A GenEO Domain Decomposition method for Saddle Point problems

Frédéric Nataf, Pierre-Henri H Tournier

A GenEO Domain Decomposition method for Saddle Point problems

F. Nataf [*1] and P.H. Tournier [†2]

[1,2]Sorbonne Université, CNRS, Université de Paris, Inria Equipe Alpines, Laboratoire Jacques-Louis Lions, F-75005 Paris, France,

December 21, 2021

Contents

[*]frederic.nataf@sorbonne-universite.fr
[†]tournier@ljll.math.upmc.fr

Abstract

We introduce an adaptive element-based domain decomposition (DD) method for solving saddle point problems defined as a block two by two matrix. The algorithm does not require any knowledge of the constrained space. We assume that all sub matrices are sparse and that the diagonal blocks are spectrally equivalent to a sum of positive semi definite matrices. The latter assumption enables the design of adaptive coarse space for DD methods that extends the GenEO theory [32] to saddle point problems. Numerical results on three dimensional elasticity problems for steel-rubber structures discretized by a finite element with continuous pressure are shown for up to one billion degrees of freedom.

1 Introduction

Solving saddle point problems with parallel algorithms is very important for many branches of scientific computing: fluid and solid mechanics, computational electromagnetism, inverse problems and optimization.

We are interested in domain decomposition (DD) methods since they are naturally well-fitted to modern parallel architectures. For specific systems of partial differential equations with a saddle point formulation, efficient DD methods have been designed, see e.g., [28, 23, 29] and [34] references therein. Also in [16], a GenEO coarse space is introduced for the P.L. Lions' algorithm and its efficiency is mathematically proved for symmetric definite positive problems. In the above article, numerical experiments are conducted on three dimensional elasticity problems for steel-rubber structures discretized by a finite element with continuous pressure. Although the method works well in practice, the method lacks theoretical convergence guarantees and also demands the design of specific absorbing conditions as interface conditions.

As for a convergence rate analysis for a discretization with a continuous pressure, the recent article [37] generalizes the theory developed in [36] to the case of nonzero pressure block but under the assumption that the discontinuities are resolved by the subdomains.

Compared to the above mentioned works, the method we propose has a provable control on the condition number for zero or non zero pressure block with a continuously discretized pressure also in the case arbitrary heterogeneities and bypasses the need for absorbing boundary conditions.

Here as in [26, 4, 9, 30], we consider the problem in the form of a two by two block matrix. Let m and n be two integers with $m < n$. Let A $n \times n$ SPD matrix and B be a sparse $m \times n$ full rank matrix of constraints and C a $m \times m$ non negative matrix (in particular, $C = 0$ is allowed), we consider the following saddle point matrix:

$$\mathcal{A} := \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}. \tag{1}$$

When the kernel of matrix B is known, very efficient multigrid methods have been designed in the context of finite element methods, see e.g., [7, 19, 18, 2, 31, 14]. Without this knowledge, it is nevertheless also possible to design efficient geometric multigrid methods as in [11] where the fine mesh is obtained by several uniform mesh refinements.

Here we do not assume any knowledge on the kernel of matrix B and we work with arbitrary meshes. The following three factor factorization, see e.g., [5]:

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} = \begin{pmatrix} I & 0 \\ BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & -(C + BA^{-1}B^T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B^T \\ 0 & I \end{pmatrix},$$

shows that solving the linear system with $\mathcal{A}$ can be performed by solving sequentially linear systems with A and one with the Schur complement $C + BA^{-1}B^T$. In order to build a scalable method, we assume that all three matrices A, B and C are sparse and that A and C are the sum of positive semi definite matrices. This is easily achieved in finite element or finite volume contexts for partial differential equations. The latter assumption enables the design of adaptive coarse space for DD methods, see [10].

More precisely, in § 2, we recall the two-level additive Schwarz method denoted M_A used to precondition the matrix A (the primal-primal block of the saddle point problem). Then in § 3, we introduce the operator $P_S := C + BM_A^{-1}B^T$ which is spectrally equivalent to the Schur complement S. Its preconditioning is studied in § 3.2. In § 4, we combine these different components to define in a compact way the parallel saddle point preconditioner. In § 5, we present weak and strong scaling experiments on large scale elasticity problems for steel-rubber structures discretized by a finite element with continuous pressure. These problems are highly heterogeneous since the Lamé-Poisson coefficients of the rubber are $(E_1, \nu_1) = (1 \times 10^7, 0.4999)$ and those of the steel are $(E_2, \nu_2) = (2 \times 10^9, 0.35)$.

2 Preconditioning of matrix A

The sparse $n \times n$ SPD matrix A is preconditioned by a two-level Schwarz type DD method:

$$M_A^{-1} := R_0^T \left(R_0 A R_0^T\right)^{-1} R_0 + \sum_{i=1}^{N} R_i^T \left(R_i A R_i^T\right)^{-1} R_i \,, \tag{2}$$

where R_0 is full rank $dim(V_0) \times n$ where V_0 denotes the space spanned by the columns of R_0^T. The following assumptions are crucial to ensure the final method is scalable:

Assumption 2.1 (dimension and structure of the coarse space)

- *The coarse space dimension, $dim(V_0)$, is $O(N)$ typically 10-20 times N.*
- *The coarse space is made of extensions by zero of local vectors.*

Using the GenEO method [32], it is possible to fix in advance two constants $0 < \lambda_m < 1 < \lambda_M$ and then build a coarse space V_0 such that M_A^{-1} is spectrally equivalent to A^{-1}:

$$\frac{1}{\lambda_M} M_A^{-1} \leqslant A^{-1} \leqslant \frac{1}{\lambda_m} M_A^{-1} \,, \tag{3}$$

The dimension of the coarse space V_0 is typically proportional to the number of subdomains. This corresponds to Assumption 2.1. More precisely, for each subdomain $1 \leqslant i \leqslant N$, let D_i be a non negative diagonal matrix that defines a discrete partition of unity, i.e.:

$$\sum_{i=1}^{N} R_i^T D_i R_i = I \,,$$

and A_i^{Neu} be a symmetric semi-definite positive matrix such that for the maximum multiplicity of the intersection of subdomains denoted k_1, we have:

$$\sum_{i=1}^{N} R_i^T A_i^{Neu} R_i \leqslant k_1 A \,. \tag{4}$$

Then, the GenEO eigenvalue problem is local to each subdomain and reads:
Find $(\lambda_{ik}, V_{ik}) \in \mathbb{R} \times \mathbb{R}^{rank(R_i)}$ such that:

$$\left(D_i R_i A R_i^T D_i\right) V_{ik} = \lambda_{ik} A_i^{Neu} V_{ik} \,. \tag{5}$$

Let $\tau > 0$ be a positive threshold, the coarse space is the vector space spanned by the vectors $R_i^T D_i V_{ik}$ for all $\lambda_{ik} > \tau$. Then inequality (3) holds with $\lambda_m := (1 + k_1\tau)^{-1}$ and $\lambda_M := k_0$ where k_0 is the maximal number of neighbours of a subdomain including itself.

Our aim in the next section is to precondition the Schur complement $-S$ of matrix $\mathcal{A}$ (eq. (1)) where

$$\boxed{S := C + B\,A^{-1}\,B^T\,.} \tag{6}$$

This is achieved by a series of spectrally equivalent matrices or preconditioners, the first one being P_S defined as follows:

$$\boxed{P_S := C + B\,M_A^{-1}\,B^T} \tag{7}$$

and the final one being N_S^{-1} introduced in § 3.3, see eq. (26). Finally in § 4.2 we will introduce the preconditioner of the saddle point matrix $\mathcal{A}$.

