NASA-CR-199040

CU-CSSC-92-15

CENTER FOR SPACE STRUCTURES AND CONTROLS

A TRANSIENT FETI METHODOLOGY FOR LARGE-SCALE PARALLEL IMPLICIT COMPUTATIONS IN STRUCTURAL MECHANICS

by

C. FARHAT, L. CRIVELLI AND F. X. ROUX

NOVEMBER 1992

COLLEGE OF ENGINEERING
UNIVERSITY OF COLORADO
CAMPUS BOX 429
BOULDER, COLORADO 80309

A TRANSIENT FETI METHODOLOGY FOR LARGE-SCALE PARALLEL IMPLICIT COMPUTATIONS IN STRUCTURAL MECHANICS

Charbel FARHAT and Luis CRIVELLI
Department of Aerospace Engineering Sciences
and Center for Space Structures and Controls
University of Colorado at Boulder
Boulder, CO 80309-0429, U. S. A.

and

Francois-Xavier ROUX
O. N. E. R. A. Groupe Calcul Parallele
29 Av. de la Division Leclerc
BP72 92322 CHATILLON Cedex, FRANCE

Explicit codes are often used to simulate the nonlinear dynamics of large-scale structural systems, even for low frequency response, because the storage and CPU requirements entailed by the repeated factorizations traditionally found in implicit codes rapidly overwhelm the available computing resources. With the advent of parallel processing, this trend is accelerating because explicit schemes are also easier to parallelize than implicit ones. However, the time step restriction imposed by the Courant stability condition on all explicit schemes cannot yet — and perhaps will never — be offset by the speed of parallel hardware. Therefore, it is essential to develop efficient and robust alternatives to direct methods that are also amenable to massively parallel processing because implicit codes using unconditionally stable time-integration algorithms are computationally more efficient when simulating low-frequency dynamics. Here we present a domain decomposition method for implicit schemes that requires significantly less storage than factorization algorithms, that is several times faster than other popular direct and iterative methods, that can be easily implemented on both shared and local memory parallel processors, and that is both computationally and communication-wise efficient. The proposed transient domain decomposition method is an extension of the method of Finite Element Tearing and Interconnecting (FETI) developed by Farhat and Roux for the solution of static problems. Serial and parallel performance results on the CRAY Y-MP/8 and the iPSC-860/128 systems are reported and analyzed for realistic structural dynamics problems. These results establish the superiority of the FETI method over both the serial/parallel conjugate gradient algorithm with diagonal scaling and the serial/parallel direct method, and contrast the computational power of the iPSC-860/128 parallel processor with that of the CRAY Y-MP/8 system.

1. Introduction

Nonlinear transient finite element problems in structural mechanics are characterized by the semi-discrete equations of dynamic equilibrium:

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{f}^{int}(\mathbf{u}, \mathbf{p}_c, \theta) = \mathbf{f}^{ext} \tag{1}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{u}$ is the vector of nodal displacements, a dot superscript indicates a time derivative, $\mathbf{f}^{int}$ is the vector of internal nodal forces, $\mathbf{p}_c$ denotes a set of control parameters, θ is a function of past history of the generalized deformation gradients, and $\mathbf{f}^{ext}$ is the vector of external nodal forces. The solution of Eq. (1) via an implicit time-integration methodology generates a nonlinear algebraic system of equations at each time step. The Newton-Raphson method and its numerous variants collectively known as "Newton-like" methods are the most popular strategies for solving the resulting nonlinear problem. All of these algorithms require the solution of a linear algebraic system of equations of the form:

$$\begin{aligned} \widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\, \Delta\mathbf{u}_{n+1}^{(k+1)} &= \mathbf{r}(\mathbf{u}_{n+1}^{(k)}) \\ &\text{where} \\ \Delta\mathbf{u}_{n+1}^{(k+1)} &= \mathbf{u}_{n+1}^{(k+1)} - \mathbf{u}_{n+1}^{(k)} \end{aligned} \tag{2}$$

and where the subscript n refers to the n-th time step, the superscript k refers to the k-th nonlinear iteration within the current time step, $\widetilde{\mathbf{K}}^t$ is a symmetric positive approximate tangent matrix that includes both mass and stiffness contributions, and $\Delta\mathbf{u}_{n+1}^{(k+1)}$ and $\mathbf{r}(\mathbf{u}_{n+1}^{(k)})$ are respectively the vector of nodal displacement increments and the vector of out-of-balance nodal forces (dynamic residuals). Most if not all finite element production codes use a direct method — that is, a method based on a factorization scheme — for solving the linearized problem (2). However, for large-scale three-dimensional structural problems, direct solvers entail memory and CPU requirements that rapidly overwhelm the largest of the computing resources that are currently available. As a result, even low frequency dynamics problems are often solved with explicit codes in order to avoid the extreme demands placed on computer resources by the repeated factorizations found in implicit time-integration methodologies. This issue has been well addressed by Hughes, Ferencz and Hallquist [1] who have proposed the well celebrated Element-By-Element (EBE) Preconditioned Conjugate Gradient (PCG) algorithm (see also Hughes, Levit and Winget [2]) as a means to alleviate the difficulties associated with direct solution methods. Moreover, with the advent of parallel processing, the popularity of explicit schemes and iterative solution strategies has been rapidly increasing (see, for example, Hajjar and Abel [3],

Farhat, Sobh and Park [4], Belytschko, Plaskacz, Kennedy and Greenwell [5], and Biffle [6]) because these methodologies (a) are easier to parallelize than implicit schemes and direct solvers, and (b) they usually induce short range interprocessor communications that are relatively inexpensive, while the factorization methods used in most implicit schemes induce long range interprocessor communications that often ruin the sought-after speed-up. However, the time step restriction imposed by the Courant stability condition on all explicit schemes cannot yet — and perhaps will never — be offset by the speed of parallel hardware, and many iterative solvers often fail when applied to systems arising from the analysis of large-scale flexible structures. Therefore, it is essential to develop efficient and robust alternatives to direct methods that are also amenable to massively parallel processing, because unconditionally stable implicit schemes are computationally more efficient than explicit time-integration methodologies at simulating low-frequency dynamics. The EBE/PCG algorithm developed by Hughes and his co-workers [1, 2] is such an alternative which additionally utilizes the structure inherent in finite element formulations and implementations. Here, we propose another alternative which is driven by substructuring concepts and which achieves a greater improvement over the CG algorithm with diagonal scaling than reported in [1] for the EBE/PCG scheme. However, unlike EBE methods, the proposed methodology requires the assembly and factorization of substructure level matrices and therefore requires more storage than the EBE/PCG solution algorithm.

A good balance between direct and iterative solution algorithms is provided by Domain Decomposition (DD) methods which usually blend both of these solution strategies. A well designed DD method requires less storage than direct solvers and converges faster than other purely iterative algorithms when applied to highly ill-conditioned systems (see, for example, the collection of papers compiled in [7]). Moreover, in their simplest form, DD methods have an explicit character because they effectively share information only between neighboring subdomains, which makes them very attractive for parallel processing. The method of Finite Element Tearing and Interconnecting (FETI) is a DD algorithm based on a hybrid variational principle that was developed by Farhat and Roux [8] for the parallel solution of self-adjoint elliptic partial differential equations. It combines a direct solver for computing the incomplete subdomain displacement fields with a PCG algorithm for extracting the dual tractions at the subdomain interfaces. This method was shown to outperform direct solvers on both serial and coarse-grained multiprocessors such as the CRAY Y-MP/8 system, and to compare favorably with other DD algorithms on a 32 processor iPSC-2 (Farhat and Roux [9]). In this paper, we present an extension of the FETI methodology to structural dynamics

problems. In particular, we construct two subdomain-by-subdomain preconditioners for the interface problem which have direct mechanical interpretations. We mathematically discuss their computational properties and numerically assess their relative performances. We also report on some recent developments in the FETI process pertaining to global residual estimators, loss of orthogonality, and numerical scalability with respect to the mesh and subdomain sizes. Finally, we analyze some serial and parallel performance results obtained on the CRAY Y-MP/8 and the iPSC-860/128 parallel processors — which are representative of today's two extreme parallel architectures — for all of the transient FETI method, a parallel active column direct solver, and a parallel implementation of the CG algorithm with diagonal scaling. These performance results show that all of the mentioned parallel algorithms exhibit a different type of bottleneck, but in most cases, the FETI method is shown to significantly outperform both the parallel CG algorithm with diagonal scaling and the parallel direct solver.

2. A transient FETI methodology

2.1. Tearing and interconnecting

Let Ω denote the volume of the structure to be analyzed and $\{\Omega^s\}_{s=1}^{s=N_s}$ denote a partitioning (tearing) of Ω into N_s substructures or subdomains. We denote by Γ_t^s and $\Gamma_I^{s,q}$, respectively, the portion of the boundary of Ω^s where surface tractions $\bar{t}^s$ are prescribed, and the interface boundary between Ω^s and a neighboring Ω^q. A weak form of the equations that govern the dynamic undamped response of the global structure can be written in a subdomain-by-subdomain form as:

$$\begin{aligned}\int_{\Omega^s} (\delta u^s \rho^s \ddot{u}^s + \delta \mathcal{E}^s \sigma^s)\, d\Omega \;=\; & \int_{\Omega^s} \delta u^s f^s d\Omega \;+\; \int_{\Gamma_t^s} \delta u^s \bar{t}^s d\Gamma \\ & - \sum_{\overline{\Omega}^s \bigcap \overline{\Omega}^q \neq \phi} \int_{\Gamma_I^{s,q}} \delta u^s \lambda^{s,q} d\Gamma \qquad (3) \\ & (s = 1, ..., N_s)\end{aligned}$$

subject to the inter-substructure continuity constraints:

$$u^s \;=\; u^q;\;\; \dot{u}^s \;=\; \dot{u}^q;\;\; \ddot{u}^s \;=\; \ddot{u}^q \qquad \text{on} \bigcup_{s=1,q=1}^{s=N_s,q=N_s} \{\overline{\Omega}^s \bigcap \overline{\Omega}^q\} \qquad (4)$$

In Eqs. (3) and (4) above, the superscripts ^{s} and ^{q} refer to Ω^s and Ω^q, respectively, a dot indicates a derivative with respect to time, σ^s and $\mathcal{E}^s$ are the

stress and strain tensors, ρ^s is the mass density, u is the displacement field, f is the forcing field, and $\lambda^{s,q}$ is a Lagrange multiplier function which represents the interface tractions that maintain equilibrium between Ω^s and a neighboring Ω^q (interconnecting). The first of the inter-substructure continuity constraints (Eqs. (4)) can be dualized as:

$$\int_{\overline{\Omega}^s \bigcap \overline{\Omega}^q} \delta\lambda^{s,q}(u^s - u^q)\, d\Gamma \;=\; 0 \tag{5}$$

If the original mesh of the global structure does not contain incompatible elements, the subdomains $\{\Omega^s\}_{s=1}^{s=N_s}$ are guaranteed to have compatible interfaces since these subdomains are obtained by partitioning the global mesh into N_s submeshes. Therefore, we assume in the sequel that discrete Lagrange multipliers are introduced at the subdomain interfaces to enforce the inter-substructure displacement continuity. The piece-wise continuous approximation of the Lagrange multipliers $\lambda^{s,q}$ in Eq. (5) is treated in Farhat and Geradin [10] for static problems. Using a standard Galerkin procedure where the displacement field is approximated by suitable shape functions as:

$$u^s \;=\; \mathbf{N}\mathbf{u}^s \tag{6}$$

and linearizing the equations of dynamic equilibrium around $\mathbf{u}_{n+1}$, Eqs. (3) and (5) are transformed into the following algebraic system:

$$\begin{aligned} \mathbf{M}^s \Delta\ddot{\mathbf{u}}_{n+1}^{s^{(k+1)}} + \mathbf{K}^{t^s} \Delta\mathbf{u}_{n+1}^{s^{(k+1)}} \;&=\; \mathbf{r}_{n+1}^{s^{(k)}} - \mathbf{B}^{s^T} \boldsymbol{\lambda}_{n+1}^{(k+1)} \qquad\qquad s = 1,\ ...,\ N_s \\ \sum_{s=1}^{s=N_s} \mathbf{B}^s \Delta\mathbf{u}_{n+1}^{s^{(k+1)}} \;&=\; 0 \end{aligned} \tag{7}$$

where the superscript ^{T} indicates a transpose, $\mathbf{M}^s$ and $\mathbf{K}^{t^s}$ are respectively the subdomain mass and tangent stiffness matrices, $\Delta\mathbf{u}_{n+1}^{s^{(k+1)}}$ and $\mathbf{r}_{n+1}^{s^{(k)}}$ are respectively the subdomain vector of nodal displacement increments and the subdomain vector of out-of-balance nodal forces, $\mathbf{B}^s$ is a boolean matrix that describes how $\overline{\Omega}^s$ is connected to the global interface, and $\boldsymbol{\lambda}_{n+1}^{(k+1)}$ is the vector of Lagrange multiplier unknowns at iteration $k+1$ and step $n+1$. For linear elastostatic problems, Eqs. (7) above corresponds to the discretization of a saddle-point variational principle whose mathematical and computational properties are analyzed in Farhat and Roux [8, 9].

