INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng (2011)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nme.3281

A simple explicit–implicit finite element tearing and interconnecting transient analysis algorithm

José A. González[1] and K. C. Park[2,3,*,†]

[1]*Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, E-41092 Seville, Spain*
[2]*Division of Ocean Systems Engineering, School of Mechanical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 305-701, Korea*
[3]*Department of Aerospace Engineering Sciences, University of Colorado at Boulder, CO 80309-429, USA*

SUMMARY

A simple explicit–implicit finite element tearing and interconnecting (FETI) algorithm (AFETI-EI algorithm) is presented for partitioned transient analysis of linear structural systems. The present algorithm employs two decompositions. First, the total system is partitioned via spatial or domain decomposition to obtain the governing equations of motions for each partitioned domain. Second, for each partitioned subsystem, the governing equations are *modally* decomposed into the rigid-body and deformational equations. The resulting rigid-body equations are integrated by an explicit integrator, for its stability is not affected by step-size restriction on account of zero-frequency contents ($\omega = 0$). The modally decomposed partitioned deformation equations of motion are integrated by an unconditionally stable implicit integration algorithm. It is shown that the present AFETI-EI algorithm exhibits unconditional stability and that the resulting interface problem possesses the same solution matrix profile as the basic FETI static problems. The present simple dynamic algorithm, as expected, falls short of the performance of the FETI-DP but offers a similar performance of implicit two-level FETI-D algorithm with a much cheaper coarse solver; hence, its simplicity may offer relatively easy means for conducting parallel analysis of both static and dynamic problems by employing the same basic scalable FETI solver, especially for research-mode numerical experiments.

Received 6 January 2011; Revised 3 July 2011; Accepted 17 July 2011

KEY WORDS: FETI; AFETI; localized Lagrange multipliers

1. INTRODUCTION

One of the most widely employed parallel solution algorithms for structural mechanics problems is the finite element tearing and interconnecting (FETI) algorithm [1] and its subsequent improved versions [2–10]. A key feature of the original static FETI algorithm is the exploitation of floating modes of each substructure in the construction of the so-called *coarse-mesh* solvers. It has been recognized that the use of the coarse-mesh solver plays a fundamental role in propagating the residual imbalance at each iteration cycle. For quasi-static problems, the floating modes emanate intrinsically from the rank deficiency of the substructural solution matrices. However, for the partitioned dynamic equilibrium equations that are implicitly time discretized, the substructural dynamic solution matrices retain full rank as long as the inertia matrices are positive definite, resulting in the unavailability of a corresponding 'dynamic' coarse-mesh solver. Consequently, a 'dynamic' coarse-mesh solver had to be derived by an asymptotic perturbation of the static coarse-mesh solver [5]. This unavailability

[*]Correspondence to: K. C. Park, Department of Aerospace Engineering Sciences, University of Colorado at Boulder, CO 80309-429, USA.

[†]E-mail: kcpark@colorado.edu

of coarse solver for the dynamic problems led to employ two different solvers, one for the static problems and the other for dynamic problems. In an effort to create a common solver applicable for both static and dynamic analyses, Farhat and his coworkers eventually succeeded in developing the FETI-DP algorithm [8, 11] whose solver is not only applicable to both static and dynamic problems but also significantly improves the solution efficiency.

Although we recommend the FETI-DP and its variants for large-scale production-level analysis, we submit that there remains a need for a simple parallel solver that is straightforward to implement and applicable to both static and dynamic problems for research-mode computations as well as educational purposes. By *simple parallel solver*, it is meant to be similar to the 1991 FETI static solver [1] or its equivalents in terms of implementation overhead and the correspondingly comparable solution efficiency. This paper presents one such a simple parallel algorithm that, when used for dynamic analysis, is labeled as the AFETI-EI algorithm [12].

The starting point of the present AFETI-EI algorithm is the use of two equilibrium equations for each substructure. One is the self-equilibrium equations or the d'Alembert–Lagrange principal (DLP) equations that describe the floating rigid motions [13]. It is emphasized that the self-equilibrium (or DLP) equations exist both for static and dynamic problems. Because the DLP equations are devoid of any nonzero structural frequency contents other than the global period of rigid-body motions, any explicit or predictor-corrector integrator can be used without incurring numerical stability issues. The remaining deformational equations of motion for each domain [14, 15] are integrated with an unconditionally stable integrator. In addition, the DLP equations for each partitioned system can be exploited for constructing the same coarse solver both for the static and dynamic problems, thus enabling one to employ the same interface solver for both the static and dynamic problems.

This paper is organized as follows. In Section 2.1, we review two building blocks for the present paper, the DLP equations and the method of localized Lagrange multipliers, employing a partitioned floating structure. An explicit integration of the rigid-body (DLP) equations, which exploits the absence of any structural frequency, thus showing it is devoid of step-size restriction, are presented and described in Section 2.4 as an important element of the present algorithm. Section 2.5 describes implicit integration of the deformational equations of motion. Section 2.6 succinctly illustrates the difference in conventional *spatially partitioned explicit–implicit* methods and the present *modally decomposed explicit–implicit* method. Whereas the step size of the former is restricted by the stability limit of the explicit integrator used, the step size of the present algorithm is free from the step-size restriction because the modally decomposed rigid-body equations correspond to zero frequency ($\omega = 0$).

Section 3 derives interface solution matrix that is applicable for both static and dynamic problems as its matrix profile of the resulting coarse solver is the same for both problems.

Section 4, specifically from Sections 4.1 through 4.3, presents a step-by-step procedure of the present AFETI-EI algorithm. An important feature of the present algorithm is the satisfaction of the interface compatibility conditions not only for the interface forces but also for the interface accelerations. A stabilization that ensures the interface kinematic compatibility conditions are satisfied at each integration step is developed and presented in Section 4.4. A summary of the present AFETI-EI algorithm is given in Section 4.5.

Stability analysis of the present AFETI-EI algorithm is presented in Section 5. It is shown that the present algorithm is unconditionally stable when the trapezoidal rule (or the Newmark algorithm with $\gamma = 1/2$ and $\beta = 1/4$) is used. A noteworthy feature of the present stability analysis is a comparison with the stabilization strategy described in Section 4.4 versus without invoking the stabilization that leads to numerical drifting.

An equivalent version of the FETI-DP algorithm is described in Section 6 for the purpose of illustrating the simplicity of the present AFETI-EI algorithm versus the computational needs required in the FETI-DP. We emphasize again that the FETI-DP algorithm significantly reduced the number of iterations per each step compared with the present AFETI-EI algorithm as well as the FETI-D algorithm.

Finally, numerical experiments, which corroborates stability and accuracy of the present AFETI-EI algorithm as well as convergence properties the same as the FETI-D algorithm with much

DOI: 10.1002/nme

economical coarse solver, are reported in Section 7. A summary and discussions of the present paper are given in Section 8.

2. KEY ELEMENTS OF THE PRESENT PAPER

The present paper utilizes two key concepts: the DLP equations [13] and the deformational equations of motion [14–20]. As these concepts may not be familiar to some of the readers, we will review them herein mainly for clarity purposes. In doing so, we begin with the equations of motion for linear structures that are first spatially partitioned into subdomains. The partitioned equations of motion for each subdomain is *modally* partitioned into rigid-body motions and deformational motions. The two sets of the modally partitioned equations of motion are treated, respectively, by explicit and implicit integrators. The difference between conventional explicit–implicit integration on the spatially partitioned equations of motion and the present modally partitioned explicit–implicit algorithm is delineated.

2.1. Equations of motion for linear structural dynamic systems

Consider a floating bar discretized by two linear elements as shown in Figure 1.

For the two-element bar, the variational equation can be expressed as

$$\delta\pi(\mathbf{u}_g) = \delta\mathbf{u}_g^T\left(\mathbf{f}_g - \mathbf{K}_g\mathbf{u} - \mathbf{M}_g\ddot{\mathbf{u}}_g\right), \tag{1}$$

$$\mathbf{u}_g = \left\{\begin{array}{c} u_g(1) \\ u_g(2) \\ u_g(3) \end{array}\right\}, \quad \mathbf{f}_g = \left\{\begin{array}{c} f_g(1) \\ f_g(2) \\ f_g(3) \end{array}\right\},$$

$$\mathbf{M}_g = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{K}_g = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix},$$

where $\left(\mathbf{u}_g, \ddot{\mathbf{u}}_g, \mathbf{f}_g\right)$ are the assembled displacement, acceleration, and force vectors, respectively; and $(\mathbf{M}_g, \mathbf{K}_g)$ are the mass and stiffness matrices.

We will now carry out two decompositions of the preceding equations of motion for linear structures: *spatial domain decomposition or partitioning* and *modal decomposition for each of the partitioned equations*.

2.2. Spatial domain decomposition

When the bar is partitioned into two elements, the variational form for the partitioned governing equations is given by

$$\delta\pi(\mathbf{u},-,\mathbf{u}_f) = \delta\mathbf{u}^T\,(\mathbf{f} - \mathbf{K}\mathbf{u} - \mathbf{M}\ddot{\mathbf{u}}) + \delta\{\boldsymbol{\lambda}^T(\mathbf{B}\mathbf{u} - \mathbf{L}_f\mathbf{u}_f)\}, \tag{2}$$

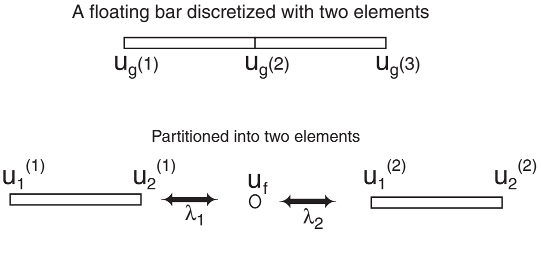

Figure 1. A floating bar partitioned into two bar elements.

$$\mathbf{u}=\left\{\begin{array}{c}\mathbf{u}^{(1)}\\ \mathbf{u}^{(2)}\end{array}\right\},\quad \mathbf{u}^{(1)}=\left\{\begin{array}{c}u_1^{(1)}\\ u_2^{(1)}\end{array}\right\},\quad \mathbf{u}^{(2)}=\left\{\begin{array}{c}u_1^{(2)}\\ u_2^{(2)}\end{array}\right\},$$

$$\mathbf{f}=\left\{\begin{array}{c}\mathbf{f}^{(1)}\\ \mathbf{f}^{(2)}\end{array}\right\},\quad \mathbf{f}^{(1)}=\left\{\begin{array}{c}f_g(1)\\ f_g(2)/2\end{array}\right\},\quad \mathbf{f}^{(2)}=\left\{\begin{array}{c}f_g(2)/2\\ f_g(3)\end{array}\right\},$$

$$\mathbf{M}=\left[\begin{array}{cc}\mathbf{M}^{(1)} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}^{(2)}\end{array}\right],\quad \mathbf{M}^{(1)}=\left[\begin{array}{cc}1 & 0\\ 0 & 1\end{array}\right],\quad \mathbf{M}^{(2)}=\left[\begin{array}{cc}1 & 0\\ 0 & 1\end{array}\right],$$

$$\mathbf{K}=\left[\begin{array}{cc}\mathbf{K}^{(1)} & \mathbf{0}\\ \mathbf{0} & \mathbf{K}^{(2)}\end{array}\right],\quad \mathbf{K}^{(1)}=\left[\begin{array}{cc}1 & -1\\ -1 & 1\end{array}\right],\quad \mathbf{K}^{(2)}=\left[\begin{array}{cc}1 & -1\\ -1 & 1\end{array}\right],$$

$$\mathbf{B}=\left[\begin{array}{cc}\mathbf{B}^{(1)} & \mathbf{0}\\ \mathbf{0} & \mathbf{B}^{(2)}\end{array}\right],\quad \mathbf{B}^{(1)}=\left[\begin{array}{cc}0 & 1\end{array}\right],\quad \mathbf{B}^{(2)}=\left[\begin{array}{cc}1 & 0\end{array}\right],$$

$$\boldsymbol{\lambda}=\left[\begin{array}{c}\lambda_1\\ \lambda_2\end{array}\right],\quad \mathbf{L}_f=\left[\begin{array}{c}1\\ 1\end{array}\right].$$

