INTERNATIONAL CENTER FOR NUMERICAL METHODS IN ENGINEERING

A Study on the Lumped Preconditioner and Memory Requirements of FETI and Related Primal Domain Decomposition Methods

Y. Fragakis

Publication CIMNE Nº-291, May 2006

A Study on the Lumped Preconditioner and Memory Requirements of FETI and Related Primal Domain Decomposition Methods

Y. Fragakis

Publication CIMNE Nº-291, May 2006

International Center for Numerical Methods in Engineering
Gran Capitán s/n, 08034 Barcelona, Spain

A study on the lumped preconditioner and memory requirements of FETI and related primal domain decomposition methods

Yannis Fragakis

International Center for Numerical Methods in Engineering (CIMNE)
Technical University of Catalonia
Edificio C1, Campus Norte, Gran Capitan s/n, Barcelona, 08034, Spain
fragayan@cimne.upc.edu

March 27, 2006

Abstract

In recent years, Domain Decomposition Methods (DDM) have emerged as advanced solvers in several areas of computational mechanics. In particular, during the last decade, in the area of solid and structural mechanics, they reached a considerable level of advancement and were shown to be more efficient than popular solvers, like advanced sparse direct solvers. The present contribution follows the lines of a series of recent publications by the authors on DDM. In these papers, the authors developed a unified theory of primal and dual methods and presented a family of DDM that were shown to be more efficient than previous methods. The present paper extends this work, presenting a new family of related DDM, thus enriching the theory of the relations between primal and dual methods. It also explores memory requirement issues, suggesting also a particularly memory efficient formulation.

1 Introduction

In the last decade Domain Decomposition Methods (DDM) have progressed significantly leading to a large number of methods and techniques, capable of giving solution to various problems of computational mechanics. In the field of solid and structural mechanics, in particular, this fruitful period has led to the extensive parallel development of two large families of methods: (a) the Finite Element Tearing and Interconnecting (FETI) methods and (b) the Balancing Domain Decomposition (BDD) methods. Both introduced at the beginning of the 90s [1,2], these two categories of methods today include a large number of variants. However, their distinct theories have led to a lack of extensive studies to interconnect them in the past. Thus, in the present decade two studies [3,4] have attempted to determine the relations between the two methods.

In particular, the studies [4,5] set the basis of a unified theory of primal and dual DDM. This effort also led to the introduction of a new family of methods, under the name "Primal class of FETI methods", or in abbreviation "P-FETI methods". These methods are derived from the Dirichlet preconditioned FETI methods. They, thus, inherit the high computational efficiency properties of these methods, while their primal flavor gives them increased efficiency and robustness in ill-conditioned problems. However, so far a primal alternative for the lumped preconditioned FETI methods has not been presented. Filling this hole is the object of the present study and even though the new formulations do not appear to exhibit all the advantages of the P-FETI formulations, they serve the purpose of diversifying our knowledge of the relations of primal and dual methods.

In modern DDM practice, the lumped preconditioner is used mostly in order to diminish memory requirements. Thus, in order to present a complete study of the lumped preconditioner in DDM, we also explore and compare the memory requirements of various DD formulations.

This paper thus presents the primal alternatives of the lumped preconditioned FETI methods and compares the efficiency of various DDM, with respect to their computational cost and memory requirements. In particular, it is organized as follows: Section 2 presents the base formulation of the introduced methods. Section 3 sets up some algebraic relations that lead to the transformation of the algorithms into a more economical form, which is derived in the section that follows. Finally, section 5 presents numerical results for comparing the new formulation with previous ones and section 6 gives some concluding statements.

2 Basic formulation of the primal alternatives of the FETI methods equipped with the lumped preconditioner

The P-FETI methods were built on the concept of preconditioning the Schur complement method with the first estimate of displacements obtained using the FETI methods. Accordingly, the primal counterparts of the lumped preconditioned methods will be obtained by similarly preconditioning the intact global problem. Thus, the following equation

$$Ku = f \Leftrightarrow L^{\mathrm{T}} K^{s} L u = L^{\mathrm{T}} f^{s} \tag{1}$$

will be preconditioned with the first displacement estimate of a FETI method. In eq. (1), K, u, and f represent the global stiffness matrix, displacement and force vectors, respectively, while

$$K^s = \begin{bmatrix} K^{(1)} & & \\ & \ddots & \\ & & K^{(n_s)} \end{bmatrix} \quad , \quad u^s = \begin{bmatrix} u^{(1)} \\ \vdots \\ u^{(n_s)} \end{bmatrix} \quad \text{and} \quad f^s = \begin{bmatrix} f^{(1)} \\ \vdots \\ f^{(n_s)} \end{bmatrix} \tag{2}$$

are the matrix block-diagonal assemblage of the corresponding quantities of the subdomains $s = 1,..., n_s$ and L is a Boolean restriction matrix, such that $u^s = Lu$. Using the original FETI formulation, usually referred to as "one-level FETI" or "FETI-1" [1], the following preconditioner for (1) is derived (this equation is obtained following an analysis almost identical to [4,section 6]):

$$\tilde{A}^{-1} = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p \tag{3}$$

where: $$\tilde{A}^{s^{-1}} = H^{\mathrm{T}} K^{s^+} H \quad , \quad H = I - B^{\mathrm{T}} QG(G^{\mathrm{T}} QG)^{-1} R^{s^{\mathrm{T}}} \quad \text{and} \quad G = BR^s \tag{4}$$

Here, R^s and K^{s^+} are the block-diagonal assemblage of subdomain zero energy modes and generalized inverses of subdomain stiffness matrices, respectively. B is a mapping matrix such that $\mathrm{null}(B) = \mathrm{range}(L)$, Q is a symmetric positive definite matrix used in the FETI-1 coarse projector (see for instance [6]), while L_p and B_p are scaled variants of L and B (see the expressions gathered from various DDM papers in [4]).

Furthermore, if we use the original dual-primal FETI (FETI-DP) formulation [7], we obtain:

$$\tilde{A}^{-1} = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p \tag{5}$$

where: $$\tilde{A}^{s^{-1}} = N_r^{s^{\mathrm{T}}} K_{rr}^{s^{-1}} N_r^s + \left(N_c^{s^{\mathrm{T}}} - N_r^{s^{\mathrm{T}}} K_{rr}^{s^{-1}} K_{rc}^s \right) L_c K_{cc}^{*^{-1}} L_c^{\mathrm{T}} \left(-K_{cr}^s K_{rr}^{s^{-1}} N_r^s + N_c^s \right) \tag{6}$$

$$K_{cc}^* = L_c^{\mathrm{T}} S_{cc}^s L_c \quad \text{and} \quad S_{cc}^s = K_{cc}^s - K_{cr}^s K_{rr}^{s^{-1}} K_{rc}^s \tag{7}$$

In eq. (6), subscripts c and r denote the restriction of the matrices to the coarse problem d.o.f. and the remaining d.o.f., respectively. Matrix N_r^s is a Boolean matrix which extracts the subdomain d.o.f. that do not belong to the coarse problem, from subdomain d.o.f. vectors, like in equation $u_r^s = N_r^s u^s$. Furthermore, matrix N_c^s is used in eq. (6), in order to extract the coarse problem d.o.f. from global d.o.f. vectors, like in equation $u_c^s = N_c^s u^s$.

Comparing the lumped preconditioned FETI-1 or FETI-DP method with the methods of this section, it is noted that the methods presented above have a significantly higher computational cost, because they operate on the full displacement vector u of the structure and also need multiplications with the full stiffness matrices of the subdomains. In order to diminish their cost, their algorithm will be transformed into a more economical version, by representing primal variables with dual variables.

3 Auxiliary equations leading to the change of variables of the algorithms

In order to perform the transformation of variables, we need to prove some relations of the introduced matrices. First, the L and B matrices satisfy the (some of the most complete studies on these equations can be found in [3,8]):

$$\text{range}(L) = \text{null}(B) \quad \text{and} \quad \text{range}(L_p) = \text{null}(B_p) \tag{8}$$

$$(L_p L^{\mathrm{T}})^2 = L_p L^{\mathrm{T}} \quad \text{and} \quad (B^{\mathrm{T}} B_p)^2 = B^{\mathrm{T}} B_p \tag{9}$$

$$L^{\mathrm{T}} L_p = I \quad \text{and} \quad L_p L^{\mathrm{T}} + B^{\mathrm{T}} B_p = I \tag{10}$$

In the following two subsections, we show that for both preconditioners (3) and (5) the following equations hold:

$$\tilde{A}^{s^{-1}} K^s L = L \tag{11}$$

$$\tilde{A}^{s^{-1}} K^s \tilde{A}^{s^{-1}} = \tilde{A}^{s^{-1}} \tag{12}$$

3.1 Auxiliary equations of the primal alternative of the lumped preconditioned FETI-1 method

In this subsection we prove that eqs. (11) and (12) hold when $\tilde{A}^{s^{-1}}$ is derived from the FETI-1 method (eq. (4)). First, H is a projector satisfying

$$R^{s^{\mathrm{T}}} H = 0 \quad , \quad H B^{\mathrm{T}} Q G = 0 \quad , \quad H^2 = H \quad \text{and} \quad H^{\mathrm{T}} L = L \tag{13}$$

The stiffness matrices satisfy the relations:

$$K^s R^s = 0 \quad \text{and} \quad H K^s = K^s \tag{14}$$

and the property: There is a matrix Y such that:

$$K^{s^+} K^s = I + R^s Y \tag{15}$$

Using eqs. (13) - (15) we obtain:

$$\tilde{A}^{s^{-1}} K^s = H^{\mathrm{T}} K^{s^+} H K^s = H^{\mathrm{T}} K^{s^+} K^s = H^{\mathrm{T}} \left(I + R^s Y \right) = H^{\mathrm{T}} + H^{\mathrm{T}} R^s Y = H^{\mathrm{T}} \tag{16}$$

Finally, fro eq. (13) and (16), we obtain the two equations that we want to prove:

$$\tilde{A}^{s^{-1}} K^s L = H^{\mathrm{T}} L = L \tag{17}$$

$$\tilde{A}^{s^{-1}} K^s \tilde{A}^{s^{-1}} = H^{\mathrm{T}} H^{\mathrm{T}} K^{s^+} H = \tilde{A}^{s^{-1}} \tag{18}$$