3 Schur complement preconditioning

3.1 First spectrally equivalent preconditioner

Note that P_S is by definition a sum of $N + 2$ positive semi definite matrices

$$P_S := B\,R_0^T\,(R_0 A R_0^T)^{-1}\,R_0\,B^T\; + C + \sum_{i=1}^{N} B R_i^T\,(R_i A R_i^T)^{-1}\,R_i B^T\,. \tag{8}$$

Since B is a sparse matrix, it is interesting to introduce, for all $0 \leqslant i \leqslant N$, $\tilde{R}_i$ the restriction operator on the support of $\Im(B\,R_i^T)$ so that

$$\tilde{R}_i^T \tilde{R}_i B R_i^T = B R_i^T. \tag{9}$$

Then by defining for $0 \leqslant i \leqslant N$,

$$\tilde{B}_i := \tilde{R}_i\,B\,R_i^T\,,$$

the operator P_S is rewritten as

$$P_S := \tilde{R}_0^T \tilde{B}_0\,(R_0 A R_0^T)^{-1}\,\tilde{B}_0^T \tilde{R}_0\; + C + \sum_{i=1}^{N} \tilde{R}_i^T \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T \tilde{R}_i\,. \tag{10}$$

We consider a partition of unity on $H := \mathbb{R}^m$ defined with local diagonal matrices $(\tilde{D}_i)_{1\leqslant i\leqslant N} \in \mathbb{R}^{dim(Im(B\,R_i^T))\times dim(Im(B\,R_i^T))}$:

$$\sum_{i=1}^{N} \tilde{R}_i^T\,\tilde{D}_i\,\tilde{R}_i = I_H\,.$$

Remark 3.1 *This partition of unity exists since*

$$B = \sum_{i=1}^{N} B R_i^T D_i R_i = \sum_{i=1}^{N} \tilde{R}_i^T \tilde{R}_i (B\,R_i^T D_i R_i)$$

is full rank.

We make the following assumption

Assumption 3.1 *There exist symmetric positive semidefinite matrices $(\tilde{C}_i)_{1\leqslant i\leqslant N}$ such that for some constant $\tilde{k}_1$*

$$C \leqslant \sum_{i=1}^{N} \tilde{R}_i^T\,\tilde{C}_i\,\tilde{R}_i \leqslant \tilde{k}_1\,C\,. \tag{11}$$

This assumption is not so restrictive. Indeed, for a minimization problem with constraints enforced exactly without penalization nor relaxation, we have $C = 0$ and the assumption is automatically satisfied. Moreover, we have:

Lemma 3.1 *If C is a diagonal matrix, Assumption 3.1 is satisfied with $\tilde{k}_1 = 1$.*

Proof If C is a diagonal matrix, it suffices to take

$$\tilde{C}_i := \tilde{R}_i C \tilde{R}_i^T \tilde{D}_i \,,$$

which is a diagonal non negative matrix. Indeed, we then have for all $\mathbf{P} \in \mathbb{R}^m$:

$$C\,\mathbf{P} = \sum_{i=1}^{N} C\,\tilde{R}_i^T \tilde{D}_i \tilde{R}_i\,\mathbf{P} = \sum_{i=1}^{N} \tilde{R}_i^T (\tilde{R}_i C\,\tilde{R}_i^T \tilde{D}_i) \tilde{R}_i\,\mathbf{P}\,.$$

■

Remark 3.2 *Note also that in the finite element case it suffices to restrict to the subdomains the variational form that defines C. In this case $\tilde{k}_1$ is the multiplicity of the intersections of the subdomains used to define the $\tilde{C}_i$'s.*

Let us define the operator M_S as the sum of a non local but low rank matrix S_0:

$$S_0 := \tilde{R}_0^T \tilde{B}_0 \,(R_0 A R_0^T)^{-1}\, \tilde{B}_0^T \tilde{R}_0 \,,$$

and of S_1 which is a sum of N local positive semi definite matrices:

$$\boxed{S_1 := \sum_{i=1}^{N} \tilde{R}_i^T (\tilde{C}_i + \tilde{B}_i \,(R_i A R_i^T)^{-1}\, \tilde{B}_i^T) \tilde{R}_i \,,}$$

that is

$$\boxed{M_S := S_0 + S_1 \,.}$$

By Assumption 3.1, the operator M_S is spectrally equivalent to P_S which is also spectrally equivalent to S. Note that we may assume that S_1 is invertible whereas it does not make sense for S_0. Note that if it is not the case, since we build a preconditioner, S_1 can be regularized by a small diagonal term with little effect on the efficiency of the preconditioner.

We consider next the construction of a preconditioner $M_{S_1}^{-1}$ to S_1 leveraging the fact that S_1 is a sum of symmetric semidefinite positive matrices. Let us stress that this property stems from the domain decomposition structure of the preconditioner for matrix A which apart from the coarse level is block diagonal.

3.2 Preconditioning of S_1

It is well known that one level domain decomposition methods are in most cases not scalable. Nevertheless, the study of a one-level method in § 3.2.1 enables the identification of a suitable coarse space that will be efficiently embedded in a scalable two-level domain decomposition method in § 3.2.2.

Our studies of the spectrum of the DD preconditioners are based on the Fictitious Space lemma which is recalled here, see [27] for the original paper and [15] for a modern presentation.

Lemma 3.2 (Fictitious Space Lemma, Nepomnyaschikh 1991) *Let H and H_D be two Hilbert spaces, with the scalar products denoted by $(\cdot,\cdot)$ and $(\cdot,\cdot)_D$. Let the symmetric positive bilinear forms $a : H \times H \to \mathbb{R}$ and $b : H_D \times H_D \to \mathbb{R}$, generated by the s.p.d. operators $A : H \to H$ and $B : H_D \to H_D$, respectively (i.e. $(Au, v) = a(u, v)$ for all $u, v \in H$ and $(Bu_D, v_D)_D = b(u_D, v_D)$ for all $u_D, v_D \in H_D$). Suppose that there exists a linear operator $\mathcal{R} : H_D \to H$ that satisfies the following three assumptions:*

(i) $\mathcal{R}$ *is surjective.*

(ii) Continuity of $\mathcal{R}$*: there exists a positive constant* c_R *such that*

$$a(\mathcal{R}u_D, \mathcal{R}u_D) \leqslant c_R \cdot b(u_D, u_D) \quad \forall u_D \in H_D \,. \tag{12}$$

(iii) Stable decomposition: there exists a positive constant c_T *such that for all* $u \in H$ *there exists* $u_D \in H_D$ *with* $\mathcal{R}u_D = u$ *and*

$$c_T \cdot b(u_D, u_D) \leqslant a(\mathcal{R}u_D, \mathcal{R}u_D) = a(u, u) \,. \tag{13}$$

We introduce the adjoint operator $\mathcal{R}^* : H \to H_D$ *by* $(\mathcal{R}u_D,\, u) = (u_D,\, \mathcal{R}^*u)_D$ *for all* $u_D \in H_D$ *and* $u \in H$*.*
Then, we have the following spectral estimate

$$c_T \cdot a(u, u) \leqslant a\left(\mathcal{R}B^{-1}\mathcal{R}^*Au,\, u\right) \leqslant c_R \cdot a(u, u) \,, \quad \forall u \in H \tag{14}$$

which proves that the eigenvalues of operator $\mathcal{R}B^{-1}\mathcal{R}^*A$ *are bounded from below by* c_T *and from above by* c_R*.*

The Fictitious Space Lemma (FSL) can also conveniently be related to the book [34]: the first assumption corresponds to equation (2.3), page 36 where the global Hilbert space is assumed to satisfy a decomposition into subspaces, the second assumption is related to Assumptions 2.3 and 2.4, page 40 and the third assumption corresponds to the Stable decomposition Assumption 2.2 page 40.

3.2.1 One-level DD for S_1

As in [10] chapter 7, we begin with a one-level Neumann-Neumann type DD method defined in terms of the Fictitious Space Lemma (FSL). This study will be the basis for constructing the two-level preconditioner in § 3.2.2. Recall the formula for the one-level preconditioner $M_{S_1,1}^{-1}$ for S_1:

$$M_{S_1,1}^{-1} := \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \left(\tilde{C}_i + \tilde{B}_i \left(R_i A R_i^T\right)^{-1} \tilde{B}_i^T\right)^{\dagger} \tilde{D}_i \tilde{R}_i \,, \tag{15}$$

where the superscript † denotes a pseudo inverse in case the operator in brackets is not invertible. For sake of simplicity, we assume that they are invertible so that the following framework enables the study of $M_{S_1,1}$ with the fictitious space lemma. Let

$$H := \mathbb{R}^m$$

and let a be the following bilinear form:

$$a : H \times H \to \mathbb{R} \quad a(\mathbf{P}, \mathbf{Q}) := (S_1\mathbf{P}, \mathbf{Q}) \,.$$

Let

$$H_D := \Pi_{i=1}^{N} \mathbb{R}^{rank(\tilde{B}_i)} \,,$$

and b be the following bilinear form:

$$b : H_D \times H_D \to \mathbb{R} \quad b((\mathbf{P}_i)_{1\leqslant i\leqslant N}, (\mathbf{Q}_i)_{1\leqslant i\leqslant N}) := \sum_{i=1}^{N} \left(\left(\tilde{C}_i + \tilde{B}_i \left(R_i A R_i^T\right)^{-1} \tilde{B}_i^T\right) \mathbf{P}_i \,,\, \mathbf{Q}_i \right) . \tag{16}$$

We define $\mathcal{R}$:

$$\begin{array}{rccc} \mathcal{R} : & H_D & \to & H \\ & (\mathbf{P}_i)_{1\leqslant i\leqslant N} & \mapsto & \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i \,. \end{array}$$

We now check the three assumptions of the FSL.

