A NUMERICAL STUDY OF FETI ALGORITHMS FOR MORTAR FINITE ELEMENT METHODS

DAN STEFANICA*

Abstract. The Finite Element Tearing and Interconnecting (FETI) method is an iterative substructuring method using Lagrange multipliers to enforce the continuity of the finite element solution across the subdomain interface. Mortar finite elements are nonconforming finite elements that allow for a geometrically nonconforming decomposition of the computational domain into subregions and, at the same time, for the optimal coupling of different variational approximations in different subregions.

We present a numerical study of FETI algorithms for elliptic self–adjoint equations discretized by mortar finite elements. Several preconditioners which have been successful for the case of conforming finite elements are considered. We compare the performance of our algorithms when applied to classical mortar elements and to a new family of biorthogonal mortar elements and discuss the differences between enforcing mortar conditions instead of continuity conditions for the case of matching nodes across the interface. Our experiments are carried out for both two and three dimensional problems, and include a study of the relative costs of applying different preconditioners for mortar elements.

Key words. FETI algorithms, mortar finite elements, Lagrange multipliers, domain decomposition

AMS(MOS) subject classifications. 65F10 65N30, 65N55

1. Introduction. The FETI method is an iterative substructuring method using Lagrange multipliers which is actively used in industrial–size parallel codes for solving difficult computational mechanics problems. This method was introduced by Farhat and Roux [25]; a detailed presentation is given in [26], a monograph by the same authors. Originally used to solve second order, self-adjoint elliptic equations, it has later been extended to many other problems, e.g., time-dependent problems [17], plate bending problems [18, 23, 42], heterogeneous elasticity problems with composite materials [44, 45], acoustic scattering and Helmholtz problems [21, 22, 27, 28], linear elasticity with inexact solvers [31], and Maxwell's equations [43, 50]. Another Lagrange multiplier based method, the dual–primal FETI method, has recently been introduced by Farhat et al. [19, 20] for two dimensional problems, and was extended to three dimensional problems by Klawonn and Widlund [33].

The FETI method is a nonoverlapping domain decomposition method and requires the partitioning of the computational domain Ω into nonoverlapping subdomains. It has been designed for conforming finite elements, and makes use of Lagrange multipliers to enforce pointwise continuity across the interface of the partition. After eliminating the subdomain variables, the dual problem, given in terms of Lagrange multipliers, is solved by a projected conjugate gradient (PCG) method. Once an accurate approximation for the Lagrange multipliers has been obtained, the values of the primal variables are obtained by solving a local problem for each subdomain; see Section 3 for more details.

It was shown experimentally in [24] that a certain projection operator used in the PCG solver plays a role similar to that of a coarse problem for other domain decomposition algorithms, and that certain variants of the FETI algorithm are numerically scalable with respect to both the subproblem size and the number of subdomains. Mandel and Tezaur later showed that for a FETI method which employs a Dirichlet preconditioner

* Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139 and Baruch College, City University of New York, 17 Lexington Avenue, New York NY 10017. Electronic mail address: dstefan@math.mit.edu URL: `http://www-math.mit.edu/~dstefan` Part of this work was carried out while the author was at the Courant Institute of Mathematical Sciences and with support in part by the National Science Foundation under Grant NSF-CCR-9732208, and in part by the U.S. Department of Energy under contract DE-FG02-92ER25127.

the condition number grows at most in proportion to $(1 + \log(H/h))^2$, if the decomposition of Ω does not have crosspoints, i.e., the points that belong to the closure of more than two subdomains, and as $C(1 + \log(H/h))^3$ in the general case; cf [41, 49]. Here, H is the subdomain diameter and h is the mesh size. Using a different preconditioner, Klawonn and Widlund obtained a FETI method which converges in fewer iterations than the classical FETI method. They proved an upper bound for the condition number of their method for elliptic problems with heterogeneous coefficients which is on the order of $(1 + \log(H/h))^2$; cf.[32].

Farhat and Rixen [44, 45] considered a Dirichlet preconditioner with a maximal number of pointwise continuity conditions at crosspoints, which results in a FETI algorithm with redundant Lagrange multipliers. It was shown in [32] that this algorithm is equivalent to using the preconditioner of Klawonn and Widlund and non–redundant multipliers for the FETI method.

In this paper, we study the numerical convergence properties of a family of FETI algorithms applied to mortar finite elements. Mortar finite elements are nonconforming finite element methods that allow for a geometrically nonconforming decomposition of the computational domain into subregions and, at the same time, for the optimal coupling of different variational approximations in different subregions. Here, optimality means that the global error is bounded by the sum of the local approximation errors on each subregion.

The importance of our study is related to the inherent advantages of mortar methods over the conforming finite elements. For example, the mesh generation is more flexible and can be made quite simple on individual subregions. This also makes it possible to move different parts of the mesh relative to each other, e.g., in a study of time dependent problems. The same feature is most valuable in optimal design studies, where the relative position of parts of the model is not fixed a priori. The mortar methods also allow for local refinement of finite element models in only certain subregions of the computational domain, and they are also well suited for parallel computing; cf. [29].

We have used geometrically nonconforming mortar finite elements. Three FETI algorithms with different preconditioners for the dual problem have been considered: the Dirichlet preconditioner of Farhat and Roux [25], the block–diagonal preconditioner of Lacour [34], and the new preconditioner of Klawonn and Widlund [32]. These algorithms have been implemented for both the classical mortar finite elements of Bernardi, Maday, and Patera [8], and for the new biorthogonal mortar elements of Wohlmuth [52, 54], in two and three dimensions. We note that a study of a FETI preconditioner for Maxwell's equations on non–matching grids has been completed by Rapetti and Toselli [43].

Our results show that the Dirichlet preconditioner does not perform well in the mortar case, since convergence is achieved only after hundreds or thousands of iterations. However, the new preconditioner performs satisfactory, i.e., the number of iterations required to achieve convergence and the condition number of the algorithms depend only weakly on the number of nodes in each subregion and is independent of the number of subregions. For each of the three preconditioners, using the biorthogonal mortars results into algorithms which require less computational effort and fewer iterations than those using the classical mortar finite elements.

We have also studied the extra computational effort, due to the complexity of the mortar conditions, required for the implementation of the FETI algorithm with new preconditioner. These costs might have been significant, in particular in the three dimensional case. We conclude that the improvement of the iteration count was enough to offset this extra cost.

In the conforming finite element case, the meshes across the interface match. Therefore, across the interface, mortar conditions may be enforced instead of continuity conditions. We have studied the differences between the FETI algorithms using both types of constraints, in terms of iteration counts and computational costs. We conclude that the new preconditioner for either continuity conditions or for biorthogonal mortars results in the best algorithms.

The rest of the paper is structured as follows. In the next section, we describe the mortar finite element method. In section 3, we present the classical FETI method and the Dirichlet preconditioner, and in section 4, we discuss the FETI algorithm for mortars with two different preconditioners. In sections 5 and 6, we present numerical comparisons of the performances of three different FETI algorithms for mortar finite elements, for two and three dimensional problems, respectively. In the last section, we discuss the differences between enforcing mortar conditions instead of continuity conditions for the case of matching nodes across the interface.

2. Mortar Finite Elements. The mortar finite element methods were first introduced by Bernardi, Maday, and Patera in [8], for low-order and spectral finite elements. A three dimensional version was developed by Ben Belgacem and Maday in [7], and was further analyzed for three dimensional spectral elements in [6]. Another family of biorthogonal mortar elements has recently been introduced by Wohlmuth [52, 54]. See also [46] for mortar *hp* finite elements, and [4, 11, 30], for mortar $H(\mathrm{curl})$ elements. Cai, Dryja, and Sarkis [12] have extended the mortar methods to overlapping decompositions.

Several domain decomposition methods for mortar finite elements have been shown to perform similarly to the case of conforming finite elements; cf. [3, 14] for iterative substructuring methods, [15, 37, 38] for Neumann-Neumann algorithms, and [36, 48] for the FETI method. For other studies of preconditioners for the mortar method, see [13] for a hierarchical basis preconditioner and [1, 2], for iterative substructuring preconditioners. Multigrid methods have also been used to solve mortar problems; cf. [9, 10, 51, 53].

2.1. 2–D Low Order Mortar Finite Elements. To introduce a mortar finite element space, the computational domain Ω is decomposed using a nonoverlapping partition $\{\Omega_i\}_{i=1:N}$, consisting of polygons,

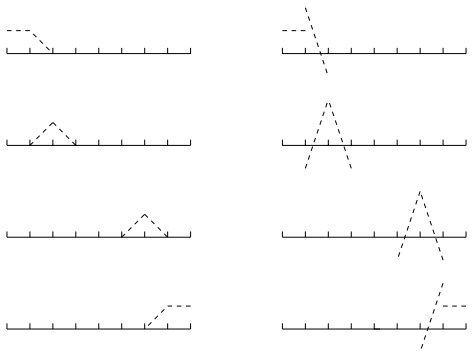
$$\overline{\Omega} = \bigcup_{i=1}^{N} \overline{\Omega}_i, \qquad \Omega_j \bigcap \Omega_k = \emptyset \quad \text{if} \quad 1 \leq j \neq k \leq N.$$

Let $\partial\Omega_D$ be the part of $\partial\Omega$ where Dirichlet conditions are imposed. If an edge of a polygon intersects $\partial\Omega_D$, we require that the entire edge belongs to $\partial\Omega_D$. The partition is said to be geometrically conforming if the intersection between the closure of any two subregions is either empty, a vertex, or an entire edge, and it is geometrically nonconforming otherwise.

The interface between the subregions $\{\Omega_i\}_{i=1:N}$, denoted by Γ, is defined as the closure of the union of the parts of $\{\partial\Omega_i\}_{i=\overline{1:N}}$ that are interior to Ω. Alternatively, Γ can be defined as the set of points that belong to the boundaries of at least two subregions.

We denote by V^h the space of low order mortar finite elements, and by $V^h(S)$ the restriction of V^h to a set S. For every subregion Ω_i, $V^h(\Omega_i)$ is a conforming element space. We do not require pointwise continuity across Γ. Instead, we choose a set of open edges $(\gamma_l)_{l=\overline{1:L}}$ of the subregions $\{\Omega_i\}_{i=1:N}$, called nonmortars, which form a disjoint

FIG. 1. *Test functions. Left: classical mortars; Right: new mortars*



partition of the interface,

$$\Gamma = \bigcup_{l=1}^{L} \overline{\gamma}_l, \quad \gamma_m \cap \gamma_n = \emptyset \ \text{ if } \ 1 \le m \ne n \le L.$$

We impose weak continuity conditions for the mortar finite element functions, in the sense that the jump of a mortar function across each nonmortar is required to be orthogonal to a space of test functions. Therefore, the mortar elements are nonconforming finite elements.

We note that a nonmortar partition of the interface is always possible; cf. Stefanica [47]. The partition is not unique, but any choice can be treated the same from a theoretical point of view.

The edges of $\{\Omega_i\}_{i=1:N}$ which are part of Γ and were not chosen to be nonmortars are called mortars and are denoted by $\{\zeta_m\}_{m=1}^{M}$. It is clear that the mortars also cover the interface.

Let γ be an arbitrary nonmortar side. It belongs to exactly one subregion, denoted by Ω_γ. Let $V^h(\gamma)$ be the restriction of $V^h(\Omega_\gamma)$ to $\overline{\gamma}$ and let $\Psi^h(\gamma)$ be a subspace of $V^h(\gamma)$ which is of codimension two. Thus, when the space $V^h(\gamma)$ is piecewise linear, $\Psi^h(\gamma)$ is given by the restriction of $V^h(\Omega_\gamma)$ to $\overline{\gamma}$, subject to the constraints that these continuous, piecewise linear functions are constant in the first and last mesh intervals of $\overline{\gamma}$; cf. Figure 1.

The mortar finite element space V^h is defined as follows: Any mortar function $v \in V^h$, vanishes at all the nodes on $\partial\Omega_D$. The restriction of v to any Ω_i is a P_1 or a Q_1 finite element function. Let $\Gamma(\gamma)$ be the union of the parts of the mortars that coincides geometrically with $\overline{\gamma}$. Let v_γ and $v_{\Gamma(\gamma)}$ be the restriction of v to γ and $\Gamma(\gamma)$, respectively. The values of v on the nonmortar γ are given by the mortar conditions

$$\int_\gamma \left(v_\gamma - v_{\Gamma(\gamma)}\right) \ \psi \ ds \ = \ 0, \quad \forall \ \psi \in \Psi^h(\gamma). \tag{1}$$

We note that the interior nodes of the nonmortar sides are not associated with genuine degrees of freedom in the finite element space V^h, while the values of v at the end points of $\overline{\gamma}$ are genuine degrees of freedom.