The stationarity of the preceding variational equation yields the following partitioned equations of motion:

$$\left[\begin{array}{ccc}\left(\mathbf{K}+\mathbf{M}\dfrac{\mathrm{d}^2}{\mathrm{d}t^2}\right) & \mathbf{B} & \mathbf{0}\\ \mathbf{B}^T & \mathbf{0} & -\mathbf{L}_f\\ \mathbf{0} & -\mathbf{L}_f^T & \mathbf{0}\end{array}\right]\left\{\begin{array}{c}\mathbf{u}\\ \boldsymbol{\lambda}\\ \mathbf{u}_f\end{array}\right\}=\left\{\begin{array}{c}\mathbf{f}\\ \mathbf{0}\\ \mathbf{0}\end{array}\right\}. \tag{3}$$

2.3. Modal decomposition of each partitioned equations

The variational energy functional derived previously can be further decomposed by dividing the total displacements into deformations and rigid-body motions. This is accomplished by using the projector

$$[\mathcal{P}_{\mathbf{R}}]=\mathbf{I}-\mathbf{M}\mathbf{R}\mathbf{M}_\alpha^{-1}\mathbf{R}^T \tag{4}$$

with the rigid-body modes $\mathbf{R}$ and the inverse of the principal mass matrix $\mathbf{M}_\alpha=\mathbf{R}^T\mathbf{M}\mathbf{R}$ introduced by Park *et al.* in [13] a (6×6) matrix for a free three-dimensional (3D) floating substructure. We can decompose now the total displacements $\mathbf{u}$ into a deformation part and a rigid-body term by using the projection

$$\mathbf{u}=[\mathcal{P}_{\mathbf{R}}]^T\,\mathbf{d}+\mathbf{R}\boldsymbol{\alpha}, \tag{5}$$

where $\boldsymbol{\alpha}$ represents the rigid-body motion amplitudes. Note that projector $[\mathcal{P}_{\mathbf{R}}]^T$ filters the deformational contribution from the total displacements and that, thanks to the idempotent property of $[\mathcal{P}_{\mathbf{R}}]$, the deformation displacements can be directly obtained from the total displacements using the relation $[\mathcal{P}_{\mathbf{R}}]^T\,\mathbf{u}=[\mathcal{P}_{\mathbf{R}}]^T\,\mathbf{d}$. Substitution of this decomposition into the variational Equation (2) leads to

$$\left[\begin{array}{cccc}\mathbf{M}_\alpha\frac{\mathrm{d}^2}{\mathrm{d}t^2} & \mathbf{0} & \mathbf{R}_b^T & \mathbf{0}\\ \mathbf{0} & \left(\mathbf{K}_{\mathcal{P}}+\mathbf{M}_{\mathcal{P}}\frac{\mathrm{d}^2}{\mathrm{d}t^2}\right) & \mathbf{B}_{\mathcal{P}} & \mathbf{0}\\ \mathbf{R}_b & \mathbf{B}_{\mathcal{P}}^T & \mathbf{0} & -\mathbf{L}_f\\ \mathbf{0} & \mathbf{0} & -\mathbf{L}_f^T & \mathbf{0}\end{array}\right]\left\{\begin{array}{c}\boldsymbol{\alpha}\\ \mathbf{d}\\ \boldsymbol{\lambda}\\ \mathbf{u}_f\end{array}\right\}=\left\{\begin{array}{c}\mathbf{R}^T\mathbf{f}\\ [\mathcal{P}_{\mathbf{R}}]\,\mathbf{f}\\ \mathbf{0}\\ \mathbf{0}\end{array}\right\}, \tag{6}$$

DOI: 10.1002/nme

$$\mathbf{K}_{\mathcal{P}} = [\mathcal{P}_{\mathbf{R}}]\,\mathbf{K}\,[\mathcal{P}_{\mathbf{R}}]^T\,, \quad \mathbf{M}_{\mathcal{P}} = [\mathcal{P}_{\mathbf{R}}]\,\mathbf{M}\,[\mathcal{P}_{\mathbf{R}}]^T\,, \quad \mathbf{R}_b = \mathbf{B}^T\mathbf{R}, \quad \mathbf{B}_{\mathcal{P}} = [\mathcal{P}_{\mathbf{R}}]\,\mathbf{B}.$$

2.4. *Explicit integration of d'Alembert–Lagrange's principal equations*

The first equation is the DLP equations of motion [13],

$$\mathbf{M}_\alpha \ddot{\boldsymbol{\alpha}} + \mathbf{R}_b^T \boldsymbol{\lambda} = \mathbf{R}^T \mathbf{f}, \tag{7}$$

which is associated with the zero-frequency characteristics. In other words, it represents rigid-body motions of each substructure.

As such, the step size to be taken in integrating the preceding DLP equation can be taken solely for accuracy considerations and not by numerical stability limitations. Consequently, the preceding DLP Equation (7) can be integrated by an explicit integration formula without having to incur numerical instability. Specifically, it can be recast for the $(n+1)$th time step as

$$\mathbf{R}_b^T \boldsymbol{\lambda}^{n+1} = \mathbf{R}^T \mathbf{f}^{n+1} - \mathbf{M}_\alpha \ddot{\boldsymbol{\alpha}}_{(p)}^{n+1}, \tag{8}$$

where $\ddot{\boldsymbol{\alpha}}_{(p)}^{n+1}$ may be obtained by a suitable explicit integration formula or simply by a predictor.

2.5. *Implicit integration of deformational equations of motion*

The second equation of (6), given by

$$\mathbf{M}_{\mathcal{P}} \ddot{\mathbf{d}} + \mathbf{K}_{\mathcal{P}} \mathbf{d} + \mathbf{B}_{\mathcal{P}} \boldsymbol{\lambda} = [\mathcal{P}_{\mathbf{R}}]\,\mathbf{f}, \tag{9}$$

is the deformational equations of motion. Hence, it is usually a stiff system, namely the ratio of its lowest and highest frequency can be very large. This means an unconditionally stable implicit time integration should be used to integrate this equation. To this end, we employ the Newmark method to integrate the deformational vector

$$\dot{\mathbf{d}}^{n+1} = \dot{\mathbf{d}}^n + \Delta t \left[(1-\gamma)\ddot{\mathbf{d}}^n + \gamma \ddot{\mathbf{d}}^{n+1} \right], \tag{10}$$

$$\mathbf{d}^{n+1} = \mathbf{d}^n + \Delta t \dot{\mathbf{d}}^n + \Delta t^2 \left[\left(\frac{1}{2} - \beta \right) \ddot{\mathbf{d}}^n + \beta \ddot{\mathbf{d}}^{n+1} \right], \tag{11}$$

to obtain the following discrete equation:

$$\bar{\mathbf{K}}_{\mathcal{P}} \ddot{\mathbf{d}}^{n+1} + \mathbf{B}_{\mathcal{P}} \boldsymbol{\lambda}^{n+1} = [\mathcal{P}_{\mathbf{R}}]\,\mathbf{g}_u^{n+1}, \tag{12}$$

$$\bar{\mathbf{K}}_{\mathcal{P}} = \left[\mathbf{M}_{\mathcal{P}} + \beta \Delta t^2 \mathbf{K}_{\mathcal{P}} \right], \quad \mathbf{g}_u^{n+1} = \mathbf{f}^{n+1} - \mathbf{K} \left[\mathbf{u}^n + \Delta t \dot{\mathbf{u}}^n + \Delta t^2 \left(\frac{1}{2} - \beta \right) \ddot{\mathbf{u}}^n \right],$$

where $(0.5 \leqslant \gamma$ and $0.25 \leqslant \beta)$ are time-integration parameters and Δt is the step size.

2.6. *Conventional explicit–implicit algorithm and the present explicit–implicit algorithm*

In conventional explicit–implicit algorithms, the stability limit of the algorithms is dictated by the criterion

$$\max(\boldsymbol{\omega}_{\text{explicit}}) \cdot \Delta t \leqslant 2, \tag{13}$$

where $\max(\boldsymbol{\omega}_{\text{explicit}})$ is the highest frequency of the explicitly partitioned domain and Δt is the integration step size.

On the other hand, for the present modally partitioned system, namely for the rigid-body modes (7) and the deformable modes (9), the corresponding frequency of the explicitly partitioned modes as can be seen from (7) is zero:

$$\max(\boldsymbol{\omega}_{\text{principal equation}}) = 0. \tag{14}$$

Hence, the only step-size restriction imposed on the integration of the DLP Equation (7) is its accuracy considerations and not stability restriction. This statement will be corroborated in Section 5.

DOI: 10.1002/nme

3. EXPLICIT–IMPLICIT INTEGRATION OF MODALLY PARTITIONED EQUATIONS OF MOTION

The preceding two equations, namely the explicitly integrated DLP Equation (8) and the implicitly integrated deformational Equations (12), are combined with the third and fourth rows of the partitioned Equations (6) to form the following discrete partitioned equation set:

$$\begin{bmatrix} \bar{\mathbf{K}}_{\mathcal{P}} & \mathbf{0} & \mathbf{B}_{\mathcal{P}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_b^T & \mathbf{0} \\ \mathbf{B}_{\mathcal{P}}^T & \mathbf{R}_b & \mathbf{0} & -\mathbf{L}_f \\ \mathbf{0} & \mathbf{0} & -\mathbf{L}_f^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{d}} \\ \ddot{\boldsymbol{\alpha}} \\ \boldsymbol{\lambda} \\ \ddot{\mathbf{u}}_f \end{Bmatrix}^{n+1} = \begin{Bmatrix} [\mathcal{P}_{\mathbf{R}}]\,\mathbf{g}_u^{n+1} \\ \mathbf{R}^T\mathbf{f}^{n+1} - \mathbf{M}_\alpha \ddot{\boldsymbol{\alpha}}_{(p)}^{n+1} \\ \mathbf{0} \\ \mathbf{0} \end{Bmatrix}. \tag{15}$$

Because $[\mathcal{P}_{\mathbf{R}}]\,\bar{\mathbf{K}}_{\mathcal{P}} = \bar{\mathbf{K}}_{\mathcal{P}}\,[\mathcal{P}_{\mathbf{R}}]^T$, the accelerations can be obtained from the first equation of (15)

$$[\mathcal{P}_{\mathbf{R}}]^T\,\ddot{\mathbf{d}}^{n+1} = \mathbf{K}_d^{-1}\,[\mathcal{P}_{\mathbf{R}}]\left(\mathbf{g}_u^{n+1} - \mathbf{B}\boldsymbol{\lambda}^{n+1}\right), \tag{16}$$

$$\mathbf{K}_d = [\mathbf{M} + \beta\Delta t^2\mathbf{K}],$$

which is substituted into the third row of (15) to yield the following dynamic flexibility equation that is solved for every time step:

$$\begin{bmatrix} \bar{\mathbf{F}}_{bb} & -\mathbf{R}_b & \mathbf{L}_f \\ -\mathbf{R}_b^T & \mathbf{0} & \mathbf{0} \\ \mathbf{L}_f^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \ddot{\boldsymbol{\alpha}} \\ \ddot{\mathbf{u}}_f \end{Bmatrix}^{n+1} = \begin{Bmatrix} \bar{\mathbf{b}}_\lambda \\ \bar{\mathbf{b}}_\alpha \\ \mathbf{0} \end{Bmatrix}^{n+1}, \tag{17}$$

$$\bar{\mathbf{F}}_{bb} = \mathbf{B}^T\bar{\mathbf{F}}\mathbf{B}, \quad \bar{\mathbf{F}} = [\mathcal{P}_{\mathbf{R}}]^T\,\mathbf{K}_d^{-1}\,[\mathcal{P}_{\mathbf{R}}] = \mathbf{K}_d^{-1} - \mathbf{R}\mathbf{M}_\alpha^{-1}\mathbf{R}^T,$$

$$\bar{\mathbf{b}}_\lambda^{n+1} = \mathbf{B}^T\bar{\mathbf{F}}\mathbf{g}_u^{n+1}, \quad \bar{\mathbf{b}}_\alpha^{n+1} = -\mathbf{R}^T\mathbf{f}^{n+1} + \mathbf{M}_\alpha\ddot{\boldsymbol{\alpha}}_{(p)}^{n+1}.$$

Because a repeated solution of this system using iterative methods could be highly inefficient for a large number of time steps, iterative resolution methods should be combined with the projection and orthogonalization technique [21], a modification of preconditioned conjugate gradient method that accelerates the solution of linear systems for repeated load cases. Thus, complete iterative resolution of the system is made only during the first time steps, where the orthogonal search directions are computed and stored. Then, knowing the search directions, the solution of system (17) becomes direct.

