can write the Schur complement preconditioner M as

$$M = \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ A_{\Gamma 1}A_{11}^{-1} & \cdots & A_{\Gamma N}A_{NN}^{-1} & I \end{pmatrix} \begin{pmatrix} A_{11} & & & \\ & \ddots & & \\ & & A_{NN} & \\ & & & \tilde{S} \end{pmatrix} \begin{pmatrix} I & & & A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix},$$

then

$$M^{-1} = \begin{pmatrix} I & & & -A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & -A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix} \begin{pmatrix} A_{11}^{-1} & & & \\ & \ddots & & \\ & & A_{NN}^{-1} & \\ & & & \tilde{S}^{-1} \end{pmatrix} \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ -A_{\Gamma 1}A_{11}^{-1} & \cdots & -A_{\Gamma N}A_{NN}^{-1} & I \end{pmatrix}.$$

A direct computation gives us

$$\begin{aligned} M^{-1}A &= \begin{pmatrix} I & & & A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix}^{-1} \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ & & & \tilde{S}^{-1}S \end{pmatrix} \begin{pmatrix} I & & & A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix} \\ &= \begin{pmatrix} I & & & X_1 \\ & \ddots & & \vdots \\ & & I & X_N \\ & & & \tilde{S}^{-1}S \end{pmatrix}, \end{aligned} \tag{10}$$

with $X_i = A_{ii}^{-1}A_{i\Gamma}(I - \tilde{S}^{-1}S)$. Then, the conclusion follows. □

According to Lemma 2.1, bounding the spectral condition number of $M^{-1}A$ requires bounding the spectral condition number of $\tilde{S}^{-1}S$. In the next section we give first the reasoning that leads to the definition of LORASC preconditioner, then we give its formal definition and prove bounds for the spectral condition number of $M^{-1}A$.

3 LORASC preconditioner

In this section we propose the LORASC preconditioner denoted by M_{LORASC}, prove upper bounds for the spectral condition number of $M_{\mathrm{LORASC}}^{-1}A$, discuss its application and its implemenation in parallel, and finally propose a different approach to obtain it based on randomized algorithms.

3.1 Algebra and analysis of LORASC preconditioner

We start from the observation that $\lambda_{max}(A_{\Gamma\Gamma}^{-1}S) \leq 1$. However, a preconditioner based only on $A_{\Gamma\Gamma}$ does not allow to lower bound the eigenvalues of $\tilde{S}^{-1}S$. To solve this problem, we use a formulation

of $\tilde{S}^{-1}$ that adds to $A_{\Gamma\Gamma}^{-1}$ a low rank matrix allowing to correct the smallest eigenvalues of $A_{\Gamma\Gamma}^{-1}S$. Our deflation method is inspired from the Wiedlandt's deflation technique explained in [20]. The correction matrix shifts the smallest eigenvalues of $A_{\Gamma\Gamma}^{-1}S$ to a prescribed lower bound $\varepsilon = \frac{1}{\tau}$. Note that since the application of M during the iterative process requires applying the inverse of $\tilde{S}$, in the following we discuss the formulation of $\tilde{S}^{-1}$ rather than the formulation of $\tilde{S}$.

In more details, we fix a threshold τ for the required spectral condition number $\kappa(\tilde{S}^{-1}S)$ which leads us to prescribe a lower bound $\varepsilon = \frac{1}{\tau}$ for the eigenvalues of $\tilde{S}^{-1}S$, as we know that $\lambda_{max}(A_{\Gamma\Gamma}^{-1}S) \leq 1$. We use the generalized eigenvalues problem

$$Su = \lambda A_{\Gamma\Gamma} u. \tag{11}$$

Let $\lambda_1, \lambda_2, ..., \lambda_i$ be the generalized eigenvalues that need to be corrected, i.e. $\lambda_k < \varepsilon$, $k \in \{1, 2, ..., i\}$, and let $v_1, v_2, ..., v_i$ be the corresponding $A_{\Gamma\Gamma}$-orthonormal generalized eigenvectors (see [22, Theorem 1.11]). The inverse of the approximation $\tilde{S}$ is defined as

$$\tilde{S}^{-1} = A_{\Gamma\Gamma}^{-1} + E_i \Sigma_i E_i^T. \tag{12}$$

where $E_i = (v_1 \quad v_2 \quad ... \quad v_i)$ and Σ_i is defined as

$$\Sigma_i = \mathrm{Diag}\left(\sigma_1, \sigma_2, \ldots, \sigma_i\right) \tag{13}$$

with $\sigma_1, \sigma_2, ..., \sigma_i$ chosen as

$$\sigma_k = \frac{\varepsilon - \lambda_k}{\lambda_k}, \qquad k \in \{1, 2, ..., i\}. \tag{14}$$

Then, one can easily prove (see Theorem 3.1 later in this section) that

$$\varepsilon \leq \lambda(\tilde{S}^{-1}S) \leq 1.$$

Hence the constructed matrix $\tilde{S}$ is a good approximation of S which ensures that the spectral condition number $\kappa(\tilde{S}^{-1}S)$ is bounded by a given tolerance $\tau = \frac{1}{\varepsilon}$.

Definition 3.1 (LORASC **preconditioner)** *Let A be an $n \times n$ symmetric positive definite matrix with a bordered block diagonal structure,*

$$A = \begin{pmatrix} A_{11} & & & A_{1\Gamma} \\ & \ddots & & \vdots \\ & & A_{NN} & A_{N\Gamma} \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & A_{\Gamma\Gamma} \end{pmatrix}. \tag{15}$$

Let $S = A_{\Gamma\Gamma} - \sum_{j=1}^{N} A_{\Gamma j} A_{jj}^{-1} A_{j\Gamma}$. Given a tolerance τ, a lower bound $\varepsilon = \frac{1}{\tau}$ for the generalized eigenvalues problem $Su = \lambda A_{\Gamma\Gamma} u$, let $\lambda_1, \lambda_2, ..., \lambda_i$ be the generalized eigenvalues smaller than ε, i.e. for all $k \in \{1, ..., i\}$ then $\lambda_k < \varepsilon$, and let $v_1, v_2, ..., v_i$ be the corresponding $A_{\Gamma\Gamma}$-orthonormal generalized eigenvectors.