3.2 Auxiliary equations of the primal alternative of the FETI-DP method

In this section we prove that eqs. (11) and (12) also hold if $\tilde{A}^{s^{-1}}$ is defined by eq. (6). First, we note that matrices introduced in eq. (6) satisfy relations:

$$N_r^s N_r^{s^{\mathrm{T}}} = I \quad , \quad N_c^s N_c^{s^{\mathrm{T}}} = I \quad , \quad N_c^s N_r^{s^{\mathrm{T}}} = 0 \quad , \quad N_r^{s^{\mathrm{T}}} N_r^s + N_c^{s^{\mathrm{T}}} N_c^s = I \tag{19}$$

$$K_{rr}^s = N_r^s K^s N_r^{s^{\mathrm{T}}} \quad , \quad K_{rc}^s = N_r^s K^s N_c^{s^{\mathrm{T}}} \quad \text{and} \quad K_{cc}^s = N_c^s K^s N_c^{s^{\mathrm{T}}} \tag{20}$$

Furthermore, we note that K^s may be decomposed in the form:

$$K^s = \begin{bmatrix} N_r^{s\mathrm{T}} & N_c^{s\mathrm{T}} \end{bmatrix} \begin{bmatrix} K_{rr}^s & K_{rc}^s \\ K_{cr}^s & K_{cc}^s \end{bmatrix} \begin{bmatrix} N_r^s \\ N_c^s \end{bmatrix} \tag{21}$$
$$= N_r^{s\mathrm{T}} K_{rr}^s N_r^s + N_r^{s\mathrm{T}} K_{rc}^s N_c^s + N_c^{s\mathrm{T}} K_{cr}^s N_r^s + N_c^{s\mathrm{T}} K_{cc}^s N_c^s$$

Using decomposition (21), eqs. (19) and doing the matrix algebra, we obtain:

$$\tilde{A}^{s^{-1}} K^s = N_r^{s\mathrm{T}} N_r^s + N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s N_c^s + \left(N_c^{s\mathrm{T}} - N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s \right) L_c K_{cc}^{*^{-1}} L_c^{\mathrm{T}} S_{cc}^s N_c^s \tag{22}$$

We note that $range(N_c^s L) = range(L_c)$, which implies that:

$$\exists X : N_c^s L = L_c X \tag{23}$$

Thus, using eqs (22) and (23) we prove eq. (11):

$$\begin{aligned} \tilde{A}^{s^{-1}} K^s L &= N_r^{s\mathrm{T}} N_r^s L + N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s N_c^s L + \left(N_c^{s\mathrm{T}} - N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s \right) L_c K_{cc}^{*^{-1}} L_c^{\mathrm{T}} S_{cc}^s L_c X \\ &= N_r^{s\mathrm{T}} N_r^s L + N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s N_c^s L + \left(N_c^{s\mathrm{T}} - N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s \right) L_c K_{cc}^{*^{-1}} K_{cc}^* X \\ &= N_r^{s\mathrm{T}} N_r^s L + N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s N_c^s L + N_c^{s\mathrm{T}} N_c^s L - N_r^{s\mathrm{T}} K_{rr}^{s^{-1}} K_{rc}^s N_c^s L \\ &= (N_r^{s\mathrm{T}} N_r^s + N_c^{s\mathrm{T}} N_c^s) L = L \end{aligned} \tag{24}$$

Finally, it suffices substituting definition (6) and eq. (22) to eq. (12) to obtain, after doing the necessary matrix algebra:

$$\tilde{A}^{s^{-1}} K^s \tilde{A}^{s^{-1}} = \tilde{A}^{s^{-1}} \tag{25}$$

REMARK 1. It can be shown that eqs. (24) and (25) also hold when the coarse d.o.f. include edge or face averages of the interface between the subdomains. In order to prove this, we generalize the definition of u_c^s and u_r^s, so that preconditioner (6) expresses all vertex, edge and face conditions of coarse displacement continuity, that are usually used in FETI-DP, PFETI-PD and BDDC literature*. Hence, we perform a change of basis in subdomain d.o.f., as follows:

$$u^s = \begin{bmatrix} N_c^{s\mathrm{T}} & N_r^{s\mathrm{T}} \end{bmatrix} \begin{bmatrix} u_c^s \\ u_r^s \end{bmatrix} \tag{26}$$

where matrix $\begin{bmatrix} N_c^{s\mathrm{T}} & N_r^{s\mathrm{T}} \end{bmatrix}$ is orthogonal and its subblocks satisfy eqs. (19), (23) and

---

* The BDDC method was introduced in [9] and it can be shown identical to the PFETI-DP with any vertex, edge or face coarse constraints [10]. In fact, this method was introduced independently in three studies: (a) as a preconditioner based on constrained energy minimization in [9] and later called BDDC, (b) as the primal derivative of the FETI-DP with only vertex constraints [7] or vertex, edge and face constraints [11] in [4] (In fact as primal alternative of the FETI-DP it was first mentioned and tested in an earlier publication [12]) and (c) as a preconditioner inspired from FETI-DP in [13]. In fact, even though the work in [13] is apparently restricted to vertex constraints and homogeneous scaling, that paper probably derived this method in the simplest and most intuitive way.

$$\exists Z : N_r^s L = L_r Z \tag{27}$$

where L_r is a Boolean restriction matrix that ensures continuity of u_r^s across the interface with the relation $u_r^s = L_r u_r$. Under the above definitions, matrix N_c^s can express all vertex, edge-averaged of face-averaged coarse d.o.f. that are used in FETI-DP, PFETI-PD and BDDC literature. Eq. (26) implies:

$$\begin{bmatrix} u_c^s \\ u_r^s \end{bmatrix} = \begin{bmatrix} N_c^s \\ N_r^s \end{bmatrix} u^s \tag{28}$$

With definition (26), we define matrices K_{rr}^s, K_{rc}^s and K_{cc}^s using eqs. (20). With the definitions that have introduced in this remark, the preconditioner (5) expresses all vertex, edge of face coarse d.o.f. compatibility conditions that are usually used in FETI-DP, PFETI-PD and BDDC literature. Furthermore, it can be easily verified that all relations used so far in section 3.2 still hold if matrices N_c^s and N_r^s are defined as in eq. (26). Following this analysis, one concludes that eqs. (24) and (25) also hold when the coarse d.o.f. include edge or face averages. It is worth noting that the change of basis (26) is similar to a change of basis used to define coarse d.o.f. in [14]. The difference in our case is that the change of basis is orthogonal because this makes our proofs straightforward.

3.3 Auxiliary equations regarding both primal methods based on FETI-1 and FETI-DP

In this subsection, we prove some relations that are based on eqs. (8) - (12) and thus hold for both preconditioners (3) and (5). First, we have:

$$L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} K^s L = L_p^{\mathrm{T}} L = I \quad , \quad B\tilde{A}^{s^{-1}} K^s L = BL = 0 \tag{29}$$

$$\begin{aligned} K\tilde{A}^{-1} &= L^{\mathrm{T}} K^s L \; L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p = L^{\mathrm{T}} K^s (I - B_p^{\mathrm{T}} B) \tilde{A}^{s^{-1}} L_p = \\ & L^{\mathrm{T}} K^s \tilde{A}^{s^{-1}} L_p - L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} L_p = I - L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} L_p \end{aligned} \tag{30}$$

$$\begin{aligned} & L^{\mathrm{T}} K^s L L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} = L^{\mathrm{T}} K^s \left(I - B_p^{\mathrm{T}} B \right) \tilde{A}^{s^{-1}} B^{\mathrm{T}} \\ & = L^{\mathrm{T}} K^s \tilde{A}^{s^{-1}} B^{\mathrm{T}} - L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} B^{\mathrm{T}} = -L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} B^{\mathrm{T}} \end{aligned} \tag{31}$$

In addition, using the above relations we have:

$$\begin{aligned} \tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} &= \tilde{A}^{s^{-1}} K^s \left(I - L L_p^{\mathrm{T}} \right) \tilde{A}^{s^{-1}} = \tilde{A}^{s^{-1}} K^s \tilde{A}^{s^{-1}} - \tilde{A}^{s^{-1}} K^s L L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} \\ &= \tilde{A}^{s^{-1}} - L L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} = B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} \end{aligned} \tag{32}$$

$$L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} = L_p^{\mathrm{T}} B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} = 0 \tag{33}$$

From eq. (32), it follows that:

$$B\tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} = B B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} = B(I - L L_p^{\mathrm{T}}) \tilde{A}^{s^{-1}} = B\tilde{A}^{s^{-1}} \tag{34}$$

and using eq. (33):

$$L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}L_pL^{\mathrm{T}}K^sB_p^{\mathrm{T}}B\tilde{A}^{s^{-1}} = L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}\left(I - B^{\mathrm{T}}B_p\right)K^sB_p^{\mathrm{T}}B\tilde{A}^{s^{-1}}$$
$$= L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}K^sB_p^{\mathrm{T}}B\tilde{A}^{s^{-1}} - L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}B^{\mathrm{T}}B_pK^sB_p^{\mathrm{T}}B\tilde{A}^{s^{-1}} = -L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}B^{\mathrm{T}}B_pK^sB_p^{\mathrm{T}}B\tilde{A}^{s^{-1}} \quad (35)$$

4 Final form of the algorithms

In this section, it will be shown that if the initial solution vector of the PCG algorithm applied for the solution of eq. (1) , with the preconditioner of eq. (3) or (5), is set equal to (In what follows, we use the notation and steps of Algorithm 1):

$$u^0 = \tilde{A}^{-1}f \quad (36)$$

then there exist suitable vectors (denoted below with the subscript "1"), such that the following variables of the PCG can be written in the forms ($k = 0,1,\ldots$):

$$z^k = -L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}B^{\mathrm{T}}z_1^k \quad , \quad p^k = -L_p^{\mathrm{T}}\tilde{A}^{s^{-1}}B^{\mathrm{T}}p_1^k \quad , \quad r^k = L^{\mathrm{T}}K^sB_p^{\mathrm{T}}r_1^k \quad \text{and} \quad q^k = L^{\mathrm{T}}K^sB_p^{\mathrm{T}}q_1^k \quad (37)$$

Eqs. (37) allow expressing the PCG vectors, which have the size of the total number of degrees of freedom (d.o.f.), with respect to vectors whose size is equal to the row size of matrix B (which in turn is equal to the number of Lagrange multipliers used in dual DDM). They thus lead to a reduction of the cost of the algorithm. Before proceeding with the proof, we will study how the linear combinations and dot products performed by the PCG algorithm are transformed due to eqs. (37).