The static flexibility equation is obtained from (17) by considering the following parametrization:

$$\triangle t \to \infty \Rightarrow \mathbf{F}_{bb}\mathbf{R}_b = \mathbf{0}, \quad \mathbf{F}_{bb} = \mathbf{B}^T\mathbf{K}^{+}\mathbf{B}, \quad \mathbf{M}_\alpha = \mathbf{0}, \quad \ddot{\boldsymbol{\alpha}}^{n+1} = \mathbf{0}, \tag{18}$$

arriving at the well-known quasi-static partitioned flexibility equation:

$$\begin{bmatrix} \mathbf{F}_{bb} & -\mathbf{R}_b & \mathbf{L}_f \\ -\mathbf{R}_b^T & \mathbf{0} & \mathbf{0} \\ \mathbf{L}_f^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \\ \mathbf{u}_f \end{Bmatrix}^{n+1} = \begin{Bmatrix} \mathbf{b}_\lambda \\ \mathbf{b}_\alpha \\ \mathbf{0} \end{Bmatrix}^{n+1}, \tag{19}$$

$$\mathbf{b}_\lambda^{n+1} = \mathbf{B}^T\mathbf{K}^{+}\mathbf{f}^{n+1}, \quad \mathbf{b}_\alpha^{n+1} = -\mathbf{R}^T\mathbf{f}^{n+1},$$

where $\mathbf{K}^{+}$ is the generalized inverse of the stiffness matrix.

Comparing the dynamic flexibility Equation (17) with the static flexibility Equation (19), we note that the solution matrix has the same matrix profile for both equations. The remaining task is to obtain an accurate prediction of the rigid-body acceleration vector, $\ddot{\boldsymbol{\alpha}}$, that appears in the dynamic flexibility equation.

4. A SIMPLE FETI-EI SOLVER FOR BOTH STATIC AND DYNAMIC PROBLEMS

We present FETI dynamic and static solution algorithms simply by constructing different right-hand sides and flexibility matrices, namely $\bar{\mathbf{F}}_{bb}$ for dynamic analysis and $\mathbf{F}_{bb}$ for static analysis. It is this property, heretofore not exploited, that constitutes a key contribution of the present paper. The benefits of this approach will be the simplicity of using the same numerical tools for statics and dynamics and the implementation simplicity for research-oriented and educational usage of the proposed algorithms.

This AFETI-EI algorithm consists of four basic steps: (1) an *explicit step* to predict the rigid-body component of the localized multipliers; (2) the *implicit step* where the self-equilibrated part of the multipliers is calculated; (3) the *correction step* where the net component of the multipliers is updated; and (4) a *substructure solution step* where the position of the substructures is updated. These four steps, summarized in Table II, are explained in the following subsections.

The algorithm uses a decomposition of the interface solution vector in the form

$$\boldsymbol{\lambda} = [\mathcal{P}_{\boldsymbol{\lambda}}]\,\boldsymbol{\lambda}_d + \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\boldsymbol{\lambda}_\alpha \tag{20}$$

with symmetric projectors

$$\left[\mathcal{P}_{\mathbf{L}_f}\right] = \mathbf{I} - \mathbf{L}_f\left(\mathbf{L}_f^T\mathbf{L}_f\right)^{-1}\mathbf{L}_f^T, \tag{21}$$

$$[\mathcal{P}_{\boldsymbol{\lambda}}] = \left[\mathcal{P}_{\mathbf{L}_f}\right] - \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}b_b\right)^{-1}\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right], \tag{22}$$

where $[\mathcal{P}_{\boldsymbol{\lambda}}]\,\boldsymbol{\lambda}_d$ represents the self-equilibrated part of the localized multipliers, whereas $\boldsymbol{\lambda}_\alpha$ contains the net component of the multipliers participating in the DLP Equation (7), that is, $\mathbf{R}_b^T\boldsymbol{\lambda} = \left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_\alpha$.

4.1. Step 1: compute initial vectors

The present dynamic algorithm needs the correct starting values of $\boldsymbol{\lambda}^0$ and the substructural accelerating vector $\ddot{\mathbf{u}}^0$. To this end, we obtain the acceleration vector for the partitioned system from (3):

$$\ddot{\mathbf{u}}^0 = \mathbf{M}^{-1}\left(\mathbf{f}^0 - \mathbf{B}\boldsymbol{\lambda}^0 - \mathbf{K}\mathbf{u}^0\right). \tag{23}$$

Differentiating twice the second row of (3) gives

$$\mathbf{B}^T\ddot{\mathbf{u}}^0 - \mathbf{L}_f\ddot{\mathbf{u}}_f^0 = 0. \tag{24}$$

Substituting (23) into the twice time-differentiated interface constraint condition (24), one obtains together with the third row of (3)

$$\begin{bmatrix} \mathbf{B}^T\mathbf{M}^{-1}\mathbf{B} & \mathbf{L}_f \\ \mathbf{L}^T & \mathbf{0} \end{bmatrix}\begin{Bmatrix} \boldsymbol{\lambda} \\ \ddot{\mathbf{u}}_f \end{Bmatrix}^0 = \begin{Bmatrix} \mathbf{B}^T\mathbf{M}^{-1}\left(\mathbf{f}^0 - \mathbf{K}\mathbf{u}^0\right) \\ \mathbf{0} \end{Bmatrix}. \tag{25}$$

Once $\boldsymbol{\lambda}^0$ is computed, the initial acceleration, $\ddot{\mathbf{u}}^0$, is obtained from (23). We are now ready to describe the present algorithm. For the static analysis, this step is not needed.

4.2. Step 2: explicit integration

The explicit integration step of the algorithm starts with the prediction of the localized Lagrange multipliers $\boldsymbol{\lambda}_{(p)}^{n+1}$ for current time step $(n+1)$. We employ the simplest choice:

$$\boldsymbol{\lambda}_{(p)}^{n+1} = \begin{cases} \boldsymbol{\lambda}^0 & \text{for the first time step, } n = 1, \\ 2\boldsymbol{\lambda}^n - \boldsymbol{\lambda}^{n-1} & \text{for subsequent steps, } n \geqslant 2. \end{cases} \tag{26}$$

DOI: 10.1002/nme

In order to proceed with the implicit integration step, a good approximation of the interface force $\boldsymbol{\lambda}_{\alpha(p)}$ that satisfies the substructure-by-substructure self-equilibrium (DLP equation) is all that is needed. This is obtained by solving

$$\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_{\alpha(p)}^{n+1}=\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\left(\mathbf{b}_\lambda-\mathbf{F}_{bb}\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}_{(p)}^{n+1}\right), \tag{27}$$

$$\mathbf{F}_{bb}=\mathbf{B}^T\mathbf{K}_d^{-1}\mathbf{B},\quad \mathbf{b}_\lambda^{n+1}=\mathbf{B}^T\mathbf{K}_d^{-1}\mathbf{g}_u^{n+1},\quad \mathbf{K}_d=[\mathbf{M}+\beta\Delta t^2\mathbf{K}].$$

Note that this is a coarse problem with a size equal to the total number of rigid-body modes that requires global communications. Once $\boldsymbol{\lambda}_{\alpha(p)}^{n+1}$ is computed, we have an approximation of $\mathbf{M}_\alpha\ddot{\boldsymbol{\alpha}}$ that is needed for the solution of (17) and is obtained by

$$\mathbf{M}_\alpha\ddot{\boldsymbol{\alpha}}_{(p)}^{n+1}=\mathbf{R}^T\mathbf{f}^{n+1}-\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_{\alpha(p)}^{n+1} \tag{28}$$

to proceed with the calculation of $\bar{\mathbf{b}}_\alpha$.

In a static problem, this step is not needed.

4.3. Step 3: implicit integration step

We now describe a step-by-step procedure for solving the partitioned flexibility Equations (17).

Step 3.1: starting $\boldsymbol{\lambda}$ vector

The starting vector is obtained by solving the least squares solution of $\boldsymbol{\lambda}$ from the second and third rows of (17), that is,

$$\begin{bmatrix}-\mathbf{R}_b^T\\ \mathbf{L}_f^T\end{bmatrix}\boldsymbol{\lambda}_0=\left\{\begin{array}{c}-\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\boldsymbol{\lambda}_{\alpha(p)}^{n+1}\\ \mathbf{0}\end{array}\right\}, \tag{29}$$

and an iterative solution scheme follows using the classical pre algorithm.

Step 3.2: iterative solution of $\boldsymbol{\lambda}$

This is, by now, a well-documented solution procedure. As such, we summarize it in Table I.

Note that this step is exactly the same for the solution of the static flexibility Equation (19).

Step 3.3: correction step

After obtaining $\boldsymbol{\lambda}$ as a combination of the predicted substructure-by-substructure net multipliers $\boldsymbol{\lambda}_{\alpha(p)}^{n+1}$ and a new self-equilibrated component of the localized Lagrange multipliers $\boldsymbol{\lambda}_d^{n+1}$, the self-equilibrium part of the interface force, $\boldsymbol{\lambda}_\alpha^{n+1}$, is corrected using the following coarse solver:

$$\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_\alpha^{n+1}=\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\left(\mathbf{b}_\lambda-\mathbf{F}_{bb}\left[\mathcal{P}_\lambda\right]\boldsymbol{\lambda}\right). \tag{30}$$

Note that in statics, this process is not needed as the corresponding localized multipliers can be directly computed from applied external forces using the second row of (19), that is,

$$\boldsymbol{\lambda}_\alpha^{n+1}=\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)^{-1}\mathbf{R}^T\mathbf{f}^{n+1}. \tag{31}$$

Thus, the total localized Lagrange multipliers are finally obtained from

$$\boldsymbol{\lambda}^{n+1}=\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}_d^{n+1}+\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\boldsymbol{\lambda}_\alpha^{n+1}, \tag{32}$$

and we proceed with the solution of the substructures.

DOI: 10.1002/nme

Table I. Classical preconditioned conjugate gradient algorithm.

```
Compute starting vector:
$\begin{bmatrix} -\mathbf{R}_b^T \\ \mathbf{L}_f^T \end{bmatrix} \boldsymbol{\lambda}_0 = \begin{Bmatrix} \mathbf{b}_\alpha \\ \mathbf{0} \end{Bmatrix}$
$\boldsymbol{\lambda}_0 = -\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)^{-1}\mathbf{b}_\alpha$
Initial residual:
$\mathbf{r}_0 = \bar{\mathbf{b}}_\lambda - \bar{\mathbf{F}}_{bb}\boldsymbol{\lambda}_0$
Initial projected residual:
$\mathbf{w}_0 = [\mathcal{P}_{\boldsymbol{\lambda}}]^T\,\mathbf{r}_0$
Iterate $k = 0, 1 \ldots$ until convergence
    Precondition:
    $\mathbf{z}_k = \tilde{\mathbf{F}}_{bb}^{+}\mathbf{w}_k$
    Project preconditioned residual:
    $\mathbf{y}_k = [\mathcal{P}_{\boldsymbol{\lambda}}]\,\mathbf{z}_k$
    Compute conjugate parameters:
    $\zeta_k = \dfrac{\mathbf{y}_k^T\mathbf{w}_k}{\mathbf{y}_{k-1}^T\mathbf{w}_{k-1}}$        $(\zeta_0 = 0)$
    $\mathbf{p}_k = \mathbf{y}_k + \zeta_k\mathbf{p}_{k-1}$        $(\mathbf{p}_0 = \mathbf{y}_0)$
    $\mathbf{q}_k = \bar{\mathbf{F}}_{bb}\mathbf{p}_k$
    $\nu_k = \dfrac{\mathbf{y}_k^T\mathbf{w}_k}{\mathbf{p}_k^T\mathbf{q}_k}$
    Update:
    $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \nu_k\mathbf{p}_k$
    Compute projected residual:
    $\mathbf{w}_{k+1} = \mathbf{w}_k - \nu_k\,[\mathcal{P}_{\boldsymbol{\lambda}}]^T\,\mathbf{q}_k$
If $|\mathbf{w}_{k+1}| > \epsilon$ , $k \leftarrow k+1$ go back to iteration loop.
```