2.2. Time integration

The linearized subdomain equations of equilibrium (7) can be rewritten in compact form as:

$$\begin{bmatrix} \overline{\mathbf{M}} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\ddot{\mathbf{u}}_{n+1}^{(k+1)} \\ \ddot{\boldsymbol{\lambda}}_{n+1}^{(k+1)} \end{bmatrix} + \begin{bmatrix} \overline{\mathbf{K}}^t & \overline{\mathbf{B}}^T \\ \overline{\mathbf{B}} & 0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_{n+1}^{(k+1)} \\ \boldsymbol{\lambda}_{n+1}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{n+1}^{(k)} \\ 0 \end{bmatrix}$$

where

$$\begin{aligned} \overline{\mathbf{M}} &= \text{block diagonal}\left[\mathbf{M}^{(1)} \quad \ldots \quad \mathbf{M}^{(N_s)}\right] \\ \overline{\mathbf{K}}^t &= \text{block diagonal}\left[\mathbf{K}^{t^{(1)}} \quad \ldots \quad \mathbf{K}^{t^{(N_s)}}\right] \\ \overline{\mathbf{B}} &= \left[\mathbf{B}^{(1)} \quad \ldots \quad \mathbf{B}^{(N_s)}\right] \\ \Delta\mathbf{u}_{n+1}^{(k+1)} &= \left[\Delta\mathbf{u}_{n+1}^{1^{(k+1)}} \quad \ldots \quad \Delta\mathbf{u}_{n+1}^{N_s^{(k+1)}}\right]^T \\ \mathbf{r}_{n+1}^{(k)} &= \left[\mathbf{r}_{n+1}^{1^{(k)}} \quad \ldots \quad \mathbf{r}_{n+1}^{N_s^{(k)}}\right]^T \end{aligned} \tag{8}$$

In a companion paper [11], we show that the constraint equation $\overline{\mathbf{B}}\,\Delta\mathbf{u}_{n+1}^{(k+1)} = 0$ introduces a destabilizing effect in the system that can be analyzed by investigating the behavior of the time-integration algorithm at infinity. In particular, we prove that the Newmark integrator without numerical dissipation ($\beta = 1/4,\ \gamma = 1/2$) presents a weak instability which is excited for any time step value. We also show that the Newmark-β damping scheme stabilizes the integration but at the cost of reducing the accuracy to only first order. Finally, we describe in [11] three implicit time-integration algorithms for integrating Eqs. (8) that are unconditionally stable and second-order accurate. These algorithms are:

(A1). A substructure version of the Hilber-Hughes-Taylor [12] algorithm with $\alpha < 0$. The numerical dissipation of this second-order accurate algorithm is shown to cure the weak instability caused by the constraints.

(A2). A substructure version of the Newmark integrator without numerical dissipation ($\beta = 1/4,\ \gamma = 1/2$) but with a constraint on the substructure acceleration increments $\overline{\mathbf{B}}\,\Delta\ddot{\mathbf{u}}_{n+1}^{(k+1)} = 0$, rather than the substructure displacement increments. In particular, we show in [11] that the latter constraint does not cause the inter-substructure displacement constraint to drift away.

(A3). A substructure version of the Newmark integrator without numerical dissipation ($\beta = 1/4,\ \gamma = 1/2$) and with a constraint on the substructure displacement increments $\overline{\mathbf{B}}\,\Delta\mathbf{u}_{n+1}^{(k+1)} = 0$, but written in terms

of the displacement field increment $\Delta\mathbf{u}_{n+1}^{(k+1)}$ and the momentum increment $\Delta\mathbf{v}_{n+1}^{(k+1)} = \overline{\mathbf{M}}\ \Delta\dot{\mathbf{u}}_{n+1}^{(k+1)}$. The latter algorithm possesses desirable features for reducing round-off error propagation.

Here, we select the time-integration algorithm **(A3)** and summarize it in order to keep this paper self-contained. Let $\Delta\mathbf{v}_{n+\frac{1}{2}}^{(k+1)}$ denote the momentum increment at iteration $k+1$ and at the midpoint between steps n and $n+1$:

$$\Delta\mathbf{v}_{n+\frac{1}{2}}^{(k+1)} = \overline{\mathbf{M}}\ \Delta\dot{\mathbf{u}}_{n+\frac{1}{2}}^{(k+1)} \tag{9}$$

The midpoint subdomain increments $\Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)}$ and $\Delta\mathbf{v}_{n+\frac{1}{2}}^{s(k+1)}$ are integrated as follows:

$$\begin{aligned} \Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} &= \frac{\Delta t}{2}\Delta\dot{\mathbf{u}}_{n+\frac{1}{2}}^{s(k+1)} \\ \Delta\mathbf{v}_{n+\frac{1}{2}}^{s(k+1)} &= \frac{\Delta t}{2}\Delta\dot{\mathbf{v}}_{n+\frac{1}{2}}^{s(k+1)} \end{aligned} \tag{10}$$

where Δt is the time step. Substituting Eqs. (10) into the dynamic equations of subdomain equilibrium (7) written at the midpoint step $n+\frac{1}{2}$ leads to:

$$\Delta\mathbf{v}_{n+\frac{1}{2}}^{s(k+1)} = \frac{2}{\Delta t}\mathbf{M}^s\Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} \tag{11}$$

and

$$\begin{aligned} (\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} &= \frac{\Delta t^2}{4}(\mathbf{r}_{n+\frac{1}{2}}^{s(k)} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)}) \qquad s = 1, ..., N_s \\ \sum_{s=1}^{s=N_s}\mathbf{B}^s\Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} &= 0 \\ \mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} &= \mathbf{u}_{n+\frac{1}{2}}^{s(k)} + \Delta\mathbf{u}_{n+\frac{1}{2}}^{s(k+1)} \end{aligned} \tag{12}$$

After Eqs. (12) are repeatedly solved for all nonlinear increments, $\mathbf{u}_{n+\frac{1}{2}}^s$ is found and the time derivative of each subdomain momentum can be obtained via back-substitution in Eq. (1) as:

$$\dot{\mathbf{v}}_{n+\frac{1}{2}}^s = \mathbf{f}^{s^{ext}}_{n+\frac{1}{2}} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}} - \mathbf{f}^{s^{int}}(\mathbf{u}_{n+\frac{1}{2}}^s, \mathbf{p}_c, \theta) \tag{13}$$

Finally, the subdomain solutions are advanced as follows:

$$\begin{aligned} \mathbf{u}_{n+1}^s &= 2\mathbf{u}_{n+\frac{1}{2}}^s + \mathbf{u}_n^s \\ \mathbf{v}_{n+1}^s &= \mathbf{v}_{n+\frac{1}{2}}^s + \frac{\Delta t}{2}\dot{\mathbf{v}}_{n+\frac{1}{2}}^s \end{aligned} \tag{14}$$

REMARK 2.2.1. Since the solution $\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)}$ of Eqs. (12) will contain round-off errors, we implicitly invoke the equilibrium condition $\sum_{s=1}^{s=N_s} \mathbf{B}^{s^T} \boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)} = 0$ and directly compute the momentum increment in assembled form as:

$$\dot{\mathbf{v}}_{n+\frac{1}{2}} = \mathbf{f}_{n+\frac{1}{2}}^{ext} - \mathbf{f}^{int}(\mathbf{u}_{n+\frac{1}{2}}, \mathbf{p}_c, \theta) \tag{15}$$

Therefore, our actual code utilizes Eq. (15) rather than Eq. (13).

The above subdomain-by-subdomain time-integration algorithm is second-order accurate and unconditionally stable (see [11] for a proof). Its computational cost is dominated by the solution of Eqs. (12) which can be reduced to the following interface problem:

$$\left(\sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}\right)\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} \tag{16}$$

For large problems and fine mesh decompositions, it is not feasible to explicitly assemble the interface operator:

$$\widetilde{\mathbf{F}}_I^t = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T} \tag{17}$$

whose size is equal to the total number of degrees of freedom on the subdomain interfaces. This implies that a direct method is not attractive to solve Eq. (16). The only efficient numerical method for solving Eq. (16) is that of conjugate gradients because once $\{(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\}_{s=1}^{s=N_s}$ have been factored in parallel, matrix-vector products of the form $\widetilde{\mathbf{F}}_I^t\mathbf{x}$ can be efficiently performed using only parallel subdomain-by-subdomain forward and backward substitutions.

The remainder of this paper focuses on the parallel solution of Eq. (16) via the PCG algorithm.

3. Spectral properties of the interface problem

The interface matrix $\sum_{s=1}^{s=N_s} \mathbf{B}^s\mathbf{K}^{t^{s^{-1}}}\mathbf{B}^{s^T}$ corresponds to the discretization of a *compact* operator which maps the normal components of the stress tensor along the subdomain interfaces, onto the traces on these interfaces of the corresponding

subdomain displacement fields. Recently, Roux [13] has shown that because of this compactness, the eigenvalues of the matrix $\sum_{s=1}^{s=N_s} \mathbf{B}^s \mathbf{K}^{t^s\,-1} \mathbf{B}^{s^T}$ are well separated and have a higher density towards the low end of their spectrum. For a fixed Δt, we can show that $\sum_{s=1}^{s=N_s} \mathbf{B}^s (\mathbf{M}^s + \frac{\Delta t^2}{4} \mathbf{K}^{t^s})^{-1} \mathbf{B}^{s^T}$ has similar eigen properties. The objectives of this section are: (a) to numerically illustrate the spectral properties of the transient interface operator $\sum_{s=1}^{s=N_s} \mathbf{B}^s (\mathbf{M}^s + \frac{\Delta t^2}{4} \mathbf{K}^{t^s})^{-1} \mathbf{B}^{s^T}$, and (b) to highlight the impact of these spectral properties on the convergence rate of the CG algorithm.

Consider the three-dimensional two-subdomain cantilever problem depicted in FIG. 1. The beam has a square cross section and a 10/1 length/height aspect ratio. Two finite element meshes are constructed using 8-node brick elements. The first mesh, M_1, contains 2400 internal degrees of freedom (d.o.f.) and 192 interface d.o.f. The second mesh, M_2, is finer: it contains 16320 internal d.o.f. and 672 interface d.o.f.

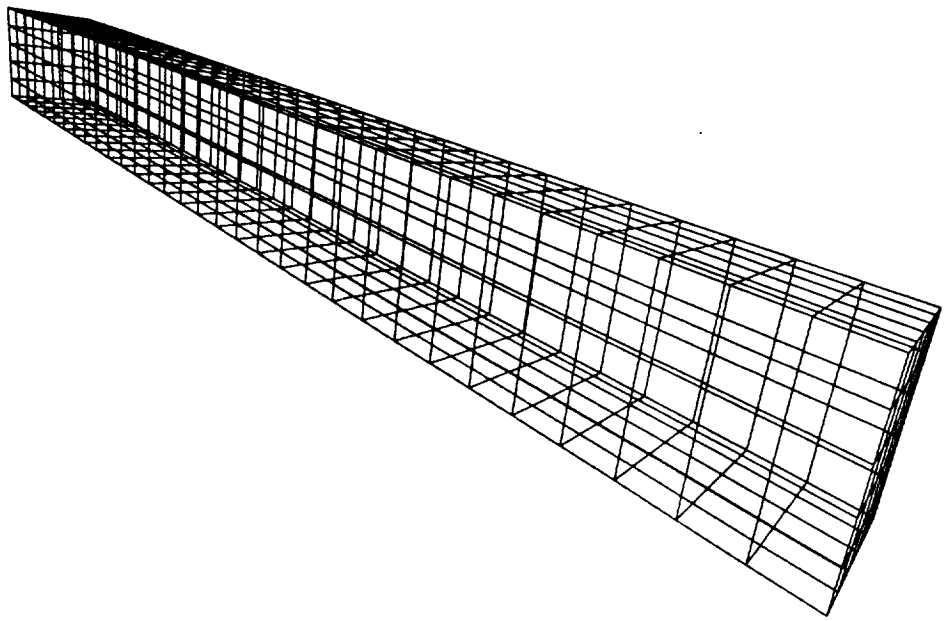

FIG. 1 *A three-dimensional cantilever problem*

The eigenvalues of the transient interface operator $\widetilde{\mathbf{F}}_I^t$ (17) associated with the above problem are computed using a consistent mass formulation and a time step equal to one tenth of the sixth period of the structure. The distribution of these eigenvalues is depicted in FIG. 2 for mesh M_1, and in FIG. 3 for mesh M_2.