The linear combinations of the PCG vectors are simply transformed to linear combinations of Lagrange multiplier vectors. For instance: using eq. (37) the PCG residual becomes:

$$r^k = r^{k-1} - \eta^{k-1}q^{k-1} = L^{\mathrm{T}}K^sB_p^{\mathrm{T}}r_1^{k-1} - \eta^{k-1}L^{\mathrm{T}}K^sB_p^{\mathrm{T}}q_1^{k-1}$$
$$= L^{\mathrm{T}}K^sB_p^{\mathrm{T}}(r_1^{k-1} - \eta^{k-1}q_1^{k-1}) \Rightarrow r_1^{k-1} = r_1^{k-1} - \eta^{k-1}q_1^{k-1} \quad (38)$$

In order to discuss the transformation of the PCG dot products, we define vectors:

$$z_2^k = B\tilde{A}^{s^{-1}}B^{\mathrm{T}}z_1^k \quad \text{and} \quad z_3^k = B_pK^sB_p^{\mathrm{T}}z_2^k \quad , \quad p_2^k = B\tilde{A}^{s^{-1}}B^{\mathrm{T}}p_1^k \quad \text{and} \quad p_3^k = B_pK^sB_p^{\mathrm{T}}p_2^k \quad (39)$$

- Initialize

$$r^0 = b - Ku^0 \quad , \quad z^0 = \tilde{A}^{-1}r^0 \quad , \quad p^0 = z^0 \quad , \quad q^0 = Kp^0 \quad , \quad \eta^0 = \frac{p^{0^{\mathrm{T}}}r^0}{p^{0^{\mathrm{T}}}q^0}$$

- Iterate $k = 1,2,\ldots$ until convergence

$$u^k = u^{k-1} + \eta^{k-1}p^{k-1} \quad , \quad r^k = r^{k-1} - \eta^{k-1}q^{k-1} \quad , \quad z^k = \tilde{A}^{-1}r^k$$
$$p^k = z^k - \sum_{i=0}^{k-1}\frac{z^{k^{\mathrm{T}}}q^i}{p^{i^{\mathrm{T}}}q^i}p^i \quad , \quad q^k = Kp^k \quad , \quad \eta^k = \frac{p^{k^{\mathrm{T}}}r^k}{p^{k^{\mathrm{T}}}q^k}$$

Algorithm 1. The PCG algorithm for solving system $Ku = f$ preconditioned with $\tilde{A}^{-1}$ (full reorthogonalization)

$$r_2^k = B_p K^s B_p^{\mathrm{T}} r_1^k \quad \text{and} \quad r_3^k = B\tilde{A}^{s^{-1}} B^{\mathrm{T}} r_2^k \quad , \quad q_2^k = B_p K^s B_p^{\mathrm{T}} q_1^k \quad \text{and} \quad q_3^k = B\tilde{A}^{s^{-1}} B^{\mathrm{T}} q_2^k \quad (40)$$

Then for instance (using eqs. (10) and (37)), the following dot product becomes:

$$\begin{aligned} p^{k^{\mathrm{T}}} q^k &= -p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} L_p L^{\mathrm{T}} K^s B_p^{\mathrm{T}} q_1^k = -p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} (I - B^{\mathrm{T}} B_p) K^s B_p^{\mathrm{T}} q_1^k \\ &= -p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} q_1^k + p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} q_1^k \end{aligned} \quad (41)$$

Here, it is needed to make the following assumption, which will also be verified in the proof that follows: $q_1^k \in \text{range}(B\tilde{A}^{s^{-1}})$. Thus, there is a vector y such that $q_1^k = B\tilde{A}^{s^{-1}} y$ and the first term of eq. (41) becomes (using eq. (34)):

$$-p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} q_1^k = -p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} K^s B_p^{\mathrm{T}} B\tilde{A}^{s^{-1}} y = -p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} y = -p_1^{k^{\mathrm{T}}} q_1^k \quad (42)$$

For the second term, there are the following three choices (taking into account the assumptions (37) and the definitions (39) - (40)):

$$p_1^{k^{\mathrm{T}}} B\tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} q_1^k = \begin{cases} p_2^{k^{\mathrm{T}}} q_2^k \\ p_1^{k^{\mathrm{T}}} q_3^k \\ p_3^{k^{\mathrm{T}}} q_1^k \end{cases} \quad (43)$$

Thus, using each of these choices and expression (42), the dot product (41) takes the following three expressions:

$$p^{k^{\mathrm{T}}} q^k = \begin{cases} p_2^{k^{\mathrm{T}}} q_2^k - p_1^{k^{\mathrm{T}}} q_1^k \\ -p_1^{k^{\mathrm{T}}} q_1^k + p_1^{k^{\mathrm{T}}} q_3^k = p_1^{k^{\mathrm{T}}} (q_3^k - q_1^k) \\ -p_1^{k^{\mathrm{T}}} q_1^k + p_3^{k^{\mathrm{T}}} q_1^k = (p_3^{k^{\mathrm{T}}} - p_1^{k^{\mathrm{T}}}) q_1^k \end{cases} \quad (44)$$

where it is noted that the last two options are more cost effective. Like dot product $p^{k^{\mathrm{T}}} q^k$ which was used here as an example, all dot products of the PCG can be expressed in the form of dot products of Lagrange multiplier vectors, like in eqs. (44).

We are now ready to proceed to the proof of eqs. (37), proving also that there are vectors y such that $q_1^k = B\tilde{A}^{s^{-1}} y$ and $r_1^k = B\tilde{A}^{s^{-1}} y$ and obtaining the transformed algorithm with respect to Lagrange multiplier vectors. We simply follow the steps of Algorithm 1. Thus, from eq. (36) it follows (using eq. (30)):

$$r^0 = f - Au^0 = f - A\tilde{A}^{-1} f = f - (I - L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B\tilde{A}^{s^{-1}} L_p) f = L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B\tilde{A}^{s^{-1}} L_p f \quad (45)$$

and:

$$r_1^0 = B\tilde{A}^{s^{-1}} L_p f \quad (46)$$

Computing the residual r^0 from the above equations, we get:

$$r^0 = L^{\mathrm{T}} K^s B_p^{\mathrm{T}} r_1^0 \Rightarrow \begin{bmatrix} r_b^0 \\ r_{i(b)}^0 \end{bmatrix} = \begin{bmatrix} L_b^{\mathrm{T}} \\ I \end{bmatrix} \begin{bmatrix} K_{bb}^s & K_{ib}^s \\ K_{ib}^s & K_{ii}^s \end{bmatrix} \begin{bmatrix} B_{p_b}^{\mathrm{T}} \\ 0 \end{bmatrix} r_1^0 = \begin{bmatrix} L_b^{\mathrm{T}} K_{bb}^s \\ K_{ib}^s \end{bmatrix} B_{p_b}^{\mathrm{T}} r_1^0 \quad (47)$$

where subscripts b and i restrict the matrices to interface or internal d.o.f. of the subdomains, respectively. In eq. (47) it is worth noting that the residual vanishes in internal

d.o.f. of the subdomains, when these d.o.f. are not adjacent to interface d.o.f., which is also observed in the lumped preconditioned FETI methods. Furthermore:

$$r_2^0 = B_p K^s B_p^{\mathrm{T}} r_1^0 = \begin{bmatrix} B_{p_b}^{\mathrm{T}} \\ 0 \end{bmatrix} \begin{bmatrix} K_{bb}^s & K_{ib}^s \\ K_{ib}^s & K_{ii}^s \end{bmatrix} \begin{bmatrix} B_{p_b}^{\mathrm{T}} \\ 0 \end{bmatrix} r_1^0 = B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}} r_1^0 \tag{48}$$

where it is worth noting that matrix $B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}}$ is equal to the lumped preconditioner of FETI-1 method (with respect to the lumped preconditioner of FETI-DP, matrix $B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}}$ is only augmented with some extra lines and columns). Then, using eq. (35) it follows that there is a y such that (here $y = L_p f$ from eq. (46))

$$\begin{aligned} z^0 &= \tilde{A}^{-1} r^0 = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p L^{\mathrm{T}} K^s B_p^{\mathrm{T}} r_1^0 = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} y \\ &= -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} y = -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} r_1^0 = -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} r_2^0 \end{aligned} \tag{49}$$

and: $$z_1^0 = r_2^0 \quad , \quad z_2^0 (= r_3^0) = B \tilde{A}^{s^{-1}} B^{\mathrm{T}} z_1^0 \quad \text{and} \quad p^0 = z^0 \Rightarrow \begin{cases} p_1^0 = z_1^0 \\ p_2^0 = z_2^0 \end{cases} \tag{50}$$

Then, using eq. (31):

$$q^0 = A p^0 = -L^{\mathrm{T}} K^s L L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} p_1^0 = L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B \tilde{A}^{s^{-1}} B^{\mathrm{T}} p_1^0 = L^{\mathrm{T}} K^s B_p^{\mathrm{T}} p_2^0 \tag{51}$$

Therefore:

$$q_1^0 (= z_3^0) = p_2^0 = B \tilde{A}^{s^{-1}} B^{\mathrm{T}} p_1^0 \Rightarrow q_1^0 \in \text{ range}(B \tilde{A}^{s^{-1}}) \quad \text{and} \quad q_2^0 (= p_3^0) = B_p K^s B_p^{\mathrm{T}} q_1^0 \tag{52}$$

Then (similar to eq. (44) and using the fact that both r_1^0, $q_1^0 \in$ range$(B \tilde{A}^{s^{-1}})$:

$$\eta^0 = \frac{p^{0^{\mathrm{T}}} r^0}{p^{0^{\mathrm{T}}} q^0} = \frac{(p_3^{0^{\mathrm{T}}} - p_1^{0^{\mathrm{T}}}) r_1^0}{(p_3^{0^{\mathrm{T}}} - p_1^{0^{\mathrm{T}}}) q_1^0} \tag{53}$$

So far, we have followed the steps of the initialisation of Algorithm 1 and have shown that eqs. (37) hold for $k = 0$. In the following, we show that if we assume that eqs. (37) hold for all previous iterations of the PCG, they will also hold for iteration k. Thus, this proof will be concluded recursively. We simply follow the steps of iteration k of the PCG. We thus have:

$$r^k = r^{k-1} - \eta^{k-1} q^{k-1} \quad , \quad r_1^k = r_1^{k-1} - \eta^{k-1} q_1^{k-1} \quad \text{and} \quad r_2^k = r_2^{k-1} - \eta^{k-1} q_2^{k-1} \tag{54}$$

which also implies that if r_1^{k-1}, $q_1^{k-1} \in$ range$(B \tilde{A}^{s^{-1}})$ then $r_1^k \in$ range$(B \tilde{A}^{s^{-1}})$.

