INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2008; **0**:1–25 *Prepared using* nmeauth.cls *[Version: 2000/01/19 v2.0]*

A Time-Parallel Implicit Method for Accelerating the Solution of Nonlinear Structural Dynamics Problems

Julien Cortial[1] and Charbel Farhat[2*]

[1] *Institute for Computational and Mathematical Engineering*
[2] *Department of Mechanical Engineering and Institute for Computational and Mathematical Engineering*
Stanford University, Mail Code 3035, Stanford, CA 94305, U.S.A.

SUMMARY

The Parallel Implicit Time-integration Algorithm (PITA) is among a very limited number of time-integrators that have been successfully applied to the time-parallel solution of linear second-order hyperbolic problems such as those encountered in structural dynamics. Time-parallelism can be of paramount importance to fast computations, for example, when space-parallelism is unfeasible as in problems with a relatively small number of degrees of freedom in general, and reduced-order model applications in particular, or when reaching the fastest possible CPU time is desired and requires the exploitation of both space- and time-parallelisms. This paper extends the previously developed PITA to the nonlinear case. It also demonstrates its application to the reduction of the time-to-solution on a Linux cluster of sample nonlinear structural dynamics problems. Copyright © 2008 John Wiley & Sons, Ltd.

KEY WORDS: nonlinear dynamics, parallel computing, PITA, time-integration, time-parallel

1. INTRODUCTION

The parallelization of the time-loop of a partial differential equation solver, or time-parallelism, has in general received little attention in the literature compared to the parallelization of its space-loop. This is because time-parallelism — where the solution is computed simultaneously at different time-instances — is harder to achieve efficiently than space-parallelism, due to the inherently sequential nature of the time-integration process. However, time-parallelism can be of paramount importance to many applications. These include those time-dependent problems where the underlying computational models have either very few spatial degrees of freedom (dof) to enable any significant amount of space-parallelism, or not enough spatial dof to efficiently exploit a given large number of processors. Problems in robotics and protein folding, and more generally, time-dependent problems arising from reduced-order models

Contract/grant sponsor: National Science Foundation; contract/grant number: 0540419
*Correspondence to: Department of Mechanical Engineering, Stanford University, Mail Code 3035, Stanford, CA 94305, U.S.A.

Received 0
Revised 0
Accepted 0

usually fall in the first category. Many structural dynamics problems, even those with hundreds of thousands of dof, can fall in the second category when massively parallel computations are desired. Accelerating the time-to-solution of the first category of problems is crucial for real-time or near-real-time applications, and time-parallelism can be a key factor for achieving this objective. With the advent of massively parallel systems with thousands of cores, time-parallelism also provides a venue for reducing the time-to-solution of fixed size problems of the second category below the minimum attainable using space-parallelism alone on such systems.

Various methods based on waveform relaxation [1, 2, 3, 4, 5, 6, 7] have been proposed for enabling time-parallelism in the solution of initial boundary value problems. Most if not all of these methods are essentially inapplicable to second-order hyperbolic problems (for example, see [8]). Other methods based on transforming an initial value problem into a boundary value one have also been proposed for the same purpose [9, 10]. Such methods take advantage of the strategies that have been perfected for space-parallelism. However, they produce time-parallel solvers that are radically different from their serial counterparts.

A natural approach for exploring time-parallelism is to partition the time-domain into *time-slices*, distribute these across the target processors, and formulate a time-integration strategy that allows for some independent computations on these time-slices. To the authors' best knowledge, this approach was first suggested in [11] for the parallel solution of scalar ordinary differential equations (ODEs). Then, it was further explored in [12, 13, 14] for implementing parallel shooting methods. This natural approach to time-parallelism was also applied in [15, 16, 17] to accelerate orbital computations. It was later set in a formal time-domain decomposition framework in [18, 19, 20, 21, 22, 23], which led to the development of the so-called "parareal" method. This computational framework was further refined in [24, 25, 26], which led to the design of the Parallel Implicit Time Algorithm (PITA).

In the time-domain decomposition framework outlined above, the boundaries of the time-slices can be interpreted as the points of a *coarse time-grid*. In all methods belonging to this computational framework, low-accuracy *seed* values are first computed at the beginning of each time-slice — that is, on the coarse time-grid. Next, these seed values are used to time-advance the solution in all time-slices in an embarrassingly parallel fashion. The generated global solution inherits however the low-accuracy property of the seed values. Furthermore, this solution is discontinuous at the boundaries of the time-slices (except the first and last point of the global time-interval). For these reasons, iterative correction steps are typically introduced in the solution process to eliminate the jumps at the time-slice boundaries and improve the accuracy of the computation. The various methods presented in [11, 12, 13, 14, 18, 19, 20, 21, 22, 23, 24, 25] differ mostly in the design of the correction steps. Many of these methods, and in particular the parareal algorithm and its variants, perform the corrections by propagating information on the coarse time-grid. These methods have been shown, both theoretically and through numerical experiments, not to be suitable for the solution of second-order hyperbolic problems. Only the PITA method proposed in [25] was shown to perform well for such hyperbolic problems without resorting to artificial damping and therefore without lowering accuracy. The PITA developed in [25] was further refined in [26] where propagating information on the coarse time-grid was shown to be ineffective for linear second-order hyperbolic problems. Instead, PITA was equipped in [26] with an alternative correction procedure that operates solely on the original time-grid and maintains computational efficiency by effectively re-utilizing previously computed information. The resulting method was then successfully demonstrated for the time-parallel solution of

various linear structural dynamics problems. The main purpose of this paper is to extend this new PITA to the nonlinear case. As it will be shown in this paper, this extension is not straightforward. To this effect, the remainder of this paper is organized as follows.

In Section 2, the notations and definitions used in this paper are laid out. In Section 3, the computational framework of PITA for linear second-order hyperbolic problems is recalled to keep this paper as self-contained as possible. In Section 4, new concepts are injected in this framework to extend it to the nonlinear case. In Section 5, the nonlinear PITA is applied to the time-parallel solution of two geometrically nonlinear structural dynamics problems on a Linux cluster and the obtained performance results are discussed. Finally, a summary of this paper and appropriate conclusions are given in Section 6.

2. NOTATIONS AND DEFINITIONS

For the purpose of presentation only, the system of nonlinear ODEs of interest is written in first-order form as follows

$$\begin{aligned} &\frac{dy}{dt} = F(y,t), \\ &y(t^0) = y^0, \end{aligned} \tag{1}$$

where $y \in \mathbb{R}^{N_{dof}}$, N_{dof} denotes the number of dof or the size of problem (1), t denotes time, and $y^0 \in \mathbb{R}^{N_{dof}}$ denotes an initial condition for y. Hence, in the context of problem (1) above, a superscrit for a time quantity does not designate a power of that quantity.

Let ITA denote a preferred sequential *implicit* time-integration algorithm for solving problem (1) in the time-domain $\Omega_t = [t^0, t^f]$. Given a constant time-step Δt, Ω_t is partitioned into N_{ts} time-slices $\{S^i = [T^i,\ T^{i+1}]\}_{i=0}^{N_{ts}-1}$ of constant size ΔT, where

$$\Delta T = J\Delta t,\ J \in \mathbb{N},\ J > 1. \tag{2}$$

Furthermore, let T^{i-} denote the right boundary of the time-slice $S^{i-1} = [T^{i-1},\ T^i]$ and T^{i+} the left boundary of the time-slice $S^i = [T^i,\ T^{i+1}]$.