4.4. *Step 4: solution of substructural acceleration*

Once the interface force vector, $\boldsymbol{\lambda}^{n+1}$, is obtained from the previous step, one is tempted to obtain the substructural accelerations by the following equation:

$$\begin{bmatrix} \mathbf{K}_{d_{ii}} & \mathbf{K}_{d_{ib}} \\ \mathbf{K}_{d_{bi}} & \mathbf{K}_{d_{bb}} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{u}}_i \\ \ddot{\mathbf{u}}_b \end{Bmatrix}^{n+1} = \begin{Bmatrix} \mathbf{g}_{u_i}^{n+1} \\ \mathbf{g}_{u_b}^{n+1} - \boldsymbol{\lambda}^{n+1} \end{Bmatrix}. \tag{33}$$

However, we do not recommend the previous updating procedure. Instead, we recommend the computation of $\ddot{\mathbf{u}}_b$ by utilizing the frame acceleration, $\ddot{\mathbf{u}}_f$, that can be obtained from the dynamic flexibility Equation (17):

$$\ddot{\mathbf{u}}_f^{n+1} = \left(\mathbf{L}_f^T\mathbf{L}_f\right)^{-1}\mathbf{L}_f^T\left(\mathbf{b}_\lambda^{n+1} - \mathbf{F}_{bb}\boldsymbol{\lambda}^{n+1}\right). \tag{34}$$

The boundary accelerations, $\ddot{\mathbf{u}}_b$, are obtained from the frame accelerations, $\ddot{\mathbf{u}}_f$, by the compatibility condition:

$$\ddot{\mathbf{u}}_b^{n+1} = \mathbf{B}^T\ddot{\mathbf{u}}^{n+1} = \mathbf{L}_f\ddot{\mathbf{u}}_f^{n+1} = \mathbf{L}_f\left(\mathbf{L}_f^T\mathbf{L}_f\right)^{-1}\mathbf{L}_f^T\left(\mathbf{b}_\lambda^{n+1} - \mathbf{F}_{bb}\boldsymbol{\lambda}^{n+1}\right). \tag{35}$$

Finally, the internal accelerations of the substructure can be computed from the first row of (33) solving the linear system

$$\mathbf{K}_{d_{ii}}\ddot{\mathbf{u}}_i^{n+1} = \mathbf{g}_{u_i}^{n+1} - \mathbf{K}_{d_{ib}}\ddot{\mathbf{u}}_b^{n+1}. \tag{36}$$

Afterwards, the velocity and displacement vectors can be updated via the Newmark method ((10)–(11)).

DOI: 10.1002/nme

It is noted that the substructural interface acceleration, $\ddot{\mathbf{u}}_b$, obtained by (35) is, in effect, a correction step to ensure that the interface accelerations satisfy the interface equilibrium. It will be shown in Section 5 that, without this correction step, the solution would gradually diverge stemming from the violation of interface equilibrium condition.

4.5. *Summary of the present AFETI-EI algorithm*

Table II provides a summary of the present AFETI-EI algorithm.

Table II. Partitioned explicit–implicit algorithm AFETI-EI. The algorithm is composed of four basic steps: explicit step, implicit step, correction step, and solution of the substructures.

```
Compute initial acceleration $\ddot{\mathbf{u}}^0$:
$\begin{bmatrix} \mathbf{B}^T\mathbf{M}^{-1}\mathbf{B} & \mathbf{L}_f \\ \mathbf{L}_f^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \ddot{\mathbf{u}}_f \end{Bmatrix}^0 = \begin{Bmatrix} \mathbf{B}^T\mathbf{M}^{-1}\left(\mathbf{f}^0 - \mathbf{K}\mathbf{u}^0\right) \\ \mathbf{0} \end{Bmatrix}$
$\ddot{\mathbf{u}}^0 = \mathbf{M}^{-1}\left(\mathbf{f}^0 - \mathbf{K}\mathbf{u}^0 - \mathbf{B}\boldsymbol{\lambda}^0\right)$

Time loop $n = 0 \ldots n_{\text{step}}$
    • Explicit step:
    Predict multipliers $\boldsymbol{\lambda}_{(p)}^{n+1}$ using:
    $\boldsymbol{\lambda}_{(p)}^{n+1} = 2\boldsymbol{\lambda}^n - \boldsymbol{\lambda}^{n-1}$
    Solve for $\boldsymbol{\lambda}_{\alpha(p)}^{n+1}$ (coarse problem)
    $\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_{\alpha(p)}^{n+1} = \mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\left(\mathbf{b}_\lambda - \mathbf{F}_{bb}\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}_{(p)}^{n+1}\right)$

    • Implicit step:
    Solve for $\boldsymbol{\lambda}$ (see Table I for detail):
    $\begin{bmatrix} \bar{\mathbf{F}}_{bb} & -\mathbf{R}_b & \mathbf{L}_f \\ -\mathbf{R}_b^T & \mathbf{0} & \mathbf{0} \\ \mathbf{L}_f^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \ddot{\boldsymbol{\alpha}} \\ \ddot{\mathbf{u}}_f \end{Bmatrix} = \begin{Bmatrix} \bar{\mathbf{b}}_\lambda^{n+1} \\ -\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\boldsymbol{\lambda}_{\alpha(p)}^{n+1} \\ \mathbf{0} \end{Bmatrix}$
    Extract $\boldsymbol{\lambda}_d^{n+1}$ from $\boldsymbol{\lambda}$:
    $\boldsymbol{\lambda}_d^{n+1} = \left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}$

    • Correction step:
    Correct $\boldsymbol{\lambda}_\alpha^{n+1}$ (coarse problem):
    $\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)\boldsymbol{\lambda}_\alpha^{n+1} = \mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]^T\left(\mathbf{b}_\lambda - \mathbf{F}_{bb}\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}_d^{n+1}\right)$
    Obtain $\boldsymbol{\lambda}^{n+1}$:
    $\boldsymbol{\lambda}^{n+1} = \left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\boldsymbol{\lambda}_d^{n+1} + \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\boldsymbol{\lambda}_\alpha^{n+1}$

    • Substructure solution:
    Compute boundary and internal accelerations:
    $\ddot{\mathbf{u}}_b^{n+1} = \mathbf{L}_f\left(\mathbf{L}_f^T\mathbf{L}_f\right)^{-1}\mathbf{L}_f^T\left(\mathbf{b}_\lambda^{n+1} - \mathbf{F}_{bb}\boldsymbol{\lambda}^{n+1}\right)$
    $\ddot{\mathbf{u}}_i^{n+1} = \mathbf{K}_{d_{ii}}^{-1}\left(\mathbf{g}_{u_i}^{n+1} - \mathbf{K}_{d_{ib}}\ddot{\mathbf{u}}_b^{n+1}\right)$
    Update $\mathbf{u}^{n+1}$ and $\dot{\mathbf{u}}^{n+1}$:
    $\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t\dot{\mathbf{u}}^n + \Delta t^2\left[\left(\frac{1}{2} - \beta\right)\ddot{\mathbf{u}}^n + \beta\ddot{\mathbf{u}}^{n+1}\right]$
    $\dot{\mathbf{u}}^{n+1} = \dot{\mathbf{u}}^n + \Delta t\left[(1-\gamma)\ddot{\mathbf{u}}^n + \gamma\ddot{\mathbf{u}}^{n+1}\right]$
Advance time $n \leftarrow n + 1$
where
    $\left[\mathcal{P}_{\mathbf{L}_f}\right] = \mathbf{I} - \mathbf{L}_f\left(\mathbf{L}_f^T\mathbf{L}\right)^{-1}\mathbf{L}_f^T$
    $\left[\mathcal{P}_{\boldsymbol{\lambda}}\right] = \left[\mathcal{P}_{\mathbf{L}_f}\right] - \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\left(\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)^{-1}\mathbf{R}_b^T\left[\mathcal{P}_{\mathbf{L}_f}\right]$
```

DOI: 10.1002/nme

5. STABILITY ANALYSIS AND ACCURACY ASSESSMENT

As alluded to in Section 2.6, the present explicit–implicit algorithm applied to the modally partitioned equations of motion (6) does not encounter stability limit imposed by (13). To this end, we express out the necessary steps summarized in Table II as given by the recurrence equation

$$\mathbf{B}\mathbf{y}^{n+1} = \mathbf{A}\mathbf{y}^{n}, \quad \mathbf{y} = \left\{ \boldsymbol{\lambda}^{T}\ \ddot{\mathbf{u}}^{T}\ \dot{\mathbf{u}}^{T}\ \mathbf{u}^{T} \right\}^{T}, \tag{37}$$

where **A** and **B** are given, respectively, by

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\delta\mathbf{I} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\delta^{2}\mathbf{I} & \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{38}$$

$$\mathbf{B}_{21} = \begin{bmatrix} -\mathbf{L}_{b}\mathbf{F}_{bb} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{B}_{22} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{K}_{d_{ii}}^{-1}\mathbf{K}_{d_{ib}} & \mathbf{I} \end{bmatrix},$$

$$\mathbf{L}_{b} = \mathbf{L}_{f}\left(\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right]\mathbf{L}_{f}\right)^{-1}\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right], \quad \delta = \frac{\Delta t}{2},$$

and

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \delta^{2}\mathbf{D} & 2\delta\mathbf{D} & \mathbf{D} \\ \mathbf{A}_{21} & \delta^{2}\mathbf{E} & 2\delta\mathbf{E} & \mathbf{E} \\ \mathbf{0} & \delta\mathbf{I} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \delta^{2}\mathbf{I} & 2\delta\mathbf{I} & \mathbf{I} \end{bmatrix}, \tag{39}$$

$$\mathbf{A}_{11} = \mathbf{G}\mathbf{C}_{2}, \ \mathbf{D} = -(\mathbf{D}_{1} + \mathbf{D}_{2})\mathbf{B}^{T}\bar{\mathbf{F}}\mathbf{K},$$

$$\mathbf{G} = \left(\mathbf{I} - \left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{R}_{b}\mathbf{A}_{\alpha}^{-1}\mathbf{R}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{F}_{bb}\right)\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\mathbf{F}_{bb}^{-1}\left[\begin{matrix} \mathbf{R}_{b} & -\mathbf{L}_{f} \end{matrix}\right]\mathbf{A}_{\alpha u}^{-1},$$

$$\mathbf{D}_{1} = \left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\mathbf{F}_{bb}^{-1} + \left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{R}_{b}\mathbf{A}_{\alpha}^{-1}\mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\left(\mathbf{I} - \mathbf{F}_{bb}\left[\mathcal{P}_{\boldsymbol{\lambda}}\right]\mathbf{F}_{bb}^{-1}\right), \ \mathbf{D}_{2} = \mathbf{G}\mathbf{C}_{1},$$

$$\mathbf{C}_{1} = \begin{bmatrix} -\mathbf{R}_{b}^{T}\mathbf{F}_{bb}^{-1} + \mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{R}_{b}\mathbf{A}_{\alpha}^{-1}\mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right] \\ \mathbf{L}_{f}^{T}\mathbf{F}_{bb}^{-1} \end{bmatrix}, \ \mathbf{C}_{2} = \begin{bmatrix} -\mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{R}_{b}\mathbf{A}_{\alpha}^{-1}\mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{F}_{bb}\left[\mathcal{P}_{\boldsymbol{\lambda}}\right] \\ \mathbf{0} \end{bmatrix},$$

$$\mathbf{A}_{\alpha} = \mathbf{R}_{b}^{T}\left[\mathcal{P}_{\mathbf{L}_{f}}\right]\mathbf{F}_{bb}\mathbf{R}_{b}\left[\mathcal{P}_{\mathbf{L}_{f}}\right], \ \mathbf{A}_{\alpha u} = \left[\begin{matrix} \mathbf{R}_{b} & -\mathbf{L}_{f} \end{matrix}\right]^{T}\mathbf{F}_{bb}^{-1}\left[\begin{matrix} \mathbf{R}_{b} & -\mathbf{L}_{f} \end{matrix}\right],$$

$$\mathbf{A}_{21} = \begin{bmatrix} -\mathbf{L}_{f}\left(\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right]\mathbf{L}_{f}\right)^{-1}\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right]\mathbf{F}_{bb} \\ \mathbf{0} \end{bmatrix}, \ \mathbf{E} = \begin{bmatrix} -\mathbf{L}_{f}\left(\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right]\mathbf{L}_{f}\right)^{-1}\mathbf{L}_{f}^{T}\left[\mathcal{P}_{\mathbf{R}}\right]\mathbf{B}^{T}\bar{\mathbf{F}}\mathbf{K} \\ \mathbf{K}_{d_{ii}}^{-1}\left[\mathbf{K}_{d_{ib}} - \mathbf{K}_{d_{ii}}\right] \end{bmatrix}.$$

We look for the solution of the recurrence Equation (37) in the form of

$$\mathbf{y}^{n+1} = \mu\mathbf{y}^{n}. \tag{40}$$

The unconditional stability of the present explicit–implicit partitioned algorithm is guaranteed if

$$|\mu| \leqslant 1 \quad \text{for all step sizes.} \tag{41}$$

DOI: 10.1002/nme

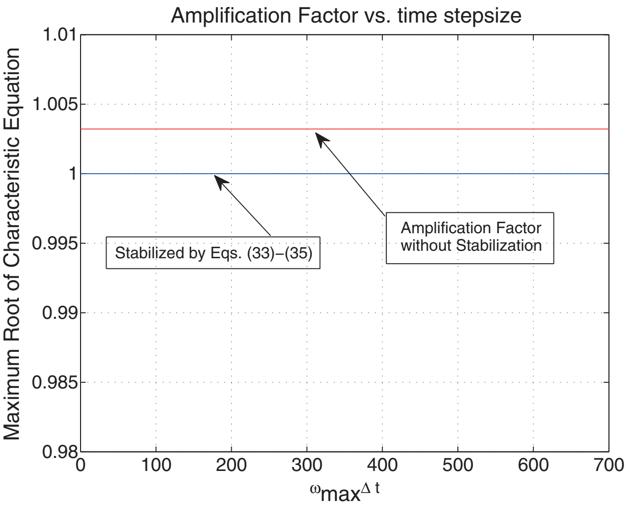

Figure 2. Solution amplification versus normalized step size for the generic model problem.