Surjectivity of $\mathcal{R}$ For any $\mathbf{P} \in H$, we have:

$$\mathbf{P} = \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \tilde{R}_i \mathbf{P}\,,$$

so that

$$\mathbf{P} = \mathcal{R}((\tilde{R}_i \mathbf{P})_{1 \leqslant i \leqslant N})\,. \tag{17}$$

Continuity of $\mathcal{R}$ On one hand, we have using k_0 the number of neighbours of a subdomain (including itself), $k_0 := \max_{1 \leqslant i \leqslant N} \#\mathcal{O}(i)$ where $\mathcal{O}(i) := \{1 \leqslant j \leqslant N \mid \tilde{R}_i\, \tilde{D}_i\, S_1\, \tilde{D}_j\, \tilde{R}_j^T \neq 0\}$:

$$\begin{aligned} a(\mathcal{R}(\mathcal{P})\,,\, \mathcal{R}(\mathcal{P})) &= \|(\textstyle\sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i)\|_a^2 \leqslant k_0 \textstyle\sum_{i=1}^{N} \|\tilde{R}_i^T \tilde{D}_i \mathbf{P}_i\|_a^2 \\ &= k_0\, (\Big(\textstyle\sum_{j \in \mathcal{O}(i)} \tilde{R}_j^T (\tilde{C}_j + \tilde{B}_j\, (R_j A R_j^T)^{-1}\, \tilde{B}_j^T) \tilde{R}_j\Big)\, \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i\,,\, \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i)\,. \end{aligned}$$

On the other hand, we have by definition:

$$b(\mathcal{P}\,,\, \mathcal{P}) = \sum_{i=1}^{N} (\,(\tilde{C}_i + \tilde{B}_i\, (R_i A R_i^T)^{-1}\, \tilde{B}_i^T)\, \mathbf{P}_i\,,\, \mathbf{P}_i)\,.$$

We can take:

$$c_R := \max_{1 \leqslant i \leqslant N} \max_{P_i \in \mathbb{R}^{rank(\tilde{B}_i)}} \frac{(\sum_{j \in \mathcal{O}(i)} \tilde{R}_i\, \tilde{R}_j^T\, (\tilde{C}_j + \tilde{B}_j\, (R_j A R_j^T)^{-1}\, \tilde{B}_j^T)\, \tilde{R}_j\, \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i\,,\, \tilde{D}_i \mathbf{P}_i)}{(\,(\tilde{C}_i + \tilde{B}_i\, (R_i A R_i^T)^{-1}\, \tilde{B}_i^T)\, \mathbf{P}_i\,,\, \mathbf{P}_i)}\,, \tag{18}$$

but we have no control on c_R which may be large. This motivates the introduction of a spectral coarse space in § 3.2.2 with the generalized eigenvalue problem (19) .

Stable decomposition Let $\mathbf{P} \in H$, we start from its decomposition (17) and estimate its b-norm

$$b(\mathcal{P}\,,\, \mathcal{P}) \;=\; \textstyle\sum_{i=1}^{N} (\,(\tilde{C}_i + \tilde{B}_i\, (R_i A R_i^T)^{-1}\, \tilde{B}_i^T)\, \tilde{R}_i \mathbf{P}\,,\, \tilde{R}_i \mathbf{P}) = a(\mathbf{P}\,,\, \mathbf{P})\,,$$

so that we can take $c_T = 1$ which is an optimal value.

3.2.2 Two-level DD for S_1

In order to control the value of c_R defined above, we introduce two two-level preconditioners. The first one is similar to what is done for Schur complement methods in [10, § 7.8.3, page 197] or in [33]. The second one is a cheaper lightweight version of the former but then not with a full control of its spectrum. In practice, in our numerical experiments both methods perform similarly in terms of iterations counts and thus with an advantage in terms of elapsed time for the second preconditioner. For both two-level methods, the generalized eigenvalue value problem in each subdomain $1 \leqslant i \leqslant N$ to be solved to build the coarse space is inferred from the definition of the constant c_R in eq. (18):

$$\begin{aligned} \tilde{D}_i\, \tilde{R}_i\, \Big(\textstyle\sum_{j \in \mathcal{O}(i)} \tilde{R}_j^T (\tilde{C}_j + \tilde{B}_j\, (R_j A R_j^T)^{-1}\, \tilde{B}_j^T) \tilde{R}_j\Big)\; \tilde{R}_i^T \tilde{D}_i\; \mathbf{P}_{i\,k} \\ = \lambda_{i\,k} (\tilde{C}_i + \tilde{B}_i\, (R_i A R_i^T)^{-1}\, \tilde{B}_i^T)\; \mathbf{P}_{i\,k}\,. \end{aligned} \tag{19}$$

This generalized eigenvalue problem contains inverses of some local matrices on both sides and in order to solve it via e.g. Arpack, we have to factorize one of them. This difficulty can be overcome since the right matrix is the inverse of the Schur complement of the extended sparse matrix (25). It is thus amenable to a factorization using only sparse matrix factorizations.

It can be solved in $O(1)$ communications. The coarse space is defined as follows. Let τ_{S_1} be a user-defined threshold; for each subdomain $1 \leqslant i \leqslant N$, we introduce a subspace $\tilde{W}_i \subset \mathbb{R}^{rank(\tilde{B}_i)}$:

$$\tilde{W}_i := \text{Span}\{\mathbf{P}_{i\,k} \mid \lambda_{i\,k} > \tau_{S_1}\}\,. \tag{20}$$

Then the coarse space $\tilde{W}_0$ is defined (with some abuse of notation)

$$\boxed{\tilde{W}_0 := \bigoplus_{1\leqslant i\leqslant N} \tilde{R}_i^T \tilde{D}_i \tilde{W}_i\,.}$$

Let Z_{S_1} be a rectangular matrix whose columns span the coarse space $\tilde{W}_0$. Let $\tilde{P}_0$ be the S_1 orthogonal projection from $\mathbb{R}^m$ on $\tilde{W}_0$ whose formula is

$$\tilde{P}_0 = Z_{S_1}(Z_{S_1}^T S_1 Z_{S_1})^{-1} Z_{S_1}^T S_1\,. \tag{21}$$

In order to avoid a too cumbersome analysis, we make the following assumption:

Assumption 3.2 *We assume that for all subdomains $1 \leqslant i \leqslant N$, $\tilde{C}_i + \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T$ is invertible.*

Finally, the first preconditioner for S_1 reads

$$\begin{array}{rcl} M_{S_1}^{-1} & := & Z_{S_1}\,(Z_{S_1}^T S_1 Z_{S_1})^{-1}\,Z_{S_1}^T + (I - \tilde{P}_0) \\ & \times & \left(\sum_{i=1}^N \tilde{R}_i^T \tilde{D}_i\,(\tilde{C}_i + \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T)^\dagger\,\tilde{D}_i \tilde{R}_i\right)(I - \tilde{P}_0^T)\,. \end{array} \tag{22}$$

If Assumption 22 is not satisfied for some subdomain i, we should incorporate the kernel of $\tilde{C}_i + \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T$ in the coarse space and make use of a pseudoinverse in the definition of the preconditioner as it is done for the FETI method, see [13] or [10] § 7.8.2 and references therein.