Continuing, like in eq. (49) we obtain (provided that $r_1^k \in$ range$(B \tilde{A}^{s^{-1}})$):

$$\begin{aligned} z^k &= \tilde{A}^{-1} r^k = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p L^{\mathrm{T}} K^s B_p^{\mathrm{T}} r_1^k = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p L^{\mathrm{T}} K^s B_p^{\mathrm{T}} B H^{\mathrm{T}} y \\ &= -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} B H^{\mathrm{T}} y = -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} B_p K^s B_p^{\mathrm{T}} r_1^k = -L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} r_2^k \end{aligned} \tag{55}$$

and: $$z_1^k = r_2^k \quad \text{and} \quad z_2^k (= r_3^k) = B \tilde{A}^{s^{-1}} B^{\mathrm{T}} z_1^k \tag{56}$$

Then, in the case of full reorthogonalization:

$$p^k = z^k - \sum_{i=0}^{k-1} \frac{z^{k^{\mathrm{T}}} q^i}{p^{i^{\mathrm{T}}} q^i} p^i \quad , \quad p_1^k = z_1^k - \sum_{i=0}^{k-1} \frac{z^{k^{\mathrm{T}}} q^i}{p^{i^{\mathrm{T}}} q^i} p_1^i \text{ and } \quad p_2^k = z_2^k - \sum_{i=0}^{k-1} \frac{z^{k^{\mathrm{T}}} q^i}{p^{i^{\mathrm{T}}} q^i} p_2^i \tag{57}$$

The dot product terms are written (using eq. (44) and assuming $q_1^k \in$ range$(B\tilde{A}^{s^{-1}})$):

$$\frac{z^{k^{\mathrm{T}}} q^i}{p^{i^{\mathrm{T}}} q^i} = \frac{z_1^{k^{\mathrm{T}}} (q_3^i - q_1^i)}{p_1^{i^{\mathrm{T}}} (q_3^i - q_1^i)} \tag{58}$$

Then, like in eq. (51) (using eq. (39)):

$$q^k = Ap^k = L^{\mathrm{T}} K^s B_p^{\mathrm{T}} p_2^k \quad , \quad q_1^k = p_2^k = B\tilde{A}^{s^{-1}} B^{\mathrm{T}} p_1^k \quad \text{and} \quad q_2^k (= p_3^k) = B_p K^s B_p^{\mathrm{T}} p_2^k \tag{59}$$

which also shows that $q_1^k \in$ range$(B\tilde{A}^{s^{-1}})$. Note that this concludes recursively the proof that r_1^k, $q_1^k \in$ range$(B\tilde{A}^{s^{-1}})$, $k = 0,1,...$ Finally, we have:

$$\eta^k = \frac{p^{k^{\mathrm{T}}} r^k}{p^{k^{\mathrm{T}}} q^k} = \frac{(p_3^{k^{\mathrm{T}}} - p_1^{k^{\mathrm{T}}}) r_1^k}{(p_3^{k^{\mathrm{T}}} - p_1^{k^{\mathrm{T}}}) q_1^k} \tag{60}$$

In eq. (58), we use vector q_3^i that has not been computed yet. This vector will be computed using eq. (54) that implies that:

$$r_3^k = r_3^{k-1} - \eta^{k-1} q_3^{k-1} \Rightarrow q_3^{k-1} = \left(1/\eta^{k-1}\right)\left(r_3^{k-1} - r_3^k\right) \tag{61}$$

Hence, using the previous equations, the final form of the algorithm is obtained as is shown in Algorithm 2 (in the case of full reorthogonalization). It is worth noting that even though the formulation is primal, the final algorithm is very similar to the algorithm of the FETI-1 (or FETI-DP) method with the lumped preconditioner. In particular:

- With the appropriate definition of $\tilde{A}^{s^{-1}}$, matrices $B\tilde{A}^{s^{-1}} B^{\mathrm{T}}$ and $B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}}$ that are used during the iterations are equal to the FETI-1 matrix operator and lumped preconditioner, respectively. Compared to the matrix operator of FETI-DP, matrix $B\tilde{A}^{s^{-1}} B^{\mathrm{T}}$ is simply augmented with some redundant columns and rows and will thus have the same eigenspectrum as the FETI-DP operator, with the exception of some zero eigenvalues. Thus, using operator $B\tilde{A}^{s^{-1}} B^{\mathrm{T}}$ and preconditioner $B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}}$ for FETI-DP, we would simply obtain the same results.
- The algorithm iterates on vectors of the size of the Lagrange multipliers.
- From the equations that compute vectors r^k and q^k $(k = 0,1,...)$ in Algorithm 2, it follows that the residuals r^k vanish in internal d.o.f. of the subdomains, when these d.o.f. are not adjacent to the interface, again as in FETI-1 with the lumped preconditioner.

On the other hand, each iteration of the present algorithms requires more linear combinations of vectors than the corresponding dual algorithms. These operations become important in the case of reorthogonalization. In this case, the required dot products $z_1^{k^{\mathrm{T}}} (q_3^i - q_1^i)$, $i = 0,...,k-1$ imply the same computational cost as in FETI algorithms, because at each iteration $q_3^k - q_1^k$ is computed and stored. However, compared to FETI methods, this algorithm requires twice as many linear combinations for computing the vectors p_1^k and p_2^k, that represent the direction vectors p^k. In total, in this algorithm

- Initialize

$$u^0 = L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} L_p f \quad , \quad \tilde{u}^0 = 0 \quad , \quad r_1^0 = B\tilde{A}^{s^{-1}} L_p f \quad , \quad r^0 = \begin{bmatrix} L_b^{\mathrm{T}} K_{bb}^s \\ K_{ib}^s \end{bmatrix} B_{p_b}^{\mathrm{T}} r_1^0$$

$$p_1^0 = z_1^0 = B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}} r_1^0 \quad , \quad q_1^0 = p_2^0 = r_3^0 = z_2^0 = B\tilde{A}^{s^{-1}} B^{\mathrm{T}} z_1^0$$

$$q^0 = \begin{bmatrix} L_b^{\mathrm{T}} K_{bb}^s \\ K_{ib}^s \end{bmatrix} B_{p_b}^{\mathrm{T}} q_1^0 \quad , \quad p_3^0 = q_2^0 = B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}} q_1^0 \quad , \quad \eta^0 = \frac{(p_3^{0^{\mathrm{T}}} - p_1^{0^{\mathrm{T}}}) r_1^0}{(p_3^{0^{\mathrm{T}}} - p_1^{0^{\mathrm{T}}}) q_1^0}$$

- Iterate $k = 1,2,\ldots$ until convergence ($\| r^k \| < \varepsilon$)

$$\tilde{u}_1^k = \tilde{u}_1^k + \eta^{k-1} p_1^{k-1} \quad , \quad r^k = r^{k-1} - \eta^{k-1} q^{k-1} \quad , \quad r_1^k = r_1^{k-1} - \eta^{k-1} q_1^{k-1}$$

$$z_1^k = r_2^k = r_2^{k-1} - \eta^{k-1} q_2^{k-1} \quad , \quad r_3^k = z_2^k = B\tilde{A}^{s^{-1}} B^{\mathrm{T}} z_1^k \quad , \quad q_3^{k-1} = \left(1/\eta^{k-1}\right)\left(r_3^{k-1} - r_3^k\right)$$

$$p_1^k = z_1^k - \sum_{i=0}^{k-1} \frac{z_1^{k^{\mathrm{T}}} (q_3^i - q_1^i)}{p_1^{i^{\mathrm{T}}} (q_3^i - q_1^i)} p_1^i \quad , \quad q_1^k = p_2^k = z_2^k - \sum_{i=0}^{k-1} \frac{z_1^{k^{\mathrm{T}}} (q_3^i - q_1^i)}{p_1^{i^{\mathrm{T}}} (q_3^i - q_1^i)} p_2^i$$

$$q^k = \begin{bmatrix} L_b^{\mathrm{T}} K_{bb}^s \\ K_{ib}^s \end{bmatrix} B_{p_b}^{\mathrm{T}} p_2^k \quad , \quad p_3^k = q_2^k = B_{p_b}^{\mathrm{T}} K_{bb}^s B_{p_b}^{\mathrm{T}} p_2^k \quad , \quad \eta^k = \frac{(p_3^{k^{\mathrm{T}}} - p_1^{k^{\mathrm{T}}}) r_1^k}{(p_3^{k^{\mathrm{T}}} - p_1^{k^{\mathrm{T}}}) q_1^k}$$

- After convergence

$$u^k = u^0 - L_p^{\mathrm{T}} \tilde{A}^{s^{-1}} B^{\mathrm{T}} \tilde{u}_1^k$$

Algorithm 2: The primal alternative of the FETI-1 and FETI-DP methods with the lumped preconditioner (full reorthogonalization)

reorthogonalization requires 50% more floating point operations than in dual methods. In addition, while in FETI methods reorthogonalization requires storing two vectors per iteration, here it is required to store the three vectors p_1^k, p_2^k and $q_3^k - q_1^k$, which implies 50% higher memory requirements for reorthogonalization in Algorithm 2.

5 Numerical results

Our numerical tests are divided in two categories: First, we compare the computational cost of lumped preconditioned FETI methods with the introduced primal alternatives and then, the memory requirements of various DD formulations.