To emphasize this aspect, we present here the matrix formulation of the mortar conditions, which will be further used in section 4.

Let $\bar{v}_\gamma$ be the vector of the interior nodal values of v on γ. For simplicity, we assume that the mesh is uniform on γ, of mesh size h. Let $\bar{v}_{\Gamma(\gamma)}$ be the vector of the values of u at the end points of γ and at all the nodes on the edges opposite γ, such that the intersection of γ and the support of the corresponding nodal basis functions is not empty. Then $\bar{v}_\gamma$ is uniquely determined by $\bar{v}_{\Gamma(\gamma)}$; the matrix formulation of the mortar conditions (1) is

$$M_\gamma \bar{v}_\gamma - N_\gamma \bar{v}_{\Gamma(\gamma)} = 0, \tag{2}$$

or, solving for $\bar{v}_\gamma$, $\bar{v}_\gamma = P_\gamma \bar{v}_{\Gamma(\gamma)}$, with $P_\gamma = M_\gamma^{-1} N_\gamma$.

We note that N_γ is a banded matrix with a bandwidth of similar size for both the classical and the new mortars. For the classical mortar method, M_γ is a tridiagonal matrix and the mortar projection matrix P_γ is a full matrix. The projection of a nodal basis function from the mortar side results in a function with support equal to γ. The nodal values of this function decay exponentially to 0 at the end points of γ, away from the nodes on γ opposite the support of the nodal basis function from the mortar side.

Since $V^h(\Omega_i) \subset H^1(\Omega_i)$, we know that $v_\gamma \in H^{1/2}(\gamma)$. Thus, the test functions space $\Psi^h(\gamma)$ may be embedded in the dual space of $H^{1/2}(\gamma)$ with respect to the L^2 inner product, and therefore $\Psi^h(\gamma) \subset H^{-1/2}(\gamma)$.

Based on this observation, a space of discontinuous piecewise linear test functions $\Psi^h_{new}(\gamma)$ for low order mortars has been developed by Wohlmuth [52].

There, the test function associated to the first interior node on γ is the constant 1 on the first mesh interval, decreases linearly from 2 to -1 on the second mesh interval, and vanishes everywhere else. A similar test function is introduced for the last interior node on γ. The test function for any other node on γ has the support on the two mesh intervals having the node as an end point; it increases linearly from -1 to 2 on the first interval and decreases from 2 to -1 on the second; cf. Figure 1.

The new mortar space has similar approximation properties as the classical mortar space; cf. [52]. A major advantage of the new mortar finite element space is that the mortar projection can be represented by a banded matrix, as opposed to the classical mortar finite element method, where the mortar projection matrix is, in general, a full matrix. More precisely, for the new mortar method, $M_\gamma = hI$ is a diagonal matrix and $P_\gamma = N_\gamma/h$ is banded. Therefore, the mortar projection of a nodal basis function on the mortar side vanishes outside the mesh intervals on the nonmortar which intersect its support.

2.2. The Three Dimensional Case. For three dimensional problems, the mortars and nonmortars are open faces of the subregions which form the nonconforming decomposition of the computational domain Ω.

To introduce the mortar finite element space, we follow the outline from the previous section. Let $\{\Omega_i\}_{i=1:N}$ be a nonoverlapping polyhedral partition of Ω. If a face or an edge of a polyhedron intersects $\partial\Omega_D$ at an interior point, then the entire face or edge is assumed to belong to $\partial\Omega_D$. The partition is said to be geometrically conforming if the intersection between the closures of any two subregions is either empty, a vertex, an entire edge, or an entire face, and it is nonconforming otherwise.

The nonmortars $\{\mathcal{F}_l\}_{l=1}^L$ are faces of the subregions which form a disjoint partition of the interface Γ. The faces of $\{\Omega_i\}_{i=1:N}$ that are part of Γ and were not chosen to be nonmortars are called mortars.

We now describe the test functions associated to an arbitrary nonmortar face $\mathcal{F}$. Let $\Gamma(\mathcal{F})$ be the union of parts of mortar faces opposite $\mathcal{F}$. The test function space $\Psi^h(\mathcal{F})$ is a subset of $V^h(\mathcal{F})$, the restriction of V^h to $\mathcal{F}$, such that the value of a test function at

a node on $\partial\mathcal{F}_l$ is a convex combination of its values at the neighboring interior nodes of $\mathcal{F}_l$. If $V^h(\mathcal{F})$ is a P_1 or a Q_1 space, then the dimension of $\Psi^h(\mathcal{F})$ is equal to the number of interior nodes of $\mathcal{F}$.

The mortar finite element space V^h consists of functions v which vanish at all the nodal points of $\partial\Omega_D$. Its restriction to any Ω_i is a P_1 or a Q_1 finite element function. The values of a mortar function $v \in V^h$ on any nonmortar face $\mathcal{F}$ are given by the mortar conditions

$$\int_{\mathcal{F}} \left(v_{\mathcal{F}} - v_{\Gamma(\mathcal{F})}\right)\ \psi\ ds\ =\ 0, \quad \forall\ \psi \in \Psi^h(\mathcal{F}).$$

We note that the values of v at all the boundary nodes of the nonmortars are genuine degrees of freedom.

A version of the new mortars for the 3–D case, based on biorthogonal test functions such as those described in section 2.1 has been developed by Wohlmuth. For details, we refer the reader to [54].

3. The Classical FETI Algorithm. In this section, we review the original FETI method of Farhat and Roux for elliptic problems discretized by *conforming finite elements*. To simplify our presentation, we only discuss the Poisson equation with mixed Neumann–Dirichlet boundary conditions. The extension of the algorithm to the case of other self-adjoint elliptic equations is straightforward.

Let $\partial\Omega = \partial\Omega_N \cup \partial\Omega_D$, where $\partial\Omega_N$ and $\partial\Omega_D$ are the parts of the boundary where Neumann and Dirichlet boundary conditions are imposed, respectively. For unique solvability, we require that $\partial\Omega_D$ has positive Lebesgue measure. Let $f \in L^2(\Omega)$. We look for a solution $u \in H^1(\Omega)$ of the mixed boundary value problem

$$\left\{\begin{array}{rcll} -\Delta u & = & f & \text{on } \Omega \\ u & = & 0 & \text{on } \partial\Omega_D \\ \frac{\partial u}{\partial n} & = & 0 & \text{on } \partial\Omega_N. \end{array}\right. \tag{3}$$

On Ω, we consider P_1 or Q_1 finite elements with mesh size h. The finite element mesh is partitioned along mesh lines into N non–overlapping subdomains $\Omega_i \subset \Omega$, $i = 1 : N$. Since the finite element mesh is conforming, the boundary nodes of the subdomains match across the interface. A subdomain Ω_i is said to be floating if $\partial\Omega_i \cap \partial\Omega_D = \emptyset$, and non–floating otherwise.

As in other substructuring methods, the first step of the FETI method consists in eliminating the interior subdomain variables, which results in a Schur complement formulation of our problem. Let $S^{(i)}$ be the Schur complement matrix of Ω_i and let f_i be the contribution of Ω_i to the load vectors. Let $S = diag_{i=1}^N S^{(i)}$ be a block–diagonal matrix, and let f be the vector $[f_1, \ldots, f_N]$. We denote by u_i the vector of nodal values on $\partial\Omega_i$ and by u the vector $[u_1, \ldots, u_N]$.

If Ω_i is a floating subdomain, then $S^{(i)}$ is a singular matrix and its kernel is generated by a vector Z_i which is equal to 1 at the nodes of $\partial\Omega_i$ and vanishes at all the other interface nodes. Let Z consisting of all the column vectors Z_i. Then

$$\text{Ker}S\ =\ \text{Range}Z. \tag{4}$$

Let B be the matrix of constraints which measures the jump of a given vector u across the interface; B will also be referred to as the Lagrange multiplier matrix. Each row of the matrix B is associated to two matching nodes across the interface, and has values 1 and -1, respectively at the two nodes, and zero entries everywhere else. A

finite element function with corresponding vector values u is continuous if and only if $Bu = 0$.

For a method without redundant constraints and multipliers, the number of pointwise continuity conditions required at crosspoints, i.e., the points that belong to the closure of more than two subdomains, and therefore the number of corresponding rows in the matrix B, is one less then the number of the subdomains meeting at the crosspoint. There exist several different ways of choosing which conditions to enforce at a crosspoint, all of them resulting in algorithms with similar properties.

An alternative suggested in [44, 45] is to connect all the degrees of freedom at the crosspoints by Lagrange multipliers and use a special scaling, resulting in a method with redundant multipliers; see section 4.3 for further details.

Let W_i be the space of the degrees of freedom associated with $\partial\Omega_i \setminus \partial\Omega_D$, and let W be the direct sum of all spaces W_i. If $U = \mathrm{Range}B$ is the space of the Lagrange multipliers, then

$$S : W \to W, \quad B : W \to U.$$

By introducing Lagrange multipliers λ for the constraint $Bu = 0$, we obtain a saddle point Schur formulation of (3),

$$\left\{ \begin{array}{lcll} Su & + & B^t\lambda & = & f \\ Bu & & & = & 0, \end{array} \right. \tag{5}$$

where B^t denotes the transpose of B.

3.1. Algebraic Formulation. In the FETI method, the primal variable u is eliminated from (5) and the resulting equation for the dual variable λ is solved by a projected conjugate gradient method.

We note that S is singular if there exist at least one floating subdomains among the subdomains Ω_i, $i = 1 : N$. Let $S^\dagger$ be the pseudoinverse of S, i.e., for any $b \perp \mathrm{Ker}S$, $S^\dagger b$ is the unique solution of $Sx = b$ such that $S^\dagger b \in \mathrm{Range}S$. The first equation in (5) is solvable if and only if

$$f - B^t\lambda \ \perp \ \mathrm{Ker}S. \tag{6}$$

If (6) is satisfied, then

$$u \ = \ S^\dagger(f - B^t\lambda) + Z\alpha, \tag{7}$$

where $Z\alpha$ is an element of $\mathrm{Ker}S = \mathrm{Range}Z$ to be determined.

Let $G = BZ$. Substituting (7) into the second equation in (5), it follows that

$$BS^\dagger B^t\lambda \ = \ BS^\dagger f + G\alpha. \tag{8}$$

An important role in the FETI algorithm is played by V, a subset of U defined by $V = \mathrm{Ker}G^t$. In other words,

$$V \ = \ \mathrm{Ker}G^t \ \perp \ \mathrm{Range}G \ = \ B\mathrm{Range}Z \ = \ B\mathrm{Ker}S. \tag{9}$$

Let $P \ = \ I - G(G^tG)^{-1}G^t$ be the orthogonal projection onto V. It is easy to see that G^tG is non–singular, by using the fact that $\mathrm{Ker}B \cap \mathrm{Range}Z = \mathrm{Ker}B \cap \mathrm{Ker}S = \emptyset$. Since $P(G\alpha) = 0$, if P is applied to (8), it results that

$$PBS^\dagger B^t\lambda \ = \ PBS^\dagger f. \tag{10}$$

We now return to the necessary condition (6). From (4), we obtain that (6) is equivalent to $f - B^t\lambda \perp \text{Range}Z$, which leads to $Z^t(f - B^t\lambda) = 0$ and therefore to

$$G^t\lambda = Z^t f. \tag{11}$$

Let $F = BS^\dagger B^t$, $d = BS^\dagger f$, and $e = Z^t f$. We concluded that we have to solve the dual problem (10) for λ, subject to the constraint (11); with the new notations,

$$PF\lambda = Pd; \tag{12}$$

$$G^t\lambda = e. \tag{13}$$

We note that, from (8), it follows that $\alpha = (G^tG)^{-1}G^t(F\lambda - d)$. Therefore, after an approximate solution for λ is found, the primal variable u is obtained from (7) by solving a Neumann or a mixed boundary problem on each floating and nonfloating subdomain, respectively, corresponding to a vector multiplication by $S^\dagger$.

The main part of the FETI algorithm consists of solving (12) for the dual variable λ, which is done by a projected conjugate gradient (PCG) method. Since λ must also satisfy the constraint (13) let

$$\lambda_0 = G(G^tG)^{-1}e \tag{14}$$

be the initial approximation. Then $G^t\lambda_0 = e$ and $\lambda - \lambda_0 \in \text{Ker}G^t = V$. If all the increments $\lambda_k - \lambda_{k-1}$, i.e., the search directions, are in V, then (13) will be satisfied.