Recall that from [10] chapter 7, we have for $\alpha := \max(1, \frac{k_0}{\tau_{S_1}})$:

$$\frac{1}{\alpha} M_{S_1}^{-1} \leqslant S^{-1} \leqslant M_{S_1}^{-1}\,.$$

A careful implementation of (22) requires two coarse solves. In order to simplify the application of $M_{S_1}^{-1}$, we can save one coarse solve by proposing an alternative definition of preconditioner $M_{S_1}^{-1}$ without the global projection $\tilde{P}_0$. We directly define it using the FSL framework. We keep definitions for H and a from the beginning of § 3.2.1. But now the space H_D is defined as:

$$H_D := \mathbb{R}^{rank(Z_{S_1})} \times \Pi_{i=1}^N \mathbb{R}^{rank(\tilde{B}_i)}\,,$$

the operator $\mathcal{R}$ is defined using for $1 \leqslant i \leqslant N$ the $(\tilde{C}_i + \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T)$ orthogonal projections $\xi_\mathbf{i}$ on $\tilde{W}_i$ and parallel to $\text{Span}\{\mathbf{P}_{i\,k} \mid \lambda_{i\,k} \leqslant \tau_{S_1}\}$:

$$\begin{array}{rccl} \mathcal{R}: & H_D & \to & H \\ & (\mathbf{P}_i)_{0\leqslant i\leqslant N} & \mapsto & Z_{S_1}\mathbf{P}_0 + \sum_{i=1}^N \tilde{R}_i^T \tilde{D}_i (I - \xi_i)\mathbf{P}_i\,, \end{array}$$

and b is the following bilinear form:

$$b : H_D \times H_D \to \mathbb{R}\,,\;\; b((\mathbf{P}_i)_{0\leqslant i\leqslant N}, (\mathbf{Q}_i)_{0\leqslant i\leqslant N}) := (S_1 Z_{S_1}\mathbf{P}_0, Z_{S_1}\mathbf{Q}_0) + \sum_{i=1}^N (\,(\tilde{C}_i + \tilde{B}_i\,(R_i A R_i^T)^{-1}\,\tilde{B}_i^T)\,\mathbf{P}_i\,,\,\mathbf{Q}_i)\,.$$

We now check the three assumptions of the FSL.

Surjectivity of $\mathcal{R}$ For any $\mathbf{P} \in H$, we have:

$$\mathbf{P} = \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P} + \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i (I - \xi_i) \tilde{R}_i \mathbf{P} \,.$$

Note that since the first sum belongs to $\tilde{W}_0$, there exists $\mathbf{P}_0$ such that $\sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P} = Z_{S_1} \mathbf{P}_0$. Then, we have:

$$\mathbf{P} = \mathcal{R}(\mathbf{P}_0, (\tilde{R}_i \mathbf{P})_{1 \leqslant i \leqslant N}) \,. \tag{23}$$

Continuity of $\mathcal{R}$ For $\mathcal{P} \in H_D$ we have:

$$\begin{aligned}
a(\mathcal{R}(\mathcal{P}) \,,\, \mathcal{R}(\mathcal{P})) &= (S_1 \mathcal{R}(\mathcal{P}) \,,\, \mathcal{R}(\mathcal{P})) \\
&\leqslant 2\,((S_1 Z_{S_1} \mathbf{P}_0 \,,\, Z_{S_1} \mathbf{P}_0)) + (S_1 \textstyle\sum_{i=1}^{N} (I - \xi_i) \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i \,,\, \sum_{i=1}^{N} (I - \xi_i) \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i)) \\
&\leqslant 2\,((S_1 Z_{S_1} \mathbf{P}_0 \,,\, Z_{S_1} \mathbf{P}_0)) + k_0 \textstyle\sum_{i=1}^{N} (S_1 (I - \xi_i) \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i \,,\, (I - \xi_i) \tilde{R}_i^T \tilde{D}_i \mathbf{P}_i)) \\
&\leqslant 2\,((S_1 Z_{S_1} \mathbf{P}_0 \,,\, Z_{S_1} \mathbf{P}_0)) + k_0 \tau_{S_1} \textstyle\sum_{i=1}^{N} ((\tilde{C}_i + \tilde{B}_i \,(R_i A R_i^T)^{-1} \,\tilde{B}_i^T) \mathbf{P}_i \,,\, \mathbf{P}_i)) \\
&\leqslant 2 \max(1, k_0 \tau_{S_1}) \, b(\mathcal{P} \,,\, \mathcal{P}) \,,
\end{aligned}$$

so that we can take $c_R = 2 \max(1, k_0 \tau_{S_1})$ and we lose only a factor of 2 compared to eq. (22).

Stable decomposition Let $\mathbf{P} \in H$, we start from its decomposition (23) and estimate its b-norm

$$\begin{aligned}
b(\mathcal{P} \,,\, \mathcal{P}) &= (S_1 \textstyle\sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P}, \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P}) \\
&\quad + \textstyle\sum_{i=1}^{N} (\,(\tilde{C}_i + \tilde{B}_i \,(R_i A R_i^T)^{-1} \,\tilde{B}_i^T)\, \tilde{R}_i \mathbf{P} \,,\, \tilde{R}_i \mathbf{P}) \\
&= (S_1 \textstyle\sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P}, \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P}) + a(\mathbf{P} \,,\, \mathbf{P}) \\
&\leqslant (\gamma + 1)\, a(\mathbf{P} \,,\, \mathbf{P}) \,,
\end{aligned}$$

where

$$\gamma := \max_{\mathbf{P}} \frac{(S_1 \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P}, \sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \xi_i \tilde{R}_i \mathbf{P})}{(S_1 \mathbf{P} \,,\, \mathbf{P})} \,.$$

We can take $c_T := 1/(1 + \gamma)$ but we have no estimate on γ.

Finally, the explicit form of this alternative preconditioner reads:

$$\begin{aligned}
M_{S_1}^{-1} &:= Z_{S_1} \,(Z_{S_1}^T S_1 Z_{S_1})^{-1} \, Z_{S_1}^T \\
&\quad + \left(\textstyle\sum_{i=1}^{N} \tilde{R}_i^T \tilde{D}_i \,(I - \xi_i)(\tilde{C}_i + \tilde{B}_i \,(R_i A R_i^T)^{-1} \,\tilde{B}_i^T)^{\dagger} \,(I - \xi_i^T) \tilde{D}_i \tilde{R}_i \right) \,.
\end{aligned} \tag{24}$$

Note that the application of $M_{S_1}^{-1}$ can be done using only sparse solvers since solving a linear system with a local Schur complement

$$(\tilde{C}_i + \tilde{B}_i \,(R_i A R_i^T)^{-1} \,\tilde{B}_i^T) \mathbf{P}_i = \mathbf{G}_i \,,$$

amounts to solving an augmented sparse system which has the form of a local saddle point system:

$$-\begin{pmatrix} R_i A R_i^T & \tilde{B}_i^T \\ \tilde{B}_i & -\tilde{C}_i \end{pmatrix} \begin{pmatrix} \mathbf{U}_i \\ \mathbf{P}_i \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{G}_i \end{pmatrix} \,. \tag{25}$$

3.3 Final Preconditioner for the Schur complement

From the spectrally equivalent preconditioner M_{S_1} to S_1, we define N_S a spectrally equivalent preconditioner to M_S and thus to S as well:

$$\boxed{N_S := S_0 + M_{S_1} \,.} \tag{26}$$

We now consider the application of the preconditioner N_S, that is for some right hand side $\mathbf{G} \in \mathbb{R}^m$, the solving in $\mathbf{P}$ of the following system:

$$N_S \, \mathbf{P} = \mathbf{G} \,, \tag{27}$$

by a Krylov solver with $M_{S_1}^{-1}$ as a preconditioner.

4 Recap

4.1 Setup for the Schur complement preconditioner

We have a setup phase which is composed of:

1. Build the two-level preconditioner M_A^{-1} for A, see eq. (2),
2. Build the two-level preconditioner $M_{S_1}^{-1}$ for S_1, see eq. (22).

Once the setup is complete, applying preconditioner N_S^{-1} can be performed following Algorithm 1

Algorithm 1 N_S^{-1} matvec product

INPUT: $\mathbf{G} \in \mathbb{R}^m$ OUTPUT: $\mathbf{P} = N_S^{-1}\,\mathbf{G}$

1. Solve eq. (27) in $\mathbf{P}$ by a Krylov method with M_S^{-1} as preconditioner.

4.2 DD solver for the saddle point system

Algorithm 2 DD saddle point solver

INPUT: $\begin{pmatrix} \mathbf{F}_U \\ \mathbf{F}_P \end{pmatrix} \in \mathbb{R}^{n+m}$ OUTPUT: $\begin{pmatrix} \mathbf{U} \\ \mathbf{P} \end{pmatrix}$ the solution to (28).