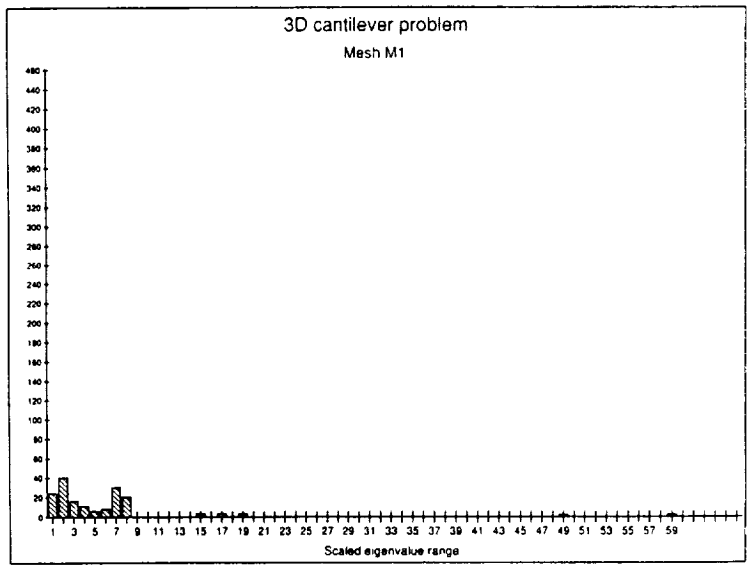

FIG. 2 *Spectral density of* $\widetilde{\mathbf{F}}_I^t$ *for mesh* M_1

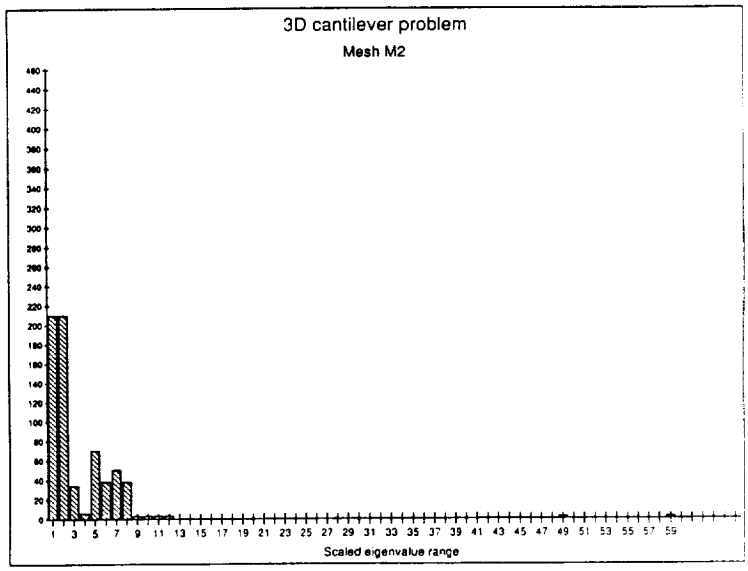

FIG. 3 *Spectral density of* $\widetilde{\mathbf{F}}_I^t$ *for mesh* M_2

The x axis of the bar diagrams depicted in FIG. 2-3 represents the eigenvalues scaled by the smallest eigenvalue, and the y axis the number of eigenvalues per interval. For both meshes, $\widetilde{\mathbf{F}}_I^t$ is shown to have a few large eigenvalues that are well separated from the small ones. Figure 3 suggests that this separation amplifies when the mesh size $h \to 0$. Indeed, one can mathematically prove that the large eigenvalues of $\widetilde{\mathbf{F}}_I^t$ stabilize when the mesh size decreases, while its small eigenvalues accumulate towards zero. Essentially, this is because $\widetilde{\mathbf{F}}_I^t$ involves the inverses of the pencils $(\mathbf{M}^s, \mathbf{K}^{t^s})$ and therefore its high modes correspond to physical modes while its low modes correspond to mesh modes.

The spectral distribution of the interface problem associated with the FETI methodology has important consequences on the rate of convergence of the CG algorithm. During the first iterations, the conjugate gradient algorithm mostly captures the eigenvectors associated with the large eigenvalues. Since the high numerical modes of $\widetilde{\mathbf{F}}_I^t$ correspond to the low physical modes of the structure and since $\widetilde{\mathbf{F}}_I^t$ has only a few relatively high eigenvalues, the CG algorithm applied to the solution of Eq. (16) quickly gives a good approximation of the displacement increments. Intuitively, one can imagine that after the few relatively high modes of the interface operator are captured, the "effective" condition number of $\widetilde{\mathbf{F}}_I^t$ — that is the ratio of its largest uncaptured eigenvalue over its smallest one — becomes significantly smaller than the original condition number of $\widetilde{\mathbf{F}}_I^t$, which accelerates the convergence of the CG algorithm. A detailed analysis of this superconvergence behavior of the CG algorithm in the presence of well separated eigenvalues can be found in the work of van der Luis and van der Vorst [14].

The impact of the spectral distribution of the transient interface operator $\widetilde{\mathbf{F}}_I^t$ on the convergence rate of the CG algorithm is highlighted in TABLE 1 which reports the number of iterations to achieve convergence for the FETI methodology described in this paper and for the classical Schur complement method. The transient interface operator resulting from the latter DD method (static condensation) is denoted here by $\widetilde{\mathbf{S}}_I^t$. While both $\widetilde{\mathbf{F}}_I^t$ and $\widetilde{\mathbf{S}}_I^t$ are shown to have identical two-norm condition numbers ($\kappa_2(\widetilde{\mathbf{F}}_I^t) = \kappa_2(\widetilde{\mathbf{S}}_I^t)$), the CG algorithm applied to $\widetilde{\mathbf{F}}_I^t$ is shown to converge twice as fast as when applied to $\widetilde{\mathbf{S}}_I^t$. This clearly demonstrates that conditioning is not always the only factor governing the convergence rate of the CG algorithm.

TABLE 1

Three-dimensional two-subdomain cantilever problem

Mesh M_1: 2400 internal d.o.f — 192 interface d.o.f.
Mesh M_2: 16320 internal d.o.f — 672 interface d.o.f.

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

mesh	$\kappa_2(\widetilde{\mathbf{F}}_I^t)$	$\kappa_2(\widetilde{\mathbf{S}}_I^t)$	# of iterations (CG-FETI)	# of iterations (CG-Schur)
M_1	58.0	58.0	14	25
M_2	55.2	55.2	14	33

Another important consequence of the spectral distribution of $\widetilde{\mathbf{F}}_I^t$ is that, for a fixed mesh partition, the rate of convergence of CG applied to the solution of Eq. (17) is weakly dependent on h, even though the condition number of $\widetilde{\mathbf{F}}_I^t$ varies as $O(1/h)$. This is because once the few h-dependent high eigenvalues of $\widetilde{\mathbf{F}}_I^t$ have been captured, the superconvergence phenomenon "kicks-in" independently of h. This is clearly demonstrated in TABLE 1 where the CG-FETI algorithm is shown to converge with the same number of iterations for both the coarse mesh M_1 and the fine mesh M_2. Similar results are reported in TABLE 2 for a four-subdomain bending plate problem. The plate is clamped at one end and subjected to a uniform pressure (FIG. 4). It is discretized with 4-node shell elements and with a consistent mass formulation. The time step is set to one tenth of the sixth period of the structure.

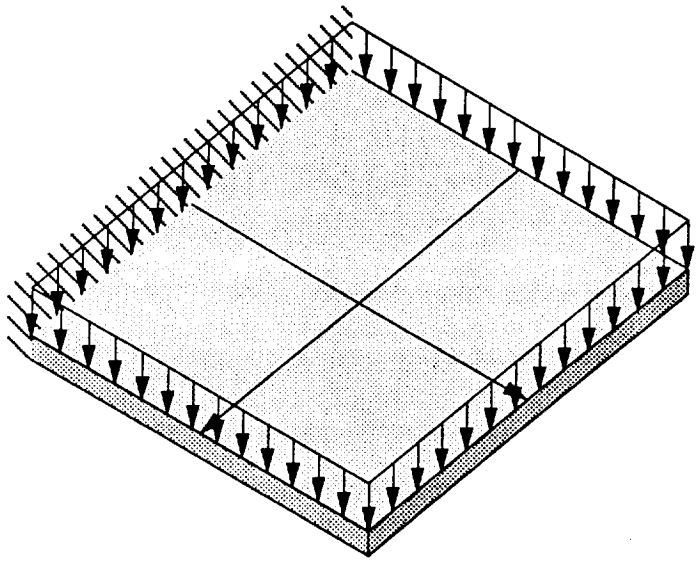

FIG. 4 *A four-subdomain bending plate problem*

TABLE 2

Four-subdomain bending plate problem

Discretization: $N \times N$ where $N = \frac{1}{h}$

Algorithm: CG-FETI

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

h	# of d.o.f.	# of interface d.o.f.	# of iterations
$\frac{1}{10}$	605	55	43
$\frac{1}{20}$	2205	105	54
$\frac{1}{40}$	8405	205	74
$\frac{1}{80}$	32805	405	75
$\frac{1}{160}$	129605	805	75

All of the above results confirm that for all practical purposes and for a fixed mesh partition, the rate of convergence of the CG algorithm applied to the solution of the interface problem associated with the transient FETI methodology can be considered independent of the mesh size h. We refer to this important property as the numerical h scalability of the FETI methodology, where "numerical" is used to differentiate from the well-established parallel scalability of the proposed computational method, and the "h" is used to remind the reader that this result is valid for a fixed mesh partition.

4. Preconditioning with a trace operator

4.1. Sum of the projections of the inverses

Because the interface operator $\widetilde{\mathbf{F}}_I^t = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}$ essentially assembles the projections on the subdomain interfaces of the inverses of the subdomain transient operators $(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})$, it is attractive to consider the following subdomain-by-subdomain parallel preconditioner:

$$\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T} \tag{18}$$

which essentially assembles the projections on the subdomain interfaces of the independent subdomain transient operators $(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})$ themselves. However, we have found that for many static problems the static equivalent of the above preconditioner did not much improve the condition number of the interface problem (Farhat and Roux [8]), and in some cases, it slightly degraded it (Roux [13]). It has also been suggested to us (Mandel [15]) that the condition number of $\widetilde{\mathbf{T}}_I^{t^{-1}}\widetilde{\mathbf{F}}_I^t$ still varies as $O(1/h)$, which should not make $\widetilde{\mathbf{T}}_I^{t^{-1}}$ an attractive preconditioner. Yet, we have recently reported on an extensive set of numerical results for realistic structural problems where the static equivalent of $\widetilde{\mathbf{T}}_I^{t^{-1}}$ was shown to significantly improve the convergence rate of the CG algorithm for static problems (see for example Farhat and Roux [8, 9, 16] and Farhat, Felippa and Militello [17]). All of these observations can be reconciled within the theory of "superconvergence" discussed in Section 3. For this purpose, we consider again the two-subdomain cantilever problem depicted in FIG. 1. The distribution of the eigenvalues of its $\widetilde{\mathbf{T}}_I^{t^{-1}}\widetilde{\mathbf{F}}_I^t$ operator computed using a consistent mass formulation and a time step equal to one tenth of the sixth period of the structure is depicted in FIG. 5 for mesh M_2.

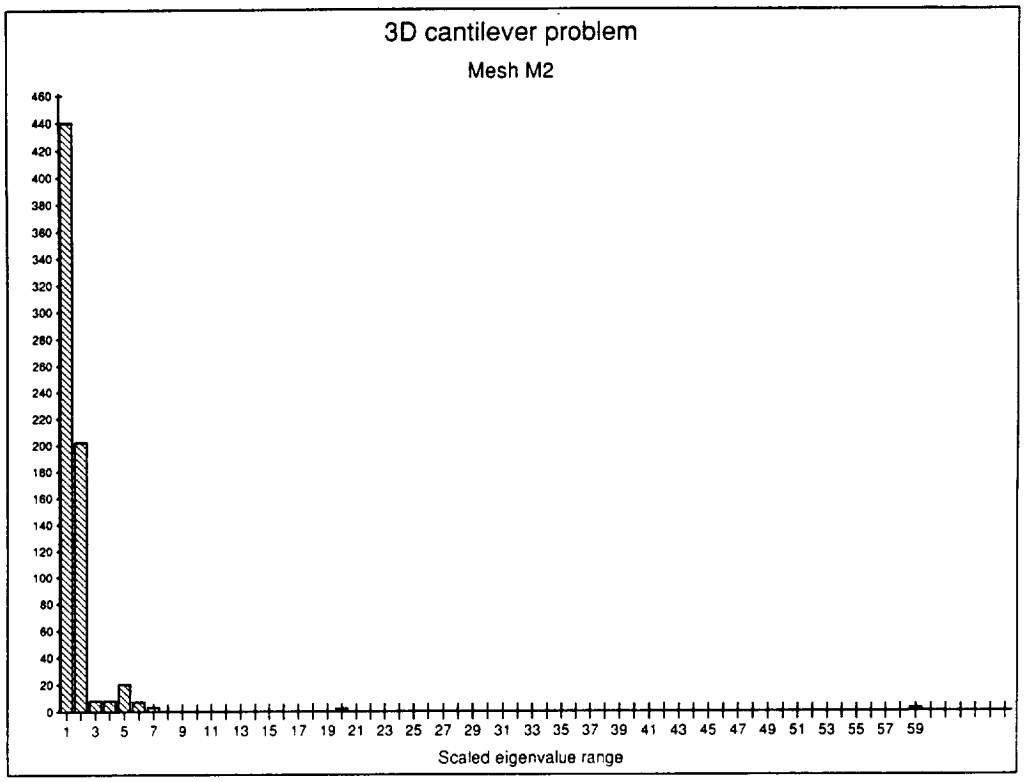

FIG. 5 *Spectral density of* $\widetilde{\mathbf{T}}_I^{t^{-1}} \widetilde{\mathbf{F}}_I^t$ *for mesh* M_2

By comparing FIG. 3 and FIG. 5, the reader can observe that $\mathbf{T}_I^{t^{-1}}$ essentially amplifies the separation between the clusters of small and large eigenvalues of the interface problem, which accelerates the convergence of the CG algorithm. Following the reasoning outlined in Section 3, the reader can also conclude from FIG. 3 and FIG. 5 that after the few large eigenvalues of the interface problem are captured, the "effective" condition number of $\mathbf{T}_I^{t^{-1}} \widetilde{\mathbf{F}}_I^t$ is much smaller than that of $\widetilde{\mathbf{F}}_I^t$. This "superconvergence" behavior is highlighted in TABLE 3 below for the three-dimensional two-subdomain cantilever problem.