One possible preconditioner for (12) is of the form PM, where

$$M = BSB^t.$$

When a vector multiplication by M is performed, N independent Dirichlet problems have to be solved in each iteration step. Therefore, M is known as the Dirichlet preconditioner. We note that the Schur complement matrix S is never computed explicitly, since only the action of S on a vector is needed.

Mandel and Tezaur [41] have shown that the condition number of this FETI method has a condition number which grows polylogarithmically with the number of nodes in each subdomain,

$$\kappa(PMPF) \leq C\left(1 + \log\frac{H}{h}\right)^3,$$

where C is a positive constant independent of h, H. If there are no crosspoints in the partition of Ω, then the bound improves to $C(1 + \log(H/h))^2$.

We conclude this section by presenting the PCG algorithm:

Projected Preconditioned Conjugate Gradient Iteration (PCG)

$\lambda_0 = G(G^tG)^{-1}e$, $r_0 = Pd - PF\lambda_0$, $n = 1$

while $(Mr_{n-1}, r_{n-1}) \geq tol$

$w_{n-1} = Pr_{n-1}$

$z_{n-1} = Mw_{n-1}$

$y_{n-1} = Pz_{n-1}$

$\beta_n = (y_{n-1}, r_{n-1})/(y_{n-2}, r_{n-2})$ $\qquad$ $(\beta_1 = 0)$

$p_n = y_{n-1} + \beta_n p_{n-1}$ $\qquad$ $(p_1 = y_0)$

$\alpha_n = (y_{n-1}, r_{n-1})/(Fp_n, p_n)$

$\lambda_n = \lambda_{n-1} + \alpha_n p_n$
$r_n = r_{n-1} - \alpha_n PFp_n$
$n = n + 1$
end

In contrast to the classical conjugate gradient algorithm, in each iteration step of the PCG algorithm, the residual and the search directions are projected onto the space V, i.e., $w_{n-1} = Pr_{n-1}$ and $y_{n-1} = Pz_{n-1}$. This projection step plays the role of a coarse problem which is solved in each iteration, and is the reason why the FETI method is numerically scalable, even though it lacks an explicit coarse space construction. We note that $r_{n-1} \in V$ at every step. Therefore, it follows that $w_{n-1} = r_{n-1}$ and thus only one projection onto V is required per iteration step. This observation is particularly important for some of the algorithms suggested in [32].

4. The FETI Algorithm for Mortars. As we have seen in Section 3, in the classical FETI algorithm the computational domain Ω is partitioned into nonoverlapping subregions, multiple degrees of freedom are introduced for the matching nodes across the interface, and pointwise continuity across the interface is enforced by a Lagrange multiplier matrix B. This methodology is very similar to that used in [5], where a saddle point formulation for the mortar finite element method has been introduced.

In fact, the FETI method can be applied without any algorithmic changes for a mortar finite element discretization of Ω, using the nonoverlapping partition $\{\Omega_i\}_{i=1:N}$ considered in Section 2. We recall that this partition may be geometrically nonconforming and the nodes across the interface do not necessarily match. To keep the presentation clear, we assume that each subregion Ω_i has a diameter of order H and that its triangulation has a mesh size of order h. The matrix S is again a block–diagonal matrix $diag_{i=1}^{N} S^{(i)}$, where the local Schur complement matrices $S^{(i)}$ are obtained from the finite element discretizations on individual subregions. We have to solve the problem

$$\begin{cases} Su + B^t\lambda = f; \\ Bu = 0, \end{cases}$$

where the matrix B enforces mortar conditions across the interface. The dual problem is obtained as in Section 3.1. It results in solving

$$PF\lambda = Pd, \tag{15}$$

with a PCG method, with the initial approximation λ_0 given by (14) and with all the search directions in V.

The price we pay for the inherent flexibility of the mortar finite elements is due to the fact that the matrix B is more complicated in the mortar case, compared to that of the classical FETI method with conforming finite elements. The matrix B has one block, B_γ, for each nonmortar side γ. We adopt the matrix formulation of the mortar conditions from section 2.1. Let M_γ and N_γ be the matrices which multiply the nonmortar and mortar nodal values in the mortar conditions across γ, respectively. Then B_γ consists of the columns of M_γ and $-N_\gamma$ for the nodes of γ and those on the mortars opposite γ, and has zero columns corresponding to all the other nodes.

We note that the mortar conditions are all associated with the interior nodes on the nonmortar sides. Therefore, the problem of choosing the crosspoints constraints does not arise in the mortar case.

In our numerical experiments, we have implemented three different preconditioners suggested in the FETI literature for the dual problem (15). In Sections 4.1–4.3, we present each of them briefly.

4.1. The Dirichlet Preconditioner. In [25], Farhat and Roux introduced the Dirichlet preconditioner for the FETI method,

$$PM = PBSB^t. \tag{16}$$

This preconditioner was shown to perform well for conforming finite elements; see, e.g., [25] for numerical results and [41] for condition number estimates.

4.2. A Block-diagonal Preconditioner. In [34, 35], Lacour suggested another preconditioner designed specifically for a mortar version of the FETI algorithm, and without a counterpart in the conforming case.

Let $diagB_\gamma B_\gamma^t$ be the block–diagonal matrix which has a block $B_\gamma B_\gamma^t$ of size equal the number of interior nodes on γ for each nonmortar γ. We note that $diagB_\gamma B_\gamma^t$ is the block–diagonal part of the matrix BB^t. In the three dimensional case, each block corresponds to a nonmortar face $\mathcal{F}$, and the block–diagonal matrix is $diagB_\mathcal{F} B_\mathcal{F}^t$. To simplify our presentation, we will use the same notation, $diagB_\gamma B_\gamma^t$, for the three dimensional block–diagonal matrix.

The preconditioner $P\overline{M}$ is defined as follows:

$$P\overline{M} \;=\; P(diagB_\gamma B_\gamma^t)^{-1} BSB^t (diagB_\gamma B_\gamma^t)^{-1}. \tag{17}$$

4.3. A New Preconditioner. In [32], Klawonn and Widlund studied a FETI method for elliptic problems with heterogeneous coefficients, discretized by conforming finite elements, and designed a new preconditioner for this type of problems. They used this preconditioner to show the connection between FETI methods and Neumann-Neumann methods, in particular the balancing method of Mandel and Brezina [39, 40].

In the case of no coefficient jump, as in our Poisson problem, the new preconditioner has the form

$$P\widehat{M} \;=\; P(BB^t)^{-1} BSB^t (BB^t)^{-1}. \tag{18}$$

Klawonn and Widlund established the following upper bound for the condition number of their FETI algorithms, which is valid for all cases, including when the partition contains crosspoints:

$$\kappa(P\widehat{M}PF) \;\leq\; C\left(1 + \log\frac{H}{h}\right)^2.$$

In the same paper, it is proven that the preconditioner $\widehat{M}$ with a minimal number of pointwise continuity conditions at the crosspoints, and therefore of Lagrange multipliers, results in a similar algorithm as the FETI method with redundant Lagrange multipliers of Farhat and Rixen [44, 45]. Since the Lagrange multipliers in the mortar case are not associated with the vertices of the subregions, a FETI algorithm with redundant multipliers cannot be implemented for mortars.

To use the new preconditioner of Klawonn and Widlund for the FETI method with mortars, we must show that the matrix BB^t is non–singular in the mortar case.

The number of columns of B is equal to the number of nodes from W, while the number of rows of B is equal to the number of Lagrange multipliers. Since each Lagrange multiplier is associated to an interior node on a nonmortar side, it results that B has fewer rows than columns. Therefore, if we show that the rank of B is equal to its number of rows, we may conclude that BB^t is non–singular. We consider the minor of B consisting of the columns corresponding to the interior nodes of the nonmortars. The

resulting block–diagonal square matrix $diag M_\gamma$ which is non–singular, since each block M_γ is a diagonally dominant matrix for the classical mortar elements, and the identity matrix for the new mortar elements.

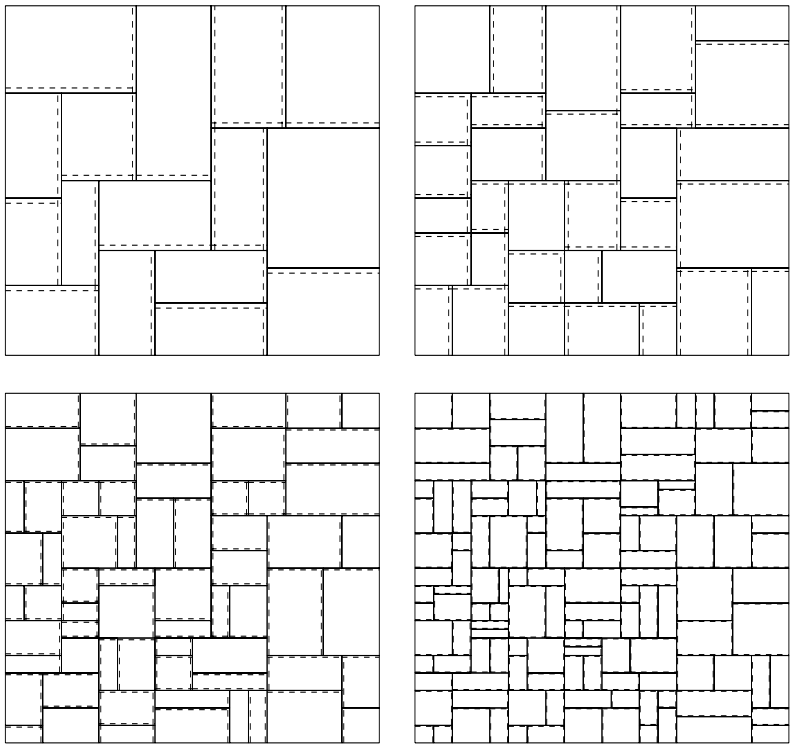
5. Numerical Results for Two Dimensional Problems. In this section, we present numerical results for the FETI method for a mortar finite element discretization of a two–dimensional problem. We have tested each of the three preconditioners of Sections 4.1–4.3, on nonconforming discretizations of the computational domain.

Our interests were three–fold:

- to compare the convergence performances of the different FETI preconditioners for mortar methods, based on iteration counts and estimates for the condition numbers;
- to apply the FETI algorithms for the new mortar finite elements, and compare the iteration counts and the flop counts to those obtained for the classical mortar finite elements;
- to analyze the extra computational effort, due to the complexity of the mortar conditions, required for the implementation of the FETI algorithm with the new preconditioner.

As the model problem in 2–D, we chose the Poisson equation on the unit square $\Omega = [0,1]^2$ with zero Dirichlet boundary conditions. The right hand side function f in (3) was selected such that the exact solution of the problem is known.

FIG. 2. *Geometrically nonconforming partitions of* Ω. *Upper left: 16 subdomains, Upper right: 32 subdomains, Lower left: 64 subdomains, Lower right: 128 subdomains.*



The computational domain Ω was partitioned into 16, 32, 64, and 128 geometrically

nonconforming rectangular subregions, respectively; see Figure 2. On each subregion, we considered Q_1 elements of mesh size h, and, to make the comparisons easier, all the subregions had diameters of the same order, H. For each partition, the number of nodes on each edge, H/h, was taken to be, *on average*, 4, 8, 16, and 32, respectively, for different sets of experiments. Across the partition interface Γ the meshes did not necessarily match. A saddle point formulation of the problem was used, and mortar conditions were enforced across Γ.

We report the iteration counts, the condition number estimates, and the flop counts of the algorithms. The PCG iteration was stopped when the residual norm had decreased by a factor of 10^{-6}. All the experiments were carried out in MATLAB.

We now present some implementation details. We did not compute the Schur complements explicitly, nor their pseudoinverses, but only the stiffness matrices for each subdomain. To multiply a vector by a Schur complement matrix, we solved, in each subregion, a Poisson problem with Dirichlet boundary conditions. To multiply a vector by $S^\dagger$, we solved a Poisson problem with mixed boundary conditions in each non–floating subregion, and with Neumann boundary conditions in each floating subregion; see, e.g., [16]. We stored only the interior–boundary and boundary–boundary blocks of the local stiffness matrix and the Cholesky factor of the interior–interior block, which is symmetric and positive definite. To have a uniquely solvable problem on the floating subregions, we required that the solution of the local Neumann problem be orthogonal to Ker$S$, i.e., to the constant functions on the subregion. A simple way of enforcing this orthogonality condition was by adding a Lagrange multiplier, and storing the LU components of the extended stiffness matrix.