For demonstration purposes, we will use the two-bar problem shown in Figure 1, as it represents a generic model problem: a rigid-body mode and a deformational mode at each partition. A sufficiency of such a generic model for stability analysis of general problems has been amply shown in previous studies [22–24]. Figure 2 shows the magnitude of the maximum amplification values $|\mu|$ versus normalized step size $\omega_{\max} \cdot \Delta t$. It should be noted that for accurate tracing of the model problem, one demands $\omega_{\max} \cdot \Delta t \leqslant 0.5$. Observe that the present explicit–implicit algorithm maintains its stability regardless of the step sizes. Also plotted in Figure 2 is the amplification factor, which is slightly larger than unity when the stabilization procedure outlined in (33)–(36) is not implemented.

It is noted that the model problem for our stability analysis corresponds to taking only one rigid-body mode and one deformational mode for multideformational general problems. For general problems, each partition consists of up to six rigid-body modes and many deformational modes. As demonstrated in Section 7, the present method is shown to be unconditionally stable for problems involving multi-degrees of freedom per partition on a number of different problems. Thus, the unconditional stability of the present *modally decomposed* explicit–implicit algorithm, as summarized in Table II, is analytically proved for the model problem and numerically demonstrated for general partitioned problems.

As for accuracy assessment, it is recalled that the solution of the rigid-body equations of motion is reduced to obtaining the rigid-body part of the Lagrange multipliers, $\boldsymbol{\lambda}_{\alpha}$, by a predictor-corrector step, as detailed in (27) and (30). It was shown in [25] that one cycle of predictor-corrector iteration achieves a second-order accuracy when one employs a second-order integration formula. This will be corroborated in Section 7.

6. AN EQUIVALENT VERSION OF FETI-DP ALGORITHM

We present a version of the FETI-DP algorithm employing the method of localized Lagrange multipliers and the decomposition of the displacement into rigid and deformation modes, not to come up with yet another version of the FETI-DP algorithm but to illustrate the added implementation complexities it entails compared with the proposed AFETI-EI algorithm. To this end, the basic partitioned discrete dynamic Equations (17) can be further partitioned into the so-called coarse nodes and the noncoarse nodes by partitioning $\boldsymbol{\lambda}$ and $\ddot{\mathbf{u}}_f$ as

$$\boldsymbol{\lambda} = \left\{ \begin{array}{c} \boldsymbol{\lambda}_r \\ \boldsymbol{\lambda}_c \end{array} \right\}, \quad \ddot{\mathbf{u}}_f = \left\{ \begin{array}{c} \ddot{\mathbf{u}}_r \\ \ddot{\mathbf{u}}_c \end{array} \right\} \tag{42}$$

DOI: 10.1002/nme

to obtain

$$\begin{bmatrix} \bar{\mathbf{F}}_{rr} & \bar{\mathbf{F}}_{rc} & -\mathbf{R}_r & \mathbf{L}_r & \mathbf{0} \\ \bar{\mathbf{F}}_{cr} & \bar{\mathbf{F}}_{cc} & -\mathbf{R}_c & \mathbf{0} & \mathbf{L}_c \\ -\mathbf{R}_r^T & -\mathbf{R}_c^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{L}_r^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_c^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda}_r \\ \boldsymbol{\lambda}_c \\ \ddot{\boldsymbol{\alpha}} \\ \ddot{\mathbf{u}}_r \\ \ddot{\mathbf{u}}_c \end{Bmatrix}^{n+1} = \begin{Bmatrix} \mathbf{b}_{\lambda_r}^{n+1} \\ \mathbf{b}_{\lambda_c}^{n+1} \\ -\mathbf{R}_b^T \boldsymbol{\lambda}_{(p)}^{n+1} \\ \mathbf{0} \\ \mathbf{0} \end{Bmatrix}. \tag{43}$$

Upon eliminating the coarse localized Lagrange multipliers $\boldsymbol{\lambda}_c$, we obtain

$$\begin{bmatrix} \bar{\mathbf{F}}_{rr}^S & -\bar{\mathbf{F}}_{rc}\bar{\mathbf{F}}_{cc}^{-1}\mathbf{L}_c & -\left(\mathbf{R}_r - \bar{\mathbf{F}}_{rc}\bar{\mathbf{F}}_{cc}^{-1}\mathbf{R}_c\right) & \mathbf{L}_r \\ -\mathbf{L}_c^T\bar{\mathbf{F}}_{cc}^{-1}\bar{\mathbf{F}}_{cr} & -\mathbf{L}_c^T\bar{\mathbf{F}}_{cc}^{-1}\mathbf{L}_c & \mathbf{L}_c^T\bar{\mathbf{F}}_{cc}^{-1}\mathbf{R}_c & \mathbf{0} \\ -\left(\mathbf{R}_r - \bar{\mathbf{F}}_{rc}\bar{\mathbf{F}}_{cc}^{-1}\mathbf{R}_c\right)^T & \mathbf{R}_c^T\bar{\mathbf{F}}_{cc}^{-1}\mathbf{L}_c & -\mathbf{R}_c^T\bar{\mathbf{F}}_{cc}^{-1}\mathbf{R}_c & \mathbf{0} \\ \mathbf{L}_r^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda}_r \\ \ddot{\mathbf{u}}_c \\ \ddot{\boldsymbol{\alpha}} \\ \ddot{\mathbf{u}}_r \end{Bmatrix}^{n+1} = \begin{Bmatrix} \mathbf{b}_{\lambda_r}^{n+1} \\ \mathbf{b}_{\mathbf{u}_c}^{n+1} \\ -\mathbf{R}_b^T \boldsymbol{\lambda}_p^{n+1} \\ \mathbf{0} \end{Bmatrix}, \tag{44}$$

$$\bar{\mathbf{F}}_{rr}^S = \bar{\mathbf{F}}_{rr} - \bar{\mathbf{F}}_{rc}\bar{\mathbf{F}}_{cc}^{-1}\bar{\mathbf{F}}_{cr}, \quad \bar{\mathbf{F}}_{cc}^{-1} = \bar{\mathbf{K}}_{cc} - \bar{\mathbf{K}}_{ci}\bar{\mathbf{K}}_{ii}^{-1}\bar{\mathbf{K}}_{ic},$$

where the subscript i denotes the remaining degrees of freedom for each substructure excluding the coarse degrees of freedom.

Hence, compared with the present basic dynamic flexibility Equation (17), the preceding Equation (44) illustrates the computational overhead and implementation complexities required for performance improvements. We reiterate that we recommend the FETI-DP algorithm for production-level computations. We do submit, nevertheless, that the present simple AFETI-EI algorithm, as summarized in Table II, would serve as a parallel solver for medium-scale research and education-oriented practices as its static analysis companion algorithm employs the same solver.

7. NUMERICAL EXPERIMENTS

In this section, we solve different example problems by the AFETI-EI algorithm developed in the paper. For comparison purposes, a localized version of the dynamic FETI-D version discussed in [5, 26] has been also implemented and herein named as AFETI-D as summarized in Table III. Time integration is performed using the Newmark method with parameters $\beta = 1/4$ and $\gamma = 1/2$ to ensure unconditional stability.

We would like to point out that the computational need per iteration for the FETI-D is somewhat more expensive than needed by the proposed AFETI-EI algorithm. This point becomes evident from Table IV, where the two projectors used for AFETI-D and AFETI-EI methods are expanded. It can be observed that the projector used for AFETI-EI method, which is the same for statics and dynamics, is much simpler. What we are looking for is that the present AFETI-EI algorithm offers a comparable number of iterations as those required by the FETI-D algorithm.

Table III. Summary of labels used to identify the different partitioned dynamic solution algorithms examined.

Label	Description
AFETI-EI	Explicit–implicit time integration algorithm of Table II
AFETI-D	Localized version of the implicit two-level FETI-D method

DOI: 10.1002/nme

Table IV. Projectors used by AFETI-EI and AFETI-D.

Method	Projector
AFETI-EI	$\left[\mathcal{P}_{\mathbf{L}_f}\right] - \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\left({\mathbf{R}_b}^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)^{-1}{\mathbf{R}_b}^T\left[\mathcal{P}_{\mathbf{L}_f}\right]$
AFETI-D	$\left[\mathcal{P}_{\mathbf{L}_f}\right] - \left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\left({\mathbf{R}_b}^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{R}_b\right)^{-1}{\mathbf{R}_b}^T\left[\mathcal{P}_{\mathbf{L}_f}\right]\mathbf{F}_{bb}\left[\mathcal{P}_{\mathbf{L}_f}\right]$

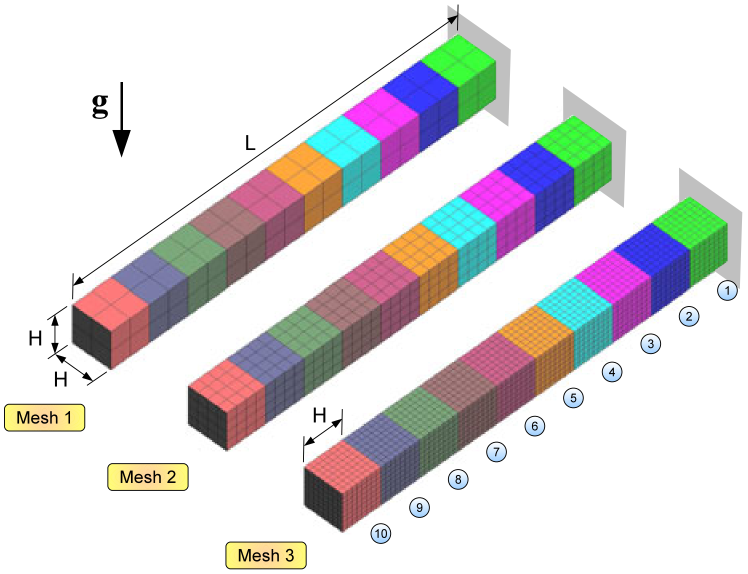

Figure 3. Three different partitioned meshes used for the three-dimensional cantilever beam problem.

In all cases, the same convergence criterion based on the relative projected residual is used:

$$\frac{\|\mathbf{w}_{k+1}\|_2}{\|\mathbf{w}_0\|_2} < \epsilon = 10^{-7}, \tag{45}$$

where $\mathbf{w}_{k+1}$ is the projected residual at iteration $(k+1)$ (Table I). The convergence results are compared for the first time step using the different dynamic algorithms proposed, and both preconditioners lumped and Dirichlet are considered.

7.1. *Three-dimensional cantilever beam*

A standard benchmark example commonly used to validate transient dynamic formulations is the bending problem of an elastic cantilever beam, in this case, approximated using 3D elastic solid elements. The longitude of the beam is $L = 10$ m, and the section is square of side $H = 1$ m. The material properties are Young modulus $E = 1 \times 10^8$ Pa, Poisson's coefficient $\nu = 0.3$, and density $\rho = 100$ kg/m^3. A Heaviside body load in the vertical direction is used to apply self-weight instantly, producing a transient bending response.