TABLE 3

Three-dimensional two-subdomain cantilever problem

Mesh M_1: 2400 internal d.o.f — 192 interface d.o.f.
Mesh M_2: 16320 internal d.o.f — 672 interface d.o.f.

Algorithm: CG-FETI

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

mesh	# of iterations $(\widetilde{\mathbf{F}}_I^t)$	# of iterations $(\widetilde{\mathbf{T}}_I^{t^{-1}}\widetilde{\mathbf{F}}_I^t)$
M_1	14	6
M_2	14	6

4.2 Mechanical interpretation and parallelism

At each step q of the PCG algorithm, the preconditioning phase using $\mathbf{T}_I^{t^{-1}}$ can be written as:

$$\textit{Solve} \quad \widetilde{\mathbf{T}}_I^t \mathbf{z}^q = \bar{\mathbf{r}}^q$$

where

$$\begin{aligned} \bar{\mathbf{r}}^q &= \sum_{s=1}^{s=N_s} \bar{\mathbf{r}}^{s^q} \\ &= \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}(\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q}) \end{aligned} \tag{19}$$

and the solution of Eq. (19) can be written as:

$$\mathbf{z}^q = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}\bar{\mathbf{r}}^{s^q} \tag{20}$$

The q-th residual $\bar{\mathbf{r}}^q$ corresponds to the jump of the displacement increments across the subdomain interfaces (see Section 7.1.). Clearly, the above preconditioner is intrinsically parallel as it involves only subdomain-by-subdomain independent computations. Moreover, it is economical because it only requires

matrix-vector products of sizes equal to the subdomain interfaces. From a mechanical viewpoint, solving Eq. (19) corresponds to finding a set of "equivalent" interface forces that can reproduce the imposed jump of the displacement increments. Therefore, the problem described by Eq. (20) is indeed an approximate inverse of the interface problem.

5. Preconditionig with a dual operator

5.1 Sum of the exact inverses

A better approximation of $\widetilde{\mathbf{F}}_I^{t^{-1}}$ can be obtained by approximating the inverse of the sum by the sum of the exact inverses — that is, by assuming that:

$$[\sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1} \approx \sum_{s=1}^{s=N_s} [\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1} \quad (21)$$

which leads to the following preconditioner:

$$\widetilde{\mathbf{D}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} [\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1} \quad (22)$$

If the subdomain mass and stiffness matrices and the subdomain displacement field are partitioned as:

$$\mathbf{M}^s = \begin{bmatrix} \mathbf{M}_{ii}^s & \mathbf{M}_{ib}^s \\ \mathbf{M}_{ib}^{s^T} & \mathbf{M}_{bb}^s \end{bmatrix}, \quad \mathbf{K}^s = \begin{bmatrix} \mathbf{K}_{ii}^s & \mathbf{K}_{ib}^s \\ \mathbf{K}_{ib}^{s^T} & \mathbf{K}_{bb}^s \end{bmatrix}, \text{ and } \mathbf{u}^s = \begin{bmatrix} \mathbf{u}_i^s \\ \mathbf{u}_b^s \end{bmatrix}$$

where the subscripts $_i$ and $_b$ respectively refer to the internal and interface boundary degrees of freedom, the inverse of $(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})$ can be written as the solution of the partitioned matrix equations:

$$\begin{bmatrix} \mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s & \mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s \\ \mathbf{M}_{ib}^{s^T} + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^{s^T} & \mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s \end{bmatrix} \begin{bmatrix} \mathbf{A}_{ii}^s & \mathbf{A}_{ib}^s \\ \mathbf{A}_{ib}^{T^s} & \mathbf{A}_{bb}^s \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{O} & \mathbf{I} \end{bmatrix} \quad (23)$$

which yields:

$$\mathbf{A}_{bb}^s = [\mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s - (\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)^T(\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)^{-1}(\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)]^{-1} \quad (24)$$

It follows that:

$$[\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1} = [[\mathbf{O} \quad \mathbf{I}] \begin{bmatrix} \mathbf{A}_{ii}^s & \mathbf{A}_{ib}^s \\ \mathbf{A}_{ib}^{T^s} & \mathbf{A}_{bb}^s \end{bmatrix} \begin{bmatrix} \mathbf{O} \\ \mathbf{I} \end{bmatrix}]^{-1} = \mathbf{A}_{bb}^{s^{-1}} \quad (25)$$

which implies that:

$$[\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1}$$
$$= (\mathbf{M}^s_{bb} + \frac{\Delta t^2}{4}\mathbf{K}^s_{bb}) - (\mathbf{M}^s_{ib} + \frac{\Delta t^2}{4}\mathbf{K}^s_{ib})^T(\mathbf{M}^s_{ii} + \frac{\Delta t^2}{4}\mathbf{K}^s_{ii})^{-1}(\mathbf{M}^s_{ib} + \frac{\Delta t^2}{4}\mathbf{K}^s_{ib}) \tag{26}$$

The right hand side of Eq. (26) above is the Schur Complement matrix associated with the subdomain transient operator $(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})$. Therefore, Eq. (26) demonstrates that the preconditioning operator:

$$\widetilde{\mathbf{D}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s}(\mathbf{M}^s_{bb}+\frac{\Delta t^2}{4}\mathbf{K}^s_{bb})-(\mathbf{M}^s_{ib}+\frac{\Delta t^2}{4}\mathbf{K}^s_{ib})^T(\mathbf{M}^s_{ii}+\frac{\Delta t^2}{4}\mathbf{K}^s_{ii})^{-1}(\mathbf{M}^s_{ib}+\frac{\Delta t^2}{4}\mathbf{K}^s_{ib}) \tag{27}$$

and the interface operator $\widetilde{\mathbf{F}}_I^t = \sum_{s=1}^{s=N_s}\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}$ are *dual* operators. Based on this duality and on the mathematical framework presented in Bjordstad and Widlund [19], it can be shown that away from crosspoints — that is, points where more than two subdomains intersect, the preconditioner $\widetilde{\mathbf{D}}_I^{t^{-1}}$ is such that the condition number of $\widetilde{\mathbf{D}}_I^{t^{-1}}\widetilde{\mathbf{F}}_I^t$ is independent of the mesh size h. Moreover, $\mathbf{D}_I^{t^{-1}}$ is also an intrinsically parallel preconditioner since it involves only subdomain-by-subdomain independent computations.

5.2 Mechanical interpretation and computational issues

The solution of the interface problem (16) via the PCG algorithm (here with the dual preconditioner) requires performing at each iteration q the following computations:

$$\begin{aligned} &\textit{Multiply} \quad \bar{\mathbf{p}}_b^q = \widetilde{\mathbf{F}}_I^t\mathbf{p}_b^q \qquad \text{(main loop of CG)} \\ &\textit{Solve} \quad \widetilde{\mathbf{D}}_I^t\mathbf{z}_b^q = \bar{\mathbf{r}}_b^q \qquad \text{(preconditioning step)} \end{aligned} \tag{28}$$

Here, $\mathbf{p}_b^q$ denotes the search direction computed during the q-th iteration, and $\bar{\mathbf{r}}_b^q$ denotes the q-th residual which represents the jump of the displacement increments across the subdomain interfaces. All computations summarized in Eqs.

(28) can be performed at the subdomain level as follows:

$$\begin{aligned}
\bar{\mathbf{p}}_b^{s^q} &= \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}\mathbf{p}_b^{s^q} \\
\mathbf{z}_b^{s^q} &= [(\mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s) \\
&\quad - (\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)^T(\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)^{-1}(\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)]\bar{\mathbf{r}}_b^{s^q}
\end{aligned} \tag{29}$$

The first of Eqs. (29) can be re-formulated as a problem with *Neumann* boundary conditions (indicated between braces {}):

$$\begin{bmatrix} \mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s & \mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s \\ \mathbf{M}_{ib}^{s^T} + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^{s^T} & \mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_i^{s^q} \\ \bar{\mathbf{p}}_b^{s^q} \end{bmatrix} = \begin{bmatrix} 0 \\ \{\mathbf{p}_b^{s^q}\} \end{bmatrix} \tag{30}$$

The solution of this problem $\bar{\mathbf{p}}_b^{s^q}$ represents the trace on the interface boundary of Ω^s of the displacement field resulting from prescribing the interface traction forces $\mathbf{p}_b^{s^q}$.

The second of Eqs. (29) can be re-formulated as a problem with *Dirichlet* boundary conditions (indicated between braces {}):

$$\begin{bmatrix} \mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s & \mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s \\ \mathbf{M}_{ib}^{s^T} + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^{s^T} & \mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s \end{bmatrix} \begin{bmatrix} \bar{\mathbf{z}}_i^{s^q} \\ \{\bar{\mathbf{r}}_b^{s^q}\} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{z}_b^{s^q} \end{bmatrix} \tag{31}$$

and which can be solved in two steps:

$$\begin{aligned}
\textit{Solve} \quad & (\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)\bar{\mathbf{z}}_i^{s^q} = -(\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)\bar{\mathbf{r}}_b^{s^q} \\
\textit{Evaluate} \quad & \mathbf{z}_b^{s^q} = (\mathbf{M}_{ib}^{s^T} + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^{s^T})\bar{\mathbf{z}}_i^{s^q} + (\mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s)\bar{\mathbf{r}}_b^{s^q}
\end{aligned} \tag{32}$$

Therefore, the preconditioning step reformulated in Eq. (31) corresponds to finding in each subdomain Ω^s the traction forces that are needed to prescribe the interface boundary displacement increment jump $\bar{\mathbf{r}}_b^{s^q}$.

The mechanical interpretation of the CG algorithm with the dual preconditioner $\widetilde{\mathbf{D}}_I^t$ is straightforward. Within each iteration, a Neumann problem is first solved: traction forces are applied at the subdomain interfaces, and a jump in the displacement increments is computed. Next, this computed jump is imposed at the subdomain interfaces and new interface traction forces are computed. The

successive iterations drive this jump to zero and upon convergence the computed Lagrange multipliers (traction forces) enforce the inter-subdomain displacement continuity.

On the computational side, it should be noted that even though the block $(\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)$ governing Eq. (30) is embedded in the system governing Eq. (32), this block is stored and factored twice. The reason is that Eqs. (30) and (32) have two different optimal numberings for the internal degrees of freedom that lead to two different matrices. For a small number of subdomains, this drawback becomes a severe issue as the cost of storing and factoring $(\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)$ is important. For finer mesh partitions, this additional storage and computational burden becomes less significant as the size of every subdomain and the cost of factoring $(\mathbf{M}_{ii}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ii}^s)$ becomes negligible when compared to the size of the global interface problem and the cost of the main CG iterations, respectively. In particular, if the tangent operator is not reconstructed at each nonlinear iteration, the computational overhead introduced by the dual preconditioner is acceptable. However, it should also be noted that an iteration with the dual preconditioner is more expensive than an iteration with the trace preconditioner as the former one involves in every subdomain a pair of forward/backward substitutions of sizes equal to the number of internal subdomain degrees of freedom, while the latter one involves in every subdomain a matrix-vector product of a size equal to the number of subdomain interface degrees of freedom.

REMARK 5.2.1. Global and/or local symmetry effects such as mesh and material symmetry, *but not necessarily load symmetry*, can accelerate the convergence of the FETI method with the dual preconditioner. For example, in the case of a two-subdomain problem, symmetry implies that $\widetilde{\mathbf{D}}_I^{t^{-1}} = 4\ \widetilde{\mathbf{F}}_I^{t^{-1}}$, and therefore the PCG algorithm converges in one iteration.

6. Loss of orthogonality

Matrices with the spectral pattern discussed in Section 3 have been shown to cause a rapid loss of orthogonality of the direction vectors (Parlett [18]). For these matrices, increasing the numerical precision does not restore the orthogonality of the search vectors because in this case the propagation of the errors is a polynomial function of the ratio between consecutive eigenvalues.

The loss of orthogonality of the search directions during the solution of the interface problem (16) has a disastrous consequence on the rate of convergence of the PCG algorithm. To remedy this problem, we introduce a reorthogonalization procedure within the PCG algorithm which however, in order to determine the new direction vector, entails at each iteration q the following additional burden:

(a) the storage of the direction vector $\mathbf{p}^q$ and the product $\widetilde{\mathbf{F}}_I^t \mathbf{p}^q$.

(b) the evaluation of q dot products of the form $\mathbf{r}_j^T[\widetilde{\mathbf{F}}_I^t \mathbf{p}^q]$ where $[\widetilde{\mathbf{F}}_I^t \mathbf{p}^q]$ is readily available and $1 \leq j < q$, and of an $n_I \times j$ matrix-vector product where n_I is the number of interface unknowns.