The LORASC *preconditioner of A is defined as*

$$M_{\mathrm{LORASC}} := (L + \tilde{D})\tilde{D}^{-1}(\tilde{D} + L^T).$$

where L is given by (7) *and $\tilde{D} = \mathrm{Block{-}Diag}(A_{11}, A_{22}, ..., A_{NN}, \tilde{S})$. The matrix $\tilde{S}$ is defined as*

$$\tilde{S}^{-1} = A_{\Gamma\Gamma}^{-1} + E_i \Sigma_i E_i^T, \tag{16}$$

where E_i, Σ_i are defined as

$$E_i = (v_1 \quad v_2 \quad ... \quad v_i), \qquad \Sigma_i = \mathrm{Diag}\left(\sigma_1, \sigma_2, \ldots, \sigma_i\right).$$

The construction of the LORASC preconditioner is completely algebraic, since no information from the underlying PDE is required. Additionally, the following theorem proves that with the choice of $\tilde{S}$ given in Equation (16), the spectral condition number of the preconditioned matrix $\kappa(M_{\mathrm{LORASC}}^{-1}A)$ is upper bounded by a user defined value τ, and this gives a robust preconditioner.

Theorem 3.1 *Let A be a symmetric positive definite matrix with a bordered block diagonal structure as in Equation* (3) *and let M_{LORASC} be the* LORASC *preconditioner defined in Definition 3.1. Then $\kappa(M_{\mathrm{LORASC}}^{-1}A) \leq \tau$, where τ is a given tolerance used in the definition of M_{LORASC}.* **Proof.** *We start by proving that $\kappa(\tilde{S}^{-1}S) \leq \tau$. Let $\lambda_1, \lambda_2, ..., \lambda_i, \lambda_{i+1}, ..., \lambda_n$ be the generalized eigenvalues of the problem 11, with $\lambda_k \geq \varepsilon$ for all $k \in \{i+1, ..., n\}$ and let $v_1, v_2, ..., v_i, v_{i+1}, ..., v_n$ be the corresponding $A_{\Gamma\Gamma}$-orthonormal generalized eigenvectors. Then, for all $k \in \{1, ..., i\}$, using the definition of E_i, and Σ_i one can prove that*

$$\tilde{S}^{-1}Sv_k = A_{\Gamma\Gamma}^{-1}Sv_k + (E_i\Sigma_i E_i^T)Sv_k = \lambda_k v_k + \big(\frac{\varepsilon - \lambda_k}{\lambda_k}\big)\lambda_k v_k = \varepsilon v_k = \frac{1}{\tau}v_k$$

where we used the fact that the vectors of the matrix E_i are $A_{\Gamma\Gamma}$-orthonormal. Similarly, for all $k \in \{i+1, ..., n\}$, one has

$$\tilde{S}^{-1}Sv_k = A_{\Gamma\Gamma}^{-1}Sv_k + (E_i\Sigma_i E_i^T)Sv_k = \lambda_k v_k + 0 = \lambda_k v_k.$$

Recall that for all $k \in \{i+1, ..., n\}$, then $\lambda_k \leq 1$. Consequently, we conclude that

$$\kappa(\tilde{S}^{-1}S) \leq \tau.$$

The conclusion follows using Lemma 2.1. □

Since our deflation technique requires the construction of low rank approximation, the efficiency of our preconditioner depends on the number of eigenvalues that need to be deflated. We will see in the numerical experiments in Section 4 that the number of deflated eigenvalues grows slowly when increasing the number of the partitions N.

3.2 Application of LORASC preconditioner

We show in the following how to obtain a simplified application of the preconditioner. The preconditioned linear system that we need to solve is

$$M_{\mathrm{LORASC}}^{-1}Ax = M_{\mathrm{LORASC}}^{-1}b,$$

where b is a given right hand side. Before calling the Krylov subspace solver we will first start by computing the vector $d := M_{\mathrm{LORASC}}^{-1}b$. We need then to solve the system

$$M_{\mathrm{LORASC}}d = b. \tag{17}$$

Recall that the matrix M_{LORASC} is factored as

$$\begin{aligned} M_{\mathrm{LORASC}} &= (L + \tilde{D})\tilde{D}^{-1}(\tilde{D} + L^T) \\ &= (L + \tilde{D}) \cdot (I + \tilde{D}^{-1}L^T) \\ &=: \quad M_L \quad \cdot \quad M_U \end{aligned}$$

The two matrices M_L and M_U are lower and upper triangular matrices, respectively. The linear system (17) can then be transformed into

$$M_L y = b, \tag{18}$$

$$M_U d = y. \tag{19}$$

To solve the system (18), we need to solve

$$\begin{pmatrix} A_{11} & & & \\ & \ddots & & \\ & & A_{NN} & \\ A_{\Gamma 1} & \cdots & A_{\Gamma N} & \tilde{S} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_N \\ y_\Gamma \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_N \\ b_\Gamma \end{pmatrix}.$$

Therefore, we first compute for all $i \in \{1, 2, ..., N\}$ the vector y_i from the system $A_{ii}y_i = b_i$, then we compute the last vector y_Γ by solving the system

$$\tilde{S}y_\Gamma = z_\Gamma, \tag{20}$$

with $z_\Gamma := b_\Gamma - \sum_{i=1}^{N} A_{\Gamma i}y_i$. Now, owing the expression of $\tilde{S}^{-1}$ (Equation (16)), the system (20) is solved by computing a vector w_Γ such that $A_{\Gamma\Gamma}w_\Gamma = z_\Gamma$ and then $y_\Gamma = w_\Gamma + (E_i\Sigma_i E_i^T)z_\Gamma$.