1. Solve $A\mathbf{G}_U = \mathbf{F}_U$ by a PCG with M_A^{-1} as a preconditioner
2. Compute $\mathbf{G}_P := \mathbf{F}_P - B\,\mathbf{G}_U$
3. Solve $(C + BA^{-1}B^T)\mathbf{P} = -\mathbf{G}_P$ by a PCG with N_S^{-1} as a preconditioner, see Algorithm 1
4. Compute $\mathbf{G}_U := \mathbf{F}_U - B^T\mathbf{P}$
5. Solve $A\mathbf{U} = \mathbf{G}_U$ by a PCG with M_A^{-1} as a preconditioner

We now consider the solving of the saddle point problem:

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} \mathbf{U} \\ \mathbf{P} \end{pmatrix} = \begin{pmatrix} \mathbf{F}_U \\ \mathbf{F}_P \end{pmatrix} \tag{28}$$

by Algorithm 2 whose Step 3 demands the matrix-vector product with matrix $C+BA^{-1}B^T$ which is done by an iterative solve for matrix A^{-1}. In order to avoid such a nested loop, block solvers have been developed, see [26, 4, 9, 38]. Following [38], solving with A^{-1} is not needed when preconditioning the original system (1) by P^{-1} where P is an inexact block factorization:

$$P := \begin{pmatrix} M_A & \\ B & -N_S \end{pmatrix} \begin{pmatrix} I & M_A^{-1}\,B^T \\ & I \end{pmatrix} \tag{29}$$

or a slightly different but symmetric variant of it introduced in eq. (2.5) of [38]:

$$P := \begin{pmatrix} I & \\ BM_A^{-1} & I \end{pmatrix} \begin{pmatrix} M_A\,(2\,M_A^{-1} - A)^{-1}\,M_A & \\ & -N_S \end{pmatrix} \begin{pmatrix} I & M_A^{-1}\,B^T \\ & I \end{pmatrix}. \tag{30}$$

Note that in both cases, the inverses of P are computable easily in our framework.

5 Numerical experiments

In this section, we perform 3D experiments to illustrate the theory and the performance of the method. We are interested in a heterogeneous elasticity problem with nearly incompressible material typically rubber-steel structures. First, we recall the definition of the coefficients and the corresponding variational formulation.

The mechanical properties of a solid are characterized by its Young modulus E and Poisson ratio ν, or alternatively by its Lamé coefficients λ and μ. They verify the following relations:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)}\,. \tag{31}$$

For the discretization, we choose a continuous pressure space and take the lowest order Taylor-Hood finite element $C0P2 - C0P1$ whose stability is proved, see e.g., [6]. The domain Ω is a beam and the variational problem consists in finding $(\boldsymbol{u}_h, p_h) \in \mathcal{V}_h := \mathbb{P}_2^3 \cap (H_0^1(\Omega))^3 \times \mathbb{P}_1$ with Dirichlet boundary conditions on the four lateral faces and Neumann boundary conditions on the other two faces such that for all $(\boldsymbol{v}_h, q_h) \in \mathcal{V}_h$

$$\begin{cases} \int_\Omega 2\mu \underline{\underline{\varepsilon}}(\boldsymbol{u}_h) : \underline{\underline{\varepsilon}}(\boldsymbol{v}_h)dx & -\int_\Omega p_h \mathrm{div}\,(\boldsymbol{v}_h)dx = \int_\Omega \boldsymbol{f}\boldsymbol{v}_h dx \\ \\ -\int_\Omega \mathrm{div}\,(\boldsymbol{u}_h) q_h dx & -\int_\Omega \frac{1}{\lambda} p_h q_h = 0. \end{cases} \tag{32}$$

Letting $\boldsymbol{u}$ denote the degrees of freedom of $\boldsymbol{u}_h$ and p that of p_h, the problem can be written in matrix form as:

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} \boldsymbol{u} \\ p \end{pmatrix} = \begin{pmatrix} \mathrm{f} \\ 0 \end{pmatrix}. \tag{33}$$

The matrix C is a mass matrix arising from the discretization of a variational form. This enables us to satisfy Assumption 3.1 with $\tilde{C}_i$ the corresponding mass matrix but only defined on subdomain $\tilde{\Omega}_i$ which is the extension by a layer of direct neighbors of subdomain Ω_i.

In the following numerical experiments, we consider a heterogeneous beam composed of 10 alternating layers of rubber material $(E_1, \nu_1) = (1 \times 10^7, 0.4999)$ and steel material $(E_2, \nu_2) = (2 \times 10^9, 0.35)$, see Fig. 1.

5.1 Software, hardware, implementation details

In the following numerical experiments, the iteration counts assess the control of the condition number via the adaptive coarse spaces. We also report timings along with iteration counts. We also showcase the fact that the size of the GenEO coarse space adapts automatically to the difficulty of the problem at hand, for example when going from a homogeneous to a heterogeneous problem. We illustrate the efficiency of the method by performing weak and strong scalability tests, using the automatic graph partitioner *Metis* [22] for the subdomain partitioning.

The problem is discretized and solved with the open-source parallel finite element software *FreeFEM* [17]. FreeFEM is domain specific language (DSL) where the problem to be solved is defined in terms of its variational formulation. Then the local matrices $(A_i^{Neu})_{1\leqslant i\leqslant N}$ (see eq. (4)) and $(\tilde{C}_i)_{1\leqslant i\leqslant N}$ (see eq. (11)) are easily obtained by restricting the corresponding variational formulations to adequate local subdomains. Note that these matrices are different from the restriction of the global matrices A and C to the local degrees of freedom. The domain decomposition algorithm presented in this paper is implemented on top of the *ffddm* framework, a set of parallel *FreeFEM* scripts implementing Schwarz domain decomposition methods. *ffddm* already implements the GenEO method [32] for SPD problems, and its building blocks are designed to simplify the implementation and prototyping of new domain decomposition methods such as the saddle point solver presented in this paper. The *ffddm* documentation is available on the FreeFem.org web page, see [35].

Numerical results are obtained on the french GENCI supercomputer *Occigen*, on the Haswell partition composed of 50544 cores of Intel Xeon E5-2690V3 processors clocked at 2.6 GHz. The interconnect is an InfiniBand FDR 14 pruned fat tree. We use Intel compilers, the Intel Math Kernel Library and Intel MPI.

As is usually done in domain decomposition methods, we assign one subdomain per MPI process. Our implementation is pure MPI and no multithreading is done ; we assign one MPI process per computing core. The mesh of the computational domain is partitioned using the automatic graph partitioner *Metis* [22] (see Figure 1). Local subdomain matrices are factorized by the sparse direct solver *MUMPS* [1]. Local eigenvalue problems are solved with *Arpack* [24] ; both libraries are interfaced with *FreeFEM*. GenEO coarse space matrices $R_0 A R_0^T$ in (2) and $Z_{S_1}^T S_1 Z_{S_1}$ in (21) are assembled and factorized in a distributed manner on a few cores (24 in most of the experiments) using the parallel solver *MUMPS*.

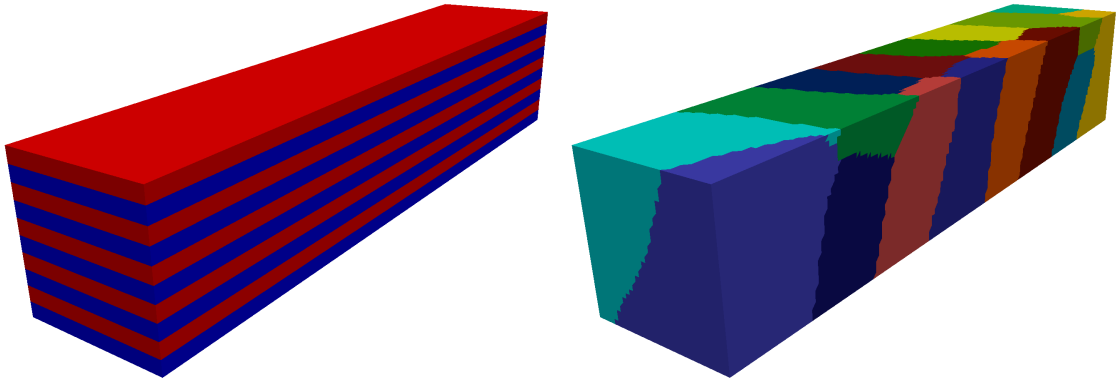

Figure 1: Heterogeneous beam composed of 10 alternating layers of rubber and steel. Coefficient distribution (left) and mesh partitioning into 16 subdomains by the automatic graph partitioner *Metis* (right).