Clearly, such a reorthogonalization procedure is not feasible if introduced during the solution via the PCG algorithm of a global finite element problem, as it would require unreasonable amounts of memory and CPU. However, it is quite affordable within the context of a domain decomposition algorithm as it applies only to the interface problem. In particular, the reader should note that the additional computational costs outlined in (b) are small compared to the cost of the pair of forward and backward substitutions that are required at each iteration q of the PCG algorithm in order to evaluate the product $\widetilde{\mathbf{F}}_I^t \mathbf{p}^q$.

The efficiency of the reorthogonalization procedure discussed above is highlighted in TABLE 4 for the four-subdomain bending plate problem introduced in Section 3.

TABLE 4

Four-subdomain bending plate problem

Algorithm: PCG-FETI

Preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-3}$

Multiprocessor: iPSC-860 (4 processors)

h	# of iterations (without reorth.)	# of iterations (with reorth.)	CPU (without reorth.)	CPU (with reorth.)
$\frac{1}{20}$	42	22	5.5 s.	3.7 s.
$\frac{1}{40}$	66	24	33.2 s.	16.5 s.

REMARK 6.1. In practice, the number of direction vectors that are stored for reorthogonalization is determined by the memory space that is available after all of the other storage requirements of the transient FETI method have been satisfied. When only a few directions can be stored, a partial reorthogonalization

is implemented. In this case, the optimal strategy consists in storing the first few directions instead of the most recent ones, because the subspace generated by the first directions is closer to the subspace associated with the highest eigenvalues.

At this point, we mention that for many of the realistic structural problems that we have solved with the FETI method, we have observed that reorthogonalization was necessary for convergence. All of the numerical results presented in the remainder of this paper were obtained with either partial or full reorthogonalization.

7. Global residual and numerical precision

7.1. Interface v.s. global convergence

At every iteration q of the PCG algorithm applied to the solution of the interface problem (16), the computed $\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q}$ and $\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}$ verify:

$$(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} = \frac{\Delta t^2}{4}(\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q}) \qquad s = 1, ..., N_s \quad (33)$$

so that:

$$\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} = \frac{\Delta t^2}{4}(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} - \frac{\Delta t^2}{4}(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q} \quad (34)$$

Using (34), the jump in the displacement increments at the subdomain interfaces can be written as:

$$\sum_{s=1}^{s=N_s} \mathbf{B}^s\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} = -\mathbf{G}\boldsymbol{\lambda} + \mathbf{L}$$

where

$$\mathbf{G} = \frac{\Delta t^2}{4}\sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T} \quad (35)$$

$$\mathbf{L} = \frac{\Delta t^2}{4}\sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}}$$

From (16) and (35), it is clear that the gradient of the interface problem (16) is indeed the jump in the displacement increments at the subdomain interfaces. Therefore, the residual of the interface problem $|| - \mathbf{G}\boldsymbol{\lambda} + \mathbf{L}||$ gives the order

of magnitude of the error in $||\Delta \mathbf{u}_{n+\frac{1}{2}}||$ and not the order of magnitude of the error in the global residual $||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||$, where $\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)}) = \mathbf{M} + \frac{\Delta t^2}{4}\mathbf{K}^t$. Moreover, since the condition number of $\widetilde{\mathbf{K}}^t$ varies as $O(1/h^2)$ while the condition number of $\widetilde{\mathbf{F}}_I^t$ varies as $O(1/h)$, the convergence criterion:

$$||-\mathbf{G}\boldsymbol{\lambda}^q + \mathbf{L}|| \;\leq\; \epsilon\,||-\mathbf{G}\boldsymbol{\lambda}^0 + \mathbf{L}|| \tag{36}$$

does not guarantee that:

$$||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{(k+1)^q} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})|| \;\leq\; \epsilon\,||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{(k+1)^0} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})|| \tag{37}$$

We have observed that for most structural problems, the relative global residual (37) is typically 10^2 to 10^3 larger than the relative interface residual (36), which clearly indicates that the convergence of the FETI methodology should be based on the global criterion (37).

Unfortunately, evaluating (37) at every PCG iteration q requires performing a global matrix-vector multiply in order to compute the global residual $||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{(k+1)^q} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||$. This would double both the computational and communication costs of a PCG iteration Therefore, we had to develop an estimator for the global residual that is accurate and computationally economical.

Using Eq. (12) and the partitioning introduced in Section 5.1, the restriction of the global residual to every subdomain Ω^s can be written as:

$$\widetilde{\mathbf{K}}^{t^s}(\mathbf{u}_{n+1}^{s^{(k)}})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} - \mathbf{r}(\mathbf{u}_{n+1}^{s^{(k)}}) = \begin{bmatrix} (\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)\,(\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb}) \\ (\mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s)\,(\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb}) \end{bmatrix} \tag{38}$$

where $\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb}$ is the jump in the displacement increments at the subdomain boundary interface. Clearly, the reaction forces $(\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)\,(\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb})$ are zero except on those degrees of freedom which are connected to the interface ones. Moreover, equilibrium suggests that $||(\mathbf{M}_{ib}^s + \frac{\Delta t^2}{4}\mathbf{K}_{ib}^s)\,(\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb})||$ and $||(\mathbf{M}_{bb}^s + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^s)\,(\overline{\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb})||$ are of the same magnitude order, so that one can reasonably approximate $||\widetilde{\mathbf{K}}^{t^s}(\mathbf{u}_{n+1}^{s^{(k)}})\,\Delta \mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} - \mathbf{r}(\mathbf{u}_{n+1}^{s^{(k)}})||$

as follows:

$$
\begin{aligned}
||\widetilde{\mathbf{K}}^{t^s}(\mathbf{u}_{n+1}^{s^{(k)}})\,\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} \;-\; \mathbf{r}(\mathbf{u}_{n+1}^{s^{(k)}})||^{est} \\
&= ||(\mathbf{M}_{bb}^{s} + \frac{\Delta t^2}{4}\mathbf{K}_{bb}^{s})\,(\overline{\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb})|| \\
&= ||\mathbf{B}^{s}(\mathbf{M}^{s} + \frac{\Delta t^2}{4}\mathbf{K}^{s})\mathbf{B}^{s^T}\,(\overline{\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}}_{bb} - \Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}{}_{bb})||
\end{aligned}
\tag{39}
$$

From (35) and (19) it follows that:

$$
\begin{aligned}
\overline{\Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}}} - \Delta\mathbf{u}_{n+\frac{1}{2}}^{s^{(k+1)^q}} &= \frac{\Delta t^2}{4}\mathbf{B}^{s}(\mathbf{M}^{s} + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}(\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q}) \\
&= \frac{\Delta t^2}{4}\bar{\mathbf{r}}^{s^q}
\end{aligned}
\tag{40}
$$

In assembled form, the above estimator becomes:

$$
\begin{aligned}
||\widetilde{\mathbf{K}}^{t}(\mathbf{u}_{n+1}^{(k)})\,\Delta\mathbf{u}_{n+\frac{1}{2}}^{(k+1)^q} \;-\; \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||^{est} \\
&= \frac{\Delta t^2}{4}\sum_{s=1}^{s=N_s}\mathbf{B}^{s}(\mathbf{M}^{s} + \frac{\Delta t^2}{4}\mathbf{K}^{s})\mathbf{B}^{s^T}(\mathbf{M}^{s} + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}(\mathbf{r}_{n+\frac{1}{2}}^{s^{(k)}} - \mathbf{B}^{s^T}\boldsymbol{\lambda}_{n+\frac{1}{2}}^{(k+1)^q}) \\
&= \frac{\Delta t^2}{4}\bar{\mathbf{r}}^{s^q}
\end{aligned}
\tag{41}
$$

which in view of Eqs. (19-20) can be written as:

$$
||\widetilde{\mathbf{K}}^{t}(\mathbf{u}_{n+1}^{(k)})\,\Delta\mathbf{u}_{n+\frac{1}{2}}^{(k+1)^q} \;-\; \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||^{est} \;=\; \frac{\Delta t^2}{4}\mathbf{z}^q \;=\; \frac{\Delta t^2}{4}\widetilde{\mathbf{T}}_I^{t^{-1}}\,\bar{\mathbf{r}}^q \tag{42}
$$

Therefore, we replace the convergence criterion (37) with:

$$
\begin{aligned}
\mathbf{z}^q \;&\leq\; \epsilon\,\mathbf{z}^0 \\
\text{— that is:} \\
\widetilde{\mathbf{T}}_I^{t^{-1}}\,\bar{\mathbf{r}}^q \;&\leq\; \epsilon\,\mathbf{T}_I^{t^{-1}}\bar{\mathbf{r}}^0
\end{aligned}
\tag{43}
$$

Clearly, if the trace preconditioner is used, the proposed estimator for the global residual does not require any additional computation.

The discrepancy between the interface and global residuals and the accuracy of the proposed global residual estimator are demonstrated in FIG. 6 for the four subdomain bending plate problem with $h = 1/40$ (Section 3).

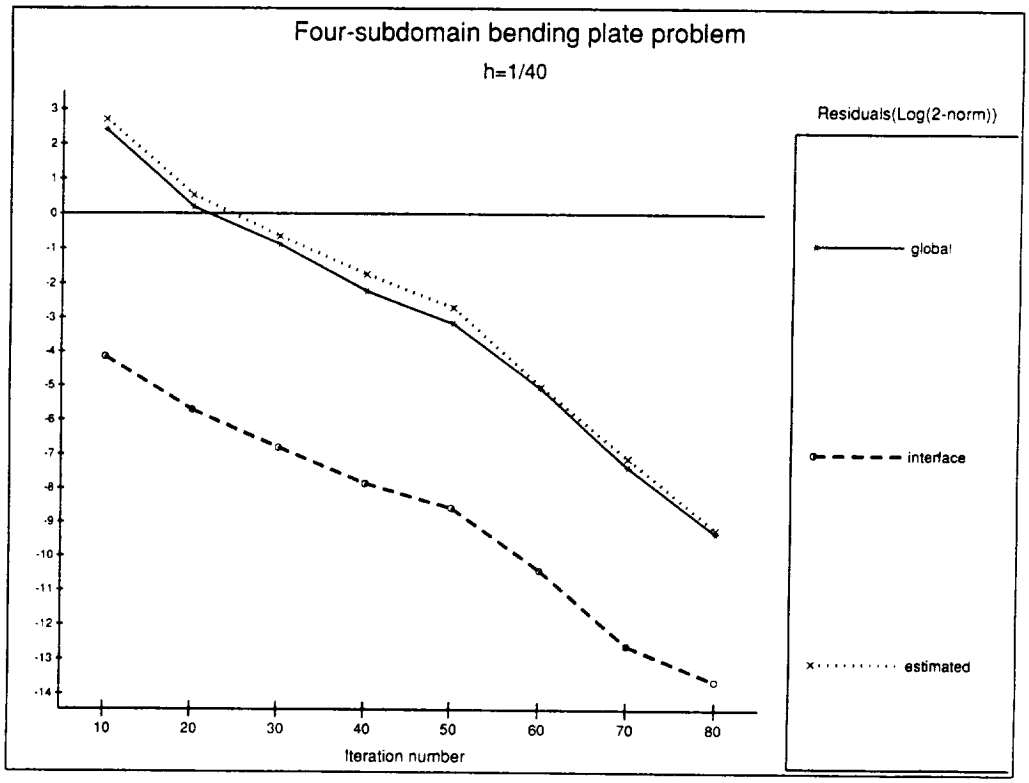

FIG. 6 *Accuracy of the global residual estimator*

7.2. Setting the tolerance for the PCG algorithm

Let $\mathbf{d}_{n+1}$, $\mathbf{u}_{n+1}^k$, and $(\mathbf{u}_{n+1}^k)^\infty$ denote respectively the exact solution of the nonlinear problem (1) at time step $n+1$, the exact solution of the linearized problem at the k-th nonlinear iteration of time step $n+1$, and the PCG solution of the linearized problem at the k-th nonlinear iteration of time step $n+1$. Our stopping criterion for the PCG algorithm is:

$$||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta\mathbf{u}_{n+\frac{1}{2}}^{(k+1)^q} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})|| \leq \epsilon\,||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\,\Delta\mathbf{u}_{n+\frac{1}{2}}^{(k+1)^0} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})|| \quad (44)$$

where q denotes the PCG iteration number. It follows that:

$$||(\mathbf{u}_{n+1}^{k+1})^\infty - \mathbf{u}_{n+1}^{k+1}|| \leq \epsilon\,||\mathbf{u}_{n+1}^k - \mathbf{d}_{n+1}|| \quad (45)$$

On the other hand, a "Newton-like" method implies that:

$$||\mathbf{u}_{n+1}^{k+1} - \mathbf{d}_{n+1}|| \leq C\, ||\mathbf{u}_{n+1}^{k} - \mathbf{d}_{n+1}||^p \tag{46}$$

where C is a real constant, and p is an integer describing the rate of convergence of the "Newton-like" method. For example, $p = 2$ for the Newton-Raphson algorithm, and $p = \frac{1+\sqrt{(5)}}{2} = 1.618$ for the Secant method [20]. From the triangular inequality for norms and from (45-46) we deduce:

$$||(\mathbf{u}_{n+1}^{k+1})^{\infty} - \mathbf{d}_{n+1}|| \leq \epsilon\, ||\mathbf{u}_{n+1}^{k} - \mathbf{d}_{n+1}|| + C\, ||\mathbf{u}_{n+1}^{k} - \mathbf{d}_{n+1}||^p \tag{47}$$

Therefore, if we desire that $||(\mathbf{u}_{n+1}^{k+1})^{\infty} - \mathbf{d}_{n+1}|| \leq \tilde{\epsilon}$, it suffices to take $\epsilon = \tilde{\epsilon}^{\frac{1}{p}}$. For $p = 2$ (Newton-Raphson) this leads to $\epsilon = \sqrt{\tilde{\epsilon}}$, and for $p = 1.618$ (Secant), this leads to $\epsilon = \tilde{\epsilon}^{0.618}$.