Next, to solve the system (19), we write

$$\begin{pmatrix} I & & & A_{11}^{-1}A_{1\Gamma} \\ & \ddots & & \vdots \\ & & I & A_{NN}^{-1}A_{N\Gamma} \\ & & & I \end{pmatrix} \begin{pmatrix} d_1 \\ \vdots \\ d_N \\ d_\Gamma \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \\ y_\Gamma \end{pmatrix},$$

then $d_\Gamma = y_\Gamma$ and the other required vectors are computed locally, for all $i \in \{1, 2, ..., N\}$, by solving $d_i = y_i - c_i$, where c_i is the solution of the system $A_{ii}c_i = A_{i\Gamma}y_\Gamma$. Note that until this stage there was no call to the Krylov subspace solver, that is the previous computation is done only once, before calling the Krylov subspace solver.

The system to solve now is $M_{\mathrm{LORASC}}^{-1}Ax = d$. Owing to (10), we need to solve

$$\begin{pmatrix} I & & & X_1 \\ & \ddots & & \vdots \\ & & I & X_N \\ & & & \tilde{S}^{-1}S \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_N \\ x_\Gamma \end{pmatrix} = \begin{pmatrix} d_1 \\ \vdots \\ d_N \\ d_\Gamma \end{pmatrix}$$

with $X_i = A_{ii}^{-1}A_{i\Gamma}(I - \tilde{S}^{-1}S)$, then the Krylov subspace solver is called to solve only the system $\tilde{S}^{-1}Sx_\Gamma = d_\Gamma$. Recall that $d_\Gamma = y_\Gamma$ and y_Γ is obtained from (20). Consequently, the system to solve is $\tilde{S}^{-1}Sx_\Gamma = \tilde{S}^{-1}z_\Gamma$. Then to apply the preconditioner $\tilde{S}^{-1}$ we need to communicate to the solver, for a given vector b_s, the result of solving the system $\tilde{S}x_s = b_s$, which is the same system (20) that we have already explained how to compute simply in practice. Finally, to compute x_i for all $i \in \{1, 2, ..., N\}$ we have

$$x_i + \big(A_{ii}^{-1}A_{i\Gamma}(I - \tilde{S}^{-1}S)\big)x_\Gamma = d_i,$$

then

$$x_i = d_i - A_{ii}^{-1}A_{i\Gamma}x_\Gamma + A_{ii}^{-1}A_{i\Gamma}d_\Gamma.$$

Consequently,

$$x_i = d_i - \bar{x}_\Gamma + \bar{d}_\Gamma,$$

where $\bar{x}_\Gamma$ and $\bar{d}_\Gamma$ are the solutions of the systems $A_{ii}\bar{x}_\Gamma = A_{i\Gamma}x_\Gamma$ and $A_{ii}\bar{d}_\Gamma = A_{i\Gamma}d_\Gamma$, respectively.

3.3 Computing LORASC preconditioner using randomized algorithms

In this section we discuss the usage of randomized algorithms instead of the generalized eigenvalues problem for the construction of LORASC preconditioner. We consider the randomized algorithms with sharp performance bounds that were analyzed in [21] in the context of low-rank matrix factorizations. Algorithm 1 is one of the algorithms studied in [21] to approximate the singular value decomposition of a given matrix B. Note that if the matrix B is SPD, then the diagonal of the resulting matrix Σ approximates the largest eigenvalues of B.

Algorithm 1 Randomized algorithm

Require: Input: $m \times n$ matrix B and desired rank l.

1: Sample an $n \times l$ test matrix G with independent mean-zero, unit-variance Gaussian entries.
2: Compute $H = (BB^*)^q BG$
3: Construct $Q \in \mathbb{R}^{m \times l}$ with columns forming an orthonormal basis for the range of H.
4: Compute $C = Q^* B$
5: Compute the SVD of $C = \hat{U}\Sigma V^*$
return the approximation $U = Q\hat{U}, \Sigma, V$

The most expensive operation in Algorithm 1 is the computation of H. However, the benifit here is that we compute a matrix-matrix product directly instead of computing several matrix-vector products when solving the generalized eigenvalues problem 11. Note also that the computational cost of the singular value decomposition of the matrix C is on the order of $O(l^2 n)$ flops. In the numerical experiments of Section 4 we discuss the choice of l and its influnce on convergence results.

In order to use Algorithm 1 taken from [21], the problem of computing the smallest eigenvalues and associated eigenvectors of $A_{\Gamma\Gamma}^{-1} S$ needs to be transformed into the problem of finding the largest eigenvalues and associated eigenvectors of a different SPD matrix. The generalized eigenvalues problem 11 can be written as follows

$$A_{\Gamma\Gamma} u - S u = A_{\Gamma\Gamma} u - \lambda A_{\Gamma\Gamma} u,$$

equivalently,

$$A_{\Gamma\Gamma}^{-1}(A_{\Gamma\Gamma} - S)u = \zeta u,$$

where $\zeta = 1 - \lambda$. Since $0 < \lambda \leq 1$, we have that $0 \leq \zeta < 1$ and the smallest eigenvalues λ_i correspond to the largest eigenvalues ζ_i. Finally, as $A_{\Gamma\Gamma}$ is SPD, it can be written as $A_{\Gamma\Gamma} = R^T R$ and we obtain

$$\begin{aligned} R^{-1}R^{-T}(A_{\Gamma\Gamma} - S)u &= \zeta u, \\ R^{-1}R^{-T}(A_{\Gamma\Gamma} - S)\cdot(R^{-1}R)u &= \zeta u, \\ R^{-T}(A_{\Gamma\Gamma} - S)R^{-1}\bar{u} &= \zeta \bar{u}, \qquad \bar{u} = Ru. \end{aligned} \tag{21}$$