We construct first three different finite element uniform discretizations, named meshes 1, 2, and 3, as shown in Figure 3, using $N_{\mathrm{el}} = 80$ $(h = 1/2)$, $N_{\mathrm{el}} = 640$ $(h = 1/4)$, and $N_{\mathrm{el}} = 5120$ $(h = 1/8)$ eight-node isoparametric elements with three degrees of freedom per node. These meshes are partitioned into $N_{\mathrm{s}} = 10$ identical subdomains in the longitudinal direction by nine plane interfaces, producing respectively $N_{\mathrm{eq}} = 513$, $N_{\mathrm{eq}} = 1425$, and $N_{\mathrm{eq}} = 4617$ localized Lagrange multipliers, size of the partitioned system to be solved every time step.

A vibration study is first performed on the basis of mesh 3 to estimate the required time-step size, obtaining the first six natural frequencies listed in Table V. The decision of time step used for time

Table V. Numerical values of the structural natural frequencies of the cantilever beam problem computed using mesh 3. The first, second, and fifth frequencies correspond to bending modes. The fourth frequency is the compression mode, and the third and sixth frequencies are torsion modes.

Mode	Frequency (Hz)
1	1.62
2	9.71
3	14.36
4	25.08
5	25.58
6	43.08

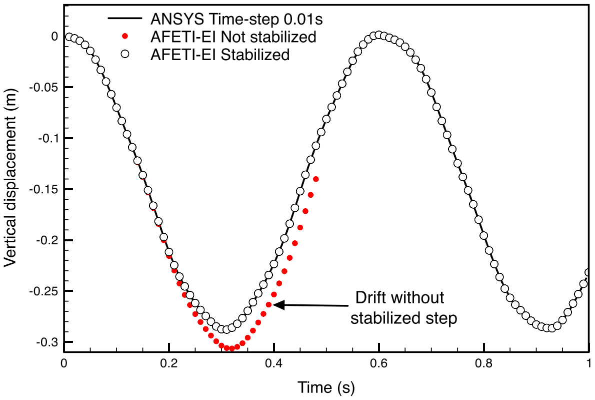

Figure 4. Cumulative error observed in the tip displacement of a cantilever beam (mesh 2 of Figure 3) when using the interface localized multipliers to solve the substructures and correct response when using the interface accelerations.

integration is based on the sixth period for the minimum $\Delta t = T_6/15 = 1 \times 10^{-3}$ s and on the first period for the maximum $\Delta t = T_1/30 = 2 \times 10^{-2}$ s. According to these limits, the time steps used are $\Delta t = 0.001$, 0.01, and 0.02 s.

The importance of the substructural acceleration computations via the filtered procedure given by (35) and (36), designated as *AFETI-EI frame displacement based solution*, is illustrated in Figure 4. On the other hand, when the acceleration computation uses (33) (labeled as *AFETI-EI lambda-based solution*), drifting occurs.

The vertical displacements at the end of the beam are calculated using AFETI-EI method for mesh 3 and plotted versus time in Figure 5 (left). These results are obtained for medium and maximum time-step sizes and compared with the solution from the commercial finite element software ANSYS (ANSYS, Inc., Canonsburg, PA, USA). It can be observed that the results are in very good agreement for the predicted range of time-step sizes.

Errors in time-integration methods can be measured in terms of period elongation and amplitude decay percentage as a function of time-step size. It is well known that the Newmark constant average acceleration method without partitioning introduces only period elongation and no amplitude decay in the solutions. By analyzing the results of Figure 5 (left) and interpolating the time-discrete solution with fourth-order splines to evaluate the period and attenuation of the solution after one cycle, it is concluded that the proposed partitioning scheme do not affect the desirable convergence characteristics of the Newmark method. The results of the analysis are represented in Figure 5 (right) revealing exactly the same percentage of period elongation than the nonpartitioned Newmark method usually measured in the literature using basic problems (see, for example, [27]).

DOI: 10.1002/nme

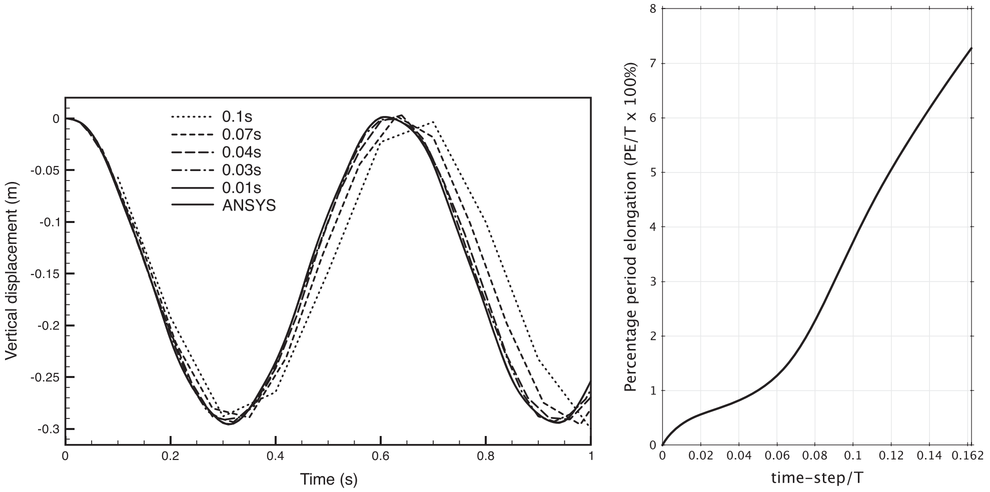

Figure 5. Dynamic response of cantilever beam (mesh 3) to self-weight body load computed with AFETI-EI algorithm. Comparison with ANSYS direct solution for different time-step sizes $\Delta t = 0.01$ s → 0.1 s (left). Time-integration error expressed in terms of percentage period elongation obtained in the solution (right).

Table VI. Results of three-dimensional cantilever beam problem with $N_s = 10$ partitions. Number of iterations for a relative residual $\epsilon < 10^{-7}$ with different time-step sizes and preconditioners for two AFETI methods.

						AFETI-D	AFETI-EI
H/h	N_{el}	N_s	Preconditioner	0.01 s	0.001 s	0.01 s	0.001 s
2	80	10	Lumped	6 iterations	6 iterations	6 iterations	6 iterations
			Dirichlet	5 iterations	4 iterations	5 iterations	5 iterations
4	640	10	Lumped	9 iterations	7 iterations	8 iterations	8 iterations
			Dirichlet	6 iterations	5 iterations	6 iterations	6 iterations
8	5120	10	Lumped	12 iterations	11 iterations	12 iterations	11 iterations
			Dirichlet	6 iterations	5 iterations	6 iterations	6 iterations

The performance results for fixed number of partitions $N_s = 10$ with increasing number of elements are reported in Table VI for different preconditioners and two time-step sizes. It is noted that the convergence of the AFETI-D method is identical to that of the AFETI-EI method. The results also show that the number of iterations needed by these methods are weakly dependent on time-step and mesh size when the lumped preconditioner is used. The use of Dirchlet preconditioner makes both methods insensitive to mesh refinements and time-step sizes for this problem.

Remark

The Courant explicit time-step limit (CFL limit) for mesh 3 is determined as $\delta t = L_e/c = 1.25 \times 10^{-4} s$, where $c = \sqrt{E/\rho}$ is the longitudinal one-dimensional wave speed and L_e the element size. Thus, the ratio of the CFL limit to the time steps used for implicit integration $\Delta t = 0.001, 0.01, 0.02$, and 0.04 s are $\Delta t/\delta t = 8$, 80, 160, and 320, respectively. This confirms the accuracy as well as the unconditional stability of the present AFETI-EI algorithm, because the DLP equations that govern rigid motions are associated with zero-frequency contents. In addition, considering that the number of iterations needed by the implicit and explicit–implicit AFETI-EI algorithm to solve one time step using the lumped preconditioner is around 10, we conclude that implicit method becomes economical with time step $\Delta t = 0.01$ s.

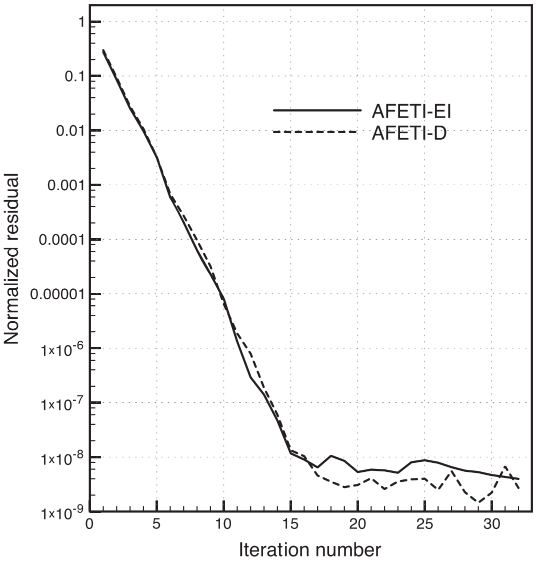

Figure 6. Evolution of the normalized residual solving first time step of cantilever beam problem (mesh 3) with different AFETI dynamic algorithms using the lumped preconditioner and time step $\Delta t = 0.02$ s.

Table VII. Number of iterations needed to drive the relative residual to a value of 10^{-7} using the lumped preconditioner for the three-dimensional cantilever beam problem. AFETI-EI uses lineal extrapolation of multipliers for prediction.

H/h	N_{el}	N_s	AFETI-D	AFETI-EI
4	640	5	8 iterations	9 iterations
		10	9 iterations	8 iterations
		20	11 iterations	10 iterations
8	5120	5	12 iterations	11 iterations
		10	12 iterations	12 iterations
		20	18 iterations	18 iterations

The performance results reported in Table VI highlight again the fact that the implicit AFETI-D algorithm has essentially the same convergence rate as the present AFETI-EI method. This conclusion is also supported by the convergence curves given in Figure 6 for the mesh 3 case with $\Delta t = 0.02$ s and using the lumped preconditioner.

Clearly, the results summarized in Table VII demonstrate that the AFETI-EI method exhibits the same convergence behavior as the AFETI-D, performing in all cases the same number of iterations plus/minus one. However, the iterations of AFETI-D method are computationally more expensive than those of AFETI-EI and involve interprocess communications to perform computations at the substructural level (see Table IV for a comparison of both projectors). The benefits of using a projector that enforce the global rigid-motion equilibrium by enforcing DLP equations at each iteration are also observed.

7.2. *Plate problem*

In order to study the behavior of the proposed AFETI-EI method in the iterative solution of fourth-order problems, we consider the plate bending problem introduced in [6]. The plate is assumed to have a square shape of side $L = 1$ m and thickness $t = 2$ mm with one side completely fixed. The material properties are given by its Young modulus $E = 690$ GPa, Poisson's coefficient $\nu = 0.3$, and density $\rho = 2800$ kg/m^3. It is discretized with four-node discrete Kirchhoff plate elements and subjected to self-weight body load, applied using a step function starting at time zero.

DOI: 10.1002/nme

The minimum time step used in the computations is set to $\Delta t = T_6/15 = 2 \times 10^{-3}$ s, and the maximum time step is selected as $\Delta t = T_1/30 = 2 \times 10^{-2}$ s, where T_6 and T_1 are the periods of the sixth and first natural modes of vibration of the plate obtained from Table VIII.

In the analysis, first, we fix the number of subdomains to $N_s = 16$, partition the plate into square substructures with straight interfaces of size $H = 0.25$ m, and consider three different meshes corresponding to element sizes $h = 1/40$, $h = 1/80$ and $h = 1/120$ named meshes 1, 2, and 3, as depicted in Figure 7.

Transient solution of point A, computed using mesh 1 with minimum and maximum time-step sizes, is presented in Figure 8 showing convergence rate of total displacement, velocity, and acceleration. Figure 9 represents the global multipliers, $\mathbf{R}_b^T \boldsymbol{\lambda}$, in the vertical and bending directions, affecting substructure 14 of the partitioned plate. Note that, because external forces are constant in this problem, these net multipliers are linearly related with the rigid-body accelerations $\ddot{\boldsymbol{\alpha}}$ through the DLP Equation (7).