5.1. Convergence properties of the FETI algorithms. We now turn to the main part of this section, a discussion of the performance of the FETI algorithms for mortar finite elements with the new preconditioner $\widehat{M}$, (18), the preconditioner $\overline{M}$, (17), and the Dirichlet preconditioner M, (16). In Table 1 we report the iteration count, the condition number approximation, and the flop count for the aforementioned preconditioners.

The FETI algorithm with the Dirichlet preconditioner M required hundreds of iterations to converge, and the computational costs were one to two orders of magnitude bigger than for the other preconditioners. The iteration count grew faster than polylogarithmically as a function of the number of nodes on each subdomain edge, H/h, and appeared to grow linearly with the number of subdomains. The Dirichlet preconditioner is therefore noncompetitive, since many domain decomposition methods have convergence rates independent of the number of subdomains. The condition numbers estimates were on the order of 10^4–10^6, unusually large for these types of algorithms. Moreover, our estimates are likely to be smaller than the actual condition numbers, since there was no convincing convergence pattern for the condition number approximation obtained in the iteration; see section 6.1 and Figure 6 therein for more details. Thus, unlike in the case of FETI algorithms with conforming finite elements, the Dirichlet preconditioner M did not yield a numerically scalable method for mortar finite element methods.

The new preconditioner $\widehat{M}$ scaled similarly to M in the conforming case. When the number of nodes on each subdomain edge, H/h, was fixed and the number of subdomains, N, was increased, the iteration count showed only a slight growth, cf. Figure 3, plot (I). When H/h was increased, while the partition was kept unchanged, the increase in the number of iterations was quite satisfactory and very similar to that of the conforming case, cf. Figure 4, plot (I). The condition number estimates exhibited a

TABLE 1
Convergence results, 2D geometrically nonconforming partition, classical mortar elements

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
16	4	10	4.14	6.9e–1	21	26.95	1.4e+0	111	7.3e+3	7.2e+0
16	8	12	5.14	8.2e+0	21	29.86	1.4e+1	240	4.2e+4	1.6e+2
16	16	13	6.44	1.5e+2	23	36.53	2.6e+2	320	6.5e+4	3.7e+3
16	32	14	7.35	3.4e+3	23	38.03	5.6e+3	348	6.8e+4	7.4e+4
32	4	11	6.53	1.9e+0	23	34.51	3.7e+0	223	1.2e+4	3.5e+1
32	8	13	7.58	1.9e+1	24	45.96	3.5e+1	455	7.1e+4	6.6e+2
32	16	14	8.86	3.5e+2	26	61.97	6.5e+2	528	1.0e+5	1.3e+4
32	32	16	9.79	9.1e+3	27	65.39	1.5e+4	569	1.2e+5	3.2e+5
64	4	14	7.23	6.1e+0	32	47.99	1.3e+1	578	9.1e+4	2.2e+2
64	8	16	8.76	5.7e+1	35	72.62	1.2e+2	1012	7.5e+5	3.5e+3
64	16	18	10.68	1.0e+3	36	91.43	2.0e+3	1266	1.2e+6	7.1e+4
64	32	20	12.40	2.5e+4	39	94.47	4.8e+4	1324	1.4e+6	1.5e+6
128	4	14	7.60	1.3e+1	36	64.53	3.0e+1	1144	9.2e+5	9.2e+2
128	8	17	9.56	1.3e+2	40	82.09	2.9e+2	1350	6.8e+5	1.0e+4
128	16	19	11.73	2.3e+3	41	96.60	4.8e+3	1436	9.9e+6	1.7e+5
128	32	21	13.03	5.7e+4	41	99.82	1.1e+5	–	–	–

similar dependence on the number of subdomains and on the number of nodes on each subdomain edge; cf. Figure 5, upper row.

The block–diagonal preconditioner $\overline{M}$ had good convergence properties as well. The iteration counts showed just a small increase when the number of nodes on each subdomain edge was increased, while the partition was kept unchanged; cf. Figure 4, plot (II). There seemed to be a stronger than desired dependence of the iteration counts on the number of subdomains, which was less than optimal; cf. Figure 3, plot (II). The condition number estimates followed a similar pattern, but were significantly larger than the condition number estimates for the new preconditioner $\widehat{M}$; cf. Figure 5, lower row.

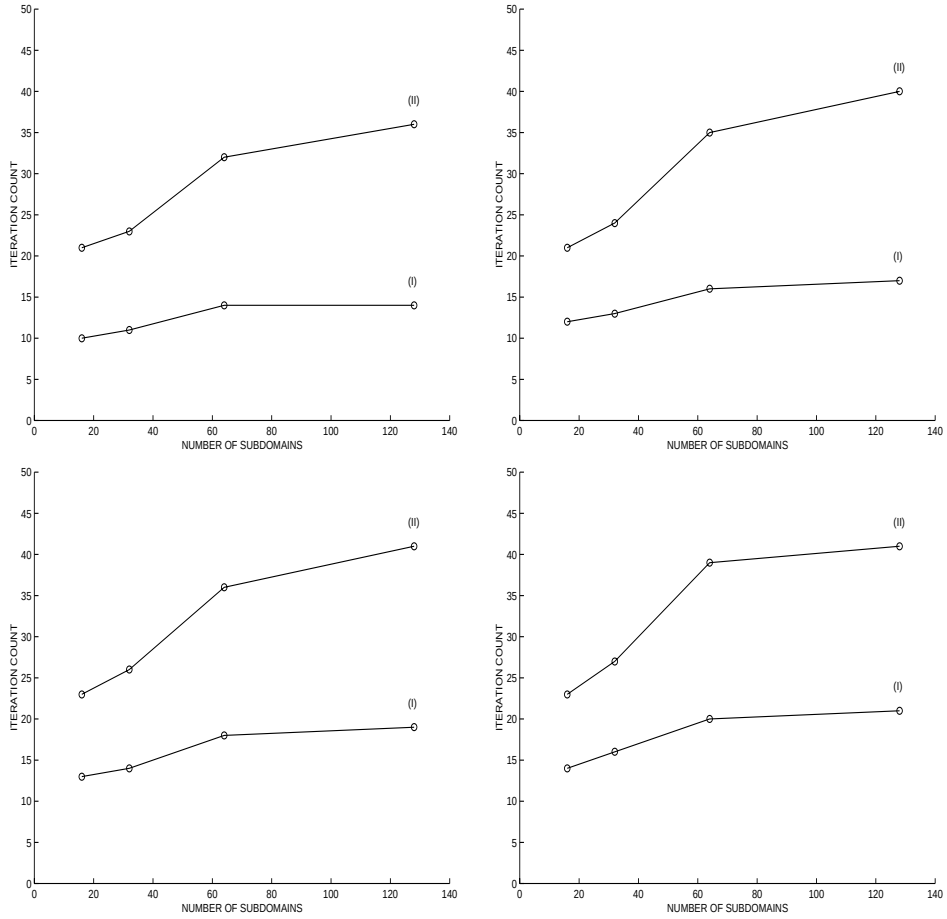
Overall, the block–diagonal preconditioner $\overline{M}$ required about twice as many iterations to convergence and twice as much computational effort as $\widehat{M}$; cf. Table 1. Therefore, even though multiplying $\overline{M}$ by a vector required less computational effort than when $\widehat{M}$ was used, the increase in the iteration count resulted in flop counts which are twice as large. This suggests that dropping the non-zero diagonal terms of BB^t relaxed the weak continuity conditions for mortar finite elements more than is optimal.

We conclude that, among the three preconditioners for FETI algorithms for mortar finite element methods analyzed here, the new preconditioner $\widehat{M}$ is the best.

5.2. New mortars vs. classical mortars. Another objective of our study was to compare the performance of the FETI algorithms for new mortar element methods with the performance of the FETI algorithms for classical mortar element methods.

We used the same nonconforming partitions of our computational domain, see Figure 2, and considered new mortar finite elements on the subdomains. As explained in section 2.1, the only difference between the two mortars methods is due to different mortar conditions across the interface Γ. This results in different Lagrange multiplier matrices B.

FIG. 3. *Dependence of the iteration count on the number of subdomains. 2D geometrically non-conforming partition, (I) = New Preconditioner, (II) = Block–diagonal Preconditioner. Upper left:* $H/h = 4$, *Upper right:* $H/h = 8$, *Lower left:* $H/h = 16$, *Lower right:* $H/h = 32$.



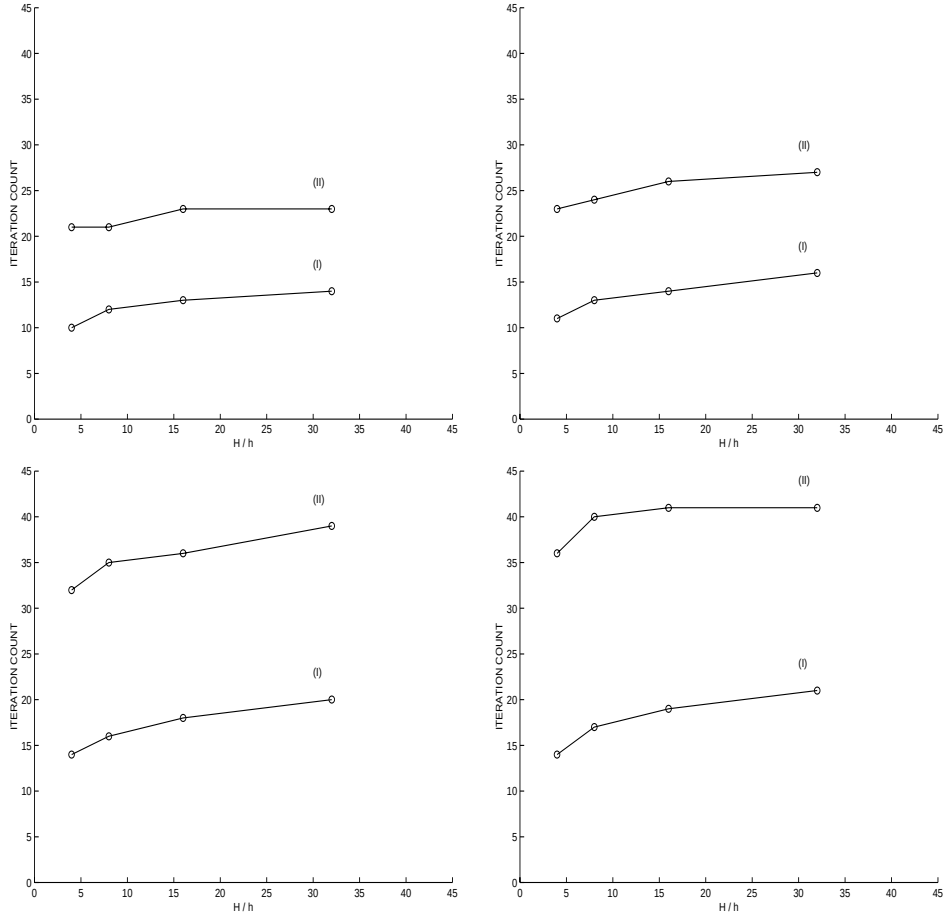
We run the same set of experiments for the new mortars discretization, for the preconditioners $\widehat{M}$, $\overline{M}$, and M. The PCG iteration was stopped when the residual norm had decreased by a factor of 10^{-6}. We report the iteration count, the condition number approximation, and the flop counts in Table 2.

Due to the inherent simplification of the Lagrange multiplier matrix B for the new mortar constraints, we expected the results for the new mortar method to be similar, but somewhat better than those for the classical mortar method. Indeed, this was confirmed by our numerical results.

The iteration counts for the new preconditioner $\widehat{M}$ were identical to those for $\widehat{M}$ for classical mortars. The condition number estimates and the flop counts were slightly smaller than in the classical mortar case, but essentially the same. Therefore, $\widehat{M}$ scaled just as well as in the classical mortar case.

For the new mortar conditions, the matrix BB^t had fewer nonzero entries outside its block–diagonal structure and fewer terms to be dropped in order to obtain $diag B_\gamma B_\gamma^t$. Therefore, the block diagonal preconditioner $\overline{M}$ was closer to $\widehat{M}$ than in the classical

FIG. 4. *Dependence of the iteration count on the number of nodes per subdomain edge. 2D geometrically nonconforming partition, (I) = New Preconditioner, (II) = Block–diagonal Preconditioner. Upper left:* $N = 16$, *Upper right:* $N = 32$, *Lower left:* $N = 64$, *Lower right:* $N = 128$.