Note that, for an eigenvector $\bar{v}_i$ that corresponds to an eigenvalue ζ_i in the system (21), then

$$R^{-T} S R^{-1} \bar{v}_i = \lambda_i \bar{v}_i. \tag{22}$$

and $v_i = R^{-1}\bar{v}_i$ is an eigenvector that corresponds to the eigenvalue λ_i of the system 11. Therefore, at this stage we need to compute/approximate the largest eigenvalues of the symmetric matrix $B = R^{-T}(A_{\Gamma\Gamma} - S)R^{-1}$. Whenever we approximate the largest eigenvalues ζ_i of the matrix B we compute the corresponding λ_i and then we construct the approximation matrix $\tilde{S}$ following (12). Numerical results in Section 4 discuss the efficiency of this approach.

4 Numerical results

In this section we analyze the efficiency of the LORASC preconditioner on a set of matrices arising from the discretization by the finite element methods of linear elasticity models, with highly heterogeneous elastic moduli, on two-dimensional (2D) and three-dimensional (3D) domains. We define the test cases and build the different matrices via FreeFem++ [23], we use METIS [24] as a graph partitioner, and we build our LORASC preconditioner using MatLab. To solve the linear systems we use CG via MatLab with the threshold 10^{-8} for the stopping criteria of the algebraic iterative resolution. The smallest eigenvalues and associated eigenvectors of the generalized eigenvalues problem are computed using either ARPACK or the randomized algorithm 1 described in Section 3.3.

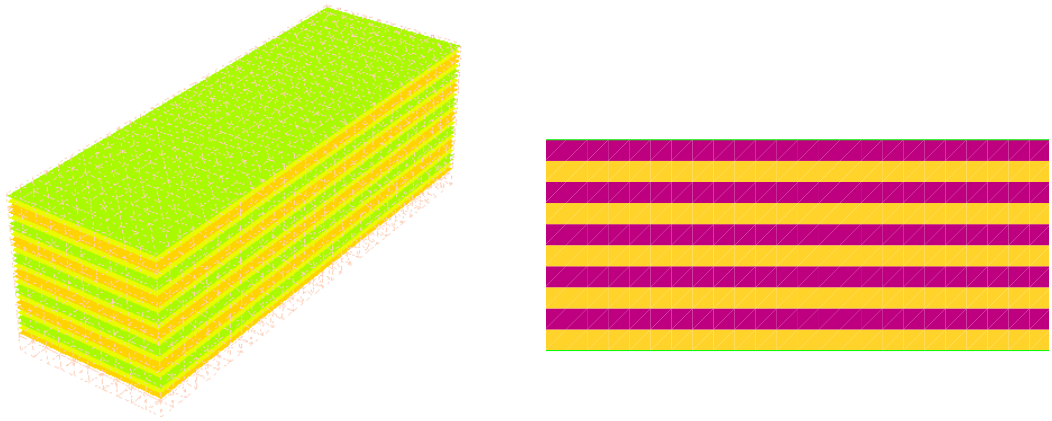
4.1 Linear elasticity models

Let Ω be a d-dimensional polygonal or polyhedral domain ($d = 2$ or 3). The linear elasticity problem in infinitesimal strain theory may be written as follows:

$$\begin{aligned} \operatorname{div}(\sigma(u)) + f &= 0 && \text{on } \Omega, && (23)\\ u &= u_D && \text{on } \partial\Omega_D, && (24)\\ \sigma(u) \cdot n &= g && \text{on } \partial\Omega_N, && (25) \end{aligned}$$

where $u \in \mathbb{R}^d$ is the unknown displacement field, the Dirichlet boundaries $\partial\Omega_D = \{(x, y, z) \in \partial\Omega, x = 0\}$ and the remaining boundaries $\partial\Omega_N$ are the Neumann boundaries, and f is some body force. The Cauchy stress tensor $\sigma(u)$ is given by Hooke's law $\sigma(u) = 2\mu\epsilon(u) + \lambda \mathrm{Tr}(\epsilon(u))I$, where Tr is the trace, μ, λ are the Lamé parameters, and are a property of the elastic material, and $\epsilon(u) = \frac{1}{2}(\nabla u + \nabla u^T)$ is the strain tensor. Note that μ and λ can be expressed in terms of Young's modulus E and Poisson's ratio ν as

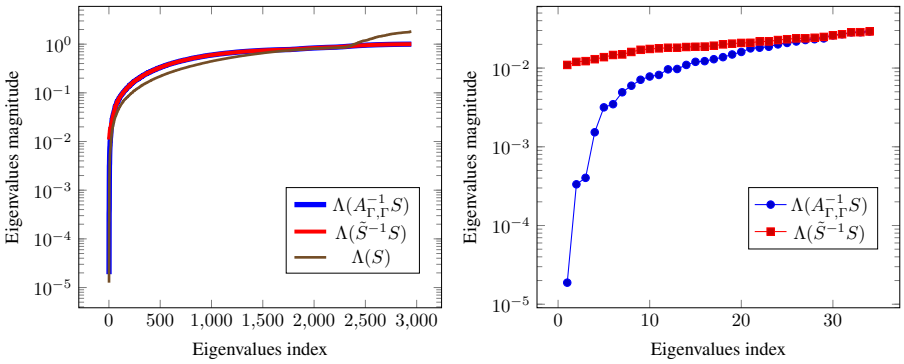
$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}.$$