For illustration purposes, we represent in Figure 2 (top) the eigenvalues of the local GenEO eigenvalue problems for both coarse spaces, V_0 for A and $\tilde{W}_0$ for S_1 (the former corresponding to eq. (5) and the latter to eq. (19)), for the heterogeneous beam problem with 16 million degrees of freedom, corresponding to the first row of Table 1. The figures show the inverse of the first 40 largest eigenvalues for 10 of the subdomains for the experiment corresponding to the first row of Table 1, so that the eigenvectors corresponding to the smallest values on the graphs (below the dashed line) will be selected to enter the coarse space. For comparison, we also solve the constant coefficient problem corresponding to an homogeneous steel (and compressible) beam and show the eigenvalues in Figure 2 (bottom).

We can see the effect of heterogeneities on the spread of eigenvalues for different subdomains compared to the homogeneous case. In addition, we retrieve the 6 eigenvalues corresponding to the rigid body modes for A in the homogeneous case, and we see that we need a larger set of eigenvectors in order to build a robust coarse space in the heterogeneous case. Figure 2 also shows that there is no need for a coarse space for S_1 in the compressible homogeneous case. A strong feature of the GenEO method is that relevant eigenvectors to enter the coarse space are selected automatically, adapting to the problem at hand and its spatial heterogeneity. Moreover, the robustness of the model does not rely on a specific partitioning, which allows the use of automatic graph partitioners such as *Metis* [22] or *Scotch* [8].

5.2 Parameters of the method

The method has a few parameters in play:

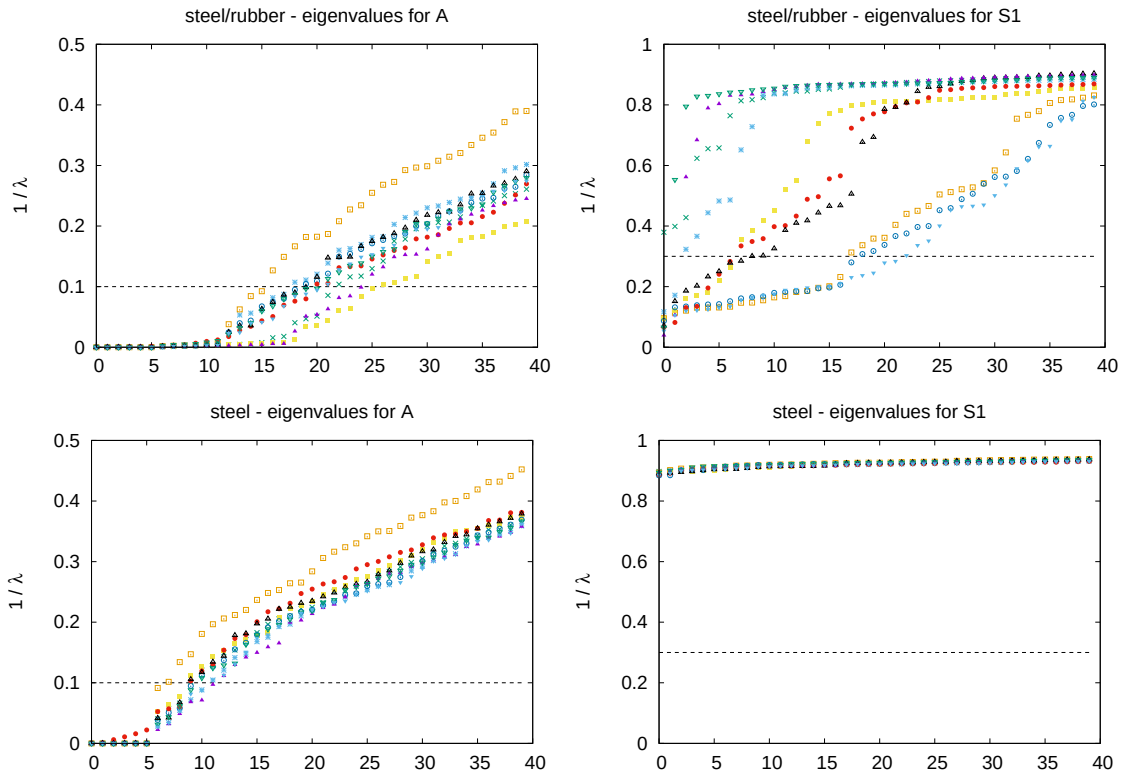

Figure 2: Top: Heterogeneous steel and rubber beam. Bottom: Homogeneous steel beam. Inverse of the eigenvalues of the local GenEO eigenvalue problems for both coarse spaces, V_0 for A (left) and $\tilde{W}_0$ for S_1 (right), for 10 of the subdomains.

- The number of layers of mesh elements in the overlap region between subdomains is 2 for the velocity blocks (corresponding to R_i in (2)) and 4 for the pressure blocks (corresponding to $\tilde{R}_i$ in (10)). This corresponds to the minimum overlap that satisfies relation (9) for a symmetric construction of the overlap between subdomains.

- For the heterogeneous beam problem, we set the threshold τ_A for selecting the local eigenvectors entering the coarse space V_0 to 10 (corresponding to 0.1 on Figure 2, left). The threshold τ_{S_1} for selecting the local eigenvectors entering the coarse space $\tilde{W}_0$ is set to 3.33 (corresponding to 0.3 on Figure 2, right).

  The selection of these thresholds is based on the fact that the iteration counts are very stable in weak and strong scaling experiments. It means that for a given physics the parameters τ_A and τ_{S_1} can be tuned for a small test case and then used in large scale experiments.

5.3 Weak scalability test for heterogeneous steel and rubber beam

Here we present weak scalability results for the heterogeneous beam composed of 10 alternating layers of rubber $(E_1, \nu_1) = (1 \times 10^7, 0.4999)$ and steel $(E_2, \nu_2) = (2 \times 10^9, 0.35)$. Local problem size is kept roughly constant as N grows, and the total number of dofs n goes from 16 million on 262 cores to 1 billion on 16800 cores. We use the original definition of $M_{S_1}^{-1}$ (22).

We report in Table 1 the iteration counts and computing times for the DD saddle point solver Algorithm 2. Note that in Algorithm 2 we replace PCG by right-preconditioned GMRES for step 3. The stopping criterion is a tolerance smaller than 10^{-5}. Moreover, we use flexible GMRES, as we solve (27) inexactly using GMRES with a tolerance of 10^{-2} in order to apply N_S^{-1}.

In order, columns correspond to: number of cores, number of dofs n, size of the coarse space for A $dim(V_0)$, size of the coarse space for S_1 $dim(\tilde{W}_0)$, setup time corresponding to the assembly and factorization of the various local and coarse operators, number of outer GMRES iterations, GMRES computing time, total computing time (setup + GMRES) and average number of inner GMRES iterations for each solution of (27). All timings are reported in seconds.

#cores	n	$dim(V_0)$	$dim(\tilde{W}_0)$	setup(s)	#It	gmres(s)	total(s)	#It N_S^{-1}
262	15 987 380	5 383	3 319	710.7	24	631.6	1342.3	11
525	27 545 495	9 959	2 669	526.6	21	519.5	1046.1	12
1 050	64 982 431	17 837	4 587	675.2	22	665.9	1341.1	11
2 100	126 569 042	32 361	7 995	689.2	25	733.8	1423.0	10
4 200	218 337 384	59 704	13 912	593.0	27	705.4	1298.4	10
8 400	515 921 881	141 421	25 949	735.8	32	1152.5	1888.3	10
16 800	1 006 250 208	260 348	41 341	819.2	29	1717.9	2537.1	12

Table 1: Weak scaling experiment for 3D heterogeneous elasticity: beam with 10 alternating layers of steel and rubber. Reported iteration counts and timings for DD saddle point algorithm 2.

Iteration counts. We first discuss iteration counts. We see that outer iteration count remains stable, between 21 and 32. The inner iteration count is also stable and remains around 11. We also observed (figures are not reported here) than the inner GMRES tolerance of 10^{-2} does not affect the outer iteration count compared to an accurate solution with a stricter tolerance of 10^{-5}, and allows a significant reduction in inner iteration count. For example, 11 iterations on average instead of 28 on 1050 cores for the same outer iteration count of 22, leading to a decrease from 1178.2 to 665.9 seconds in GMRES timing.

Timings. In terms of setup timings, the computing time remains relatively stable, with roughly 15% increase for a factor of 64 in problem size. Around 60% of the setup time is spent in the solution of the eigenvalue problems (19) for S_1.

The solution time stays relatively stable up to 4200 cores, where it starts to degrade. This can be related to the increased cost of the coarse space solves with matrices $R_0 A R_0^T$ and $Z_{S_1}^T S_1 Z_{S_1}$ as their size increases: total time spent in coarse space solves is 14.7, 62.1 and 679.3 seconds on 262, 4200 and 16800 cores respectively. A possible improvement would be to use a multi-level method to solve the coarse problems iteratively.