In the remainder of this paper, the sampling of Ω_t by Δt is referred to as the *fine* time-grid. Similarly, the sampling of the same time-domain by ΔT is referred to as the *coarse* time-grid. The correspondence between these two time-grids is given by

$$T^i = t^{Ji} = i\Delta T = Ji\Delta t,\ 0 \le i \le N_{ts}. \tag{3}$$

Hence, ΔT and Δt are referred to in the sequel as the coarse and fine time-steps, respectively, and J as their ratio.

The following additional notations are borrowed from [24]:

- A quantity associated with a k-th iteration is designated by the subscript k.
- A quantity associated with a l-th Newton iteration is designated by the subscript (l).
- A quantity evaluated on the fine time-grid at the time-instance $t^i = i\Delta t$ is denoted by a lower case and the superscript i.
- A quantity evaluated on the coarse time-grid at the time-instance $T^i = t^{Ji} = i\Delta T = Ji\Delta t$ is denoted by an upper case and the superscript i.

- A function defined in the time-slice S^i is denoted by a lower case, the superscript i, and a pair of parentheses ().

For example, the initial approximation of $Y^i = y(T^i)$ is denoted by Y^i_0.

Finally, in all algorithmic descriptions, the symbol $\oplus$ is used to designate time-parallel steps and the symbol $\ominus$ is used to designate time-sequential ones.

REMARK 1. The concept of a coarse time-grid has also recently appeared in the context of space-time multiscale methods [27] where it plays however a different role.

3. PITA FOR LINEAR SECOND-ORDER HYPERBOLIC PROBLEMS

For parabolic problems such as the unsteady heat equation and first-order hyperbolic problems such as the Euler flow equations, the standard PITA described in [24] performs well. For linear second-order hyperbolic systems such as those governing linear wave propagation and structural dynamics problems, the appropriate PITA is that described in [26] and recalled herein to make this paper as self-contained as possible.

The standard semi-discrete system of ODEs governing a linear structural dynamics problem can be written as

$$\begin{aligned} &M\ddot{u} + D\dot{u} + Ku = 0, \\ &u(t^0) = u^0, \\ &\dot{u}(t^0) = \dot{u}^0, \end{aligned} \tag{4}$$

where $u(t)$ is the time-dependent displacement vector, M, D and K are the usual mass, damping and stiffness constant symmetric matrices, and a dot denotes a time-derivative. This system of ODEs can be re-written in first-order form (1) as follows

$$\begin{aligned} &\frac{dy}{dt} = F(y,t) = Ay, \\ &y(t^0) = y^0, \end{aligned} \tag{5}$$

where

$$y = \begin{pmatrix} \dot{u} \\ u \end{pmatrix}, \quad \text{and} \quad A = \begin{pmatrix} -M^{-1}D & -M^{-1}K \\ I & 0 \end{pmatrix}. \tag{6}$$

The PITA proposed in [26] for the time-parallel solution of problem (5) above can be described as follows

- $\ominus$ Step 0: *Estimate the initial values of the seeds* $Y^i_0,\ 0 \le i \le N_{ts} - 1$ *(at the beginning of each time-slice* S^i*), for instance, by solving problem* (5) *on the coarse time-grid.*

For $k = 0, 1, ...$ until convergence

- $\oplus$ Step 1: *Apply the ITA on each time-slice of the time-decomposed fine time-grid to the solution of the problem*

$$\begin{aligned} &\frac{dy^i_k()}{dt} = F\big(y^i_k(), t\big) = Ay^i_k(),\ \text{in } S^i, \\ &y^i_k(T^i) = Y^i_k,\ 0 \le i \le N_{ts} - 1. \end{aligned} \tag{7}$$

- $\oplus$ Step 2: *Evaluate on the coarse time-grid the jumps*

$$\Delta_k^i = y_k^{i-1}(T^i) - Y_k^i,\ 1 \leq i \leq N_{ts} - 1. \tag{8}$$

Stop if all these jumps are sufficiently small, in which case convergence is reached.

- $\ominus$ Step 3: *Compute the projection matrix*

$$P_k = L_k \left(L_k^T Q L_k\right)^{-1} L_k^T Q, \tag{9}$$

where the superscript T denotes here and throughout the remainder of this paper the transpose operation, Q is the metric

$$Q = \begin{pmatrix} M & 0 \\ 0 & K \end{pmatrix}, \tag{10}$$

and L_k is a matrix whose columns span the subspace

$$\mathcal{L}_k = \text{Span}\left\{Y_l^m; 0 \leq l \leq k,\ 0 \leq m \leq N_{ts} - 1\right\}. \tag{11}$$

- $\ominus$ Step 4: *Compute on the coarse time-grid the correction coefficients*

$$C_k^i = c_k(T^{i-}),\ 1 \leq i \leq N_{ts} - 1, \tag{12}$$

where c_k is the solution on the decomposed fine time-grid of the (projected) correction problem

$$\begin{aligned} &\frac{dc_k}{dt} = \nabla_y F(y_k, t) c_k = A c_k, \\ &c_k(T^0) = 0, \\ &c_k(T^{i+}) = P_k\left(c_k(T^{i-}) + \Delta_k^i\right),\ 1 \leq i \leq N_{ts} - 1. \end{aligned} \tag{13}$$

- $\oplus$ Step 5: *Update the seed values*

$$Y_{k+1}^i = y_k^{i-1}(T^i) + C_k^i = Y_k^i + \Delta_k^i + C_k^i,\ 0 \leq i \leq N_{ts} - 1. \tag{14}$$

REMARK 2. Step 4 outlined above shows that the corrections of the seed values are obtained by solving on the fine time-grid the same ODE governing the main problem (5), but with different initial conditions designed to propagate on the fine time-grid only a relevant component of the jumps. A priori, this is a computationally prohibitive proposition. However, the reader can also observe the following. The N_{ts} *local* problems (7) — to be solved in an embarrassingly parallel fashion — and the N_{ts} problems associated with the solution of (13) on the N_{ts} time-slices — to be solved sequentially — differ only in their initial conditions. Therefore, it is shown in [25, 26] that the projector P_k is such that the correction coefficients C_k^i can be computed without explicitly solving the the N_{ts} problems (13) on the fine time-grid.

REMARK 3. In [24], it is shown that problems (13) arise from the solution by Newton's method of the nonlinear correction problem. Here, it is noted that for linear problems, the accuracy of the initial values of the seed values Y_0^i is not critical for the convergence of the Newton iterations. The key issue is rather the quick convergence of the subspace $\mathcal{L}_k$ to a subspace containing the frequency content of the intial condition y_0. However, for nonlinear problems, the accuracy of Y_0^i can be expected to affect the convergence of Newton's iterations.