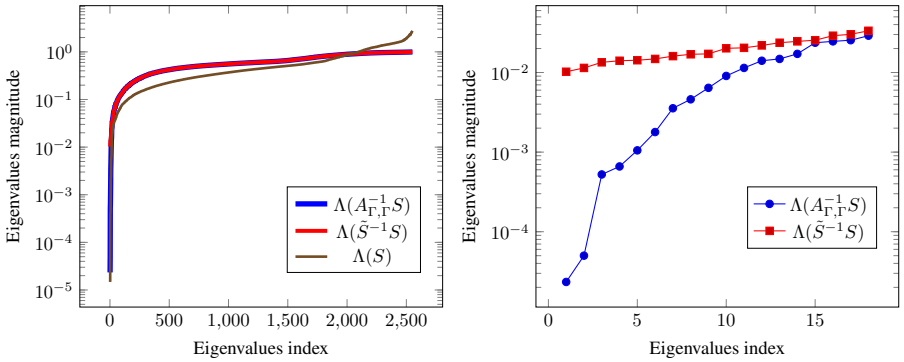
(a) 3D case (b) 2D case

Figure 1: The distribution of the Young's modulus.

The numerical efficiency and robustness of our preconditioner is tested for the two- and three-dimensional systems of linear elasticity on a rectangular and parallelepiped domain, respectively. The domains are discretized with a triangular mesh and $\mathbb{P}_1$ finite elements. The Young's modulus E and Poisson's ratio ν take two values, $(E_1, \nu_1) = (2 \cdot 10^{11}, 0.25)$, and $(E_2, \nu_2) = (10^7, 0.45)$, the distribution is shown in Figure 1.



(a) 2D, $size(S) = 5902$ (b) zoom on values in < 0.03



(c) 3D, $size(S) = 5082$ (d) zoom on values in < 0.03

Figure 2: Spectrum of the Schur complement matrix S and the preconditioned operators $A_{\Gamma,\Gamma}^{-1}S$ and $\tilde{S}^{-1}S$, for the 2D (top) and 3D (bottom) test cases. For LORASC, the parameter $\tau = 10^2$.

In what follows we refer to the number of partitions as N_p, the number of nonzero elements as nnz, the number of deflated eigenvalues as n_{EV}. We also display in the following tables N_{mult}, the number of matrix-vector multiplications used by ARPACK to compute the required eigenvalues with a fixed tolerance of 10^{-3}, and $\text{iter}_{B^{-1}}$, the number of iterations of the algebraic resolution of CG with the application of the preconditioner B^{-1} from equation (9) with $\tilde{S}^{-1} = B^{-1}$, where B^{-1} can be either the LORASC preconditioner or a block diagonal preconditioner. In the block diagonal preconditioner, $\tilde{S}^{-1} = A_{\Gamma,\Gamma}^{-1}$.

We start by illustrating the effect of LORASC preconditioner on small eigenvalues. Figure 2 shows the spectrum of the Schur complement matrix S and the preconditioned operators $A_{\Gamma,\Gamma}^{-1}S$ and $\tilde{S}^{-1}S$. The construction of LORASC uses a threshold $\tau = 10^2$, i.e. the lower bound for the eigenvalues is $\varepsilon = 0.01$. The results are presented for two different matrices of size $n \times n$: at the top, the matrix corresponds to a two-dimensional system of linear elasticity with $n = 50702$ and 604200 non zero elements, on the bottom, the matrix corresponds to a three-dimensional system with $n = 18513$ and 618747 nonzero elements. As expected, we remark that LORASC deflates the small eigenvalues, and thus the eigenvalues of the preconditioned matrix are lower bounded by 10^{-2}. This leads to a good convergence rate as it will be seen later.

4.2 Generalized eigenvalues problem

We first assess the efficiency of the LORASC preconditioner computed by using the generalized eigenvalues problem detailed in Section 3.1. As mentioned previously, the smallest eigenvalues and associated eigenvectors of the generalized eigenvalues problem are computed using ARPACK with a fixed tolerance of 10^{-3}. The numerical results focus on testing the behavior of the LORASC preconditioner in terms of weak and strong scalability. In the weak scaling experiments, the size of the problem increases proportionally with the partitions number. In the strong scaling experiments, the problem size is fixed while the number of partitions increases.

4.2.1 Two-dimensional test case

Table 1 collects the results of the number of deflated eigenvalues n_{EV} and the number of matrix-vector multiplications N_{mult} required to compute these eigenvalues via ARPACK. The results are given for weak scaling with two different thresholds $\tau = 10^2, \tau = 2 \cdot 10^2$. The corresponding lower bounds for the eigenvalues of the preconditioned matrix are $\varepsilon = 10^{-2}, \varepsilon = 5 \cdot 10^{-3}$, respectively. The last column gives the required number of iterations to solve the global system without our deflation technique, that is only a block diagonal preconditioner is used and thus the inverse of the Schur complement S is approximated by $A_{\Gamma,\Gamma}^{-1}$. We remark that with LORASC, few eigenvalues need to be deflated when $\varepsilon = 5 \cdot 10^{-3}$. Even if more eigenvalues need to be deflated for $\tau = 10^2$, the size of the deflation space increases slightly with the number of partitions. To compare the block diagonal preconditioner with LORASC, we compare the num-

			$\tilde{S}^{-1}, \varepsilon = 0.01$			$\tilde{S}^{-1}, \varepsilon = 0.005$			$A_{\Gamma,\Gamma}^{-1}$
n	N_p	nnz	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	$\text{iter}_{A_{\Gamma,\Gamma}^{-1}}$
13122	8	155520	3	83	67	2	83	76	86
25452	16	302700	6	83	71	3	83	86	138
50702	32	604200	13	160	67	7	132	77	346
101202	64	1207200	23	294	55	14	160	65	538
202202	128	2413200	52	540	41	28	233	54	742