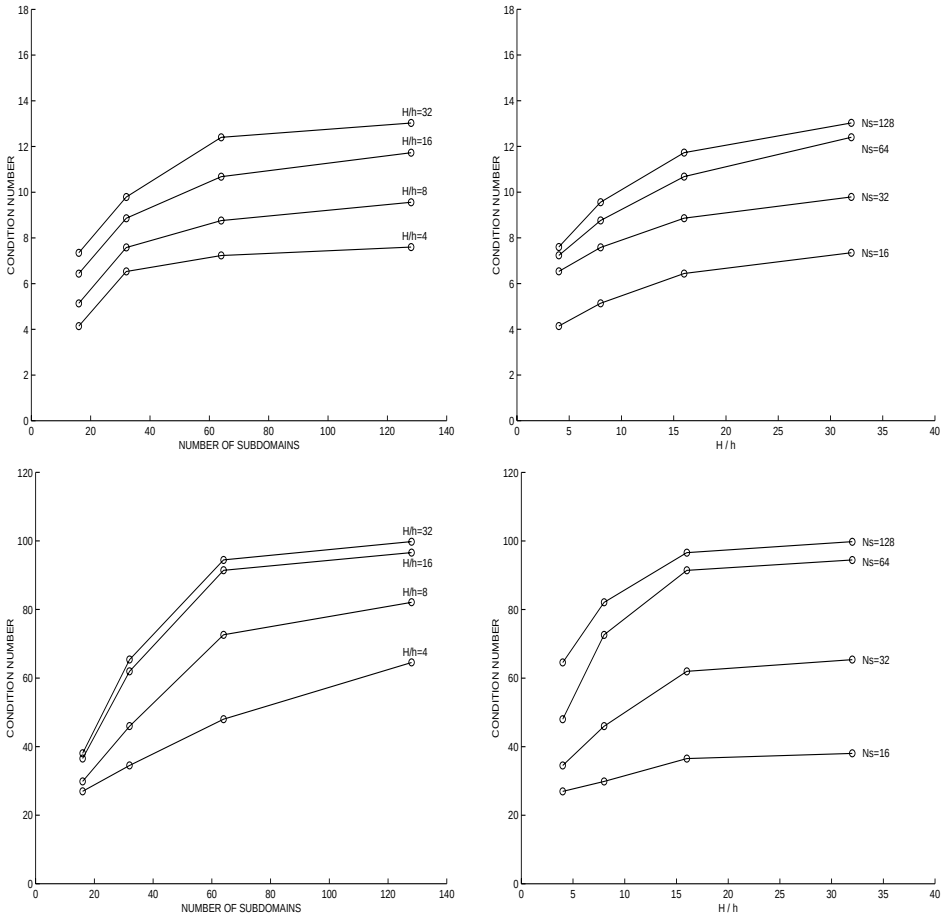
mortar case. This resulted in lower iteration counts and condition numbers for $\overline{M}$ than in the classical mortar case, and, consequently, in lower flop counts.

Once again, the iteration counts increased moderately with H/h and seemed to have a stronger than desired dependence on the number of subdomains. Overall, the block–diagonal preconditioner still performed worse than $\widehat{M}$ for the new mortars, with iteration counts one and a half times higher and with significantly bigger condition numbers.

An even greater improvement over the classical mortar case was obtained for the Dirichlet preconditioner M. The number of iterations required for M in the new mortar case was less than half the number of iterations required in the classical mortar case, and a similar improvement can be observed for the flop counts. The condition number estimates were about an order of magnitude less than for the classical mortar case.

Despite these improvements, the FETI algorithm with the Dirichlet preconditioner still required hundreds of iterations to converge and the iteration count appeared to grow linearly with the number of subdomains. The condition number estimates were on the order of 10^3–10^5, much higher than desired. Therefore, as in the classical mortar case, applying the preconditioner M for the FETI method did not result in a scalable

FIG. 5. *Upper Left: Dependence of the condition number on N for the new preconditioner; Upper Right: Dependence of the condition number on H/h for the new preconditioner; Lower Left: Dependence of the condition number on N for the block–diagonal preconditioner; Lower Right: Dependence of the condition number on H/h for the block–diagonal preconditioner.*



algorithm.

Once again, among the three preconditioners for FETI algorithms methods, the new preconditioner $\widehat{M}$ was the best. There was no significant improvement when using the new mortar elements instead of the classical mortar elements for the FETI algorithm with optimal preconditioner $\widehat{M}$.

5.3. Complexity study of the preconditioners. The last topic of this section is an analysis of how expensive is it to apply the preconditioners $\widehat{M}$ and $\overline{M}$, compared to applying the Dirichlet preconditioner M.

In each iteration step, we compute one vector multiplication by the preconditioner, which requires solving two systems with the matrix BB^t, and $diagB_\gamma B_\gamma^t$, respectively.

In section 4.2, we mentioned that $diagB_\gamma B_\gamma^t$ is obtained from BB^t by eliminating the non–zero entries outside the diagonal blocks. These entries are of two types. Some correspond to Lagrange multipliers associated to the first and last interior points of the nonmortars. Other occur because there are nodal basis functions associated to points on

TABLE 2
Convergence results, 2D geometrically nonconforming partition, new mortar elements

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
16	4	10	4.20	6.8e–1	18	18.40	1.2e+0	55	897	3.5e+0
16	8	12	5.15	8.1e+0	19	20.85	1.3e+1	70	908	4.7e+1
16	16	13	6.50	1.5e+2	20	25.40	2.3e+2	70	1018	8.0e+2
16	32	14	7.43	3.4e+3	21	30.11	5.1e+3	80	1090	2.0e+5
32	4	11	6.55	1.9e+0	19	23.58	3.0e+0	91	913	1.4e+1
32	8	13	7.61	1.9e+1	20	34.45	2.9e+1	101	1565	1.4e+2
32	16	14	8.87	3.5e+2	22	48.14	5.5e+2	114	1873	2.9e+3
32	32	16	9.80	9.1e+3	23	50.63	1.3e+4	117	2108	6.6e+4
64	4	14	7.29	6.0e+0	27	32.49	1.0e+1	278	1.0e+4	1.1e+2
64	8	16	8.69	5.6e+1	28	43.37	9.6e+1	292	1.6e+4	1.0e+3
64	16	18	10.83	1.0e+3	29	56.19	1.6e+3	297	1.9e+4	1.7e+4
64	32	20	12.57	2.5e+4	31	70.74	3.9e+4	312	2.2e+4	3.1e+5
128	4	14	7.60	1.3e+1	30	41.78	2.4e+1	614	1.0e+5	4.9e+2
128	8	17	9.57	1.3e+2	33	52.36	2.4e+2	677	1.3e+5	4.8e+3
128	16	19	11.75	2.2e+3	35	62.54	4.1e+3	755	1.6e+5	8.9e+4
128	32	21	13.07	5.6e+4	36	66.66	9.7e+4	–	–	–

the mortar sides, the support of which intersects more than one nonmortar. However, in the two dimensional case, there are relatively few such non-zero entries; see Figure 7.

It is easy to see that $diagB_\gamma B_\gamma^t$ has a band width of order H/h, the number of interior nodes on an arbitrary nonmortar. The matrix BB^t is also banded, but in this case the band depends on the ordering of the nodes on the interface, and it is possible to have band width of order $1/h$. Therefore, multiplying a vector by $(BB^t)^{-1}$ is potentially an expensive operation. To minimize the effect of these vector multiplications, we computed the Cholesky factorizations of BB^t and $diagB_\gamma B_\gamma^t$ just once, and stored the factors. Then, solving systems with BB^t or $diagB_\gamma B_\gamma^t$ only amounted to one back and one forward solve.

Our results showed that the costs of a vector multiplication by $(BB^t)^{-1}$ were between two and ten times smaller than those associated with $(diagB_\gamma B_\gamma^t)^{-1}$. However, due to the sparsity pattern of BB^t, even the costs associated to $(BB^t)^{-1}$ were relatively small compared to those for other operations performed during one iteration, e.g., multipling a vector by the Schur complement, or by its pseudoinverse; cf. Table 3. These low relative costs result into very similar flop counts per iteration step for the two preconditioners, almost identical for the case of many nodes per subdomain edge, i.e., $H/h = 16$ and $H/h = 32$.

As expected, the costs associated to $(BB^t)^{-1}$ in each iteration step decreased significantly, from six percent to less than .05 percent, when the partition was fixed and H/h increased, since the costs of multiplying S and $S^\dagger$ by a vector rose much faster than those corresponding to $(BB^t)^{-1}$.

From the flop counts reported in Table 1, it is clear that the improvement of the iteration count easily offsets the small extra costs due to the complexity of $\widehat{M}$ and $\overline{M}$.

TABLE 3

Complexity study of one iteration step for the new preconditioner and the block–diagonal preconditioner, 2D geometrically nonconforming partition

		New Preconditioner			Block–diagonal Preconditioner		
N	H/h	Mflops for $(BB^t)^{-1}$	Mflops per iteration	Ratio	Mflops for $(diagB_\gamma B_\gamma^t)^{-1}$	Mflops per iteration	Ratio
16	4	3.9e–3	6.9e–2	.06	1.4e–3	6.7e–2	.02
16	8	9.7e–3	6.8e–1	.02	4.6e–3	6.7e–1	.007
16	16	2.2e–2	1.1e+1	.002	1.1e–2	1.1e+1	.001
16	32	4.6e–2	2.4e+2	.0002	2.4e–2	2.4e+2	.0001
32	4	1.1e–2	1.7e–1	.07	3.2e–3	1.6e–1	.02
32	8	3.1e–2	1.5e+0	.02	9.9e–3	1.5e+0	.007
32	16	6.5e–2	2.5e+1	.003	2.4e–2	2.5e+1	.001
32	32	1.4e–1	5.7e+2	.0003	5.2e–2	5.7e+2	.00009
64	4	4.1e–2	4.3e–1	.10	6.9e–3	3.9e–1	.02
64	8	1.1e–1	3.5e+0	.03	2.2e–2	3.4e+0	.007
64	16	2.3e–1	5.6e+1	.004	5.4e–2	5.6e+1	.001
64	32	4.7e–1	1.2e+3	.0004	1.2e–1	1.2e+3	.00009
128	4	1.2e–1	9.3e–1	.13	1.4e–2	8.2e–1	.02
128	8	3.1e–1	7.4e+0	.04	4.4e–2	7.2e+0	.006
128	16	6.8e–1	1.2e+2	.006	1.1e–1	1.2e+2	.0009
128	32	1.4e+0	2.7e+3	.0005	2.3e–1	2.7e+3	.00009

6. Numerical Results for Three Dimensional Problems. In this section, we report numerical results for the FETI method for mortar finite element discretizations of a three–dimensional problem. As before, we compare the performance of different FETI preconditioners, and discuss the effects of using the new mortar finite elements instead of the classical ones. We include a study of the costs of applying the new preconditioner for the for classical mortar elements and more details on the convergence rate of the condition number approximation for some of our algorithms.

As the model problem in 3–D, we chose the Poisson equation on the unit cube $\Omega = [0,1]^3$ with zero Dirichlet boundary conditions. The right hand side was selected such that the exact solution is known.

The computational domain Ω was partitioned into 8, 16, and 32 nonconforming parallelepipeds, respectively. We chose these partitions such that in each case there exist floating subdomains, i.e., interior subdomains.

The subdomains of the partition had diameter of order H, and Q_1 elements of mesh size h were used in each subdomain. The number of nodes on each edge was, *on average*, 4, 8, and 16. Across the partition interface Γ the meshes did not match, and mortar conditions for three dimensional elements were enforced; see Section 2.2. This results in a Lagrange multipliers matrix B, which, as explained for the two dimensional case, plays a very important role in all FETI algorithms.

We report the iteration counts, the condition number approximations, and the flop counts of the algorithms. The PCG iteration was stopped when the residual norm had decreased by a factor of 10^{-6}. All the experiments were carried out in MATLAB.

TABLE 4
Convergence results, 3D geometrically nonconforming partition, classical mortar elements

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
8	4	11	4.31	1.8e+0	33	55	3.9e+0	866	2.2e+6	9.9e+1
8	8	14	6.54	4.9e+1	33	70	7.3e+1	7984	4.4e+7	1.7e+4
8	16	16	7.90	1.4e+3	38	81	2.4e+3	–	–	–
16	4	13	6.77	6.6e+0	36	75	9.7e+0	2985	1.2e+7	7.7e+2
16	8	15	8.20	1.6e+2	37	85	1.7e+2	14169	2.7e+8	6.3e+4
16	16	17	9.28	4.1e+3	50	176	6.5e+3	–	–	–
32	4	14	8.29	3.1e+1	44	141	2.7e+1	4156	3.5e+7	2.4e+3
32	8	16	9.77	1.1e+3	55	350	5.5e+2	–	–	–
32	16	17	10.69	2.7e+4	69	523	2.9e+4	–	–	–

6.1. Convergence properties of the FETI algorithms. We did not compute the Schur complements explicitly, but only stored those components of the stiffness matrices which were relevant for the multiplication of a vector by the Schur complement matrix and by the pseudoinverse of the Schur complement. We tested the performance of the same preconditioners as in the two dimensional case, i.e., the new preconditioner $\widehat{M}$, cf. (18), the preconditioner $\overline{M}$, cf. (17), and the Dirichlet preconditioner M, cf. (16). We report the iteration count, the condition number estimate, and the flop count of the algorithms in Table 4.