4. EXTENSION TO NON-LINEAR SECOND-ORDER HYPERBOLIC PROBLEMS

4.1. Family of nonlinear structural dynamics problems

Consider the following nonlinear finite element equations of dynamic equilibrium of a given system

$$\begin{aligned} &M\ddot{u} + f^{int}(\dot{u}, u) = f^{ext}(t), \\ &u(t^0) = u^0, \\ &\dot{u}(t^0) = \dot{u}^0, \end{aligned} \tag{15}$$

where $M \in \mathbb{R}^{N'_{dof} \times N'_{dof}}$ is the usual mass matrix, $N'_{dof} \in \mathbb{N}$ denotes the number of dofs associated with the chosen semi-discretization, $u \in \mathbb{R}^{N'_{dof}}$ is its displacement vector, $f^{int} : \mathbb{R}^{N'_{dof}} \times \mathbb{R}^{N'_{dof}} \mapsto \mathbb{R}^{N'_{dof}}$ is a nonlinear function, and $f^{ext} \in \mathbb{R}^{N'_{dof}}$ is a prescribed force vector. Let $K(\dot{u}, u) = \nabla_u f^{int}(\dot{u}, u)$ and $D(\dot{u}, u) = \nabla_{\dot{u}} f^{int}(\dot{u}, u)$ denote the tangent stiffness and damping matrices, respectively. The above problem can be recast into the equivalent first-order form (1) by introducing

$$y = \begin{pmatrix} \dot{u} \\ u \end{pmatrix}, \tag{16}$$

where $y \in \mathbb{R}^{N_{dof}}$ and $N_{dof} = 2N'_{dof}$, and defining the function F as

$$F(y, t) = \Phi(y) + b(t), \tag{17}$$

where

$$\Phi(y) = \begin{pmatrix} -M^{-1} f^{int}(y) \\ \begin{pmatrix} I_{N'_{dof}} & 0_{N'_{dof}} \end{pmatrix} y \end{pmatrix}, \; b(t) = \begin{pmatrix} M^{-1} f^{ext}(t) \\ 0 \end{pmatrix}, \tag{18}$$

$0_{N'_{dof}}$ denotes the $\mathbb{R}^{N'_{dof} \times N'_{dof}}$ null matrix, and $I_{N'_{dof}}$ denotes the $\mathbb{R}^{N'_{dof} \times N'_{dof}}$ identity matrix. The gradient of F with respect to y is

$$\nabla_y F\big(y(t)\big) = \nabla_y \Phi\big(y(t)\big) = \begin{pmatrix} -M^{-1} D\big(y(t)\big) & -M^{-1} K\big(y(t)\big) \\ I_{N'_{dof}} & 0_{N'_{dof}} \end{pmatrix}. \tag{19}$$

4.2. Nonlinear PITA framework

When an implicit scheme is chosen for time-integrating the nonlinear problem (15), a nonlinear system of algebraic equations arises at each time-step. Usually, this system is solved by a Newton-like iterative method. Since such a method typically requires several iterations to achieve convergence within a specified tolerance, it follows that in this case, the solution of PITA's *linear* correction problem (13) — even when performed on the fine time-grid — is several times computationally more economical than the solution on the same fine time-grid of the target *nonlinear* problem (15) and therefore is a relatively affordable computational overhead. Nevertheless, the extension to nonlinear problems of the linear PITA framework described in Section 3 is not straightforward.

To begin, the time-decomposed nonlinear problem (7) and the auxiliary correction problem (13) no longer share the same governing differential equation. Furthermore, the convergence of the Newton process that is intrinsic to the PITA framework can in this case

strongly depend on the initial guess. In the nonlinear case, the Jacobian $\nabla_y F\big(y_k(t)\big)$ (Eq. (19)) is a time-dependent function. Therefore, the linearized ODE system (13) does not have constant coefficients and changes at each iteration, and the *local* orthogonal projector P_k (9) should perhaps depend on the time-slice S^i.

Let L_k^i and Q_k^i denote the full-rank matrices depending on the iteration k and the time-slice S^i, and which are associated with a subspace $\mathcal{L}_k^i$ and a chosen metric, respectively, to be specified later in this paper. The Q_k^i-orthogonal projector onto $\mathcal{L}_k^i$ is

$$P_k^i = L_k^i({L_k^i}^T Q_k^i L_k^i)^{-1} {L_k^i}^T Q_k^i. \tag{20}$$

Using this projector and following the ideas exposed in Section 3, the following nonlinear PITA framework is proposed for the time-parallel solution of the nonlinear structural dynamics equations (15).

- $\ominus$ Step 0: *Estimate the initial values of the seeds* $Y_0^i,\ 0 \le i \le N_{ts}-1$.

For $k = 0, 1, ...$ until convergence

- $\oplus$ Step 1: *Apply the ITA on each time-slice of the time-decomposed fine time-grid to the solution of the problem*

$$\begin{aligned} &\frac{dy_k^i()}{dt} = F\big(y_k^i(), t\big),\ \text{in } S^i, \\ &y_k^i(T^i) = Y_k^i,\ 0 \le i \le N_{ts}-1. \end{aligned} \tag{21}$$

  This step involves at each time-instance the solution of a nonlinear system of algebraic equations.
- $\oplus$ Step 2: *Evaluate on the coarse time-grid the jumps*

$$\Delta_k^i = y_k^{i-1}(T^i) - Y_k^i, \quad 1 \le i \le N_{ts}-1. \tag{22}$$

  Stop if all these jumps are sufficiently small.
- $\ominus$ Step 3: *Compute on the coarse time-grid the correction coefficients*

$$C_k^i = c_k(T^{i-}),\ 1 \le i \le N_{ts}-1, \tag{23}$$

  where c_k is the solution on the decomposed fine time-grid of the linearized (projected) correction problem

$$\begin{aligned} &\frac{dc_k}{dt} = \nabla_y F\big(y_k(t)\big) c_k, \\ &c_k(T^0) = 0, \\ &c_k(T^{i+}) = P_k^i\big(c_k(T^{i-}) + \Delta_k^i\big),\ 1 \le i \le N_{ts}-1. \end{aligned} \tag{24}$$

- $\oplus$ Step 4: *Update the seeds*

$$Y_{k+1}^i = y_k^{i-1}(T^i) + C_k^i = Y_k^i + \Delta_k^i + C_k^i, \quad 0 \le i \le N_{ts}-1. \tag{25}$$

To complete the description of the above nonlinear PITA, it remains to discuss an efficient computation of the sequential correction step (Step 3) in order to make it an affordable computational overhead, and to specify the full-rank matrices L_k^i and Q_k^i that determine the projectors P_k^i as well as the initialization procedure (Step 0).

4.3. Efficient computation of the sequential correction step

4.3.1. Fundamental property. Let G denote the nonlinear propagator associated with a chosen one-step ITA. In the context of the solution of problem (21), and using the implicit function theorem, G can be defined by

$$y_k^{j+1} = G(y_k^j, t^j) \Leftrightarrow \Gamma(y_k^{j+1}; y_k^j, t^j) = 0, \tag{26}$$

where

$$\begin{aligned} \Gamma : \mathbb{R}^{N_{dof}} \times \mathbb{R}^{N_{dof}} \times \mathbb{R}^+ &\rightarrow \mathbb{R}^{N_{dof}} \\ (\zeta, \xi, \tau) &\mapsto \Gamma(\zeta; \xi, \tau). \end{aligned} \tag{27}$$

In other words, G advances the solution state from time t^j to time $t^{j+1} = t^j + \Delta t$.