Table 1: Weak scaling results for 2D test cases. The results for LORASC are given in the columns $\tilde{S}^{-1}$ for two values of the parameter ε. The results for the block diagonal preconditioner are given in the column $A_{\Gamma,\Gamma}^{-1}$.

ber of iterations of CG preconditioned by the block diagonal preconditioner with those of LORASC added to the number of matrix-vector multiplications required by ARPACK to compute the smallest eigenvalues that need to be deflated ($N_{\text{mult}}+$ $\text{iter}_{\tilde{S}^{-1}}$). This comparison allows to estimate the gain obtained with our deflation methods with respect to a block diagonal preconditioner in terms of matrix-vector operations coming from both the construction of LORASC (approximating eigenvalues/eigenvectors through ARPACK) and the iterative CG solver. This gain becomes more important when we increase the number of partitions and the problem size, and reaches a factor of roughly 2 for the choice of $\varepsilon = 5 \cdot 10^{-3}$. We also mention that we have tried another values of the parameter ε, however the detailed results are not presented here. We found that for $\varepsilon = 10^{-1}$, the required number of multiplications N_{mult} increases significantly, and the overall gain obtained is not important. For $\varepsilon = 10^{-3}$, the number of iteration of CG preconditioned by LORASC becomes important, and no significant gain is obtained when comparing with the number of iterations of CG with the block diagonal preconditioner.

Table 2 displays strong scaling results in terms of number of deflated eigenvalues and number of iterations of preconditioned CG by LORASC and by the block diagonal preconditioner. We observe that the size of the deflation space is increasing slowly with the number of partitions. An important gain is

n	N_p	nnz	$\tilde{S}^{-1}, \varepsilon = 0.01$			$\tilde{S}^{-1}, \varepsilon = 0.005$			$A_{\Gamma,\Gamma}^{-1}$
			n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	$\text{iter}_{A_{\Gamma,\Gamma}^{-1}}$
202202	2	2413200	6	98	55	4	83	74	105
202202	4	2413200	9	102	53	6	98	69	185
202202	8	2413200	16	151	50	10	107	64	246
202202	16	2413200	21	185	46	13	119	65	322
202202	32	2413200	28	230	45	16	151	62	401
202202	64	2413200	36	372	44	21	185	59	521
202202	128	2413200	52	540	41	28	233	54	742

Table 2: Strong scaling results for 2D test cases. The results for LORASC are given in the columns $\tilde{S}^{-1}$ for two values of the parameter ε. The results for the block diagonal preconditioner are given in the column $A_{\Gamma,\Gamma}^{-1}$.

obtained by LORASC preconditioner with respect to the block diagonal preconditioner, and this gain is more significant when the partition number is increased and reaches a factor of roughly 2 for the choice of $\varepsilon = 5 \cdot 10^{-3}$.

4.2.2 Three-dimensional test case

As for the two-dimensional test case we start by illustrating the behavior of the LORASC preconditioner in terms of weak scaling. Table 3 displays the number of deflated eigenvalues and the number of matrix-vector multiplications performed by ARPACK for the two different thresholds $\tau = 10^2, \tau = 2 \cdot 10^2$. The number of matrix-vector multiplications required in the case of $\varepsilon = 10^{-2}$ and $\varepsilon = 2 \cdot 10^{-3}$ increases slightly. Overall an important gain is obtained by LORASC with respect to the block diagonal preconditioner that uses $A_{\Gamma,\Gamma}^{-1}$ only to approximate $\tilde{S}^{-1}$.

n	N_p	nnz	$\tilde{S}^{-1}, \varepsilon = 0.01$			$\tilde{S}^{-1}, \varepsilon = 0.005$			$A_{\Gamma,\Gamma}^{-1}$
			n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	$\text{iter}_{A_{\Gamma,\Gamma}^{-1}}$
4719	2	153057	0	0	71	0	0	71	71
9438	4	312372	5	92	65	3	83	89	113
18513	8	618747	10	111	63	8	95	84	207
36663	16	1231497	15	132	60	11	111	76	267
72963	32	2456997	42	325	55	24	230	64	592

Table 3: Weak scaling results for 3D test cases. The results for LORASC are given in the columns $\tilde{S}^{-1}$ for two values of the parameter ε. The results for the block diagonal preconditioner are given in the column $A_{\Gamma,\Gamma}^{-1}$.

Strong scaling results are presented in Table 4. We note that the size of the deflation space and the number of matrix-vector multiplications performed by ARPACK increases slightly for the two choices of the parameter ε. When comparing the number of matrix-vector multiplications needed to deflate the required smallest eigenvalues plus the number of iterations of LORASC ($N_{\text{mult}}+ \text{iter}_{\tilde{S}^{-1}}$), with the number of iterations of the block diagonal preconditioner ($\text{iter}_{A_{\Gamma,\Gamma}^{-1}}$), we observe that LORASC outperforms the diagonal block precondiitoner. The gain obtained by LORASC is more important when the partition number increases, and a factor of roughly 2 is obtained for $\varepsilon = 5 \cdot 10^{-3}$.

Finally, we mention that similar results with the 2D case are obtained when we use more rough ($\varepsilon = 10^{-1}$) or smooth ($\varepsilon = 10^{-3}$) lower bound, for both weak and strong scaling. That is, no significant

gain is obtained by LORASC with respect to the block diagonal preconditioner.

			$\tilde{S}^{-1}, \varepsilon = 0.01$			$\tilde{S}^{-1}, \varepsilon = 0.005$			$A_{\Gamma,\Gamma}^{-1}$
n	N_p	nnz	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	n_{EV}	N_{mult}	$\text{iter}_{\tilde{S}^{-1}}$	$\text{iter}_{A_{\Gamma,\Gamma}^{-1}}$
72963	2	2456997	6	83	58	4	83	74	87
72963	4	2456997	13	119	65	8	110	75	168
72963	8	2456997	23	202	61	13	119	74	322
72963	16	2456997	32	249	61	18	159	69	465
72963	32	2456997	42	325	55	24	230	64	592

Table 4: Strong scaling results for 3D test cases. The results for LORASC are given in the columns $\tilde{S}^{-1}$ for two values of the parameter ε. The results for the block diagonal preconditioner are given in the column $A_{\Gamma,\Gamma}^{-1}$.