5.4 Strong scalability test for heterogeneous steel and rubber beam

Strong scalability results for the heterogeneous beam composed of 10 alternating layers of rubber and steel are presented in Table 2. The problem size is 27.5 million and the strong scaling test ranges from 525 to 4200 cores. Iteration counts and computing times for the DD saddle point solver Algorithm 2 are reported.

#cores	n	$dim(V_0)$	$dim(\tilde{W}_0)$	setup(s)	#It	gmres(s)	total(s)	#It N_S^{-1}
525	27 545 495	9 959	2 669	526.6	21	519.5	1046.1	12
1 050	27 545 495	15 078	4 082	265.7	21	224.7	490.4	11
2 100	27 545 495	23 172	6 453	168.8	23	131.1	299.9	10
4 200	27 545 495	37 768	11 152	103.8	23	91.3	195.1	9

Table 2: Strong scaling experiment for 3D heterogeneous elasticity: beam with 10 alternating layers of steel and rubber. Reported iteration counts and timings for DD saddle point Algorithm 2.

Iteration counts. Outer iteration count remains stable, with a slight increase from 21 to 23. Inner iteration counts remains also stable, even slightly decreasing from 12 to 9.

Timings. We see that the setup timing decreases accordingly as the subdomains shrink in size, from 526.6 seconds on 525 cores to 103.8 seconds on 4200 cores; the speedup efficiency with respect to 525 cores ranges from 99% on 1050 cores to 63% on 4200 cores.

We see a similar trend for the solution time, ranging from 519.5 seconds on 525 cores to 91.3 seconds on 4200 cores ; the speedup efficiency with respect to 525 cores ranges from 116% on 1050 cores to 71% on 4200 cores. This decrease in efficiency can be explained by the increased relative cost of coarse space solves as subdomains get smaller: from 3% on 525 cores to 30% on 4200 cores. The added overlap also plays a greater role in the loss of efficiency as subdomains get smaller.

		MUMPS			DD saddle point solver			
n	#cores	setup(s)	solve(s)	total(s)	setup(s)	#It	gmres(s)	total(s)
139 809	16	7.1	0.1	7.2	27.1	18	19.7	46.8
1 058 312	32	85.7	0.8	86.5	166.2	20	137.2	303.4
1 058 312	65	71.0	0.6	71.6	91.0	21	77.1	168.1
1 058 312	131	63.2	0.5	63.7	59.7	24	49.7	109.4
3 505 582	55	477.8	3.7	481.5	404.1	24	430.1	834.2
3 505 582	110	392.3	2.3	394.6	242.5	23	212.8	455.3
3 505 582	221	387.0	2.1	389.1	134.8	23	109.4	244.2
3 505 582	442	453.9	2.2	456.1	88.2	24	68.6	156.8
8 235 197	262	OOM	/	/	278.5	25	264.3	542.8
8 235 197	525	1622.1	6.1	1628.2	172.1	24	136.0	308.1
8 235 197	1050	1994.3	7.4	2001.7	136.5	25	99.7	236.2

Table 3: Comparison with the parallel sparse direct solver *MUMPS* for 3D heterogeneous elasticity: beam with 10 alternating layers of steel and rubber. Reported timings for four discretization levels while also varying the number of cores (OOM means the computation ran out of available memory).

In Table 3, we compare the performance of the solver to the parallel sparse direct solver *MUMPS* for the heterogeneous steel and rubber beam test case with four discretization levels, while also varying the number of cores. As we can see, *MUMPS* is comparatively more efficient for smaller problems, with for example a total time of 86.5 seconds compared to 303.4 seconds for our saddle point solver for 1 million unknowns on 32 cores. However, as expected, we see a large increase in memory and computational cost as the size of the system gets larger: for 8.2 million unknowns, *MUMPS* runs out of memory on 262 cores and solves the problem in 1628.2 seconds on 525 cores, compared to 308.1 seconds for the DD solver. Moreover, we can see from Table 3 that the DD saddle point solver offers much better strong scalability.

In Figure 3, we plot the convergence history of GMRES for both compressible homogeneous and heterogeneous steel-rubber cases with 27.5 million unknowns on 525 cores (left), with a comparison to the standard one-level Additive Schwarz Method (ASM) from PETSc [3] on the global problem (right), illustrating the difficulty of the test case at hand. Our saddle point solver needs 15 and 21 iterations to converge for the homogeneous and heterogeneous cases respectively, compared to 856 and 2880 for the one-level method, with significant undesired plateaux. For the same physical test case, iteration counts are better that in [16] but timings are not as good. As mentioned in the introduction, our method has the advantage of a provable convergence estimate and to not depend on the design of

specific absorbing conditions for the elasticity system.

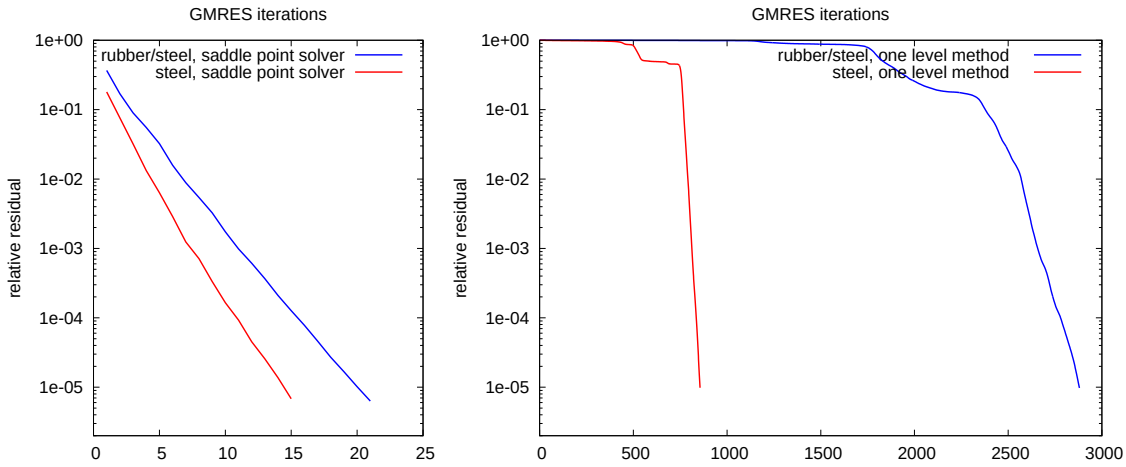

Figure 3: GMRES convergence history of the saddle point solver (left) compared to the one-level Additive Schwarz Method (right) for the homogeneous steel beam and heterogeneous rubber/steel beam problems discretized with 27.5 million unknowns (corresponding to the first row of Table 2), on 525 cores.

We also performed comparisons with the Geometric Algebraic Multigrid (GAMG) preconditioner from PETSc. We were not able to find a suitable tuning of parameters for GAMG for the saddle point formulation. However, we performed comparisons between GAMG and standard GenEO for the velocity formulation on the homogeneous beam, varying the Poisson ratio ν from 0.48 to 0.499. The GenEO threshold τ is set to 3.33, and we select at most 80 eigenvectors in each subdomain. Even though GAMG is faster for $\nu \leqslant 0.49$, we can see that GenEO is more robust as ν increases. In particular, GAMG fails to converge in 2000 iterations for $\nu \geqslant 0.495$.

131 cores	GAMG		DD saddle point solver				
ν	#It	total(s)	$dim(V_0)$	setup(s)	#It	gmres(s)	total(s)
0.48	60	67.1	10 480	200.7	24	11.2	212.0
0.485	109	89.0	10 480	199.5	27	12.7	212.2
0.49	210	137.0	10 480	202.0	32	15.0	217.0
0.495	>2000	/	10 480	199.9	43	20.2	220.1
0.499	>2000	/	10 480	199.2	99	48.6	247.7
525 cores	GAMG		DD saddle point solver				
ν	#It	total(s)	$dim(V_0)$	setup(s)	#It	gmres(s)	total(s)
0.48	56	25.5	41 766	60.4	18	5.0	65.4
0.485	60	26.1	41 984	60.9	20	5.3	66.2
0.49	116	33.3	42 000	60.4	23	5.9	66.3
0.495	>2000	/	42 000	60.4	32	7.6	68.1
0.499	>2000	/	42 000	60.6	95	20.3	81.0

Table 4: GAMG versus standard GenEO for the velocity formulation on the homogeneous beam discretized with 7.9 million unknowns, using 131 and 525 cores. Reported iteration counts and timings for different values of the Poisson ratio ν ranging from 0.48 to 0.499.