First, consider the computation of the correction function c_k via the solution on the fine time-grid of the linearized problem (24). The corresponding linearized propagator derived from (26) is

$$\begin{aligned} & c_k^{j+1} = \nabla_y G(y_k^j, t^j) c_k^j \\ & \Leftrightarrow \nabla_\zeta \Gamma(y_k^{j+1}; y_k^j, t^j) c_k^{j+1} + \nabla_\xi \Gamma(y_k^{j+1}; y_k^j, t^j) c_k^j = 0 \\ & \Leftrightarrow c_k^{j+1} = -\left(\nabla_\zeta \Gamma(y_k^{j+1}; y_k^j, t^j)\right)^{-1} \nabla_\xi \Gamma(y_k^{j+1}; y_k^j, t^j) c_k^j. \end{aligned} \tag{28}$$

Next, suppose that a Newton-like method is chosen for solving the nonlinear equation (26) that arises at each time-step within Step 1 of the nonlinear PITA. Such a solution procedure can be described as

- Initialize $y_k^{j+1,(0)} = y_k^j$.
- For $l = 0, 1, \ldots,$ until convergence is achieved
  - evaluate the error $\Gamma(y_k^{j+1,(l)}; y_k^j, t^j)$
  - if the error is small enough, set $y_k^{j+1} = y_k^{j+1,(l)}$ and stop
  - otherwise, compute and factor the Jacobian matrix $\nabla_\zeta \Gamma(y_k^{j+1,(l)}; y_k^j, t^j)$
  - compute the increment

$$\delta y_k^{j+1,(l)} = -\left(\nabla_\zeta \Gamma(y_k^{j+1,(l)}; y_k^j, t^j)\right)^{-1} \Gamma(y_k^{j+1,(l)}; y_k^j, t^j) \tag{29}$$

  - update the iterate

$$y_k^{j+1,(l+1)} = y_k^{j+1,(l)} + \delta y_k^{j+1,(l)}.$$

From Eq. (28) and Eq. (29), it follows that the Jacobians involved in both of these equations — and therefore the Jacobians involved in both Step 3 and Step 1, respectively — are related. More specifically,

$$\nabla_\zeta \Gamma(y_k^{j+1,(l)}; y_k^j, t^j) \rightarrow \nabla_\zeta \Gamma(y_k^{j+1}; y_k^j, t^j) \text{ when } l \rightarrow \infty \tag{30}$$

at a quadratic rate when Γ is a sufficiently smooth function. Therefore, if l^* denotes the number of iterations for convergence of the Newton or Newton-like procedure associated with Eq. (29), $\nabla_\zeta \Gamma(y_k^{j+1,(l^*)}; y_k^j, t^j)$ is identical to $\nabla_\zeta \Gamma(y_k^{j+1}; y_k^j, t^j)$. However, the Newton-like loop described above is such that at convergence, the most recently updated matrix is that associated with $\nabla_\zeta \Gamma(y_k^{j+1,(l^*-1)}; y_k^j, t^j)$. Hence, for the sake of computational efficiency, this matrix is chosen here as a good approximation of that associated with $\nabla_\zeta \Gamma(y_k^{j+1}; y_k^j, t^j)$.

4.3.2. Computational strategy. The fundamental property of the nonlinear PITA highlighted in Section 4.3.1 suggests two different approaches for reducing the computational overhead associated with serial Step 3 where the correction coefficients C_k^i are evaluated.

The first approach consists in: (a) storing the tangent matrices associated with $\nabla_\zeta \Gamma(y_k^{j+1,(l^*)}; y_k^j, t^j)$, $1 \leq j \leq JN_{ts}-1$ (see Section 4.3.1), after these matrices have been factored in Step 1, and (b) reusing them in Step 3 to compute the correction coefficients c_k^{j+1} as outlined in Eq. (28). Unfortunately, this computational strategy is memory prohibitive except for very small-sized problems such as those arising from reduced-order models.

Let

$$L_k^i = \left[L_{k1}^i,\ L_{k2}^i,\ \cdots,\ L_{k\dim(\mathcal{L}_k^i)}^i\right], \tag{31}$$

$$U_k^i = C_k^i + \Delta_k^i,\ 1 \leq i \leq N_{ts}-1, \tag{32}$$

and let

$$\overline{G}_k^i = \left(\nabla G(y_k^{J(i+1)-1}) \cdots \nabla G(y_k^{Ji+j}) \cdots \nabla G(y_k^{Ji})\right), \tag{33}$$

denote the *linear* propagator associated with the solution on the decomposed fine time-grid of problem (24) in the time-slice S^i. From Eq. (20) and Eq. (32), it follows that problem (24) can be re-written as

$$\begin{aligned} &\frac{dc_k}{dt} = \nabla_y F\big(y_k(t)\big) c_k, \\ &c_k(T^0) = 0, \\ &c_k(T^{i+}) = L_k^i \alpha_k^i, \end{aligned} \tag{34}$$

where $\alpha_k^i \in \mathbb{R}^{\dim(\mathcal{L}_k^i)}$ is given by

$$\alpha_k^i = \left[\alpha_{k1}^i,\ \alpha_{k2}^i,\ \cdots,\ \alpha_{k\dim(\mathcal{L}_k^i)}^i\right]^T = \left({L_k^i}^T Q_k^i L_k^i\right)^{-1} {L_k^i}^T Q_k^i U_k^i. \tag{35}$$

Using the definition (33), the solution of the linearized problem (34) above at time $t = T^{(i+1)-}$ can be written as

$$C_k^{i+1} = c_k\big(T^{(i+1)-}\big) = \overline{G}_k^i L_k^i \alpha_k^i = \sum_{l=1}^{\dim(\mathcal{L}_k^i)} \alpha_{kl}^i \overline{G}_k^i L_{kl}^i, \tag{36}$$

where for each $1 \leq i \leq N_{ts}-1$, $\overline{G}_k^i(L_{kl}^i)$ can be pre-computed in Step 1 while advancing the solution $y_k^i()$, at the mere additional cost of J pairs of forward/backward substitutions. The latter assertion derives from Eq. (33) and the fundamental property of the nonlinear PITA established in Section 4.3.1.

Hence, the second approach for computing efficiently the sequential correction step, which is the approach adopted in this paper for performing Step 3 of the nonlinear PITA, consists in: (a) propagating during Step 1 the subspace $\mathcal{L}_k^i$ on the decomposed fine time-grid to compute

$$\overline{G}_k^i \mathcal{L}_k^i = \left(\nabla G(y_k^{J(i+1)-1}) \cdots \nabla G(y_k^{Ji+j}) \cdots \nabla G(y_k^{Ji})\right) \mathcal{L}_k^i,\ 0 \leq i \leq N_{ts}-1,\ 0 \leq j \leq J-1, \tag{37}$$

and (b) reconstructing the values of the coefficients C_k^i as

$$C_k^{i+1} = c_k(T^{(i+1)-}) = \sum_{l=1}^{\dim(\mathcal{L}_k^i)} \left[\left({L_k^i}^T Q_k^i L_k^i\right)^{-1} {L_k^i}^T Q_k^i \big(C_k^i + \Delta_k^i\big)\right] \overline{G}_k^i L_{kl}^i,\ 0 \leq i \leq N_{ts}-2. \tag{38}$$

This reduces the computational burden associated with the straightforward implementation of Step 3 of the nonlinear PITA to the mere cost of $J(N_{ts}-1)\dim(\mathcal{L}_k^i)$ pairs of forward/backward substitutions.

4.4. Selection of the subspace $\mathcal{L}_k^i$

A simple choice for $\mathcal{L}_k^i$ that is inspired by that of the linear PITA is

$$\mathcal{L}_k^i = \mathcal{L}_k = \text{Span}\left\{Y_l^j;\ 0 \le l \le k,\ 0 \le j \le N_{ts}-1\right\}, \tag{39}$$

for any time-slice S^i. However, this *global basis update* strategy can rapidly become computationally inefficient since $\dim(\mathcal{L}_k) \approx kN_{ts}$.