4.3 Randomized algorithms

In this section we assess the performance of our LORASC preconditioner when the low rank approximation is computed by using the randomized algorithm 1 disscused in Section 3.3, with $q = 2$ (see [21, Algorithm 3]).

4.3.1 Two-dimensional test case

Table 5 collects the weak scaling results. As explained previousely in Section 3.3, the parameter $l = k+p$ in the table is the desired rank. In our case it corresponds to the computed eigenvalues that will be then filtered in order to deflate just the required eigenvalues following the choice of the threshold τ, related to the lower bound ε. The results in Table 5 show that with the choice of $k = p$ which corresponds to

		$l = k + p, k = p = \varepsilon \times size(S)$				$l = k + p, k = p = 10^{-1} size(S)$		
		$\varepsilon = 0.01$		$\varepsilon = 0.005$			$\varepsilon = 0.01$	$\varepsilon = 0.005$
n	N_p	l	iter	l	iter	l	iter	iter
13122	8	12	86	6	86	119	65	75
25452	16	28	138	14	138	280	70	85
50702	32	59	228	30	228	590	61	79
101202	64	122	395	62	395	1222	54	64
202202	128	250	663	125	663	2495	42	55

Table 5: 2D: Randomized algorithm (weak scaling)

$0.1 \times size(S)$, we obtain the same rate of convergence for the resolution of the linear elasticity model that we have had with the method based on solving the generalized eigenvalues problem. However, if we choose a smaller rank such as $k = p = \varepsilon \times size(S)$, the precision of the approximation of the required eigenvalues is lost and the deflation method will not be efficient. This leads to a slow convegence, close to the convergence obtained with the block diagonal preconditioner that uses $A_{\Gamma,\Gamma}^{-1}$ only. Therefore, to ensure the efficiency of the method we need to choose an important size of the low rank approximation $l = k + p$ with $k = p = 0.1 \times size(S)$. Note that, however, an important difference with the generalized eigenvalues method where we have to perform an important number of matrix-vector multiplications. In the case of the randomized algorithm we perform the multipliction of the matrix to approximate with a

		$l = k + p, k = p = \varepsilon \times size(S)$				$l = k + p, k = p = 10^{-1} size(S)$		
		$\varepsilon = 0.01$		$\varepsilon = 0.005$			$\varepsilon = 0.01$	$\varepsilon = 0.005$
n	N_p	l	iter	l	iter	l	iter	iter
202202	2	4	79	2	79	40	56	76
202202	4	12	170	6	170	121	53	72
202202	8	28	223	14	223	284	51	62
202202	16	63	299	32	299	631	49	62
202202	32	102	391	51	391	1024	45	60
202202	64	164	504	82	504	1639	44	61
202202	128	250	663	125	663	2495	42	55

Table 6: 2D: Randomized algorithm (strong scaling)

set of vectors at once, and this operation is more efficient than the matrix-vector multiplication on modern architectures, especially if we deal with matrices of large size.

The strong scaling is investigated in Table 6. As for weak scaling, we observe that when we choose a sufficient size of the low rank approximation we obtain similar results with those obtained with the generalized eigenvalues method. That is, the same rate of convergence is obtained and the number of iterations of CG preconditioned with LORASC does not increase when the partition number increases, which ensures good scalability properties. The sensibilty of the randomized algorithm with respect to the size of the low-rank approximation can be also remarked.

4.3.2 Three-dimensional test case

Table 7 displays the weak scaling results of our LORASC preconditioner on the three-dimensional system of linear elasticity. The randomized algorithm 1 of Section 3.3 is used for the construction of LORASC. The sensibility of the randomized algorithm can be again observed, in the sense that a small size of the low-rank approximation leads to a loss of precision of the deflation method, which which in turn leads to a slow convergence of our preconditioner. On the other hand, the choice of the desired rank l as $l = k + p$ with $k = p = 0.1 \times size(S)$ leads to similar results with those of LORASC based on solving the generalized eigenvalues problem via ARPACK.

		$l = k + p, k = p = \varepsilon \times size(S)$				$l = k + p, k = p = 10^{-1} size(S)$		
		$\varepsilon = 0.01$		$\varepsilon = 0.005$			$\varepsilon = 0.01$	$\varepsilon = 0.005$
n	N_p	l	iter	l	iter	l	iter	iter
4719	2	8	71	4	71	73	71	71
9438	4	21	113	11	113	211	69	85
18513	8	51	191	26	191	508	66	79
36663	16	108	241	54	241	1082	62	75
72963	32	220	563	110	563	2192	57	61

Table 7: 3D: Randomized algorithm (weak scaling)

Finally, Table 8 displays weak scaling results. A similar behavior is obtained, the sensibility of the randomized algorithm is obserevd as for the previous test cases, and a fast convergence of CG preconditioned with LORASC is ensured by choosing a sufficient size of the low rank approximations.

		$l = k + p, k = p = \varepsilon \times size(S)$				$l = k + p, k = p = 10^{-1} size(S)$		
		$\varepsilon = 0.01$		$\varepsilon = 0.005$			$\varepsilon = 0.01$	$\varepsilon = 0.005$
n	N_p	l	iter	l	iter	l	iter	iter
72963	2	7	78	4	78	73	56	71
72963	4	22	137	11	137	218	64	73
72963	8	58	268	29	268	581	59	72
72963	16	109	399	55	399	1089	58	68
72963	32	220	563	110	563	2192	57	61

Table 8: 3D: Randomized algorithm (strong scaling)

References

[1] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," Journal of Research of the National Bureau of Standards, vol. 49, pp. 409–436, Dec. 1952.