As in the two dimensional case, the FETI algorithm with the Dirichlet preconditioner M did not scale well and required thousands of iterations to converge. Since it soon became clear that M was not an optimal preconditioner, and due to significant computational costs, we only performed tests for every partition of Ω in the case of 4 nodes on each edge, and for the 8 and the 16 subdomains partitions for the case of 8 nodes on each edge of the subdomains.

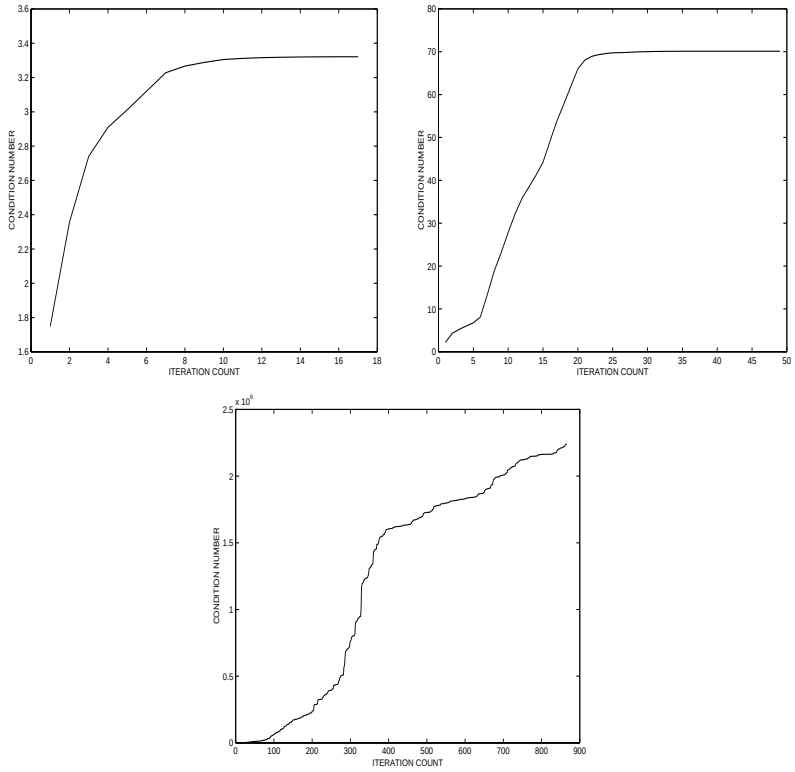
The iteration count seemed to be a linear function of the number of subdomains, and it grew by an order of magnitude when H/h was doubled while keeping the 8 and the 16 subdomain partitions fixed. The computational costs were also at least two orders of magnitude bigger than those for the other preconditioners, and deteriorated as the number of nodes per subdomain edge increased.

The condition number estimates followed a similar dependence pattern on H/h and the number of subdomains. They were on the order of 10^6–10^8, much worse than even in the 2–D case. In Figure 6, we present the convergence pattern of the condition number estimates for the 8 subdomains partition with $H/h = 4$. For M, the PCG iteration was stopped when the residual norm had decreased by a tolerance factor of 10^{-6}, while for $\widehat{M}$ and $\overline{M}$ the tolerance was set at 10^{-10}.

For the Dirichlet preconditioner, there was no clear convergence pattern for the condition number estimates. This suggests that the (extremely large) condition number approximations reported in Table 4 are just lower bounds for the actual condition number. For the other two preconditioners, convergence was achieved early in the iteration count. The estimates reported in Table 4 are within one percent of the condition number corresponding to a tolerance of 10^{-10}.

The new preconditioner $\widehat{M}$ scaled similarly to the 2–D case, and to the Dirichlet preconditioner in the conforming case. The number of iterations grew very slowly when

FIG. 6. *Convergence pattern for the condition number, 3D geometrically nonconforming partition, $N = 8$, $H/h = 4$. Top left: new preconditioner, $tol = 10^{-10}$, Top right: block–diagonal preconditioner, $tol = 10^{-10}$, Bottom: Dirichlet preconditioner, $tol = 10^{-6}$.*



the number of nodes on each subdomain edge (i.e., H/h) was fixed and the number of subdomains was increased. When the partition was kept unchanged and H/h was increased, the iteration count increased slightly, and it seemed to have a polylogarithmic dependence on H/h.

The convergence analysis for the block–diagonal preconditioner $\overline{M}$ is particularly interesting in the three dimensional case; $\overline{M}$ was a possible alternative to $\widehat{M}$ since it required significantly less computational effort per iteration step. However, our results showed a much stronger than desired dependence of the iteration count for $\overline{M}$ on the number of nodes on each subdomain edge. This dependence grew stronger as the number of subdomains in the partition increased. The number of iterations increased with the number of subdomains, another undesirable property. The condition number estimates followed a similar pattern, and were significantly larger than those corresponding to $\widehat{M}$. Their relatively large values, on the order of 10^2, and their dependence on the number of subdomains were unsatisfactory.

We refer the reader to section 6.2 for a more detailed comparison between $\overline{M}$ and $\widehat{M}$.

6.2. New mortars vs. classical mortars. In this section, we compare the performance of the FETI algorithms for new mortar element methods with that of the FETI algorithms for classical mortar element methods.

Using the same nonconforming partitions of Ω as before, we introduced new mortar

TABLE 5
Convergence results, 3D geometrically nonconforming partition, new mortar elements

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
8	4	11	4.46	1.7e+0	29	42	3.3e+0	272	2.8e+4	3.0e+1
8	8	14	6.52	4.8e+1	30	55	6.5e+1	648	4.1e+4	1.3e+3
8	16	16	7.80	1.4e+3	37	66	2.4e+3	720	4.7e+4	4.5e+4
16	4	13	6.69	6.4e+0	33	59	8.8e+0	817	2.2e+5	2.0e+2
16	8	15	8.17	1.6e+2	34	74	1.5e+2	1306	3.0e+5	5.7e+3
16	16	17	9.19	4.1e+3	43	115	5.6e+3	1870	3.5e+5	2.4e+5
32	4	14	8.02	3.1e+1	41	113	2.3e+1	989	4.2e+5	5.3e+2
32	8	16	9.28	1.1e+3	46	219	4.5e+2	1503	6.2e+5	1.4e+4
32	16	17	10.26	2.3e+4	53	331	3.6e+4	–	–	–

finite elements on the subdomains and run the same set of experiments as before. We report the results in Table 5.

The iteration counts for the new preconditioner were identical to those for classical mortars. The condition number estimates were slightly smaller than in the classical mortar case, but essentially the same. The flop counts were between one percent and five percent smaller than in the classical mortar case. In other words, $\widehat{M}$ scaled as well as in the classical mortar case.

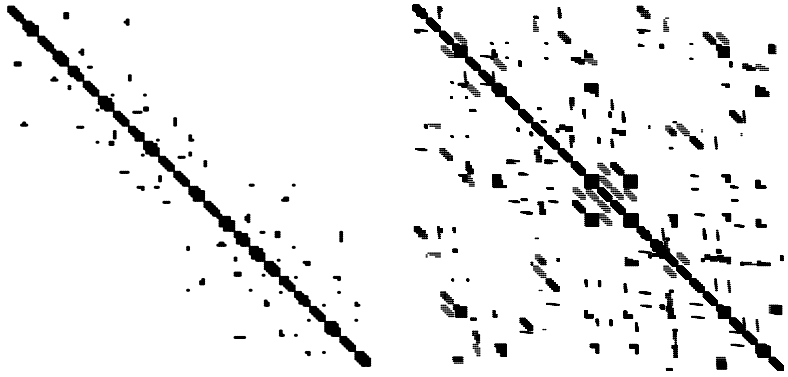
As explained in the 2–D case, the new mortar conditions resulted in simpler mortar conditions. An important consequence was that the off–diagonal entries of BB^t were fewer and smaller in absolute value than in the classical mortar case. Therefore, the block diagonal preconditioner $\overline{M}$ was closer to $\widehat{M}$ than in the classical mortar case.

This generated a clear improvement for the iteration count and for the condition number estimate of the block–diagonal preconditioner $\overline{M}$. The number of iterations decreased from the classical mortar case, in particular when the iteration count for $\overline{M}$ was higher than desired, e.g., for the partition of Ω into 32 subdomains. It also resulted in a decrease in the flop counts. However, the iteration count appeared to depend on the number of subdomains, when the number of nodes on each edges was fixed. The dependence of the iteration count on H/h seemed to be stronger than polylogarithmic.

The improvement generated by the new mortar method was even more significant for the Dirichlet preconditioner. The iteration count decreased to hundreds of iterations, instead of thousands as was the case for the classical mortar method. The condition number estimates were lower by two orders of magnitude, in a range of order 10^4–10^5, and the flop counts were one order of magnitude bigger than for the other preconditioners. However, the FETI algorithm with the Dirichlet preconditioner did not scale as a good domain decomposition method.

6.3. Complexity study of the preconditioners. A comparison between the block–diagonal and the new preconditioner showed that using the preconditioner $\overline{M}$ resulted in a method which converged in about three times as many iterations than when $\widehat{M}$ was used. We recall that, in the 2–D case, the number of iterations for the method with $\overline{M}$ was only about twice as large as that for $\widehat{M}$. This appears to be due to the fact that, in the 3–D case, there are many nodes, e.g., the nodes on the wire baskets of the subdomains, which influence several nonmortar conditions. Therefore, the block diagonal structure of BB^t is no longer as dominant, and many non–zero entries of BB^t

FIG. 7. *Sparsity pattern of BB^t. Left: 2D partition, $N = 16$, $H/h = 8$, Right: 3D partition, $N = 16$, $H/h = 8$.*



need to be dropped; see Figure 7 for the differences in the sparsity pattern of BB^t for the 2–D and 3–D cases.

TABLE 6
Complexity study of one iteration step for the new preconditioner and the block–diagonal preconditioner, 3D geometrically nonconforming partition

		New Preconditioner			Block–diagonal Preconditioner		
N	H/h	Mflops for $(BB^t)^{-1}$	Mflops per iteration	Ratio	Mflops for $(diagB_\gamma B_\gamma^t)^{-1}$	Mflops per iteration	Ratio
8	4	3.2e–2	1.5e–1	.22	4.8e–3	1.2e–1	.04
8	8	5.8e–1	2.7e+0	.21	9.2e–2	2.2e+0	.04
8	16	6.8e+0	6.9e+1	.10	1.2e+0	6.4e+1	.02
16	4	1.3e–1	3.9e–1	.34	1.1e–2	2.7e–1	.04
16	8	2.1e+0	6.6e+0	.32	1.8e–1	4.6e+0	.04
16	16	2.2e+1	1.5e+2	.15	1.9e+0	1.3e+2	.02
32	4	6.9e–1	1.3e+0	.55	3.0e–2	6.0e–1	.05
32	8	1.1e+1	2.1e+1	.54	5.0e–1	9.9e+0	.05
32	16	1.2e+2	3.8e+2	.32	5.4e+0	2.7e+2	.02

The flop counts for $\overline{M}$ were less than twice as large as to those required for the convergence of the new preconditioner, even if the FETI method with $\overline{M}$ required about three times as many iterations as that with $\widehat{M}$. Moreover, for partitions with many subdomains and a small number of nodes on each edge, e.g., for $N = 32$ and $H/h = 4$ or $H/h = 8$, the complexities of the mortar conditions and of the Lagrange multiplier matrix B, are higher relative to those of the Schur complement and its pseudoinverse. In these cases, the flop count for the block–diagonal preconditioner was less than that for the new preconditioner, despite the difference in iteration count.

This suggested that the costs of applying $(BB^t)^{-1}$ were significant in the three dimensional case, and this was confirmed by our results. In Table 6, we present the costs of applying $(BB^t)^{-1}$ and $diagB_\gamma B_\gamma^t$ twice during an iteration step, relative to the total flop count for one iteration step.