Alternatively, one can construct a *local basis update* strategy where the initial basis associated with $\mathcal{L}_0^i$ is chosen for any time-slice S^i as

$$\mathcal{L}_0^i = \mathcal{L}_0 = \text{Span}\left\{Y_0^l,\ 0 \le l \le N_{ts}-1\right\},\ 0 \le i \le N_{ts}-1, \tag{40}$$

and the subsequent bases are enriched at each iteration by adding exclusively local data — thus minimizing the amount of global communication between the time-slices — as follows:

$$\begin{aligned}\mathcal{L}_k^i = \mathcal{L}_{k-1}^i &+ \text{Span}\left(\left\{y_{k-1}^{i-1}(t^{J(i-1)+j}),\ 1 \le j \le J\right\} \cup \left\{Y_k^{i-1}\right\}\right) \\ &+ \text{Span}\left(\left\{y_{k-1}^{i}(t^{Ji+j}),\ 1 \le j \le J\right\} \cup \left\{Y_k^{i}\right\}\right),\ 0 \le i \le N_{ts}-1\end{aligned} \tag{41}$$

The above local basis update strategy is justified by the following observation. Since the problem of interest (15) is nonlinear, its solution at a time t^n that preceeds the time-slice S^i by more than one or two time-slices is unlikely to contain useful information for the correction of the solution within S^i. Therefore, using data from the current and previous time-slices only for correcting the solution within S^i enables in principle a more efficient parallel implementation of the nonlinear PITA without sacrificing the sought-after fast convergence rate.

4.5. Selection of the metric Q_k^i

By analogy with the linear case, Q_k^i is chosen here as

$$Q_k^i = Q^i = \begin{pmatrix} M & 0 \\ 0 & K(Y_0^i) \end{pmatrix} \tag{42}$$

in order for the chosen metric to posses the following properties that are desirable from a computational viewpoint: (1) time-slice locality, (2) invariance with respect to the iteration number, and (3) recovery of the metric used by the linear PITA in the particular case when problem (15) is linear.

5. APPLICATIONS

Here, the nonlinear PITA described in this paper is applied to the time-parallel solution of two geometrically nonlinear structural dynamics problems. These problems are of the academic

type; they have the merit of being easy to reproduce by the interested reader. The objective is to demonstrate the potential of the time-parallel PITA framework for reducing the CPU time-to-solution. Both considered problems have a relatively small number of dof. For this reason, and because the main purpose of this paper is to demonstrate time-parallelism, spatial parallelism is not combined with time-parallelism for their solution. However, as stated in the introduction of this paper, the PITA framework easily combines spatial and temporal parallelisms. For both considered problems, the midpoint rule is chosen as the ITA because of its good numerical stability for nonlinear problems and its second-order time-accuracy. All numerical computations are performed on a Linux cluster.

5.1. Pressure "blast" of a spherical cap

The three-dimensional structure considered here is the spherical cap shown in Fig. 1. It is assumed to be made of a linearly elastic material with a Young modulus $E = 7.24 \times 10^7$ and a Poisson ratio $\nu = 0.3$, and to be clamped at its boundary. Because of the two planes of symmetry of this structure, only a quarter of its domain is modeled using 900 three-noded shell elements, which generates $N'_{dof} = 2611$ unconstrained dof. At $t^0 = 0\,\mathrm{s}$, a uniform pressure $p = 1.14 \times 10^3$ per unit area is applied on the entire surface of the spherical shell. At this magnitude level, the dynamic response of the structure is in the nonlinear regime (see Fig. 2). All computational results associated with this problem are reported for the vertical displacement dof at the center of the cap (see tracked dof in Fig. 1).

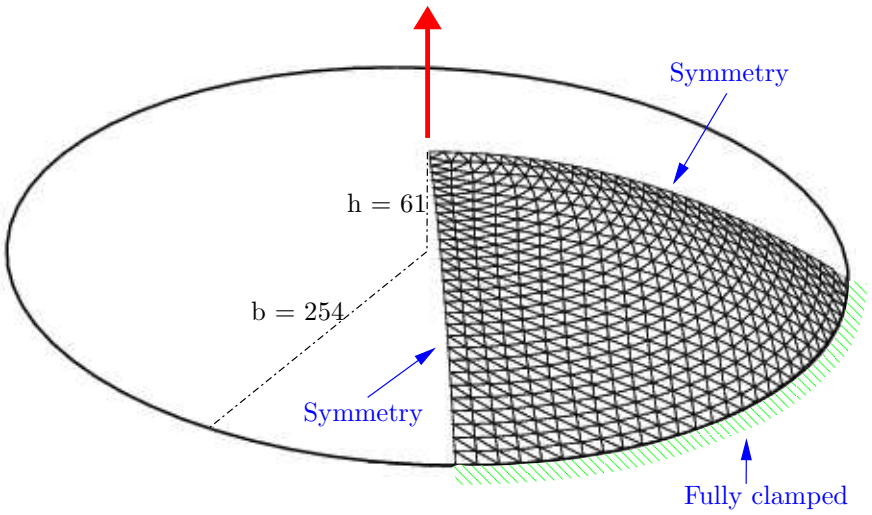

Figure 1. Spherical cap and finite element mesh of a quarter of the symmetrical domain (the red arrow designates the tracked dof).

First, a series of computations are performed using the sequential ITA to determine the convergence time-step on the fine time-grid. This time-step is found to be equal to $\Delta t = 0.1\,\mathrm{s}$ for this problem (see Fig. 3).

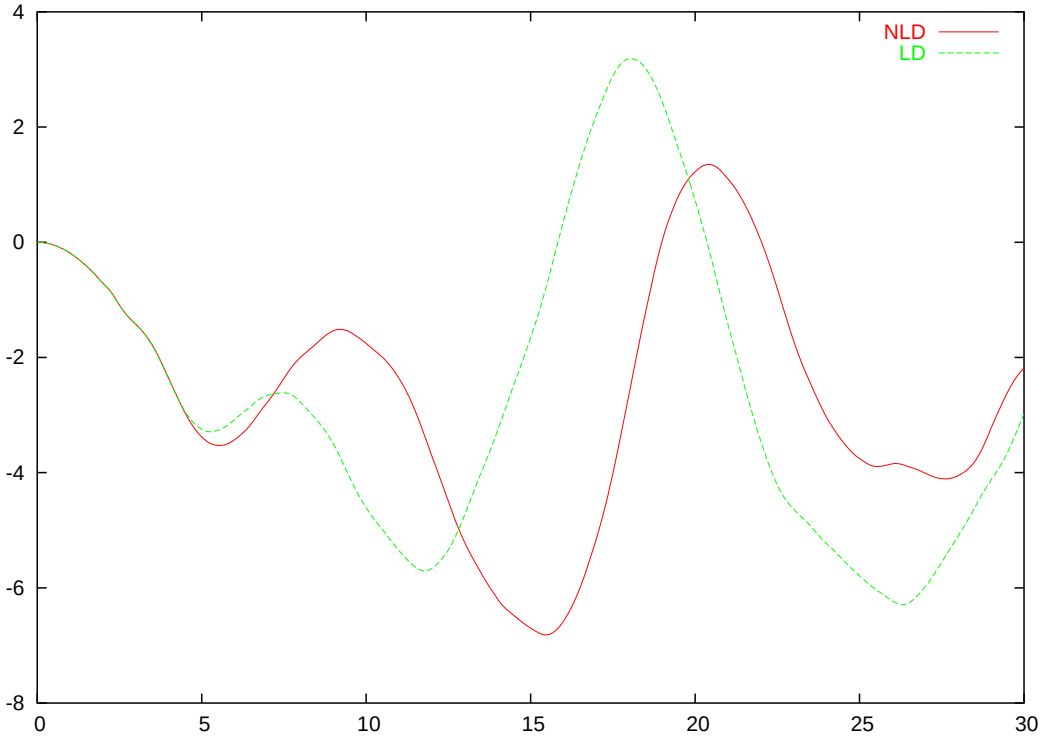

Figure 2. Spherical cap: comparison of the linear and nonlinear dynamic responses (tracked dof, $\Delta t = 0.1\,\mathrm{s}$).