[2] A. Sluis and H. Vorst, "The rate of convergence of conjugate gradients," Numerische Mathematik, vol. 48, no. 5, pp. 543–560, 1986.

[3] R. A. Nicolaides, "Deflation of conjugate gradients with applications to boundary value problems," SIAM Journal on Numerical Analysis, vol. 24, pp. 355–365, Apr. 1987.

[4] Y. Saad, M. Yeung, J. Erhel, and F. Guyomarc'h, "A deflated version of the conjugate gradient algorithm," SIAM J. Sci. Comput., vol. 21, no. 5, pp. 1909–1926 (electronic), 2000. Iterative methods for solving systems of algebraic equations (Copper Mountain, CO, 1998).

[5] M. Gutknecht, "Spectral deflation in Krylov solvers: A theory of coordinate space based methods," Electron.Trans. Numer. Anal., vol. 39, pp. 156–185, 2012.

[6] J. M. Tang, R. Nabben, C. Vuik, and Y. A. Erlangga, "Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods," J. Sci. Comput., vol. 39, pp. 340–370, 2009.

[7] C. Wagner and G. Wittum, "Adaptive filtering," Numerische Mathematik, vol. 78, no. 2, pp. 305–328, 1997.

[8] Y. Achdou and F. Nataf, "An iterated tangential filtering decomposition," Numerical Linear Algebra with Applications, vol. 10, no. 5-6, pp. 511–539, 2003. Preconditioning, 2001 (Tahoe City, CA).

[9] J. Appleyard and I. Cheshire, "Nested factorization," in Seventh SPE Symposium on Reservoir Simulation, pp. 315–324, 1983. paper number 12264.

[10] Q. Niu, L. Grigori, P. Kumar, and F. Nataf, "Modified tangential frequency filtering decomposition and its Fourier analysis," Numerische Mathematik, vol. 116, no. 1, pp. 123–148, 2010.

[11] R. Fezzani, L. Grigori, F. Nataf, and K. Wang, "Block filtering decomposition," Numerical Linear Algebra with Applications Journal, 2014. to appeari.

[12] L. Qu, L. Grigori, and F. Nataf, "Parallel design and performance of nested filtering factorization preconditioner," in Proceedings of the ACM/IEEE Supercomputing SC12 Conference, 2013.

[13] M. Gu, X. S. Li, and P. S. Vassilevski, "Direction-preserving and schur-monotonic semiseparable approximations of symmetric positive definite matrices," SIAM J. Matrix Anal. Appl., vol. 31, pp. 2650–2664, Sept. 2010.

[14] A. Brandt, J. Brannick, K. Kahl, and I. Livshits, "Bootstrap AMG," SIAM Journal on Scientific Computing, vol. 33, no. 2, pp. 612–632, 2011.

[15] Y. Efendiev, J. Galvis, R. Lazarov, S. Margenov, and J. Ren, "Robust two-level domain decomposition preconditioners for high-contrast anisotropic flows in multiscale media," Comput. Methods Appl. Math., vol. 12, no. 4, pp. 415–436, 2012.

[16] F. Nataf, H. Xiang, V. Dolean, and N. Spillane, "A coarse space construction based on local Dirichlet-to-Neumann maps," SIAM J. Sci. Comput., vol. 33, no. 4, pp. 1623–1642, 2011.

[17] F. Nataf, F. Hecht, P. Jolivet, and C. Prud'Homme, "Scalable Domain Decomposition Preconditioners For Heterogeneous Elliptic Problems," in SC13 - International Conference for High Performance Computing, Networking, Storage and Analysis, (Denver, Colorado, United States), p. 11 p., ACM, Nov. 2013.

[18] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl, "Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps," Numer. Math., vol. 126, no. 4, pp. 741–770, 2014.

[19] A. Napov and Y. Notay, "An algebraic multigrid method with guaranteed convergence rate," SIAM J. Sci. Comput., vol. 34, no. 2, pp. A1079–A1109, 2012.

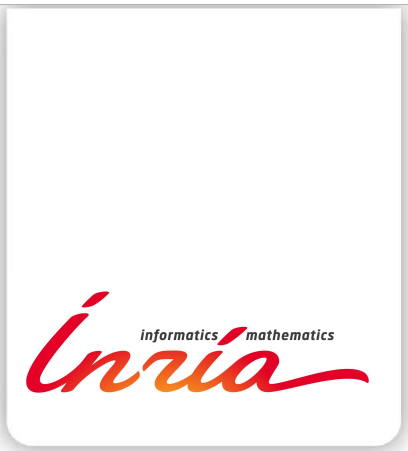
[20] Y. Saad, "Numerical solution of large nonsymmetric eigenvalue problems," Comput. Phys. Comm., vol. 53, no. 1-3, pp. 71–90, 1989. Practical iterative methods for large scale computations (Minneapolis, MN, 1988).

[21] R. Witten and E. Candes, "Randomized algorithms for low-rank matrix factorizations: sharp performance bounds," arXiv preprint arXiv:1308.5697, 2013.

[22] M. Lucchesi, Masonry constructions: mechanical models and numerical applications, vol. 39. Springer, 2008.

[23] F. Hecht, "New development in freefem++," J. Numer. Math., vol. 20, no. 3-4, pp. 251–265, 2012.

[24] G. Karypis and V. Kumar, "A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices," University of Minnesota, Department of Computer Science and Engineering, 1998.



RESEARCH CENTRE
PARIS – ROCQUENCOURT

Domaine de Voluceau, - Rocquencourt
B.P. 105 - 78153 Le Chesnay Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399