We report in Table IX the convergence rates achieved by the transient algorithms for one time step. Numerical experiments show once again that the number of iterations needed by the AFETI-D and AFETI-EI methods are coincidental, and results of both algorithms are grouped in the same column. For a fixed number of substructures $N_s = 16$ and an increasing problem size, the AFETI dynamic methods equipped with Dirichlet preconditioner show to converge in a similar number of iterations. This is not the case when using the lumped preconditioner, where the convergence is more affected by element size. The same observation is made by Farhat *et al.* [7] demonstrating that the Dirichlet preconditioner is necessary to achieve the optimal conditioning result for plate and shell problems. However, it is computationally more expensive than the lumped preconditioner, which on the other hand, does not guarantee this optimal convergence rate.

Next, we consider a series of discretizations where the size of each substructure is fixed to 15 × 15 elements. We vary the number of substructures N_s between 25 and 64 and therefore vary the total

Table VIII. Numerical values of the structural natural frequencies of the square plate problem.

Mode	Frequency (Hz)
1	1.66
2	4.07
3	10.18
4	13.01
5	14.82
6	25.95

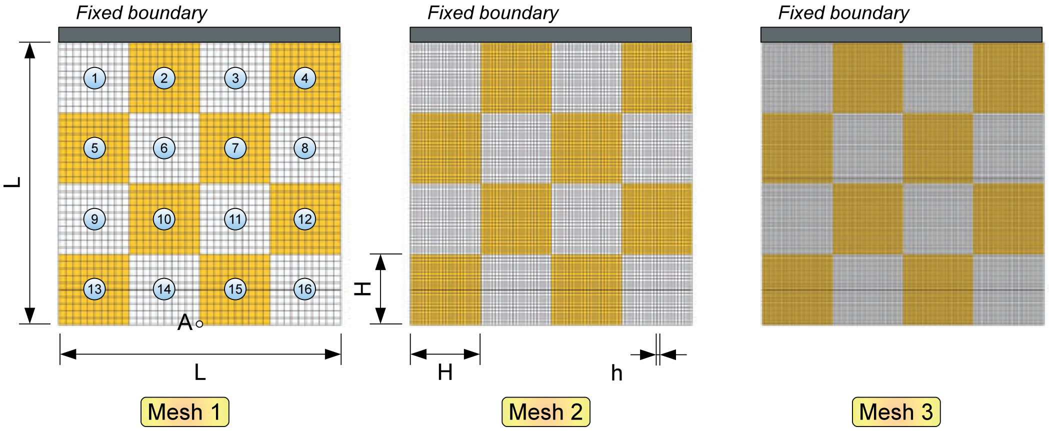

Figure 7. Discretizations used for the plate problem with fixed number of partitions $N_s = 16$ and refinement of the substructural meshes with subdomains used for the plate problem.

DOI: 10.1002/nme

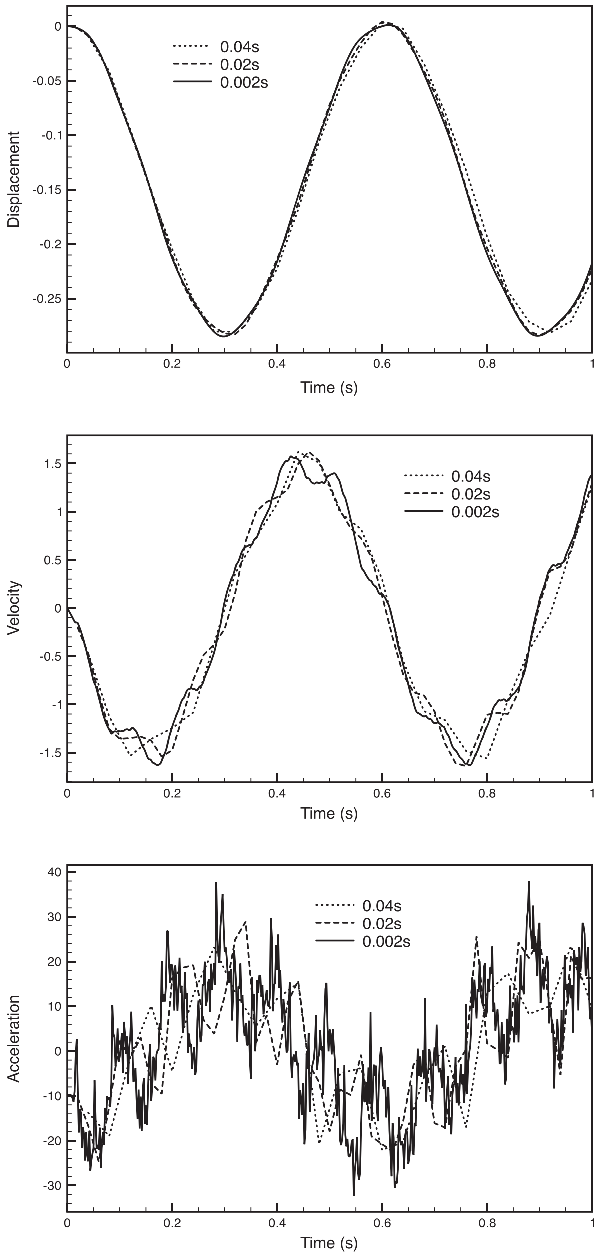

Figure 8. Dynamic response of the plate (mesh 1) to self-weight body load computed with AFETI-D and AFETI-EI algorithms. Vertical displacement, velocity, and acceleration of point A computed with time-step sizes $\Delta t = 0.002, 0.02$, and 0.04 s

number of elements in the mesh between 5625 and 14499. We report in Table X the convergence rates achieved by the present AFETI-EI and implicit AFETI-D method, the same in every case.

In this case, it can be observed that the convergence rate is seriously affected by the number of subdomains, even when using the Dirichlet preconditioner. This effect, well known in plate and shell problems [6, 7, 28], is associated to a growing condition number of the system with increasing number of subdomains. This type of behavior appearing in the discretization of biharmonic problems cannot be attributed to the proposed time-integration techniques. Solutions to this problem are extensions to enforce continuity of the transversal displacement field at the subdomain cross-points through the preconditioned conjugate gradient iterations [29, 30]. We defer this heterogeneity issue to a forthcoming paper.

DOI: 10.1002/nme

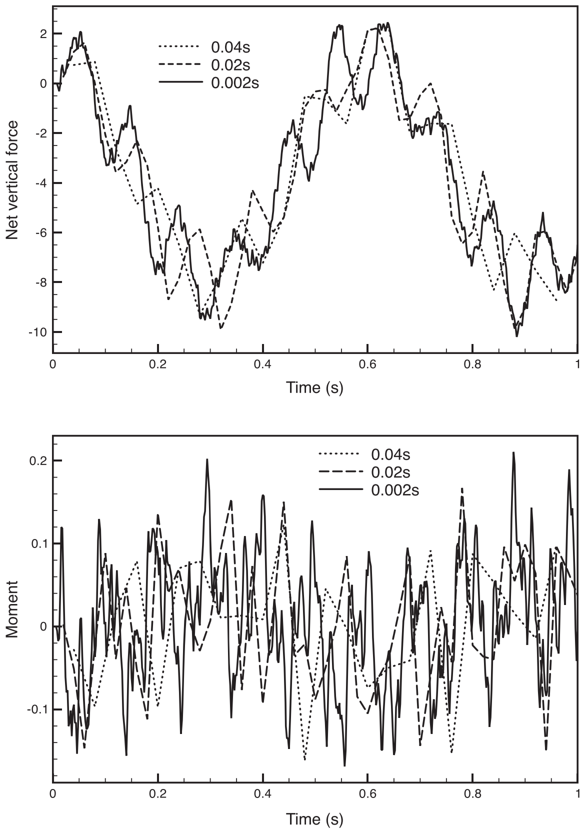

Figure 9. Resultant of multipliers acting on subdomain 14 of the plate problem (mesh 1). Total vertical force and total bending moment applied by the multipliers in the partition represented for different time-step sizes $\Delta t = 0.002, 0.02$, and 0.04 s

Table IX. Number of iterations performed to solve a time step of the plate problem with $\Delta t = 0.02$ s using different preconditioners. The number of partitions is fixed, increasing the number of elements per subdomain.

H/h	L/H	N_{el}	N_{s}	Preconditioner	AFETI-EI/D
10	4	1600	16	Lumped	70 iterations
				Dirichlet	36 iterations
20	4	6400	16	Lumped	114 iterations
				Dirichlet	38 iterations
30	4	14400	16	Lumped	144 iterations
				Dirichlet	40 iterations

Table X. Number of iterations performed to solve a time step of the plate problem with $\Delta t = 0.02$ s using different preconditioners. The number of partitions is increased maintaining the number of elements per subdomain.

H/h	L/H	N_{el}	N_{s}	Preconditioner	AFETI-EI/D
15	5	5625	25	Lumped	123 iterations
				Dirichlet	54 iterations
15	6	8100	36	Lumped	147 iterations
				Dirichlet	66 iterations
15	8	14400	64	Lumped	198 iterations
				Dirichlet	89 iterations

DOI: 10.1002/nme

7.3. Ring problem

In our last example, a more realistic engineering problem is considered. A ring of radius $R = 1$ m, thickness $t = 2$cm, and height $h = 4$cm is subject to a circumferential impulsive load along half of its perimeter and modeled as an initial velocity distribution, as represented in Figure 10. This impulsive load will excite all the structural symmetrical and antisymmetrical modes. The ring is modeled using a regular mesh of elastic eight-node brick elements, with eight elements through the thickness, three elements in the vertical direction, and a variable number of elements N_θ in the circumferential direction.

The elastic material properties correspond to steel, Young modulus $E = 210$ GPa, Poisson's coefficient $\nu = 0.3$, and density $\rho = 7850$ kg/m^3, and the distribution with angle θ of the initial velocity in the circumferential direction is given by

$$\dot{\mathbf{u}}^0(r,\theta) = v_0\cos(\theta) \quad \text{at} \quad r = R + \frac{t}{2} \quad , \quad \theta \in \left[\frac{\pi}{2}, \frac{3\pi}{2}\right], \tag{46}$$

where $v_0 = -10$ m/s corresponds to the maximum value of the initial velocity occurring at $\theta = \pi$. The explicit time-step size for this problem is $\delta t = 0.48 \times 10^{-6}$ s, computed on the basis of CFL condition using the minimum size of the element through the thickness of the ring.

Transient solution, using $N_\theta = 60$ elements along the perimeter with $N_\mathrm{s} = 4$ partitions, is presented in Figure 11 with the horizontal displacement of point A versus time for two different time-step sizes $\Delta t = 10^{-4}$ and 10^{-3} s, corresponding approximately to 200 and 2000 times the critical time step.

Convergence results obtained using the AFETI-EI algorithm endowed with a lumped preconditioner are presented in Table XI for different number of subdomains and mesh discretizations. We can observe that, for a fixed number of subdomains N_s, convergence is totally independent of the number of elements used in the circumferential direction. This is because we are maintaining constant the size of the interface. When the mesh is fixed and the number of substructures in the circumferential direction is increased to $N_\mathrm{s} = 8, 14$, and 32 subdomains, enlarging at the same time the size of the interface problem, it is found that the convergence rate is less affected than in the previous cantilever beam example problem. This effect can be explained considering that convergence is not only affected by the size of the interface but also by the aspect ratio between element and interface sizes. In this case, the size of the interface is increased but the element-interface aspect ratio is maintained constant. Note that the solution time of this problem in a parallel environment will decrease with number of subdomains N_s because the size of the substructures to be solved is smaller.

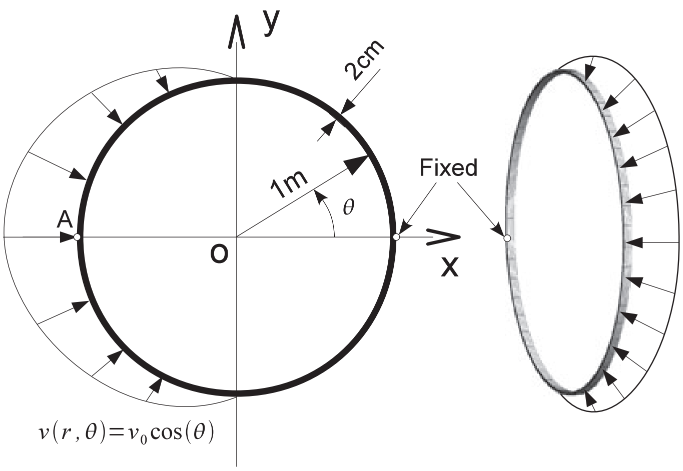

Figure 10. Ring under circumferential impulsive load.