Next, the nonlinear PITA is applied to the prediction of the transient response of the cap in the time-interval $[0\,\mathrm{s},\ 30\,\mathrm{s}]$. For this purpose, the fine time-grid is partitioned into $N_{ts} = 30$ time-slices and the nonlinear PITA is equipped with $\Delta t = 0.1\,\mathrm{s}$, $\Delta T = 1\,\mathrm{s}$ ($J = 10$), the global basis update strategy (39), and the proposed local metric (42). Fig. 4 reports the evolution of the jumps on the coarse time-grid of the iterate solutions, and Fig. 5 suggests convergence of the nonlinear PITA after three iterations. In general, one can state that the PITA has converged when the difference between the solution it has produced and that of its underlying sequential ITA (using the same fine time-step) is of the same order of magnitude or smaller than the time-discretization error intrinsic to the ITA. Here, the latter error is estimated by computing a reference solution of the considered problem using the ITA with a very small time-step. For this reasonable criterion, the results reported in Fig. 6 confirm the convergence in three iterations of the nonlinear PITA suggested by Fig. 5.

To highlight the importance of the metric chosen for the PITA projector (20), the nonlinear PITA computation described above is repeated using the same parameters except for the metric matrix which is set to $Q_k^i = I$. In this case, numerical instabilities develop during the iterative process (Fig. 7) and the nonlinear PITA diverges at the fourth-iteration.

On a single CPU of a Linux cluster, the chosen ITA solves the problem described above in $T^{ITA} = 58\,\mathrm{s}$. Using 30 CPUs of the same cluster — that is, one per time-slice — the nonlinear

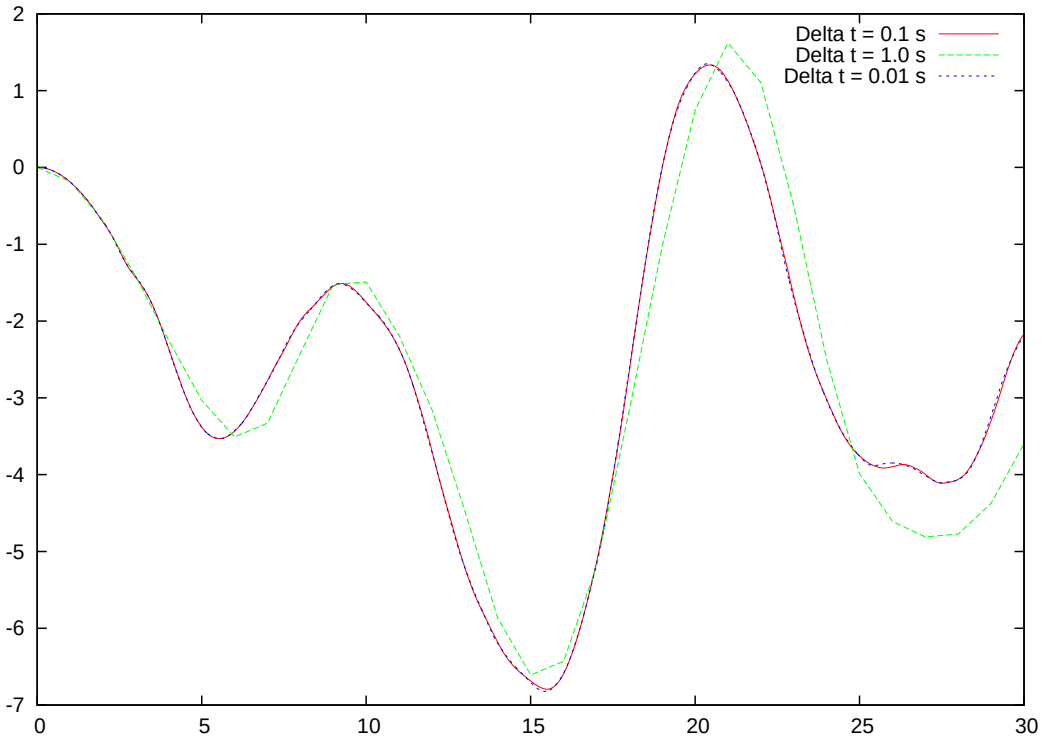

Figure 3. Spherical cap: convergence time-step of the chosen nonlinear (sequential) ITA (tracked dof).

PITA produces the converged solution in $T_{k=2}^{PITA} = 23\,\mathrm{s}$. Hence, for this nonlinear problem, PITA accelerates the time-to-solution by a factor greater than 2.5. This result is consistent with the expectation set in [24, 25, 26] for time-parallelism: to reduce the sheer time-to-solution of a given problem (or to "squeeze the most out of a given parallel computing system") and not necessarily to deliver a great parallel efficiency.

Further analysis of the above CPU performance results reveals that 12 s of the 23 s consumed by the nonlinear PITA for the solution of the problem considered here are spent in the initialization step (Step 0), which in this case is performed by applying the chosen ITA on the coarse time-grid. This CPU cost represents more than the half of the total CPU cost of the nonlinear PITA and therefore severely hampers the otherwise achievable speed-up by PITA. Indeed, if for this problem the CPU cost associated with Step 0 was negligible, the nonlinear PITA would have accelerated the time-to-solution of the ITA by a factor

$$\frac{T^{ITA}}{T_{k=2}^{PITA} - T_{Step0}^{PITA}} \geq 5. \tag{43}$$

For this reason, a faster approach for computing the initial values of the seeds is highlighted in the next section.

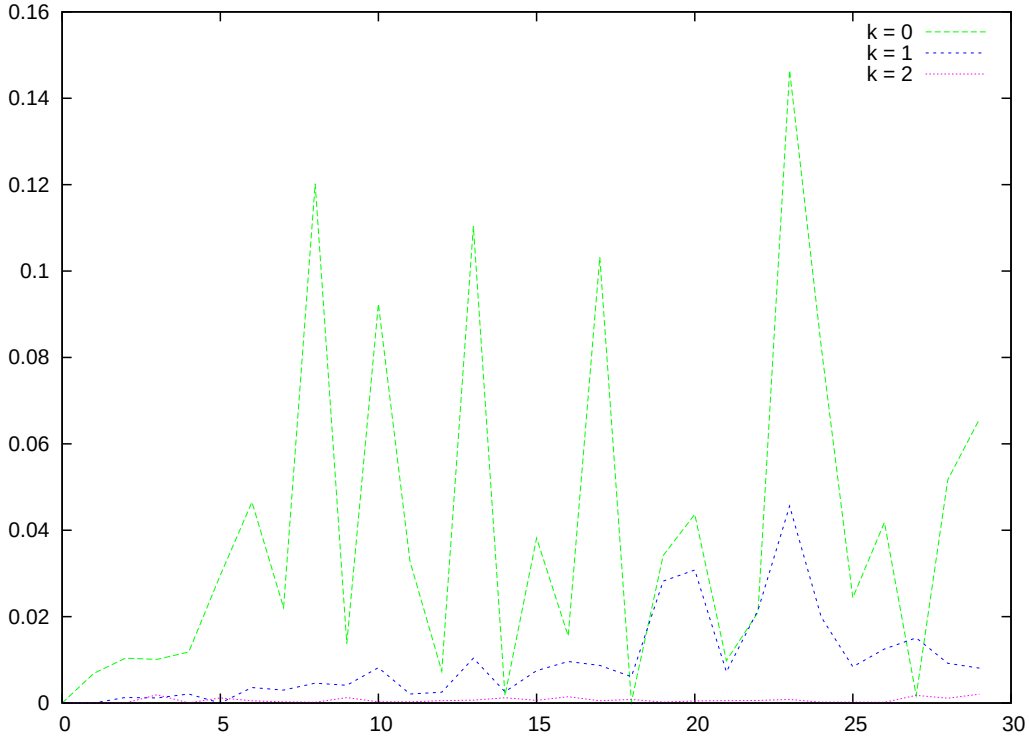

Figure 4. Spherical cap: convergence to zero of the jumps associated with the nonlinear PITA (tracked dof, $\Delta t = 0.1\,\mathrm{s}$).