The costs associated to $(BB^t)^{-1}$ were between 10 and 55 percent of those for one

iteration step. This was much higher than for the two dimensional case, when the relative costs were at most 13 percent, and as low as .02 percent; cf. Table 3. The costs associated to $(diagB_\gamma B_\gamma^t)^{-1}$ were much smaller, at most 5 percent of those for one iteration step. This was the reason why the flop counts per iteration were significantly lower for the block–diagonal preconditioner than for the new preconditioner.

The dependence of the relative cost of applying $(BB^t)^{-1}$ on the number of subdomains N and the number of nodes on each edge H/h was similar to that for the two dimensional case. It increased when H/h was kept fixed while the partition became more complex, and decreased when the partition was kept unchanged and H/h was increased. These results are consistent with the increased costs of multiplying a vector by the Schur complement and the pseudoinverse of the Schur complement when H/h increases, and the increased complexity of the Lagrange multiplier matrix B when the partition had more subdomains.

7. Continuity and Mortar Conditions for Matching Meshes. In the classical FETI algorithm, the underlying partition of Ω is geometrically conforming, the meshes across the interface match and continuity conditions are enforced across the interface; cf. section 3. We note that it is also possible to require mortar matching across the interface. In this section, we compare the performance of the resulting FETI algorithms for these two different types of matchings.

We considered both two and three dimensional problems. For mortar finite elements, we tested FETI algorithms with all three preconditioners, while for conforming finite elements we only used the new preconditioner and the Dirichlet preconditioner. The block–diagonal preconditioner is identical to the new preconditioner for continuity matchings, since BB^t is a block–diagonal matrix.

The results for the classical mortar methods and the new mortar methods were once again very similar, except for the case of the Dirichlet preconditioner, where the new mortars provided a significant improvement. However, our main goal was to compare the performance of continuity matchings versus mortar matchings. Therefore, we only present here only the results for the new mortar methods, which always resulted in better algorithms than the classical mortar methods.

7.1. The Two Dimensional Case. For the two dimensional experiments, the computational domain Ω, the unit square, was partitioned into 4×4, 6×6, 8×8, and 11×11 congruent squares, and Q_1 elements were used in each square. The meshes match across Γ, and non-redundant pointwise continuity conditions, or mortar conditions, were used across Γ for comparison purposes. Except for the different partitions, the experiments have the same parameters as in section 5.

We report the iteration count, the condition number estimate, and the flop count for the FETI algorithms with new mortar finite elements in Table 7.

When new mortar conditions were used across the interface, computing the Lagrange multiplier matrix B was very simple for matching nodes. In particular, no computations of integrals resulting from the mortar conditions (1) were necessary. The new mortar conditions are equivalent to continuity conditions for all matchings except for those corresponding to the first and last interior nodes on the nonmortar sides, where the end point nodes are involved as well. Therefore, B was very similar for the two types of matchings, and BB^t was very close to twice the identity matrix. Almost no extra work was required when a system with the matrix BB^t had to be solved.

Another consequence of matching meshes was that BB^t had very few nodes outside its block–diagonal structure $diagB_\gamma B_\gamma^t$. Therefore, we expected the convergence results for the new preconditioner and for the block–diagonal preconditioner to be similar.

TABLE 7
Convergence results, 2D geometrically conforming partition, matching grids and new mortar constraints across the interface

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
16	4	5	2.18	3.4e–1	5	2.41	3.4e–1	5	3.93	3.3e–1
16	8	5	2.42	2.9e+0	6	2.49	3.5e+0	6	3.92	3.4e+0
16	16	6	3.18	7.2e+1	7	3.27	8.5e+1	7	5.11	8.5e+1
16	32	6	3.59	1.3e+3	7	4.31	1.6e+3	8	6.73	1.8e+3
36	4	7	2.34	1.5e+0	8	3.71	1.6e+0	8	3.76	1.6e+0
36	8	9	2.90	1.4e+1	10	4.70	1.6e+1	11	4.81	1.7e+1
36	16	9	3.70	2.5e+2	10	4.62	2.8e+2	12	6.24	3.3e+2
36	32	10	4.22	6.9e+3	11	5.14	7.6e+3	13	8.11	9.0e+3
64	4	8	2.42	3.5e+0	9	3.93	3.8e+0	9	4.00	3.8e+0
64	8	9	3.09	2.7e+1	11	5.02	3.4e+1	12	5.20	3.5e+1
64	16	11	3.98	7.3e+2	11	5.37	7.2e+2	13	6.77	8.6e+2
64	32	12	4.53	1.4e+4	13	5.55	1.5e+4	15	8.79	1.8e+4
121	4	9	2.45	8.6e+0	10	4.08	9.0e+0	10	4.18	8.9e+0
121	8	10	3.14	6.5e+1	12	5.20	7.7e+1	13	7.10	8.3e+1
121	16	11	4.04	1.2e+3	13	5.73	1.4e+3	15	8.29	1.6e+3
121	32	13	4.66	3.7e+4	14	5.84	4.0e+4	17	9.19	4.8e+4

Indeed the results from Table 7 show that $\widehat{M}$ and $\overline{M}$ behaved similarly in terms of iteration counts and condition number estimates, which were just slightly higher for the block diagonal preconditioner $\overline{M}$. Both preconditioners scaled very well with the number of subdomains and the number on nodes on each edge. The computational costs for one iteration step were almost identical, which resulted in better flop counts for $\widehat{M}$, even when the iteration counts differed by only one iteration.

In contrast with the other algorithms, the Dirichlet preconditioner M performed very well for two dimensional problems with matching nodes. The iteration counts were very small, comparable to those corresponding to $\widehat{M}$ and $\overline{M}$. However, since BB^t is close to twice the identity matrix, the computational costs per iteration do not show a relevant improvement for M. The Dirichlet preconditioner yielded higher flop counts than the other two preconditioners.

We now turn our attention to the case when pointwise continuity was enforced across the interface; cf. Table 8.

As expected, both the Dirichlet preconditioner and the new preconditioner had very good scaling properties. In particular, we note that the condition number estimates were almost constant when H/h was kept fixed and the number of subdomains was changed. However, $\widehat{M}$ converged in less than half the number of iterations necessary for M, and the same was true for the computational costs.

For continuity matchings, the vector matrix multiplication by $B^t(BB^t)^{-1}B$ is very easy to compute, since it is close to an operator from the balancing algorithm; see [32]. It is possible to write the PCG algorithm with the new preconditioner such that only the product of a vector by $B^t(BB^t)^{-1}B$ and not by $(BB^t)^{-1}$ needs to be computed.

Using the data from Table 7 and Table 8, we can address the main topic of this section, finding the best FETI algorithm for conforming partitions of Ω. We compared

TABLE 8
Convergence results, 2D geometrically conforming partition, matching grids and continuity constraints across the interface

		New Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops
16	4	7	2.17	4.7e–1	18	23.02	1.2e+0
16	8	8	2.91	4.6e+0	19	28.11	1.1e+1
16	16	10	3.90	1.2e+2	20	33.75	2.4e+2
16	32	11	4.47	2.4e+3	21	39.01	4.7e+3
36	4	8	2.18	1.6e+0	24	23.23	4.8e+0
36	8	10	2.93	1.6e+1	26	28.15	4.1e+1
36	16	12	3.91	3.3e+2	26	34.03	7.2e+2
36	32	13	4.50	9.0e+3	28	39.13	1.9e+4
64	4	8	2.19	3.4e+0	25	23.34	1.0e+1
64	8	10	2.95	2.9e+1	28	28.19	8.2e+1
64	16	12	3.90	7.9e+2	28	34.03	1.8e+3
64	32	13	4.51	1.5e+4	29	39.33	3.4e+4
121	4	9	2.21	8.1e+0	27	23.35	2.4e+1
121	8	10	2.99	6.4e+1	29	28.23	1.8e+2
121	16	12	3.92	1.3e+3	29	34.10	3.1e+3
121	32	13	4.54	3.7e+4	30	39.44	8.5e+4

the new preconditioner for new mortar matchings and for continuity conditions. The iteration counts and the condition number estimates were slightly lower for the new mortar case. The flop counts were also better for mortar matchings, since the Lagrange multiplier matrices had similar structure for the two types of matchings and the costs per iteration step were almost identical.

We conclude that the new mortar matchings represent an improvement over the continuity matchings. This might be due in part to the fact that the mortar matching conditions corresponding to the first and last interior points on the nonmortars replace the continuity constraints at the crosspoints and their neighboring nodes.

7.2. The Three Dimensional Case. For the three dimensional experiments, the unit cube was partitioned into $2 \times 2 \times 2$, $2 \times 2 \times 4$, and $2 \times 4 \times 4$ geometrically conforming, non–congruent parallelepipeds; Q_1 meshes were considered in each subdomain such that the meshes across the interface matched. Across Γ, non-redundant pointwise continuity conditions, or biorthogonal mortar conditions, were enforced by using Lagrange multipliers.

For the mortar matchings, we present convergence results only for the new mortar method. We run the same set of experiments as in section 6, for all three preconditioners and for 4, 8, and 16 nodes on each subdomain edge. We report the results in Table 9.

For the three dimensional case, the Lagrange multiplier matrix B was no longer very close to a multiple of the identity. The mortar matching conditions were different than the continuity matchings for all the interior nodes on the nonmortar faces with neighbors on the boundary of the face.

Once again, the new preconditioner $\overline{M}$ scaled well. The iteration count and the condition number estimate depended only weakly on H/h and on N, the number of subdomains in the partition of Ω. A similar behavior was observed for the block–diagonal

TABLE 9
Convergence results, 3D geometrically conforming partition, matching grids and new mortar constraints across the interface

		New Precond			Block–diag Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops	Iter	Cond	Mflops
8	4	8	3.13	7.8e–1	12	4.03	9.8e–1	127	2.9e+4	1.1e+1
8	8	9	3.92	1.9e+1	12	4.78	2.2e+1	212	3.6e+4	3.8e+2
8	16	11	4.41	6.8e+2	13	5.99	7.7e+2	301	4.2e+4	1.8e+4
16	4	8	5.31	2.4e+0	12	7.34	2.3e+0	141	6.5e+4	2.3e+1
16	8	10	7.00	5.5e+1	12	9.39	4.7e+1	233	9.0e+4	8.7e+2
16	16	11	8.32	1.4e+3	13	11.89	1.6e+3	341	1.2e+5	4.1e+4
32	4	9	5.70	3.9e+0	12	11.55	4.0e+0	400	1.6e+5	1.3e+2
32	8	11	8.24	8.8e+1	14	14.99	8.7e+1	630	2.0e+5	3.8e+3
32	16	12	10.31	2.2e+3	15	18.62	2.8e+3	–	–	–

TABLE 10
Convergence results, 3D geometrically conforming partition, matching grids and continuity constraints across the interface

		New Precond			Dirichlet Precond		
N	H/h	Iter	Cond	Mflops	Iter	Cond	Mflops
8	4	6	1.77	4.6e–1	21	75.54	1.5e+0
8	8	8	2.50	1.4e+1	25	80.30	4.4e+1
8	16	9	3.17	5.3e+2	27	84.25	1.6e+3
16	4	8	3.34	1.4e+0	31	84.12	5.4e+0
16	8	9	4.91	3.4e+1	35	90.73	1.3e+2
16	16	11	6.30	1.3e+3	36	94.35	4.3e+3
32	4	8	4.13	2.6e+0	38	95.51	1.2e+1
32	8	10	6.76	6.0e+1	41	98.12	2.4e+2
32	16	12	8.73	2.2e+3	43	99.82	7.9e+3

preconditioner, at somewhat higher iteration counts. However, due to the relative complexity of BB^t, which was no longer very close to its block–diagonal structure $diagB_\gamma B_\gamma^t$, the preconditioner $\overline{M}$ required less computational effort per iteration than $\widehat{M}$. The difference in the iteration counts has been thus compensated, the two preconditioners resulting in algorithms with very close flop counts.

Unlike in the two dimensional case with matching nodes, the Dirichlet preconditioner M required hundreds of iterations to converge and did not have good scalability properties. The condition number estimates were on the order of 10^4–10^5, and the flop counts were at least one order of magnitude greater than for the other preconditioners.

The convergence results for pointwise continuity matchings across the interface are reported in Table 10.

The new preconditioner yielded a scalable algorithm with very low iteration counts and condition numbers. The Dirichlet preconditioner also resulted in a scalable algorithm, but required at least three times as many iterations as $\widehat{M}$ for convergence. The condition number estimates were much larger as well, but depended weakly on changes of parameters H/h and N. The complexity of the matrix B is compensated by the

improvement in the iteration counts. The flop counts for $\widehat{M}$ are at least half of those for M.