5.1 Comparison of the computational cost of dual and primal formulations originating from lumped preconditioned FETI methods

We have implemented the FETI-1 and FETI-DP methods with the lumped preconditioner and their primal alternatives in our Matlab DDM code and we consider the 3-D elasticity problem of Fig. 1. Details on our DDM implementations can be found in [4,5] This cubic structure is composed of five layers of two different materials and is discretized with $28\times28\times28$ 8-node brick elements. Additionally, it is pinned at the four

corners of its left surface. Various ratios E_A/E_B of the Young modulus and ρ_A/ρ_B of the density of the two materials are considered, while their Poisson ratio is set equal to $\nu_A = \nu_B = 0.30$. An optimal decomposition of this heterogeneous problem must generate subdomains with good aspect ratios, while preserving the material interfaces when partitioning the model [6]. Hence, two decompositions of this heterogeneous model of 73,155 d.o.f. in 100 subdomains are considered: In the first decomposition (Fig. 2a), the model has been partitioned in subdomains with good aspect ratios without taking into account the material interfaces (Decomposition P1). In the second decomposition (Fig. 2b), the five layers of different materials have been partitioned independently, thus generating a decomposition which preserves the material interfaces but produces subdomains of suboptimal aspect ratio in the thin layers (Decomposition P2).

Table 1 presents the iterations required by primal and dual formulations of the lumped preconditioned FETI-1 method. The results show that like in the case of comparing dual and primal formulations of the Dirichlet preconditioned FETI methods, the iterations of the two formulations of the lumped preconditioned FETI-1 methods are comparable. More precisely, it is noted that in the more ill-conditioned cases the primal method requires slightly fewer iterations (up to 11%) than the dual one. In fact, judging also from many other tests that we have performed comparing the two formulations of FETI-1 and FETI-

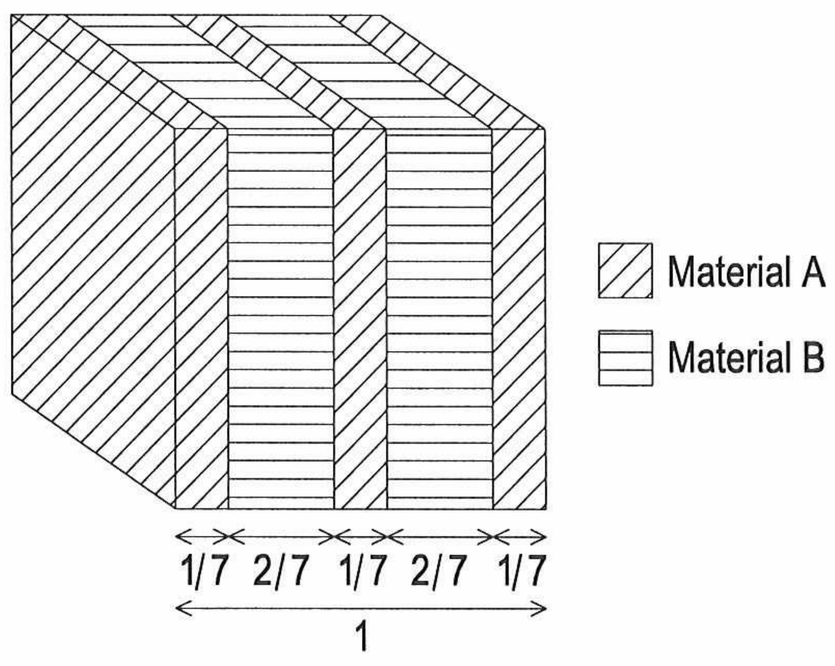
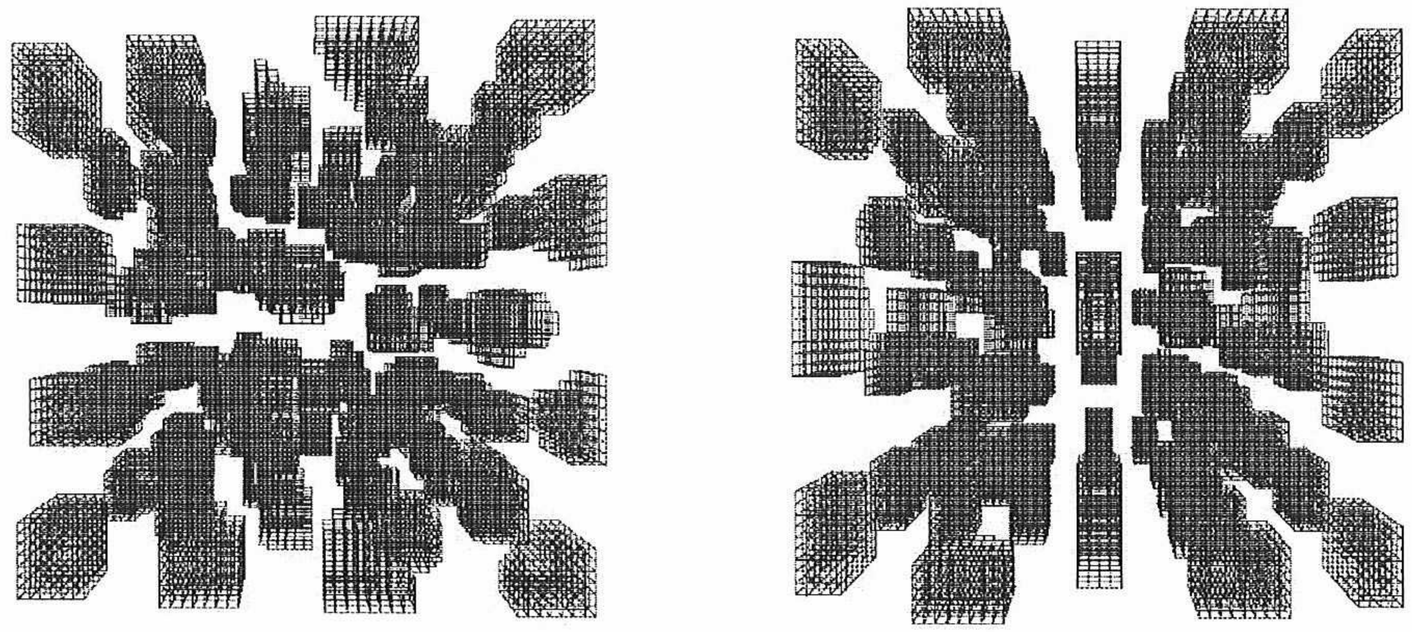


Figure 1. A cubic structure composed of two materials

Table 1

Number of iterations (Tolerance: 10^{-3}) of the lumped preconditioned FETI-1 method and its primal alternative for the solution of the example of Fig. 1

Ratio of Young moduli	Type of decomposition	Dual formulation	Primal formulation
10^0	P1	25	24
10^3	P1	44	41
10^3	P2	25	24
10^6	P1	53	47
10^6	P2	30	26



(a) (b)

Figure 2. Two decompositions of the cubic test problem in 100 subdomains:
(a) Optimal aspect ratio partitioning (Decomposition P1),
(b) Layered partitioning (Decomposition P2)

DP with the lumped preconditioner, it appears that the difference between the number of iterations of primal and dual formulations in ill-conditioned problems is more pronounced in the case of the lumped preconditioner than in the case of the Dirichlet preconditioner. A probable explanation is that the lumped preconditioned methods lead by themselves to more ill-conditioned systems than the Dirichlet ones.

On the other hand, bearing in mind that the primal formulation implies a 50% higher reorthogonalization cost, we conclude that statistically the primal formulation will be probably slower than the dual one for well-conditioned problems and probably faster for ill-conditioned problems with relatively low reorthogonalization cost. In addition, in the case of the lumped preconditioner, our results do not show the increased robustness (measured in terms of the maximum achievable solution accuracy in ill-conditioned problems) of the primal formulation that has been seen in the case of the P-FETI formulations. A probable explanation of this observation is given by the increased operations required in each iteration of the primal algorithm as opposed to the dual one and also by the fact that due to setting the initial solution vector equal to eq. (36), the initial residual of the primal methods is equal to the initial residual of the dual methods (see the residual of eq. (45), which is equal to the initial residual of the FETI methods). Thus, contrary to the P-FETI formulations, the residuals of the primal formulations of the lumped preconditioned FETI methods begin from relatively high values, as in the dual formulations.

5.2 Comparison of the memory requirements of dual and primal formulations

In the following tests, the memory requirements of selected primal and dual formulations are compared. Among the primal and dual formulations of lumped preconditioned methods we choose the dual formulation because it will usually require less memory than the primal formulation, since the primal formulation has a 50% higher overhead for storing the direction vectors. Among formulations originating from Dirichlet preconditioned methods, the best candidate is probably the primal formulation because it results in a statistically smaller number of iterations (the primal formulation also requires

storing direction vectors of the size of the number of interface d.o.f., while the dual formulation requires storing Lagrange multiplier sized vectors, that can be more than 25% lengthier in 3D problems [5]).

We will also test another P-FETI configuration, that reduces the memory required to store the factorized subdomain stiffness matrices K^s. Usually, the largest part of the memory required by FETI and P-FETI methods is occupied by this matrix. Given that in P-FETI formulations the factorized K^s matrix is involved in preconditioning operations, we can quite safely diminish the accuracy of the related operations. Here, we will test the results obtained from storing the factorized K^s in single precision. In particular, the exact strategy we use is: (a) Build and store K^s in double precision, then (b) Perform its factorization in double precision, (c) Ovewrite the factorized K^s with a copy of it in single precision, thus releasing half of the required memory, (d) Perform the required forward and backward substitutions in double precision, loading in memory the entries of the factorized K^s in double precision, whenever each entry is needed in the forward and backward substitution algorithm.

Thus, in the tests that follow we will compare this configuration with the lumped preconditioned FETI methods and P-FETI methods (with a fully double precision preconditioner). We suppose that the stiffness matrix $K^{(s)}$ of each subdomain is stored in skyline format. Note that at first we store the factorized $K^{(s)}$ in full double precision. Then, each processor (in the case of parallel processing) overwrites this matrix with a copy of it in single precision, thus releasing half the memory required. Afterwards, each processor does the same for the next subdomain assigned to it. Single precision preconditioners for DDM were tested in [15]. It was there noted that loading in double precision matrix entries that were stored in single precision does not add computational cost to the process.