5.2. Impact on a clamped-clamped plate

Fig. 8 shows a steel plate ($E = 2 \times 10^{11}\,\mathrm{N/m^2}$, $\nu = 0.3$) of length 1.0 m, width 0.2 m and thickness 0.02 m that is clamped at both ends. It is discretized by $160 \times 4 \times 2 = 1280$ 8-noded solid elements and 7155 unconstrained dof. At $t^0 = 0\,\mathrm{s}$, a line force of magnitude $F = 8 \times 10^4\,\mathrm{N/m}$ is applied at its middle, along its width. At this magnitude level, the dynamic response of the structure is in the nonlinear regime (see Fig. 9). All computational results associated with this problem are reported for the vertical displacement dof at the center of the plate and for the time-interval $[0\,\mathrm{s},\ 1.5 \times 10^{-2}\,\mathrm{s}]$.

The computational time-step on the fine time-grid is set to $\Delta t = 5.0 \times 10^{-5}\,\mathrm{s}$. Fig. 10 which compares the structural responses predicted by the chosen ITA for this time-step, a ten-fold larger one $\Delta t = 5.0 \times 10^{-4}\,\mathrm{s}$ and a ten-fold smaller time-step $\Delta t = 5.0 \times 10^{-6}\,\mathrm{s}$ shows that the value $\Delta t = 5.0 \times 10^{-5}\,\mathrm{s}$ is well-chosen.

Next, the time-interval $[0\,\mathrm{s},\ 1.5 \times 10^{-2}\,\mathrm{s}]$ is partitioned into $N_{ts} = 30$ slices so that $\Delta T = 5 \times 10^{-4}\mathrm{s}$ and $J = 10$. The nonlinear PITA is equipped with the local basis update strategy defined by (40) and (41) and the local metric (42), and is applied to the prediction of the vibrations of the plate. In this case, it converges in four iterations as shown in Fig. 11.

Next, the nonlinear PITA computation described above is repeated using the global basis

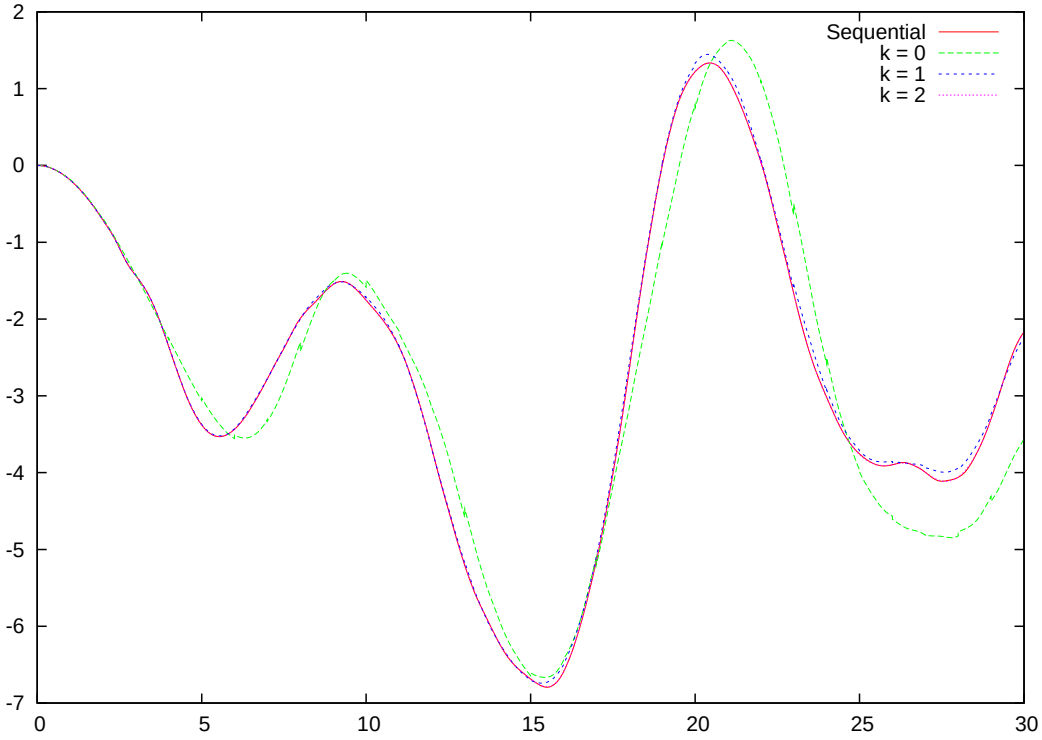

Figure 5. Spherical cap: convergence of the nonlinear PITA (tracked dof, $\Delta t = 0.1\,\mathrm{s}$).

update strategy (39). Fig. 12, which reports the magnitude of the jumps on the coarse time-grid of the solutions at the fourth iteration, reveals that when equipped with either considered local basis update strategy, the nonlinear PITA behaves almost identically. This highlights the advantage of the local basis update strategy given that it is computational more efficient that its global counterpart.

The solution of this problem by the chosen ITA on a single processor of the considered Linux cluster consumes $T^{ITA} = 172.5\,\mathrm{s}$. Using 30 processors of the same cluster and the chosen ITA on the coarse-time grid to generate the initial seeds, the solution of the same problem by the nonlinear PITA consumes $T^{PITA}_{k=3} = 81\,\mathrm{s}$ when the local basis update strategy is selected, and 83 s when the global one is chosen. Each of these two CPU timings includes 33 s spent on generating the initial solution during Step 0. Hence, in either case, PITA accelerates the time-to-solution by a factor greater than 2. To improve this speed-up factor, the nonlinear PITA computation is repeated and this time, the seeds $Y^i_0,\ 0 \leq i \leq N_{ts} - 1$ are first computed on a coarser mesh with $80 \times 4 \times 2 = 640$ 8-noded solid elements (3555 unconstrained dof) then interpolated on the original mesh. Using this strategy for implementing Step 0 which was first discussed in [26], the solution of the problem described above by the nonlinear PITA consumes $T^{PITA}_{k=3} = 69\,\mathrm{s}$ on 30 processors, which corresponds to a speed-up factor greater than 2.5.