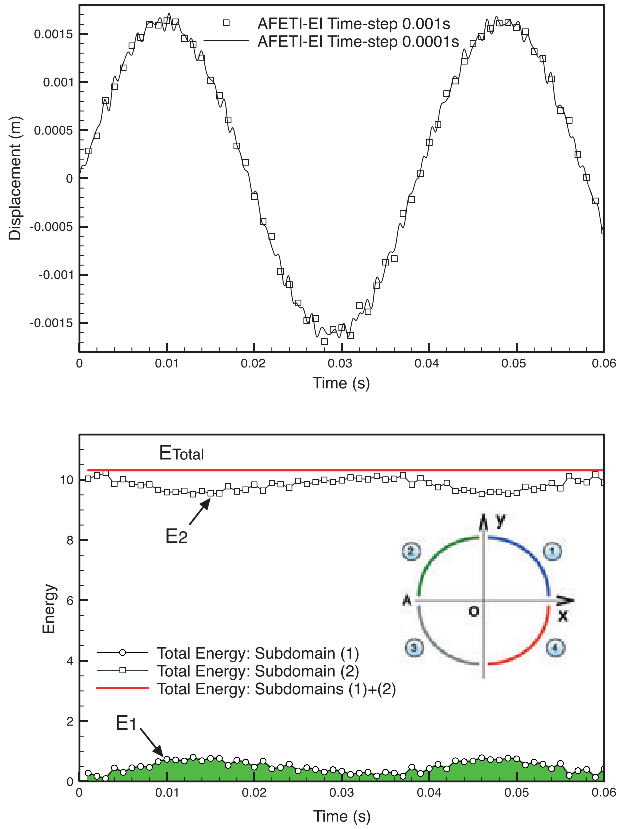

Figure 11. Solution computed with AFETI-EI algorithm using $N_s = 4$ partitions and $N_e = 1440$ elements. Dynamic response of the ring at point A. Conservation of total energy: potential plus kinetic energy of subdomains 1 and 2 of the ring (right).

Table XI. Number of iterations needed to drive the relative residual to a value of 10^{-7} using the lumped preconditioner for the ring problem. Column N_θ represents the total number of elements used in the circumferential direction. AFETI-EI uses lineal extrapolation of the multipliers in the explicit phase, and the time step used is $\Delta t = 0.001$ s.

N_θ	N_{el}	N_s	AFETI-EI
64	1536	8	11 iterations
		16	12 iterations
		32	14 iterations
128	3072	8	11 iterations
		16	10 iterations
		32	13 iterations
256	6144	8	10 iterations
		16	11 iterations
		32	13 iterations

However, it is expected that this optimal behavior exhibited by the AFETI-EI algorithm can be deteriorated in more complicated problems. Solving more realistic engineering situations, instead of academic problems with simple geometry and loads, is found to be also a cause of discrepancy between theoretical and practical convergence rates [31]. Specially, a large number of substructures using a type of discretization with bad subdomain aspect ratio combined with complicated geometry and loads still represents a challenge in domain decomposition methods.

8. CONCLUSIONS

An explicit–implicit FETI solution algorithm (AFETI-EI) has been developed, which enjoys unconditional stability and employs the same FETI-based iterative algorithm originally developed for static FETI algorithm [1]. This is accomplished by applying an explicit scheme to the modally decomposed rigid-body equations of motion (labeled as the DLP equations [13]), whereas the deformational partitions are treated by an implicit scheme. The AFETI-EI method is composed of four basic steps: explicit prediction of the rigid-body component of the localized Lagrange multipliers, implicit FETI solution of the deformational component of the localized Lagrange multipliers, correction of the rigid-body component of the Lagrange multipliers, and solution of the substructures. Conclusions and findings of this work are the following.

1. The present discrete dynamic flexibility Equation (17) consists of three coupled equations: the substructural discrete compatibility equation, the dynamic substructural self-equilibrium equation, and the interface force balance equation. A distinct feature of the present discrete dynamic flexibility equation, compared with classical FETI methods, is the enforcement of the dynamic substructural self-equilibrium condition using the substructural rigid-body modes.
2. The DLP equations that govern the rigid-body motions are characterized by zero frequency; as such, it is treated by a predictor-corrector scheme without causing numerical instability. Hence, a common solution strategy is applicable for the solution of both the dynamic and static flexibility equations.
3. This discrete dynamic partitioned flexibility Equation (17) reduces to the static direct flexibility Equation (19) previously derived by algebraic partitioning in [16, 17] and by variational framework in [14].

   For the updating the substructural acceleration, we have devised a filtering scheme exploiting the presence of the interface frame Boolean matrix, $\mathbf{L}_\mathrm{f}$, emanating from the method of localized Lagrange multipliers (Figure 1). The use of the filtered interface acceleration ($\ddot{\mathbf{u}}_\mathrm{f}$) prevents drifting in the time-integration process. This is not self-evident when employing the method of classical Lagrange multipliers.
4. Numerical experiments show that AFETI-EI method performs exactly like AFETI-D method. This is not surprising because both methods are based on the same principle, iterating on self-equilibrated multiplier fields at the implicit step. However, the AFETI-EI projector is local and much simpler to evaluate, making AFETI-EI cheaper when compared with AFETI-D. The price to pay for this simplicity is the addition of an explicit step and a correction step for the solution of the rigid-body components of the localized Lagrange multipliers.

ACKNOWLEDGEMENTS

The first author has been partially funded by the local government of Andalucía (*Junta de Andalucía*, Spain) in the framework of the project *Proyectos Investigación de Excelencia de la Junta de Andalucía* with contract number P08-TEP-03804 and the Spanish Ministry of Science (*Ministerio de Educación y Ciencia* under contract DPI2010-19331). The second author has been partially supported by WCU (World Class University) Program through the Korea Science and Engineering Foundation funded by the Ministry of Education, Science and Technology, Republic of Korea (Grant Number R31-2008-000-10045-0).

REFERENCES

1. Farhat C, Roux FX. A method of finite-element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering* 1991; **32**:1205–1227.
2. Crivelli L, Farhat C. Implicit transient finite element structural computations on MIMD systems-FETI vs. direct solvers. *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 34th and AIAA/ASME Adaptive Structures Forum, La Jolla, CA, Apr. 19-22, 1993, Technical Papers. Pt. 1 (A93-33876 13-39), p 118-130*, 1993; 118–130.
3. Farhat C, Crivelli L. A transient FETI methodology for large-scale parallel implicit computations in structural mechanics. *International Journal for Numerical Methods in Engineering* 1994; **37**:1945–1975.

DOI: 10.1002/nme

4. Farhat C, Hemez F, Mandel J. Improving the convergence rate of a transient substructuring iterative method using the rigid body modes of its static equivalent. *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 36th and AIAA/ASME Adaptive Structures Forum, New Orleans, LA*, 1995; 996–1010.
5. Farhat C, Chen PS, Mandel J. A scalable Lagrange multiplier based domain decomposition method for implicit time-dependent problems. *International Journal for Numerical Methods in Engineering* 1995; **38**:3831–3853.
6. Farhat C, Mandel J. The two-level FETI method for static and dynamic plate problems. Part I: An optimal iterative solver for biharmonic systems. *Computer Methods in Applied Mechanics and Engineering* 1998; **155**:129–151.
7. Farhat C, Chen PS, Mandel J, Roux FX. The two-level FETI method. Part II: Extension to shell problems, parallel implementation and performance results. *Computer Methods in Applied Mechanics and Engineering* 1998; **155**:153–179.
8. Farhat C, Lessoine M, LeTallec P, Pierson K, Rixen D. FETI-DP: a dualprimal unified FETI method. Part I: a faster alternative to the two-level FETI method. *International Journal for Numerical Methods in Engineering* 2001; **50**:1523–1544.
9. Kanapady R, Tamma K. A-scalability and an integrated computational technology and framework for non-linear structural dynamics. Part 1: Theoretical developments and parallel formulations. *International Journal for Numerical Methods in Engineering* 2003; **58**(15):2265–2293.
10. Kanapady R, Tamma K. A-scalability and an integrated computational technology and framework for non-linear structural dynamics. Part 2: Implementation aspects and parallel performance results. *International Journal for Numerical Methods in Engineering* 2003; **58**(15):2295–2323.
11. Farhat C, Li J, Avery P. A FETI-DP method for the parallel iterative solution of indefinite and complex-valued solid and shell vibration problems. *International Journal for Numerical Methods in Engineering* Jan. 2005.
12. Rixen DJ, Farhat C, Tezaur R, Mandel J. Theoretical comparison of the FETI and Algebraically Partitioned FETI methods and performance comparisons with a direct sparse solver. *International Journal for Numerical Methods in Engineering* 1999; **46**:501–533.
13. Park KC, Felippa CA, Ohayon R. The d'Alembert–Lagrange principal equations and applications to floating flexible structures. *International Journal for Numerical Methods in Engineering* 2009; **77**(8):72–1099.
14. Park KC, Felippa CA. A variational framework for solution method developments in structural mechanics. *Journal of Applied Mechanics* 1998; **65**(1):242–249.
15. Park KC, Felippa CA. A variational principle for the formulation of partitioned structural systems. *International Journal for Numerical Methods in Engineering* 2000; **47**:395–418.
16. Park KC, Justino MR, Felippa CA. An algebraically partitioned FETI method for parallel structural analysis: algorithm description. *International Journal for Numerical Methods in Engineering* 1997; **40**:2717–2737.
17. Justino MR, Park KC, Felippa CA. An algebraically partitioned FETI method for parallel structural analysis: performance evaluation. *International Journal for Numerical Methods in Engineering* 1997; **40**:2739–2758.
18. Park KC, Felippa CA, Gumaste UA. A localized version of the metod of Lagrange multipliers and its applications. *Computational Mechanics* 2000; **24**:476–490.
19. Felippa CA, Park KC. A direct flexibility method. *Computer Methods in Applied Mechanics and Engineering* 1997; **149**:319–337.
20. Park KC, Felippa CA, Ohayon R. Partitioned formulation of internal fluid–structure interaction problems via localized Lagrange multipliers. *Computer Methods in Applied Mechanics and Engineering* 2001; **190**(24–25):2989–3007.
21. Farhat C, Crivelli L, Roux FX. Extending substructure based iterative solvers to multiple load and repeated analyses. *Computer Methods in Applied Mechanics and Engineering* 1994; **117**:195–209.
22. Park KC. Partitioned transient analysis procedures for coupled-field problems: stability analysis. *Journal of Applied Mechanics* 1980; **47**:370–376.
23. Felippa CA, Park KC. Staggered transient analysis procedures for coupled-field mechanical systems: formulation. *Computer Methods in Applied Mechanics and Engineering* 1980; **24**:61–111.
24. Ross M, Sprague MA, Felippa CA, Park KC. Treatment of acoustic fluid–structure interaction by localized Lagrange multipliers and comparison to alternative interface coupling methods. *Computer Methods in Applied Mechanics and Engineering* 2009; **198**(9–12):986–1005.
25. Park KC, Felippa CA. Partitioned transient analysis procedures for coupled-field problems: accuracy analysis. *Journal of Applied Mechanics* 1980; **47**:919–926.
26. Farhat C, Chen PS, Risler F, Roux FX. A unified framework for accelerating the convergence of iterative substructuring methods with Lagrange multipliers. *International Journal for Numerical Methods in Engineering* 1998; **42**:257–288.
27. Bathe KJ. *Finite Element Procedures*. Prentice-Hall: Englewood Cliffs, NJ, 1996.
28. Mandel J, Tezaur R, Farhat C. A scalable substructuring method by lagrange multipliers for plate bending problems. *SIAM Journal on Numerical Analysis* Jan. 1999.
29. Rixen DJ, Farhat C. A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems. *International Journal for Numerical Methods in Engineering* 1999; **44**:489–516.
30. Gumaste UA, Park KC, Alvin KF. A family of implicit partitioned time integration algorithms for parallel analysis of heterogeneous structural systems. *Computational Mechanics* 2000; **24**:463–475.
31. Farhat C, Mandel J, Roux FX. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering* 1994; **115**:365–385.