In the sequel, we fix $\tilde{\epsilon} = 10^{-6}$. Therefore, we choose $\epsilon = 10^{-4}$, which accommodates both the Newton-Raphson and Secant methods.

8. Serial performance analysis and numerical H non-scalability

8.1 Nonlinear transient analysis of a space antenna connector

As a first large-scale example, we consider the nonlinear transient analysis of a mechanical joint designed for connecting three space antennaes. The corresponding finite element (FIG. 7) mesh contains 11284 nodes, 21888 3-node shell elements with 6 d.o.f. per node, and a total of 67704 d.o.f. The time step Δt is chosen as $T_3/15 = 0.0123s.$, where T_3 denotes the third fundamental period of the structure. After the nodes are renumbered with the Reverse Cuthill-McKee (RCM) scheme [21], the average column height of the skyline of the undecomposed matrix $(\mathbf{M} + \frac{\Delta t^2}{4}\mathbf{K}^t)$ is 616 and its envelope size is 41,674,976. All computations discussed in this section are carried out on a CRAY Y-MP processor, and all performance results are reported for the solution of the displacement increments (Eqs. (12)) in one nonlinear step.

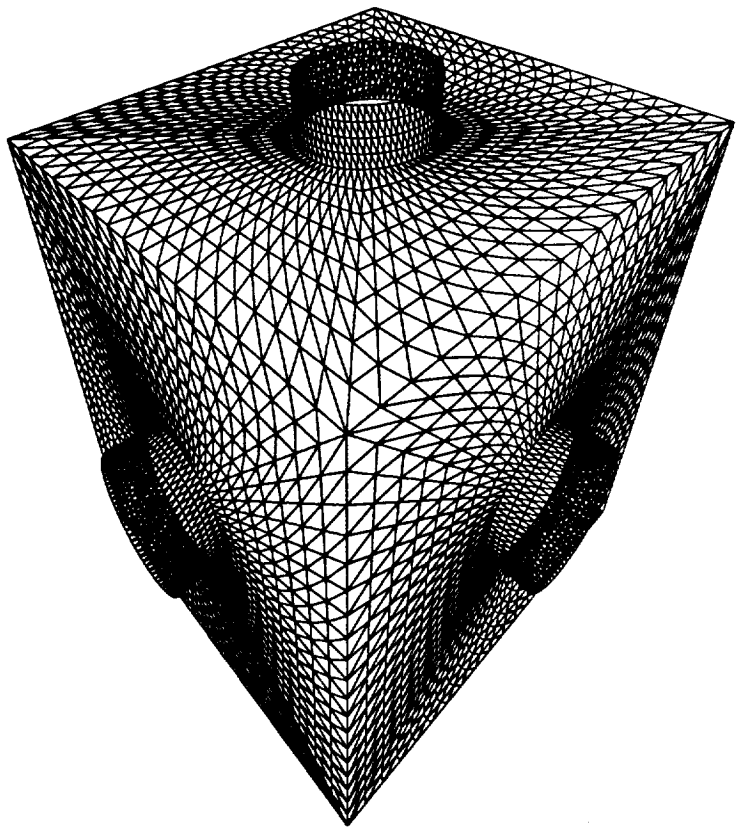

FIG. 7 *Finite element discretization of a space antenna connector (from a finer mesh)*

8.2 Reference serial performance results

First, performance results are reported for the direct method (LDL^T factorization) and the CG algorithm with diagonal scaling (Jacobi-PCG) applied to the un-decomposed problem (TABLE 5). These results will later serve as reference serial performance results. Memory consumption is measured in millions of 64 bit words (MW). MFLOPS (Million FLoating-point OPerations per Second) are reported in order to distinguish and independently assess the numerical and implementational performances. A sparse data structure is used for the Jacobi-PCG algorithm.

TABLE 5

Space antenna connector - Reference serial performance results

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s$.

Convergence criterion for Jacobi-PCG: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

CRAY Y-MP (single processor)

algorithm	memory	# of iterations	MFLOPS	CPU
direct (LDL^T)	42 MW	1	220	253.0 s.
Jacobi-PCG	4.2 MW	3320	80	345.6 s.

It is interesting to note that for the above problem, the direct method on the CRAY Y-MP outperforms the Jacobi-PCG algorithm essentially because it vectorizes better. The MFLOP rate of LDL^T is 2.75 times better than that of Jacobi-PCG, but its solution time is only 1.36 times faster.

8.3 FETI performance results

Next, we report on the performance results of the FETI method with the trace (TABLE 6) and dual (TABLE 7) preconditioners, for various numbers of subdomains. The growth of the interface problem with the number of subdomains is depicted in FIG. 8. All local subdomain problems are solved using the same implementation of the LDL^T algorithm as in the un-decomposed case (TABLE 5). FAC designates the computational phase where the subdomain operators are factored. All mesh partitions are obtained using the mesh decomposer described in Farhat [22].

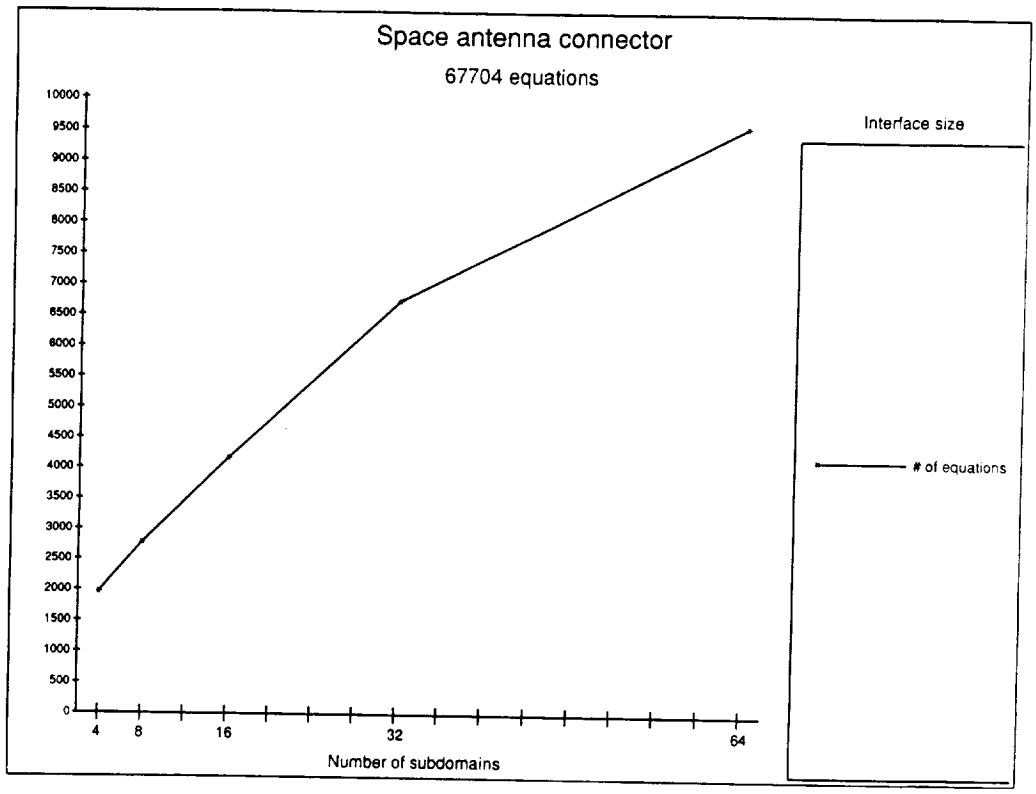

FIG. 8 *Growth of the interface problem*

TABLE 6

Space antenna connector - FETI($\widetilde{\mathbf{T}}_I^{t^{-1}}$) *serial performance results*

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s.$

Preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

CRAY Y-MP (single processor)

N_s	memory	CPU FAC	MFLOPS FAC	# of itr.	CPU /itr. PCG	CPU PCG	MFLOPS PCG	CPU TOT
4	18 MW	19.0 s.	165	159	0.43 s.	68.4 s.	130	87.4 s.
8	16 MW	13.6 s.	150	200	0.39 s.	78.0 s.	110	91.6 s.
16	12 MW	9.6 s.	130	267	0.36 s.	96.1 s.	105	105.7 s.
32	10 MW	6.7 s.	105	456	0.34 s.	155.0 s.	90	161.7 s.
64	9 MW	5.0 s.	85	717	0.33 s.	236.6 s.	75	241.6 s.

TABLE 7

Space antenna connector - FETI($\widetilde{\mathbf{D}}_I^{t^{-1}}$) serial performance results

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s.$

Preconditioner: $\widetilde{\mathbf{D}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} [\mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}]^{-1}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

CRAY Y-MP (single processor)

N_s	memory	CPU FAC	MFLOPS FAC	# of itr.	CPU /itr. PCG	CPU PCG	MFLOPS PCG	CPU TOT
4	31 MW	19.0 s.	165	111	.77 s.	85.5 s.	130	104.5 s.
8	26 MW	13.6 s.	150	141	.70 s.	98.7 s.	110	112.3 s.
16	20 MW	9.6 s.	130	194	.65 s.	126.1 s.	105	135.7 s.
32	15 MW	6.7 s.	105	381	.61 s.	232.4 s.	90	239.1 s.
64	12 MW	5.0 s.	85	595	.60 s.	357.0 s.	75	362.8 s.

Several observations are worthy discussing:

- As expected, the FETI method with the dual preconditioner requires more memory than the FETI method with the trace preconditioner. On the other hand, the FETI method with the trace preconditioner offers substantial memory savings over the global direct method.
- The computational cost of both FETI algorithms is dominated by the solution of the interface problem via the PCG method, especially for fine mesh partitions. When the number of subdomains is increased, the subdomain vectors become shorter and the MFLOP rate of the PCG algorithm on the CRAY Y-MP deteriorates. This MFLOP rate is slightly higher than that of the Jacobi-PCG algorithm for the undecomposed problem (TABLE 5) but is only half as high as that of the global direct method (TABLE 5). Therefore,

the total CPU timings reported in TABLES 6-7 highlight the intrinsic computational superiority of the FETI methodology over both the direct and Jacobi-PCG global algorithms. In the best case (4 subdomains), the FETI method with the trace preconditioner is three times faster than the global direct solver and four times faster than the global Jacobi-PCG algorithm.

- The dual preconditioner reduces the number of iterations performed by the trace preconditioner by a factor of 1.4 in the [4-16] subdomains range, and by a factor of 1.2 in the [32-64] subdomains range. However, a CG iteration with the dual preconditioner is 1.8 times more expensive than a CG iteration with the trace preconditioner, so that the trace preconditioner is overall more efficient.
- The performance of both FETI algorithms deteriorates when the number of subdomain is increased (FIG. 9). We refer to this phenomenon as the numerical H non-scalability of the FETI methodology, Clearly, the superconvergence behavior discussed in Section 3 seems to disappear in the case of fine mesh partitions and the number of iterations seems to grow linearly with the interface size (FIG. 8). However, the reader should note that for the above structural dynamics problem, increasing the number of subdomains from 4 to 64 increases the number of iterations by a factor of 4.5, but increases the CPU time by a factor of 2.7 only (TABLE 6). This is because the cost of a PCG iteration decreases with the size of a subdomain. Moreover, since the vector speed drops from 130 MFLOPS in the 4 subdomain case to 75 MFLOPS in the 64 subdomain case, increasing the number of subdomains from 4 to 64 actually increases the operation count of the FETI method by a factor of $2.7 \times 75/130 = 1.6$ only.