In our first test we compare the lumped preconditioned FETI-1 method to the P-FETI1 with K^{s^+} stored in single or double precision, in the 3D elasticity problem of Fig. 1. Table 2 shows the obtained results for a well-conditioned configuration of this problem ($E_A/E_B = 10^0$), while Table 3 reports on a more ill-conditioned case ($E_A/E_B = 10^6$). In these results, subdomain and coarse problems are solved with a skyline solver. The memory requirements reported in Tables 2 and 3 ignore the small percentages of memory required by DDM for node and element data bases and subdomain and interface mapping. These memory costs depend on implementation, they are practically the same for all the DDM that are here tested and usually constitute a very small percentage of total memory requirements. In both tables, we note that the storage of K^{s^+} in single precision does not increase the CPU time of P-FETI1, nor diminishes the maximum attainable solution accuracy. These two configurations require the same number of iterations and since single precision matrix entries can be loaded in memory in double precision without adding significant computational cost, the two configurations have the same CPU time. Finally, it is worth noting that the memory requirements of the P-FETI1 with K^{s^+} stored in single precision are more than 20% less than those of the lumped preconditioned FETI-1.

Tables 4 and 5 perform the same comparison for P-FETIDP configurations in the case of the shell problem of Fig. 3. Here, we leave out the lumped preconditioned configurations because they are known to have poor performance in fourth-order problems. Fig. 3 depicts a semi-cylindrical panel with a radius of 0.5, a length of 1.6 and a thickness of $t = 10^{-3}$ or $t = 10^{-4}$. Moreover, the Young modulus is 1×10^6 and the Poisson ratio 0.30. The panel is modeled with a structured mesh of 131×131 nodes and is discretized with triangular TRIC

Table 2

Comparison of the lumped preconditioned FETI-1 and P-FETI1 with K^s stored in single or double precision for the solution of the example of Fig. 1 ($E_A/E_B = 10^0$, decomposition P1, tolerance: 10^{-3})

Method	Memory (MB)				Iter.	Max. Acuracy
	$K_{ii}^{s^{-1}}$	K^{s^+}	Reorth.	Total		
FETI-1 (Lumped)	–	123	16	165	25	2.3E-10
P-FETI1	23	123	8	194	21	4.5E-11
P-FETI1(Single prec.)	23	62	8	132	21	4.4E-11

Table 3

Comparison of the lumped preconditioned FETI-1, P-FETI1 with K^s stored in single or double precision for the solution of the example of Fig. 1 ($E_A/E_B = 10^6$, decomposition P1, tolerance: 10^{-3})

Method	Memory (MB)				Iter.	Max. Acuracy
	$K_{ii}^{s^{-1}}$	K^{s^+}	Reorth.	Total		
FETI-1 (Lumped)	–	123	34	183	53	9.3E-7
P-FETI1	23	123	15	200	37	4.0E-7
P-FETI1(Single prec.)	23	62	15	139	37	3.7E-7

shell elements [16]. Furthermore, it is fixed on 16 nodes along its two linear edges as shown in Fig. 3. This model of 102,870 d.o.f. is decomposed in 130 subdomains (Fig. 4). The results of Tables 4 and 5 verify our conclusions from the previous example. The single precision storage practically does not affect the efficiency of P-FETIDP. Only in the more ill-conditioned case of $t = 10^{-4}$ we observe a small performance deterioration, while in both tables, memory gains are higher than 20%.

Coming back to the second-order problems, where the lumped preconditioned dual formulations usually need less memory than primal formulations, we will perform an approximate parametric analysis, in order to determine when the primal formulation with K^{s^+} stored in single precision will have less memory requirements than the lumped preconditioned dual formulation. Compared to the primal one, the lumped dual formulation does not need to store matrices $K_{ii}^{s^{-1}}$ and K_{ib}^{s}, while it will have a higher reorthogonalization memory overhead. On the contrary, in the primal formulation the single precision storage of K^{s^+} saves half the memory required for storing this matrix. Thus, if we require half the memory of K^{s^+} to be less than the memory of $K_{ii}^{s^{-1}}$, then the primal formulation with single precision storage of K^{s^+} will need less memory than the lumped preconditioned dual formulation. We will check when this condition holds in the following parametric model problem.

Suppose a 3D (2D) problem, discretized with 6-node brick (4-node quadrilateral) elements and a subdomain s of this problem with a cubic (square) shape. Suppose also that

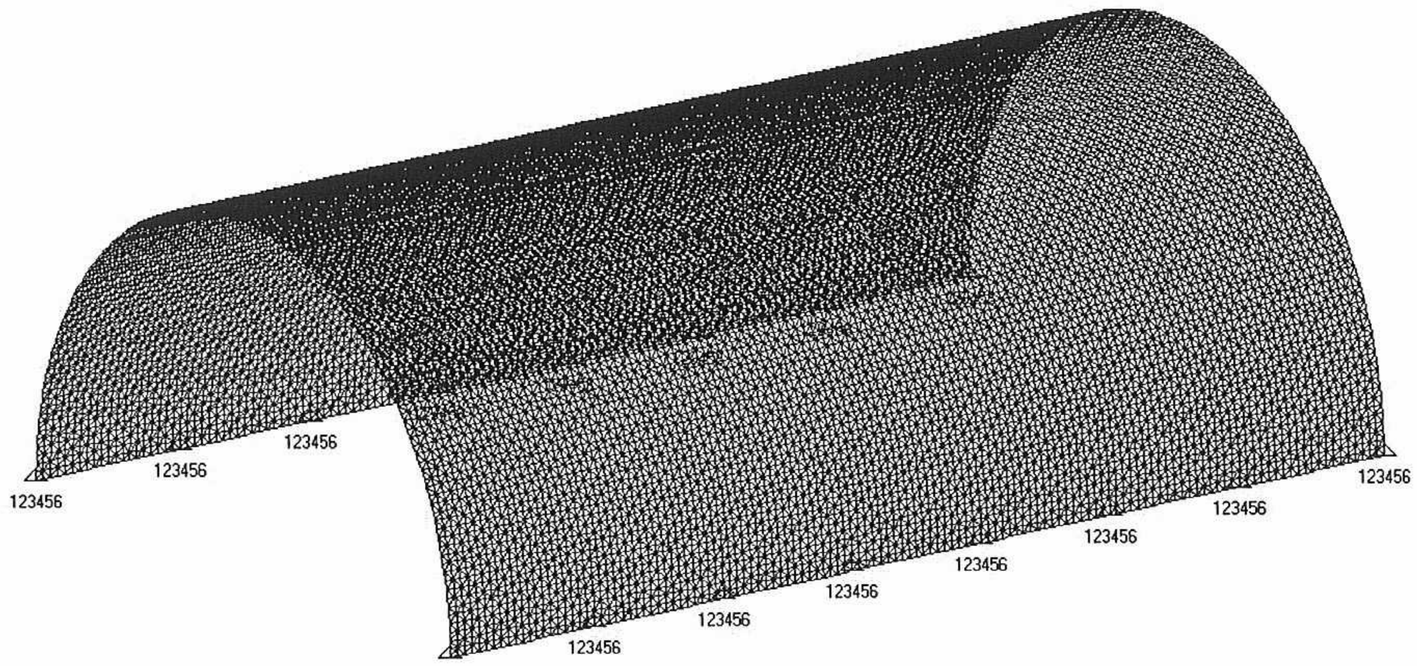


Figure 3. A semi-cylindrical panel, discretized with triangular shell elements

all surface (perimeter) nodes of the subdomain are interface nodes and that the number of nodes per edge of the subdomain is n, the number of d.o.f. per node is m, while the dimension of the problem is d. Table 6 gives an approximation of the size, the mean skyline width and the number of stored entries of matrices $K^{(s)}$ and $K_{ii}^{(s)}$ (assuming a skyline storage). Requiring one half of the memory needed to store $K^{(s)}$ to be higher than the memory required to store $K_{ii}^{(s)}$, gives:

$$\frac{1}{2}m^2 n^{2d-1} > m^2 (n-2)^{2d-1} \Leftrightarrow n < \frac{2^{\frac{2d-1}{}}}{2^{\frac{1}{2d-1}}-1} \Leftrightarrow \begin{cases} n < 8.4 & \text{for } d=2 \\ n < 15.3 & \text{for } d=3 \end{cases} \tag{62}$$

In the case a 2D elasticity problem with 2 d.o.f. per node, inequality (62) implies that the number of d.o.f. mn^d of s must be smaller than $2 \cdot 8.4^2 = 141$. In the case a 3D elasticity problem with 3 d.o.f. per node, the number of subdomain d.o.f. must be smaller than $3 \cdot 15.3^3 = 10745$. Given that in modern DDM, optimal decompositions tend to create subdomains with sizes around 1000 d.o.f., we conclude that usually, inequality (62) will not hold for 2D elasticity problems, while it will hold for 3D elasticity problems. It is thus

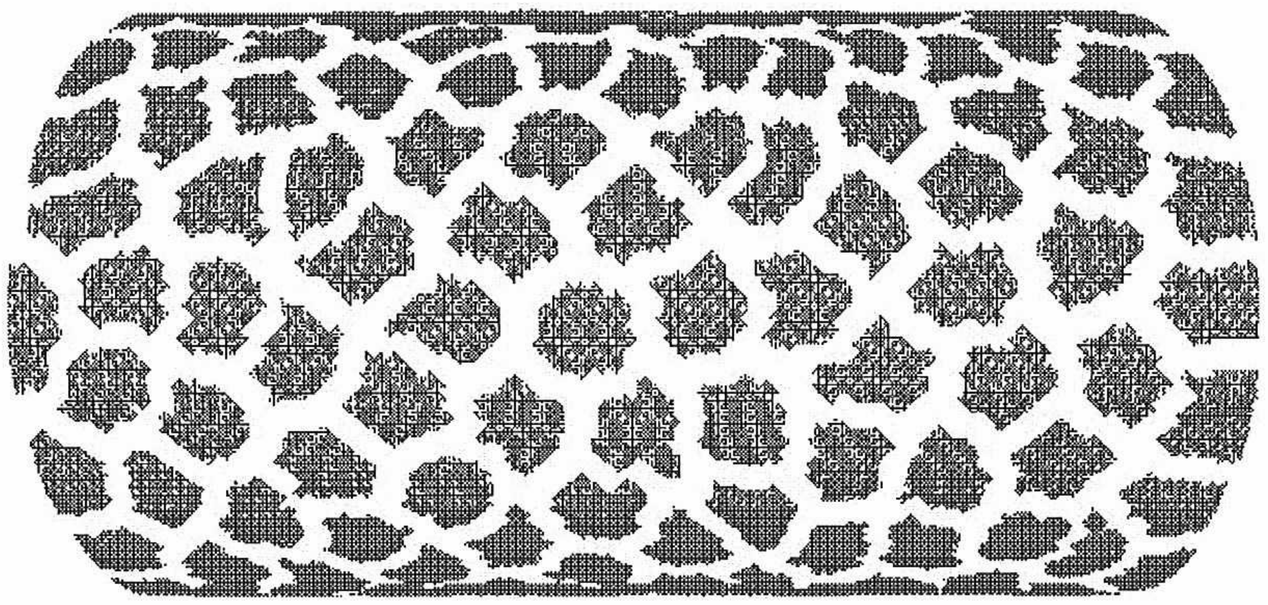


Figure 4. A decomposition of the semi-cylindrical panel in 130 subdomains – Top view