6 Conclusion and outlook

Under the assumption that the diagonal block matrices of a saddle point problem are spectrally equivalent to a sum of positive semi definite matrices, we have introduced an adaptive domain decomposition (DD) method. For this, two coarse spaces are built by solving generalized eigenvalue problems, one for the primal unknowns and the second one for the dual unknowns. The robustness of the method was assessed on a notoriously difficult three dimensional elasticity problem for a steel-rubber structure discretized with continuous pressure.

Several issues deserve further investigations. First a multilevel method with more than two levels would enable even larger and possibly faster simulations. Also, the tests were performed with FreeFem scripts using the standalone *ffddm* [35] framework. The integration of the method in the C++/MPI library *hpddm* [21] could lead to faster codes and a more general diffusion of the saddle point preconditioner. In a different setting, the design of adaptive coarse space is strongly connected to multiscale finite element (MFE) methods (see e.g., [12, 20, 25] and references therein) and this work could be used in designing MFE methods for saddle point problem.

Acknowledgment

This work was granted access to the HPC resources of OCCIGEN@CINES under the allocation 2020-067730 granted by GENCI.

Code reproducibility

Our numerical results can be reproduced running the script `https://github.com/FreeFem/FreeFem-sources/blob/master/examples/ffddm/elasticity_saddlepoint.edp` available in the FreeFem distribution starting from version 4.10.

References

[1] Patrick R. Amestoy, Iain S. Duff, Jean-Yves L'Excellent, and Jacko Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM J. Matrix Analysis and Applications*, 23(1):15–41, 2001.

[2] Douglas N. Arnold, Richard S. Falk, and Ragnar Winther. Multigrid in h (div) and h (curl). *Numerische Mathematik*, 85(2):197–217, Apr 2000.

[3] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[4] Michele Benzi, Gene H. Golub, and Jörg Liesen. Numerical solution of saddle point problems. *Acta Numer.*, 14:1–137, 2005.

[5] Michele Benzi and Andrew J. Wathen. Some preconditioning techniques for saddle point problems. In *Model order reduction: theory, research aspects and applications*, volume 13 of *Math. Ind.*, pages 195–211. Springer, Berlin, 2008.

[6] Franco Brezzi and Michel Fortin. *Mixed and hybrid finite element methods*, volume 15. Springer Science & Business Media, 2012.

[7] J Cahouet and J-P Chabard. Some fast 3d finite element solvers for the generalized stokes problem. *International Journal for Numerical Methods in Fluids*, 8(8):869–895, 1988.

[8] C. Chevalier and F. Pellegrini. PT-SCOTCH: a tool for efficient parallel graph ordering. *Parallel Computing*, 6-8(34):318–331, 2008.

[9] Eric de Sturler and Jörg Liesen. Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. I. Theory. *SIAM J. Sci. Comput.*, 26(5):1598–1619, 2005.

[10] Victorita Dolean, Pierre Jolivet, and Frédéric Nataf. *An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation.* SIAM, 2015.

[11] Daniel Drzisga, Lorenz John, Ulrich Rude, Barbara Wohlmuth, and Walter Zulehner. On the analysis of block smoothers for saddle point problems. *SIAM Journal on Matrix Analysis and Applications*, 39(2):932–960, 2018.

[12] Yalchin Efendiev, Juan Galvis, and Thomas Y Hou. Generalized multiscale finite element methods (gmsfem). *Journal of Computational Physics*, 251:116–135, 2013.

[13] Charbel Farhat and Francois-Xavier Roux. A method of Finite Element Tearing and Interconnecting and its parallel solution algorithm. *Int. J. Numer. Meth. Engrg.*, 32:1205–1227, 1991.

[14] Patrick E. Farrell, Lawrence Mitchell, and Florian Wechsung. An Augmented Lagrangian Preconditioner for the 3D Stationary Incompressible Navier–Stokes Equations at High Reynolds Number. *SIAM J. Sci. Comput.*, 41(5):A3073–A3096, 2019.

[15] M. Griebel and P. Oswald. On the abstract theory of additive and multiplicative Schwarz algorithms. *Numer. Math.*, 70(2):163–180, 1995.

[16] Ryadh Haferssas, Pierre Jolivet, and Frédéric Nataf. An additive schwarz method type theory for lions's algorithm and a symmetrized optimized restricted additive schwarz method. *SIAM Journal on Scientific Computing*, 39(4):A1345–A1365, 2017.

[17] F. Hecht. New development in Freefem++. *J. Numer. Math.*, 20(3-4):251–265, 2012.

[18] R. Hiptmair. Multigrid method for Maxwell's equations. *SIAM J. Numer. Anal.*, 36(1):204–225, 1998.

[19] Ralf Hiptmair. Multigrid method for h (div) in three dimensions. *Electron. Trans. Numer. Anal*, 6(1):133–152, 1997.

[20] Patrick Jenny, Seong H Lee, and Hamdi A Tchelepi. Adaptive multiscale finite-volume method for multiphase flow and transport in porous media. *Multiscale Modeling & Simulation*, 3(1):50–64, 2005.

[21] Pierre Jolivet and Frédéric Nataf. Hpddm: High-Performance Unified framework for Domain Decomposition methods, MPI-C++ library. https://github.com/hpddm/hpddm, 2014.

[22] G. Karypis and V. Kumar. METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. Technical report, Department of Computer Science, University of Minnesota, 1998. http://glaros.dtc.umn.edu/gkhome/views/metis.

[23] Axel Klawonn. An optimal preconditioner for a class of saddle point problems with a penalty term. *SIAM Journal on Scientific Computing*, 19(2):540–552, 1998.

[24] Richard B Lehoucq, Danny C Sorensen, and Chao Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, volume 6. SIAM, 1998.

[25] Chupeng Ma, Robert Scheichl, and Tim Dodwell. Novel design and analysis of generalized fe methods based on locally optimal spectral approximations. *arXiv preprint arXiv:2103.09545*, 2021.

[26] Malcolm F. Murphy, Gene H. Golub, and Andrew J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 2000.

[27] Sergey V. Nepomnyaschikh. Mesh theorems of traces, normalizations of function traces and their inversions. *Sov. J. Numer. Anal. Math. Modeling*, 6:1–25, 1991.

[28] Joseph E Pasciak and Jun Zhao. Overlapping schwarz methods in h (curl) on polyhedral domains. *Journal of Numerical Mathematics*, 10(3):221–234, 2002.

[29] Luca F. Pavarino and Olof B. Widlund. Balancing Neumann-Neumann methods for incompressible Stokes equations. *Comm. Pure Appl. Math.*, 55(3):302–335, 2002.

[30] T. Rees and M. Wathen. An element-based preconditioner for mixed finite element problems. *SIAM Journal on Scientific Computing*, 2020.

[31] S. Reitzinger and J. Schöberl. An algebraic multigrid method for finite element discretizations with edge elements. *Numerical Linear Algebra with Applications*, 9(3):223–238, 2002.

[32] Nicole Spillane, Victorita Dolean, Patrice Hauret, Frédéric Nataf, Clemens Pechstein, and Robert Scheichl. Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps. *Numer. Math.*, 126(4):741–770, 2014.

[33] Nicole Spillane and Daniel Rixen. Automatic spectral coarse spaces for robust finite element tearing and interconnecting and balanced domain decomposition algorithms. *Internat. J. Numer. Methods Engrg.*, 95(11):953–990, 2013.

[34] Andrea Toselli and Olof Widlund. *Domain Decomposition Methods - Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, 2005.

[35] Pierre-Henri Tournier and Frédéric Nataf. FFDDM: Freefem domain decomposition methd. https://doc.freefem.org/documentation/ffddm/index.html, 2019.

[36] Xuemin Tu and Jing Li. A FETI-DP type domain decomposition algorithm for three-dimensional incompressible Stokes equations. *SIAM J. Numer. Anal.*, 53(2):720–742, 2015.

[37] O Widlund, Stefano Zampini, S Scacchi, and L Pavarino. Block feti–dp/bddc preconditioners for mixed isogeometric discretizations of three-dimensional almost incompressible elasticity. *Mathematics of Computation*, 90(330):1773–1797, 2021.

[38] Notay Y. Convergence of some iterative methods for symmetric saddle point linear systems. *SIMAX*, 40:122–146, 2019.