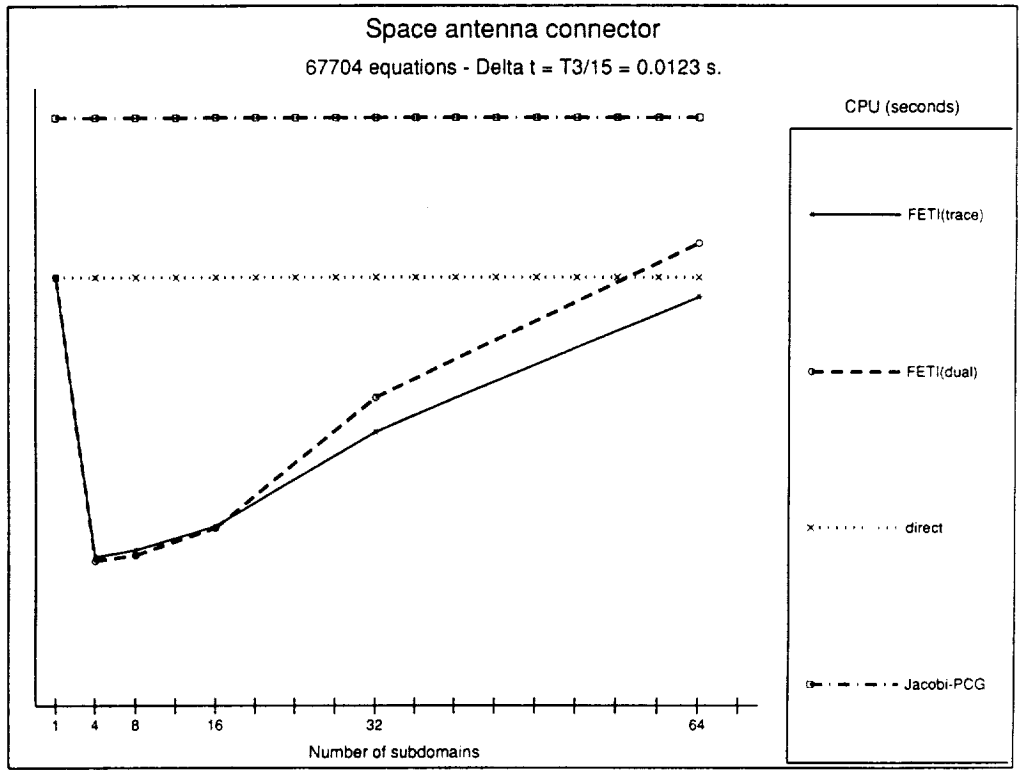

FIG. 9 *Highlighting the H non-scalability*

9. Performance results

The FETI methodology, the CG algorithm with diagonal scaling (Jacobi-PCG), and the LDL^T direct method are implemented on both a CRAY Y-MP/8 and an iPSC-860/128 parallel processors. Different optimal data structures are used for these different algorithms. All mesh partitions are obtained using the mesh decomposer described in [22].

On both parallel processors, the Jacobi-PCG algorithm is implemented in a subdomain-by-subdomain rather than element-by-element fashion in order to perform fewer floating-point operations. Within each subdomain, the transient tangent operator is assembled in sparse format.

The implementation of the direct method on the CRAY Y-MP/8 multiprocessor is described in Farhat [23]. The parallel skyline solver implemented on the iPSC-860/128 system is an improved version of the parallel LDL^T algorithm presented in Farhat and Wilson [24]. In this improved version, a ring rather than

a tree communication algorithm [25] is used for broadcasting at each factorization step the pivotal column to all of the processors. The distributed data structure described in [24] is augmented with a pointer array that identifies the last active column of each structural equation. On the CRAY Y-MP/8 system, the forward and backward substitutions are serialized. On the iPSC-860/128 multiprocessor, only the backward substitution is serialized. These serializations are performed because of well-known mapping, synchronization, and communication bottlenecks in the parallel solution of skyline triangular systems [23, 24].

The performances of the above parallel solvers are assessed with the nonlinear transient analysis of two structural systems: (a) the space antenna connector described in Section 8.1, and (b) a stiffened wing panel from the V22 tiltrotor aircraft (based on the panel described in Davis, Krishnamurthy, Stroud and McCleary [26]) (FIG. 10). The finite element model depicted in FIG. 10 contains 9486 nodes, 9136 4-node shell elements with 6 d.o.f. per node, and a total of 54216 d.o.f.

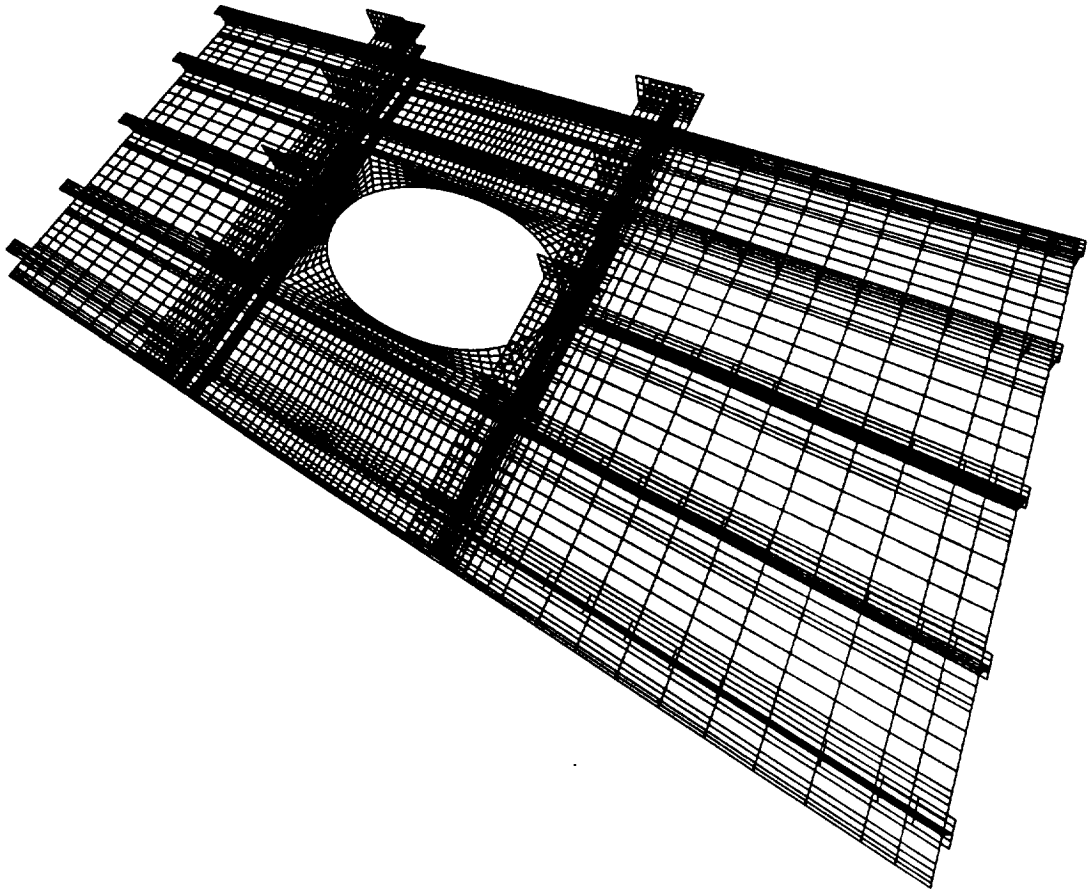

FIG. 10 *Finite element discretization of a stiffened wing panel*

For both structural problems, the time step Δt is set to $T_3/15$, where T_3 is the third fundamental period of the structure. This choice for Δt is reasonable for implicit schemes and leads to significantly larger time steps than those imposed by the Courant stability condition on explicit schemes (TABLE 8).

TABLE 8

Time step comparisons

problem	explicit	implicit
space antenna connector	$5.70\ 10^{-7}$ s.	$1.23\ 10^{-2}$ s.
stiffened wing panel	$1.09\ 10^{-8}$ s.	$1.79\ 10^{-3}$ s.

It is important to note that the condition number of the interface problem $\widetilde{\mathbf{F}}_I^t = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}$ strongly depends on the time step Δt. For a sufficiently small time step, $\widetilde{\mathbf{F}}_I^t$ is "mass dominated" and therefore is relatively well-conditioned. For larger time steps such as those selected for the above two applications (TABLE 8), $\widetilde{\mathbf{F}}_I^t$ is "stiffness dominated" and is relatively ill-conditioned.

9.1 Performance results on the CRAY Y-MP/8 system

The performance results measured on the CRAY Y-MP/8 system for both structural problems are summarized in TABLES 9-10. The number of subdomains N_s is set equal to the number of processors N_p. Samples of the deformed configurations are shown in FIG. 11-12.

For both structural problems, the time step Δt is set to $T_3/15$, where T_3 is the third fundamental period of the structure. This choice for Δt is reasonable for implicit schemes and leads to significantly larger time steps than those imposed by the Courant stability condition on explicit schemes (TABLE 8).

TABLE 8

Time step comparisons

problem	explicit	implicit
space antenna connector	$5.70\ 10^{-7}$ s.	$1.23\ 10^{-2}$ s.
stiffened wing panel	$1.09\ 10^{-8}$ s.	$1.79\ 10^{-3}$ s.

It is important to note that the condition number of the interface problem $\widetilde{\mathbf{F}}_I^t = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})^{-1}\mathbf{B}^{s^T}$ strongly depends on the time step Δt. For a sufficiently small time step, $\widetilde{\mathbf{F}}_I^t$ is "mass dominated" and therefore is relatively well-conditioned. For larger time steps such as those selected for the above two applications (TABLE 8), $\widetilde{\mathbf{F}}_I^t$ is "stiffness dominated" and is relatively ill-conditioned.

9.1 Performance results on the CRAY Y-MP/8 system

The performance results measured on the CRAY Y-MP/8 system for both structural problems are summarized in TABLES 9-10. The number of subdomains N_s is set equal to the number of processors N_p. Samples of the deformed configurations are shown in FIG. 11-12.

TABLE 9

Space antenna connector - performance results on the CRAY Y-MP/8 system

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s.$

FETI preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

N_p	N_s	# of itr. FETI	# of itr. Jacobi-PCG	CPU FETI	CPU Jacobi-PCG	CPU direct
1	1	—	3320	—	345.6 s.	253.0 s.
4	4	159	3320	24.3 s.	93.9 s.	70.3 s.
8	8	200	3320	13.7 s.	50.2 s.	39.5 s.

TABLE 10

Stiffened wing panel - performance results on the CRAY Y-MP/8 system

54216 equations - $\Delta t = \frac{T_3}{15} = 0.00179s.$

FETI preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

N_p	N_s	# of itr. FETI	# of itr. Jacobi-PCG	CPU FETI	CPU Jacobi-PCG	CPU direct
1	1	—	4775	—	418.6 s.	259.7 s.
4	4	300	4775	36.5 s.	114.0 s.	72.1 s.
8	8	396	4775	20.9 s.	60.8 s.	40.6 s.

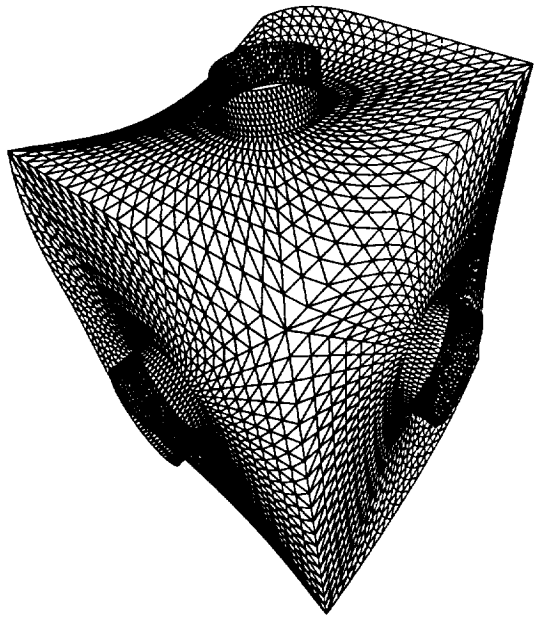

FIG. 11 *Sample of the transient response of the space antenna connector*

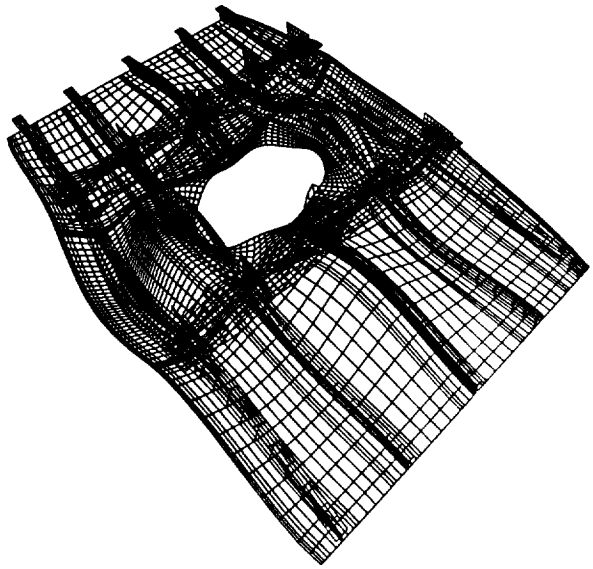

FIG. 12 *Sample of the transient response of the stiffened wing panel*

All algorithms are reported to achieve good speed-ups. This is essentially because the CRAY Y-MP/8 system has a shared memory and a relatively small number of processors. For the antenna connector problem, the FETI method is shown to outperform the direct solver by a factor of three and the Jacobi-PCG algorithm by a factor of four. The wing panel structure apparently generates a stiffer problem: the FETI method converges with a larger number of iterations than for the connector problem, but still outperforms the direct solver by a factor of two and the Jacobi-PCG algorithm by a factor of three.