Table 4

Comparison of the P-FETIDP with K^{s^+} stored in single or double precision for the solution of the example of Fig. 2 ($t=10^{-3}$, tolerance: 10^{-3})

Method	Memory (MB)				Iter.	Max. Acuracy
	$K_{ii}^{s^{-1}}$	K^{s^+}	Reorth.	Total		
P-FETIDP	39	67	10	140	48	3.3E-8
P-FETIDP (Single prec.)	39	33	10	106	48	3.5E-8

Table 5

Comparison of the P-FETIDP with K^{s^+} stored in single or double precision for the solution of the example of Fig. 2 ($t=10^{-4}$, tolerance: 10^{-3})

Method	Memory (MB)				Iter.	Max. Acuracy
	$K_{ii}^{s^{-1}}$	K^{s^+}	Reorth.	Total		
P-FETIDP	39	67	30	160	139	3.4E-5
P-FETIDP (Single prec.)	39	33	31	127	141	5.2E-5

Table 6

Approximate estimate of the size, mean skyline width and number of stored entries of matrices $K^{(s)^+}$ and $K_{ii}^{(s)^{-1}}$ of a square (cubic) subdomain s (n: number of nodes per edge of s, m: number of d.o.f. per node, d: problem dimension)

Matrix	Matrix size	Skyline width	Number of entries
$K^{(s)^+}$	mn^d	mn^{d-1}	$m^2 n^{2d-1}$
$K_{ii}^{(s)^{-1}}$	$m(n-2)^d$	$m(n-2)^{d-1}$	$m^2(n-2)^{2d-1}$

safe to conclude that statistically the primal formulation with single precision storage of the factorized subdomain stiffness matrices will require less memory than the lumped preconditioned dual formulation in 3D second-order problems, while the contrary will hold in 2D second-order problems.

6 Summary and conclusions

The present paper studies the uses of the lumped preconditioner in modern DDM practice. In the beginning of the 90^s, the lumped preconditioner constituted the first choice for second-order problems. Later on however, when the advantages of using large number of subdomains per processor were discovered, the use of the lumped preconditioner became limited to the case where there was not enough memory available for solving large second-order problems.

Recently, a unified primal and dual DDM study [4] left an open issue regarding the lumped preconditioner. In particular, this paper introduced the primal alternatives of the Dirichlet preconditioned FETI methods, thus creating the open question of the existence of

a primal alternative of the lumped preconditioned methods. The present work thus presents these primal alternatives. However, the new methods do not seem to share all the advantages of the primal offspring of the Dirichlet preconditioner. In particular, they are less robust than the dual formulations and are slightly slower in well-conditioned problems. Their principal value lies in the fact that they add a new level of completion to the theory of the relations of primal and dual methods. The fact that a primal algorithm can be turned into an algorithm that uses dual operators and vectors appears to be new in DDM literature. On passing, we also note that the same transformations used in this paper can be used for the P-FETI and the BDD methods in order to transform them into algorithms that operate on dual quantities.

Furthermore, it is also worth noting that the analysis of sections 2-4, which leads to the primal alternatives of the lumped preconditioned FETI methods, also holds in the case where the discussed FETI methods are applied to problems of implicit dynamics. While the FETI-DP and its primal alternatives can be directly applied to implicit dynamic problems (see for instance [5]), the FETI-1 and its primal alternatives degenerate to the formulations deprived of coarse problems [17]. It is quite straightforward to prove that these FETI variants for Implicit Dynamics equipped with the lumped preconditioner, as well as the FETI variant introduced in [18] and revisited in [5], all have primal alternatives that are derived following the same analysis as in sections 2-4.

The second part of the presented work checks if indeed the lumped preconditoner is the best choice for minimum memory consumption in second-order problems. In particular, the fact that primal formulations move many operations of the dual formulations into the preconditioner is exploited in order to save important amounts of memory. Our parametric study shows that the lumped preconditioned dual methods will probably require less memory in 2D second-order problems, while they are overcome by primal formulations with subdomain stiffness matrices stored in single precision in 3D second-order problems. Presented results show 20-25% saving of memory in 3D elasticity and shell problems. In these problems, primal methods with single precision stored subdomain stiffness matrices in their preconditioner seem to be the methods of choice for memory saving. Furthermore, even though there can be no proof that they will be fully effective in highly ill-conditioned problems, in some considerably ill-conditioned tests performed in this paper they appear successful. More parts of the preconditioner could also be stored in single precision, trading off robustness for memory saving, but further memory saving would be considerably less important than that obtained from the subdomain stiffness matrices.

Acknowledgements

The authors would like to thank Dimos Charmpis for several useful discussions that contributed to this work. Furthermore, during this project the first author was supported from the program: “Support for mobility of Spanish and foreign professors and researchers” of the Spanish ministry of educacion and science.

References

[1] C. Farhat, F.-X. Roux, A method of finite element tearing and interconnecting and its parallel solution algorithm, Internat. J. Numer. Methods Engrg., 32, 1205-1227 (1991)

[2] J. Mandel, Balancing domain decomposition, Commun. Appl. Numer. Meth., 9, 233-241 (1993)

[3] A. Klawonn, O. B. Widlund, FETI and Neumann-Neumann iterative substructuring methods: connection and new results, Communications on pure and applied Mathematics, 54, 57-90 (2001)

[4] Y. Fragakis, M. Papadrakakis, The mosaic of high performance Domain Decomposition Methods for Structural Mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods, Comput. Methods Appl. Mech. Engrg., 192, 3799-3830 (2003)

[5] Y. Fragakis, M. Papadrakakis, The mosaic of high performance Domain Decomposition Methods for Structural Mechanics – Part II: Formulation enhancements, multiple right-hand sides and implicit dynamics, Comput. Methods Appl. Mech. Engrg., 193, 4611-4662 (2004)

[6] M. Bhardwaj, D. Day, C. Farhat, M. Lesoinne, K. Pierson, D. Rixen. Application of the FETI Method to ASCI Problems: Scalability results on one-thousand processors and discussion of highly heterogeneous problems, Internat. J. Numer. Methods Engrg., 47, 513-536 (2000)

[7] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, D. Rixen, FETI-DP : A dual-primal unified FETI method – part I: A faster alternative to the two-level FETI method, Internat. J. Numer. Methods Engrg., 50, 1523-1544 (2001)

[8] J. Mandel, C. R. Dohrmann, R. Tezaur, An algebraic theory for primal and dual substructuring methods by constraints, Appl. Numer. Math., 54, 167–193 (2005), Proceedings of the 6th IMACS International Symposium on Iterative Methods, Denver, CO, March 27-30, 2003

[9] C. R. Dohrmann, A preconditioner for substructuring based on constrained energy minimization, SIAM J. Sci. Comput., 25(1), 246–258 (2003)

[10] J. Mandel, Private communication, 2005

[11] C. Farhat, M. Lesoinne, K. Pierson, A scalable dual-primal domain decomposition method, Numer. Lin. Alg. Appl., 7(7-8), 687-714 (2000)

[12] Y. Fragakis, M. Papadrakakis, A unified framework for formulating domain decomposition methods in structural mechanics, Technical Report, National Technical University of Athens, Greece, March, 2002, Available from http://users.ntua.gr/fragayan

[13] J.-M. Cros, A preconditioner for the Schur complement domain decomposition method, Proceedings of the 14th International Conference on Domain Decomposition Methods (DD14), Cocoyoc, Mexico, January 6-11, 2002, Available from http://www.ddm.org

[14] J. Li, O. B. Widlund, FETI-DP, BDDC, and block Cholesky methods, Internat. J. Numer. Methods Engrg., 66, 250-271 (2006)

[15] D. C. Charmpis, M. Papadrakakis, Enhancing the performance of the FETI method with preconditioning techniques implemented on clusters of networked computers, Computational Mechanics, 30(1), 12-28 (2002)

[16] J. Argyris, L. Tenek, M. Papadrakakis, C. Apostolopoulou, Postbuckling performance of the TRIC natural mode triangular shell element for isotropic and laminated composite shells, Comput. Methods Appl. Mech. Engrg., 166, 211-231 (1998)

[17] C. Farhat, L. Crivelli, A transient FETI methodology for large-scale parallel implicit computations in Structural Mechanics, Internat. J. Numer. Methods Engrg., 37, 1945-1975 (1994)

[18] C. Farhat, P.-S. Chen, J. Mandel, A scalable Lagrange multiplier based Domain Decomposition Method for time-dependent problems, Int. J. Numer. Meth. Engng., 38, 3831-3853 (1995)