We conclude this section by discussing the differences between the two types of matchings for three dimensional problems. The new preconditioner resulted into the best algorithms for both new mortar and continuity matchings. The iteration counts and the condition number estimates were slightly lower for the continuity case. The computational effort per iteration required in the mortar case is greater than in the continuity case, since BB^t is no longer very close to a multiple of the indentity. Coupled with lower iteration counts, this results in better flop counts for the continuity matching algorithms.

REFERENCES

[1] Yves Achdou and Yuri A. Kuznetsov. Substructuring preconditioners for finite element methods on nonmatching grids. *East-West J. Numer. Math.*, 3(1):1–28, 1995.

[2] Yves Achdou, Yuri A. Kuznetsov, and Olivier Pironneau. Substructuring preconditioners for the Q_1 mortar element method. *Numer. Math.*, 71(4):419–449, 1995.

[3] Yves Achdou, Yvon Maday, and Olof B. Widlund. Iterative substructuring preconditioners for mortar element methods in two dimensions. *SIAM J. Numer. Anal.*, 36:551–580, 1999.

[4] Faker Ben Belgacem, Yvon Maday, and Annalisa Buffa. The mortar finite element method for Maxwell equations in 3D. *C. R. Acad. Sci. Paris*, 329:903–908, 1999.

[5] Faker Ben Belgacem. The mortar element method with Lagrange multipliers. *Numer. Math.*, 84(2):173–197, 1999.

[6] Faker Ben Belgacem and Yvon Maday. Non-conforming spectral element method for second order elliptic problem in 3D. Technical Report R93039, Laboratoire d'Analyse Numérique, Université Pierre et Marie Curie – Centre National de la Recherche Scientifique, 1993.

[7] Faker Ben Belgacem and Yvon Maday. The mortar element method for three dimensional finite elements. *RAIRO Modél. Math. Anal. Numér.*, 31:289–302, 1997.

[8] Christine Bernardi, Yvon Maday, and Anthony T. Patera. A new non conforming approach to domain decomposition: The mortar element method. In Haim Brezis and Jacques-Louis Lions, editors, *Collège de France Seminar*. Pitman, 1994. This paper appeared as a technical report about five years earlier.

[9] Dietrich Braess, Wolfgang Dahmen, and Christian Wieners. A multigrid algorithm for the mortar finite element method. *SIAM J. Numer. Anal.*, 37:48–69, 2000.

[10] Dietrich Braess, Maksymilian Dryja, and Wolfgang Hackbusch. Multigrid method for nonconforming FE–discretisations with application to nonmatching grids. *Computing*, 63:1–25, 1999.

[11] Annalisa Buffa, Yvon Maday, and Francesca Rapetti. A sliding mesh-mortar method for a two-dimensional eddy currents model of electric engines. Technical Report R99002, Laboratoire d'Analyse Numérique, Université Pierre et Marie Curie – Centre National de la Recherche Scientifique, 1999.

[12] Xiao-Chuan Cai, Maksymilian Dryja, and Marcus Sarkis. Overlapping non-matching grid mortar element methods for elliptic problems. *SIAM J. Numer. Anal.*, 36:581–606, 1999.

[13] Mario A. Casarin and Olof B. Widlund. A hierarchical preconditioner for the mortar finite element method. *ETNA*, 4:75–88, June 1996.

[14] Maksymilian Dryja. Additive Schwarz methods for elliptic mortar finite element problems. In K. Malanowski, Z. Nahorski, and M. Peszynska, editors, *Modeling and Optimization of Distributed Parameter Systems with Applications to Engineering*. IFIP, Chapman & Hall, London, 1996.

[15] Maksymilian Dryja. An iterative substructuring method for elliptic mortar finite element problems with a new coarse space. *East-West J. Numer. Math.*, 5(2):79–98, 1997.

[16] Maksymilian Dryja, Barry F. Smith, and Olof B. Widlund. Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer. Anal.*, 31(6):1662–1694, December 1994.

[17] Charbel Farhat, Po-Shu Chen, and Jan Mandel. A scalable Lagrange multiplier based domain decomposition method for time-dependent problems. *Int. J. Numer. Meth. Eng.*, 38:3831–3853, 1995.

[18] Charbel Farhat, Po-Shu Chen, and François-Xavier Roux. The two-level FETI method - part II: Extensions to shell problems, parallel implementation, and performance results. *Comput. Methods Appl. Mech. Eng.*, 155:153–180, 1998.

[19] Charbel Farhat, Michel Lesoinne, Patrick Le Tallec, Kendall Pierson, and Daniel Rixen. FETI-DP: A Dual-Primal unified FETI method – part I: A faster alternative to the two-level FETI method. Technical Report U–CAS–99–15, University of Colorado at Boulder, Center for Aerospace Structures, August 1999. To appear in Int. J. Numer. Meth. Eng.

[20] Charbel Farhat, Michel Lesoinne, and Kendall Pierson. A scalable Dual-Primal domain decomposition method. Technical report, University of Colorado at Boulder, Center for Aerospace Structures, 2000. To appear in Numer. Lin. Alg. Appl.

[21] Charbel Farhat, Antonini P. Macedo, and Michel Lesoinne. A two-level domain decomposition method for the iterative solution of high frequency exterior Helmholtz problems. *Numer. Math.*, 85:283–308, 1999.

[22] Charbel Farhat, Antonini P. Macedo, Michel Lesoinne, François-Xavier Roux, Frédéric Magoulès, and Armel de La Bourdonnaie. A non-overlapping domain decomposition method for the exterior Helmholtz problem. In Jan Mandel, Charbel Farhat, and Xiao-Chuan Cai, editors, *Tenth International Conference of Domain Decomposition Methods*, volume 218 of *Contemporary Mathematics*, pages 42–66. AMS, 1998.

[23] Charbel Farhat and Jan 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. Eng.*, 155:129–152, 1998.

[24] Charbel Farhat, Jan Mandel, and François-Xavier Roux. Optimal convergence properties of the FETI domain decomposition method. *Comput. Methods Appl. Mech. Eng.*, 115:367–388, 1994.

[25] Charbel Farhat and François-Xavier Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *Internat. J. Numer. Meth. Eng.*, 32:1205–1227, 1991.

[26] Charbel Farhat and François-Xavier Roux. Implicit parallel processing in structural mechanics. In J. Tinsley Oden, editor, *Computational Mechanics Advances*, volume 2 (1), pages 1–124. North-Holland, 1994.

[27] Leopoldo Franca, Charbel Farhat, Antonini P. Macedo, and Michel Lesoinne. Residual-free bubbles for the Helmholtz equation. *Internat. J. Numer. Meth. Eng.*, 40:4003–4009, 1997.

[28] Leopoldo Franca and Antonini P. Macedo. A two-level finite element method and its application to the Helmholtz equation. *Internat. J. Numer. Meth. Eng.*, 43:23–32, 1998.

[29] R. H. W. Hoppe, Y. Iliash, Y. Kuznetsov, Y. Vassilevski, and B.I. Wohlmuth. Analysis and parallel implementation of adaptive mortar finite element methods. *East-West J. Numer. Math.*, 6(4):223–248, 1998.

[30] Ronald H.W. Hoppe. Mortar edge element methods in R^3. *East-West J. Numer. Math.*, 7(3):159–173, 1999.

[31] Axel Klawonn and Olof B. Widlund. A domain decomposition method with Lagrange multipliers for linear elasticity. Technical Report 780, Computer Science Department, Courant Institute of Mathematical Sciences, February 1999. To appear in SIAM J. Sci. Comput.

[32] Axel Klawonn and Olof B. Widlund. FETI and Neumann–Neumann iterative substructuring methods: Connections and new results. Technical Report 796, Computer Science Department, Courant Institute of Mathematical Sciences, December 1999. To appear in Comm. Pure Appl. Math.

[33] Axel Klawonn and Olof B. Widlund. FETI-DP methods for three-dimensional elliptic problems with heterogeneous coefficients. Technical report, Computer Science Department, Courant Institute of Mathematical Sciences, 2000. In preparation.

[34] Catherine Lacour. *Analyse et Resolution Numérique de Méthodes de Sous-Domaines Non Conformes pour des Problèmes de Plaques.* PhD thesis, Université Pierre et Marie Curie, Paris, 1997.

[35] Catherine Lacour. Iterative substructuring preconditioners for the mortar finite element method. In Petter Bjørstad, Magne Espedal, and David Keyes, editors, *Ninth International Conference of Domain Decomposition Methods*, 1997.

[36] Catherine Lacour and Yvon Maday. Two different approaches for matching nonconforming grids: the mortar element method and the FETI method. *BIT*, 37:720–738, 1997.

[37] Patrick Le Tallec. Neumann-Neumann domain decomposition algorithms for solving 2D elliptic problems with nonmatching grids. *East-West J. Numer. Math.*, 1(2):129–146, 1993.

[38] Patrick Le Tallec, Taoufik Sassi, and Marina Vidrascu. Three-dimensional domain decomposition methods with nonmatching grids and unstructured grid solvers. In David E. Keyes and Jinchao Xu, editors, *Seventh International Conference of Domain Decomposition Methods in Scientific and Engineering Computing*, volume 180 of *Contemporary Mathematics*, pages 61–74. AMS, 1994. Held at Penn State University, October 27-30, 1993.

[39] Jan Mandel. Balancing domain decomposition. *Comm. Numer. Meth. Engrg.*, 9:233–241, 1993.

[40] Jan Mandel and Marian Brezina. Balancing domain decomposition: Theory and computations in two and three dimensions. Technical Report UCD/CCM 2, Center for Computational

Mathematics, University of Colorado at Denver, 1993.

[41] Jan Mandel and Radek Tezaur. Convergence of a substructuring method with Lagrange multipliers. *Numer. Math.*, 73:473–487, 1996.

[42] Jan Mandel, Radek Tezaur, and Charbel Farhat. A scalable substructuring method by Lagrange multipliers for plate bending problems. *SIAM J. Numer. Anal.*, 36:1370–1391, 1999.

[43] Francesca Rapetti and Andrea Toselli. A FETI preconditioner for two dimensional edge element approximations of maxwell's equations on non-matching grids. Technical Report 797, Computer Science Department, Courant Institute of Mathematical Sciences, January 2000.

[44] Daniel Rixen and Charbel Farhat. Preconditioning the FETI method for problems with intra- and inter-subdomain coefficient jumps. In Petter Bjørstad, Magne Espedal, and David Keyes, editors, *Ninth International Conference of Domain Decomposition Methods*, 1997.

[45] Daniel Rixen and Charbel Farhat. A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems. *Int. J. Numer. Meth. Engng.*, 44:489–516, 1999.

[46] Padmanabhan Seshaiyer and Manil Suri. Uniform *hp* convergence results for the mortar finite element method. *Math. Comp.*, 69:481–500, 1997.

[47] Dan Stefanica. *Domain Decomposition Methods for Mortar Finite Elements.* PhD thesis, Courant Institute of Mathematical Sciences, September 1999. Technical Report, Department of Computer Science, Courant Institute of Mathematical Sciences, New York University.

[48] Dan Stefanica and Axel Klawonn. The FETI method for mortar finite elements. In C-H. Lai, P. Bjørstad, M. Cross, and O. Widlund, editors, *Eleventh International Conference on Domain Decomposition Methods*, pages 121–129. ddm.org, 1999.

[49] Radek Tezaur. *Analysis of Lagrange multiplier based Domain Decomposition Methods.* PhD thesis, University of Colorado at Denver, 1998.

[50] Andrea Toselli and Axel Klawonn. A FETI domain decomposition method for Maxwell's equations with discontinuous coefficients in two dimensions. Technical Report 788, Computer Science Department, Courant Institute of Mathematical Sciences, September 1999.

[51] Christian Wieners and Barbara Wohlmuth. A general framework for multigrid methods for mortar finite elements. Technical Report 415, Math.-Nat. Fakultät, Universität Augsburg, 1999.

[52] Barbara Wohlmuth. A mortar finite element method using dual spaces for the Lagrange multiplier. Technical Report 407, Math.-Nat. Fakultät, Universität Augsburg, 1998.

[53] Barbara Wohlmuth. Multigrid methods for saddlepoint problems arising from mortar finite element discretizations. Technical Report 413, Math.-Nat. Fakultät, Universität Augsburg, 1999.

[54] Barbara I. Wohlmuth. Discretization methods and iterative solvers based on domain decomposition, November 1999. Mathematisch–Naturwissenschaftliche Fakultät, Universität Augsburg, Habilitationsschrift.