9.2. Performance results on the iPSC-860/128 multiprocessor

The performance results measured on an iPSC-860/128 Touchstone Gamma multiprocessor are summarized in TABLE 11 for the antenna connector problem and in TABLE 12 for the wing panel problem. A minimum of 32 and 64 processors are needed to accommodate the memory requirements of the FETI method and the direct solver, respectively. Our current implementation of the FETI algorithm on on this parallel processor allows only one processor per subdomain. Therefore, it is important to note that the FETI methodology is benchmarked here in the worst conditions, given that its intrinsic performance deteriorates in the presence of large numbers of subdomains (Section 8.3).

TABLE 11

Space antenna connector - performance results on the iPSC-860/128 system

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s.$

FETI preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

N_p	N_s	# of itr. FETI	# of itr. Jacobi-PCG	CPU FETI	CPU Jacobi-PCG	CPU direct
32	32	456	3320	144.0 s.	229.4 s.	—
64	64	717	3320	144.1 s.	144.8 s.	943.3 s.

TABLE 12

Stiffened wing panel - performance results on the iPSC-860/128 system

54216 equations - $\Delta t = \frac{T_3}{15} = 0.00179s.$

FETI preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

N_p	N_s	# of itr. FETI	# of itr. Jacobi-PCG	CPU FETI	CPU Jacobi-PCG	CPU direct
64	64	705	4775	145.0 s.	227.0 s.	776.2 s.
128	128	1128	4775	136.4 s.	160.9 s.	845.4 s.

For the space antenna connector problem, the FETI method is 1.6 times faster than the Jacobi-PCG algorithm for 32 processors and 32 subdomains, and 6.5 times faster than the direct solver for 64 processors. For the stiffened wing panel problem, the FETI method is also 1.6 times faster than the Jacobi-PCG algorithm for 64 processors and 64 subdomains, and 5.4 and 6.3 times faster than the direct solver for 64 and 128 processors, respectively. However, for 128 processors and subdomains in the second problem, the FETI method is only 18% faster than the Jacobi-PCG scheme. This is partly because the superconvergence phenomenon disappears in the presence of fine mesh partitions, and partly because when the mesh size h is kept constant and the number of subdomains is increased, the local problems become smaller and the interface problem becomes larger, and the FETI method with a trace preconditioner converges towards a Jacobi-PCG scheme.

The parallel direct solver exhibits poor performance results for both structural problems. This is essentially because the bandwidth of the system of equations associated with each problem is out-of-balance with the number of processors needed to store $\widetilde{\mathbf{K}}^t$ in a distributed skyline format. We remind the reader that the parallelism in direct skyline solvers scales with the bandwidth of the

problem and not with its size. Table 13 reports the repartition of the CPU timings between compute, send and receive, for both the FETI method and parallel direct solver.

TABLE 13

Space antenna connector - compute v.s. send v.s. receive on the iPSC-860/128 system

algorithm	N_p	compute	send	receive	dot product	total CPU time
FETI ($\widetilde{\mathbf{T}}_I^{t^{-1}}$)	64	91.1 s.	1.0 s.	9.0 s.	43.0 s.	144.1 s.
direct (factor)	64	54.4 s.	50.7 s.	623.0 s.	—	728.1 s.
direct (forward)	64	1.4 s.	0.1 s.	20.0 s.	—	21.5 s.
direct (backward)	64	0.4 s.	1.3 s.	192.0 s.	—	193.7 s.
direct (total)	64	56.2 s.	52.1 s.	835.0 s.	—	943.3 s.

Clearly, the elapsed CPU time reported for the parallel direct solver is essentially spent in synchronization and interprocessor communication. On the other hand, the FETI method is shown to be communication efficient. The dot products performed during the solution of the interface problem are timed separately because they are implemented via calls to the iNTEL system routine **gdsum**. We could not evaluate the repartition of these timings between compute, send and receive. If we can assume that they are equally distributed between computation and communication, then we can conclude that only 21.8% of the total CPU time reported for the FETI method is spent in synchronization interprocessor communication.

The performance results summarized in TABLES 9-13 also demonstrate that the iPSC-860/128 Touchstone Gamma multiprocessor cannot compete with the CRAY Y-MP/8 system as far as direct solvers are concerned. However, for explicit-like computations such as those characterizing the FETI and Jacobi-PCG algorithms, 32 processors of the iPSC-860/128 system are shown to outperform one CRAY Y-MP processor.

10. Circumventing the numerical H non-scalability

The H non-scalability property characterizing the FETI methodology and discussed in Section 8 requires keeping the number of subdomains relatively small in order to obtain the best possible performance. However, increasing the number of subdomains while keeping the mesh size constant seems a priori attractive because: (a) it reduces the cost of the local problems, and (b) it also reduces the

cost per iteration of the interface problem. Nevertheless, the results reported in TABLE 6 clearly indicate that refining beyond a certain number of subdomains the mesh partition does not significantly reduce any further the computational costs of neither the subdomain nor the interface problems. This can be explained by the fact that when the number of subdomains is increased, the subdomain bandwidth which governs the computational costs of both the subdomain and interface problems does not decrease as fast as the subdomain size. As an example, consider the $N \times N \times N$ uniform discretization of a cubic region. A uniform mesh partitioning of this volume produces N_s subdomains each containing N^3/N_s nodes and each having a local bandwidth equal to $N^2/N_s^{2/3}$ (assuming 1 d.o.f. per node). If the number of subdomains N_s is increased, the size of each local problem decreases as N_s^{-1} while its local bandwidth decreases as $N_s^{-2/3}$. Therefore, even strictly from a computational complexity viewpoint and independently from rate of convergence considerations, it is not necessarily advantageous to introduce a large number of subdomains.

With the advent of massively parallel processors, the above discussion implies that one should allocate more than one processor to a given subdomain. The benefits of this strategy are illustrated in TABLE 14 for the space antenna connector problem. For 4 subdomains and 8 processors, the elapsed CPU time for the FETI method on the CRAY Y-MP/8 multiprocessor is 11.2 seconds, down from 13.7 seconds for 8 subdomains and 8 processors. Of course, significantly better performance improvements can be expected for the range [32-128] processors on the iPSC-860/128 system.

TABLE 14

Space antenna connector - circumventing the H non-scalability of FETI

67704 equations - $\Delta t = \frac{T_3}{15} = 0.0123s$.

Preconditioner: $\widetilde{\mathbf{T}}_I^{t^{-1}} = \sum_{s=1}^{s=N_s} \mathbf{B}^s(\mathbf{M}^s + \frac{\Delta t^2}{4}\mathbf{K}^{t^s})\mathbf{B}^{s^T}$

Convergence criterion: $\frac{||\widetilde{\mathbf{K}}^t(\mathbf{u}_{n+1}^{(k)})\ \Delta\mathbf{u}_{n+1}^{(k+1)} - \mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2}{||\mathbf{r}(\mathbf{u}_{n+1}^{(k)})||_2} \leq 10^{-4}$

CRAY Y-MP/8

N_p	N_s	# of itr.	CPU
1	4	159	87.4 s.
4	4	159	24.3 s.
8	4	159	11.2 s.

11. Closure

A memory efficient domain decomposition (DD) method for implicit transient dynamics has been presented. It is based on the concept of Finite Element Tearing and Interconnecting (FETI) developed by Farhat and Roux [8]. Two subdomain-by-subdomain preconditioners with direct mechanical interpretations have been formulated. The first preconditioner, called here a trace preconditioner, is constructed as the sum of the projections on the subdomain interfaces of the inverses of the subdomain transient operators. The second preconditioner, called here a dual preconditioner, is constructed as the sum of the exact inverses of the subdomain transient operators. When preconditioned with the trace operator, the interface problem behaves as if its condition number is independent of the mesh size. When preconditioned with the dual preconditioner, the interface problem has a condition number that is independent of the mesh size. Therefore, from a mathematical viewpoint, the dual preconditioner is optimal. However, the interface preconditioner is cheaper and computationally more efficient. For coarse mesh partitions, we have shown that the proposed FETI method can outperform on the CRAY Y-MP/8 multiprocessor the parallel direct solver and the parallel conjugate gradient algorithm with diagonal scaling by factors of 3 and 4, respectively. For fine mesh partitions, say more than 32 subdomains, the performance of the FETI algorithm degrades — and this is the case for most DD methods. Therefore, when working with a massively parallel processor, we recommend to keep the number of subdomains as small as possible and allocate more than one processor per subdomain. However, even in the worst case of one processor per subdomain, the FETI method still outperforms the parallel direct solver on the iPSC-860/128 system because of its low communication costs. Finally, the results we have reported for two realistic structural dynamics problems indicate that 32 processors of an iPSC-860 Gamma system can outperform a CRAY Y-MP processor.

Acknowledgments

The first and third authors acknowledge partial support by the National Science Foundation under Grant ASC-8717773. and by the NASA Langley Research Center under Grant NAG-1536427. The second author acknowledges the support of the National Science Foundation under Grant NSF 91-127.

References

[1] T.J.R. Hughes, R.M. Ferencz and J.O. Hallquist, Large-scale vectorized implicit calculations in solid mechanics on a Cray X-MP/48 utilizing EBE precon-

ditioned conjugate gradients, Comput. Meths. Appl. Mech. Engrg. 61 (1987) 215-248.

[2] T.J.R. Hughes, I. Levit and J.M. Winget, An element-by-element solution algorithm for problems of structural and solid mechanics, Comput. Meths. Appl. Mech. Engrg. 36 (1983) 241-254.

[3] J.F. Hajjar and J.F. Abel, Parallel processing of central difference transient analysis for three-dimensional nonlinear framed structures, Commu. Appl. Numer. Meths. 5 (1989) 39-46.

[4] C. Farhat, N. Sobh and K.C. Park, Transient finite element computations on 65,536 processors: the Connection Machine, Internat. J. Numer. Meths. Engrg. 30 (1990) 27-55.

[5] T. Belytschko, E.J. Plaskacz, J.M. Kennedy and D.M. Greenwell, Finite element analysis on the Connection Machine, Comput. Meths. Appl. Mech. Engrg. 81 (1990) 229-254.

[6] J.H. Biffle, Indirect solution of state problems using concurrent vector processing computers, in: A. K. Noor, ed., Parallel Computations and Their Impact on Mechanics, AMD-86 (ASME, New York, 1987) 253-277.

[7] Domain decomposition methods, ed. by T.F. Chan, R. Glowinski, J. Periaux, and O.B. Widlund (SIAM, 1989).

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

[9] C. Farhat and F. X. Roux, An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems, SIAM J. Sc. Stat. Comp. 13 (1992) 379-396.

[10] C. Farhat and M. Geradin, Using a reduced number of Lagrange multipliers for assembling parallel incomplete field finite element approximations, Comput. Meths. Appl. Mech. Engrg. 97 (1992) 333-354.

[11] C. Farhat, L. Crivelli and M. Geradin, Time integration of incomplete hybrid field formulations - Part I: stability theory, *(submitted for publication)*

[12] T. J. R. Hughes, The Finite Element Method, Prentice Hall, N. J. (1987).

[13] F. X. Roux, Dual and spectral properties of Schur and saddle-point domain decomposition methods, in: D. Keyes and B. Voigt, ed., Proc. Fifth SIAM Conference on Domain Decomposition Methods for Partial Differential Equations, (SIAM,1991).

[14] A. van der Luis and A. van der Vorst, The rate of convergence of conjugate gradients, Numerische Mathematik 48 (1986) 543-560. (1986)

[15] J. Mandel, *Private Communication* (1992).

[16] C. Farhat, A Lagrange multiplier based divide and conquer finite element algorithm, J. Comput. Sys. Engrg. 2 (1991) 149-156.

[17] C. Farhat, C. Felippa and M. Militello, A hybrid substructuring method and an adaptive refinement scheme for the distributed solution of three-dimensional structural problems, in: Proc. Eighth International Conference on Vehicle Structural Mechanics and Computer Aided Engineering (1992) 179-199.

[18] B. N. Parlett, The Symmetric Eigenvalue Problem, Prentice Hall, N. J. (1980).

[19] P. E. Bjordstad and O. B. Widlund, Iterative methods for solving elliptic problems on regions partitioned into substructures, SIAM J. of Num. Anal. 23 (1986).

[20] G. Dahlquist and A. Bjorck, Numerical Methods, Prentice Hall, N. J. (1974).

[21] W. Chan and A. George, A linear time implementation of the Reverse Cuthill Mc Kee algorithm," BIT. 20 (1980) pp. 8-14.

[22] C. Farhat, A simple and efficient automatic FEM domain decomposer, Comput. & Struct. 28 (1988) pp. 579-602.

[23] C. Farhat, Redesigning the skyline solver for parallel/vector supercomputers, Internat. J. High Speed Comput. 2 (1988) pp. 223-238.

[24] C. Farhat and E. Wilson, A parallel active column equation solver, Comput. & Struct. 28 (1988) pp. 289-304.

[25] M. A. Baddourah *Private Communication* (1992).

[26] D. D. Davis, T. Krishnamurthy, W. J. Stroud and S. L. McCleary, An accurate nonlinear finite element analysis and test correlation of a stiffened composite wing panel, American Helicopter Society National Technical Specialists' Meeting on Rotorcraft Structures, Williamsburg, Virginia, Oct. 29-31 (1991).