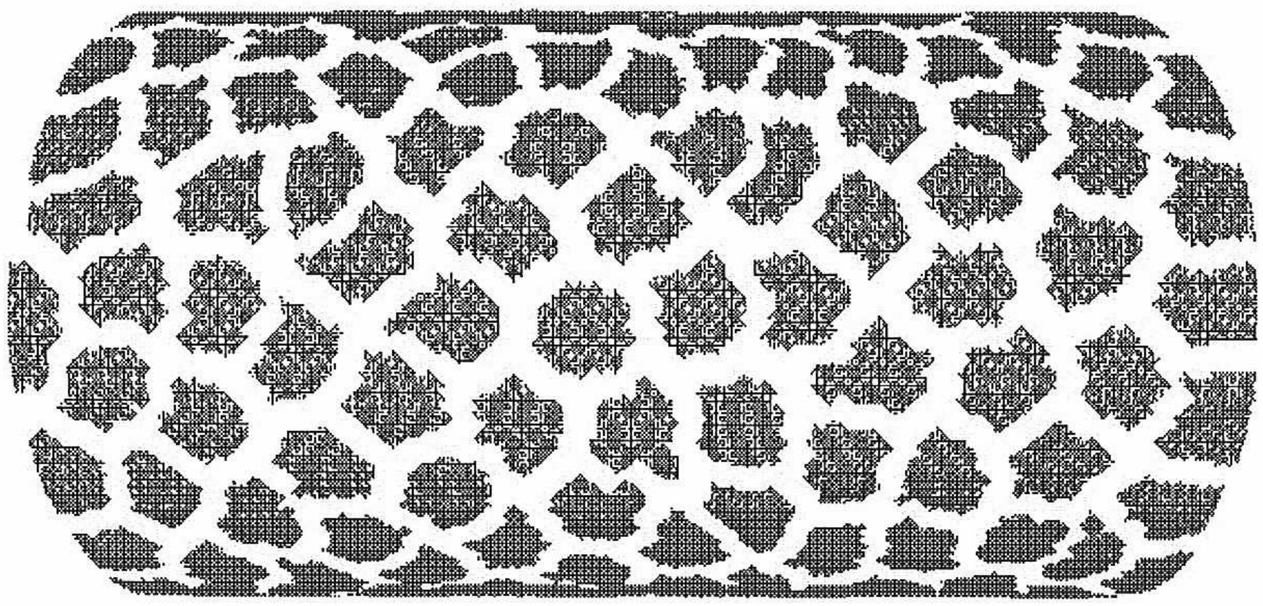


Figure 4. A decomposition of the semi-cylindrical panel in 130 subdomains – Top view

7 Concluding remarks

The final section of this paper aims at summing up what has been shown in this work and then studying the consequences of these findings. The main concept behind this work has been the fact that the definition of an operator that estimates subdomain displacements from subdomain forces can be used to build a primal and a dual DDM, that are strongly connected, provided that the operator satisfies two conditions. In particular, the first condition allows the primal formulation of the problem to be turned into a dual one. Then, a second condition guarantees that the two resulting DDM, the primal and the dual one, will have the same non-zero and non-unit eigenvalues. This suggests that the two formulations will probably have similar convergence properties and iteration counts. Some of the most popular DDM since the beginning of the 90's in structural and solid mechanics, have been inserted in this general framework, by proving that these two conditions hold for them. In particular, this has been proven when applying the methods FETI-1, FETI-2 and FETI-DP or their primal alternatives BDD and BBDC for the static analysis of structural problems. It is also worth noting that it would be simple to extend the proofs performed in this work to the applications of these methods for implicit dynamic structural problems The only things that change are the subdomain stiffness matrices that are substituted with the corresponding matrices of the implicit dynamic analysis and the zero energy mode projections that must be removed.

Hence, after summing up the general theoretical framework that has been set up here, we can investigate its consequences. A first and obvious but not unimportant consequence of this framework is that the setting of section 2 allows a very modular programming of primal and dual methods. If this general setting of primal and dual methods is programmed then by programming separately the estimate operator $\tilde{K}^{+}$ for some DDM, both primal and dual formulations are directly obtained. Furthermore, in the version of this formulations where internal d.o.f. of the subdomains are condensed, recent results show that the primal formulation, while it has similar performance to the dual one in well-conditioned problems, it is statistically faster and more robust in ill-conditioned ones [4,5]. Furthermore, with reference to the case where internal d.o.f. are not condensed, a recent study [9] proves that if the two conditions that have been set in the present paper hold, then the algorithm of the primal formulation can be made to operate on dual variables, instead of primal ones. The

Table 3

Condition nr. and nr. of iterations (Tolerance: 10^{-3}) of some DDM for the solution of the example of Fig. 3 ($t = 10^{-3}$)

Method	Condition. nr. of both formulations	Nr. of iterations	
		Primal formulation	Dual formulation
FETI-1	2.2E+6	131	135
FETI-2	5.4E+1	35	36
FETI-DP	2.0E+3	48	49

Table 4

Condition nr. and nr. of iterations (Tolerance: 10^{-3}) of some DDM for the solution of the example of Fig. 3 ($t = 10^{-4}$)

Method	Condition. nr. of both formulations	Nr. of iterations	
		Primal formulation	Dual formulation
FETI-1	1.9E+8	319	330
FETI-2	5.3E+2	91	–
FETI-DP	7.1E+4	139	145

main gain from this transformation is that the primal algorithm that would be excessively costly because it operates on the full displacement vector of the structure (thus practically inhibiting for instance the process of reorthogonalization in the PCG algorithm), is now converted to an algorithm, which operating on dual variables has comparable computational cost to the pure dual formulation. Hence, when internal d.o.f. are not condensed, the results of [9] show that the primal and dual formulations have comparable efficiency. However, in this case, the dual formulation turns out to be more robust and our tests show that in most problems it will probably be faster than the primal one.

Hence, in order to draw a general conclusion from comparing the primal and dual formulations, it is necessary to discuss when it is favourable to condense the internal d.o.f. of the subdomains. In modern DDM practice it has been noted that usually the condensation of internal d.o.f. leads to higher computational efficiency, while, avoiding the condensation can probably lead to less memory-consuming solutions in large-scale second-order problems. However, the results of [9] also suggest that the primal formulation requires less memory in fourth-order problems and in many second-order problems. Consequently, the general picture at the moment , with respect to computational cost, robustness and memory requirement, seems to be in favour of the primal formulation, at least for the majority of the cases. However, since the beginning of the 90's, the dual

methods have been implemented in many other areas beyond the pure static and dynamic analysis of structures. Therefore, for the primal formulation this work will have to be adapted when possible, or simply redone. Thus, the best choice today is probably to be armed with both options.

This paper comes to offer a small piece to the long series of works that have gradually led to today's understanding of the concept of duality in DDM for structural and solid mechanics. Before ending it, it is thus probably suitable to remind some of the most important steps in the efforts that have led here. A large step in this process was made in the beginning of the 90's, with the introduction of the FETI method, which was a dual method that quickly gained a lot of popularity. Since then, the major advances in the dual methods, like the introduction of the FETI-2 and FETI-DP methods were closely followed by similar advances in the area of the primal methods. While more and more advances were appearing that suggested that there were connections missing between primal and dual formulations, the first studies of these connections came forward. Today, the international research community of DDM has gone a long way since the introduction of the first dual methods and it can be said with a lot of certainty there is a lot more to come.

Acknowledgements

This paper owes a lot to the advances brought by the papers [6,8]. During this project the author was supported from the program: "Support for mobility of Spanish and foreign professors and researchers" of the Spanish ministry of educacion and science.

References

[1] C. Farhat, F.-X. Roux, A method of finite element tearing and interconnecting and its parallel solution algorithm, Internat. J. Numer. Methods Engrg., 32, 1205-1227 (1991)

[2] J. Mandel, Balancing domain decomposition, Commun. Appl. Numer. Meth., 9, 233-241 (1993)

[3] A. Klawonn, O. B. Widlund, FETI and Neumann-Neumann iterative substructuring methods: connection and new results, Communications on pure and applied Mathematics, 54, 57-90 (2001)

[4] Y. Fragakis, M. Papadrakakis, The mosaic of high performance Domain Decomposition Methods for Structural Mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods, Comput. Methods Appl. Mech. Engrg., 192, 3799-3830 (2003)

[5] Y. Fragakis, M. Papadrakakis, The mosaic of high performance Domain Decomposition Methods for Structural Mechanics – Part II: Formulation enhancements, multiple right-hand sides and implicit dynamics, Comput. Methods Appl. Mech. Engrg., 193, 4611-4662 (2004)

[6] J. Mandel, C. R. Dohrmann, R. Tezaur, An algebraic theory for primal and dual substructuring methods by constraints, Appl. Numer. Math., 54, 167–193 (2005), Proceedings of the 6th IMACS International Symposium on Iterative Methods, Denver, CO, March 27-30, 2003

[7] C. R. Dohrmann, A preconditioner for substructuring based on constrained energy minimization, SIAM J. Sci. Comput., 25(1), 246–258 (2003)

[8] J. Li, O. B. Widlund, FETI-DP, BDDC, and block Cholesky methods, Internat. J. Numer. Methods Engrg., 66, 250-271 (2006)

[9] Y. Fragakis, A study on the lumped preconditioner and memory requirements of FETI and related primal domain decomposition methods, Internat. J. Numer. Methods Engrg. (Submitted in 2006)

[10] J. Mandel, Private communication, 2005

[11] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, D. Rixen, FETI-DP : A dual-primal unified FETI method – part I: A faster alternative to the two-level FETI method, Internat. J. Numer. Methods Engrg., 50, 1523-1544 (2001)

[12] C. Farhat, M. Lesoinne, K. Pierson, A scalable dual-primal domain decomposition method, Numer. Lin. Alg. Appl., 7(7-8), 687-714 (2000)

[13] Y. Fragakis, M. Papadrakakis, A unified framework for formulating domain decomposition methods in structural mechanics, Technical Report, National Technical University of Athens, Greece, March, 2002, Available from http://users.ntua.gr/fragayan

[14] J.-M. Cros, A preconditioner for the Schur complement domain decomposition method, Proceedings of the 14th International Conference on Domain Decomposition Methods (DD14), Cocoyoc, Mexico, January 6-11, 2002, Available from http://www.ddm.org

[15] M. Bhardwaj, D. Day, C. Farhat, M. Lesoinne, K. Pierson, D. Rixen. Application of the FETI Method to ASCI Problems: Scalability results on one-thousand processors and discussion of highly heterogeneous problems, Internat. J. Numer. Methods Engrg., 47, 513-536 (2000)

[16] C. Farhat, J. Mandel, The two-level FETI method for static and dynamic plate problems – part I: an optimal iterative solver for biharmonic systems, Comput. Methods Appl. Mech. Engrg., 155, 129-152 (1998)

[17] C. Farhat, P.-S. Chen, J. Mandel, F. X. Roux, The two-level FETI method – part II: extension to shell problems, parallel implementation and performance results, Comput. Methods Appl. Mech. Engrg., 155, 153-180 (1998)

[18] J. Argyris, L. Tenek, M. Papadrakakis, C. Apostolopoulou, Postbuckling performance of the TRIC natural mode triangular shell element for isotropic and laminated composite shells, Comput. Methods Appl. Mech. Engrg., 166, 211-231 (1998)