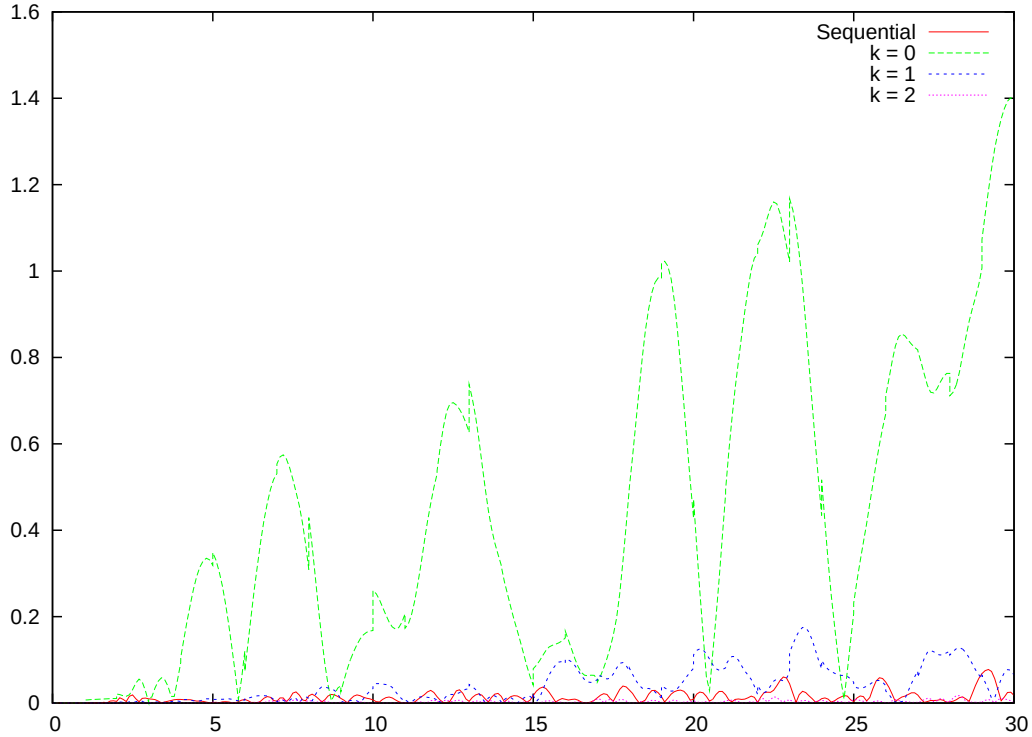

Figure 6. Convergence of the proposed nonlinear PITA framework: comparison of the approximation error of the PITA (difference between the PITA and ITA solutions) and the time-discretization error of the ITA (tracked dof, $\Delta t = 0.1\,\mathrm{s}$).

6. SUMMARY AND CONCLUSIONS

In applications involving a relatively small number of degrees of freedom (i.e. robotics, multibody dynamics, protein folding, etc.), the spatial computations within each time-step offer little prospect for parallelism or for utilizing efficiently a large number of available processors. This lack of scalabilty becomes a significant drawback when the spatial domain is well resolved but massively parallel ressources are available for reducing the time-to-solution. It becomes even more critical when real-time response is to be achieved. In both cases, time-parallelism becomes desirable even if realized at the expense of parallel efficiency. However, time-stepping is sequential in nature. Therefore, parallelizing a computation across the time domain has always been a challenge. In a previous work, a unique Parallel Implicit Time-Integration (PITA) methodology was developed for the case of linear structural dynamics problems. In this paper, a nonlinear extension of this PITA framework is presented. Its application to the time-parallelization of sample transient dynamic plate and shell problems reveals a potential for significantly reducing the time-to-solution associated with nonlinear structural dynamics analyses.

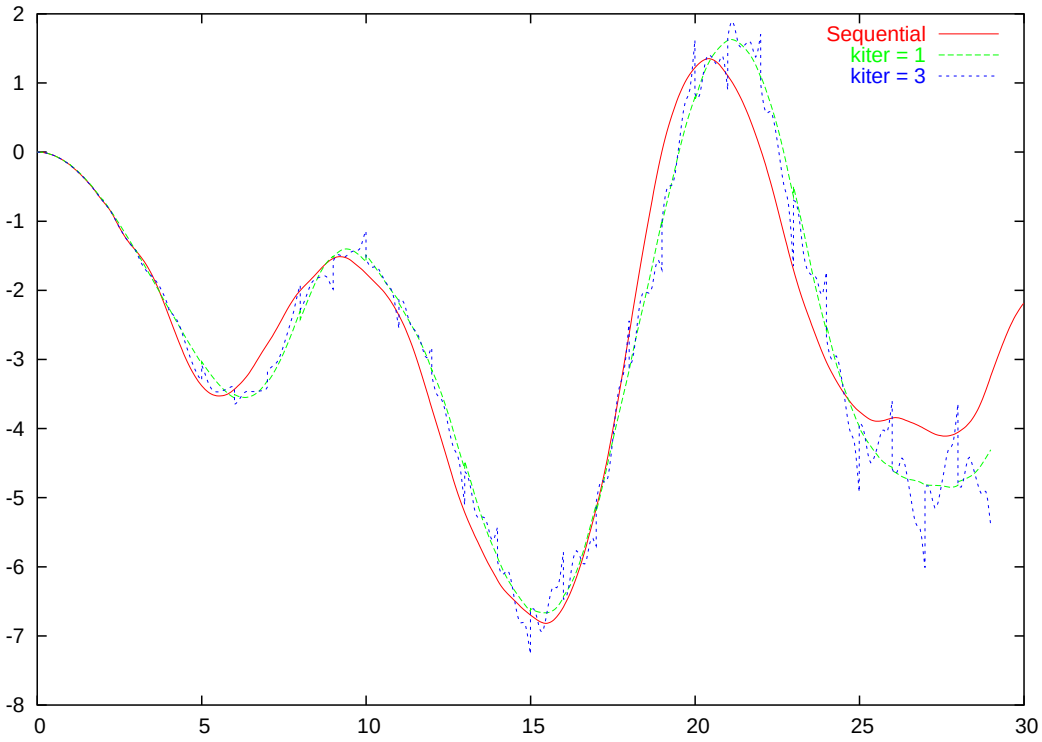

Figure 7. Spherical cap: numerical instabilities experienced by the nonlinear PITA when equipped with $Q^i = I$ (tracked dof, $\Delta t = 0.1\,\mathrm{s}$).

REFERENCES

1. J. White, A. Sangiovanni-Vincentelli, F. Odeh, and A. Ruehli. Waveform relaxation: Theory and practice. *Trans. Soc. Comput. Simul.*, 2:95–133, 1985.
2. A. Lumsdaine and D. Wu. Krylov subspace acceleration of waveform relaxation. *SIAM J. Numer. Anal.*, 41(1):90–111, 2003.
3. Stefan Vandewalle and Robert Piessens. On dynamic iteration methods for solving time-periodic differential equations. *SIAM Journal on Numerical Analysis*, 30(1):286–303, 1993.
4. K. Burrage. Parallel methods for initial value problems. *Appl. Numer. Math.*, 11(1–3):5–25, 1993.
5. S. Vandewalle and E. Van de Velde. Space-time concurrent multigrid waveform relaxation. *Ann. Numer. Math.*, 1:347–360, 1994.
6. G. Horton, S. Vandewalle, and P. Worley. An algorithm with polylog parallel complexity for solving parabolic partial differential equations. *SIAM J. Sci. Comput.*, 16:531–541, 1995.
7. G. Horton and S. Vandewalle. A space-time multigrid method for parabolic PDEs. *SIAM J. Sci. Comput.*, 16:848–864, 1995.
8. S. Ta'asan and H. Zhang. Fourier-Laplace analysis of multigrid waveform relaxation method for hyperbolic equations. *BIT*, 36(4):831–841, 1996.
9. P. Amodio and L. Brugnano. Parallel implementation of block boundary value methods for ode's. *J. Comput. Appl. Math.*, 78:197–211, 1997.
10. L. Brugnano and D. Trigiante. *Solving differential problems by multistep initial and boundary value methods*, volume 6 of *Stability and Control: Theory, Methods and Applications*. Gordon and Breach Science Publishers, Amsterdam, 1998.
11. J. Nievergelt. Parallel methods for integrating ordinary differential equations. *Commun. ACM*, 7(12):731–

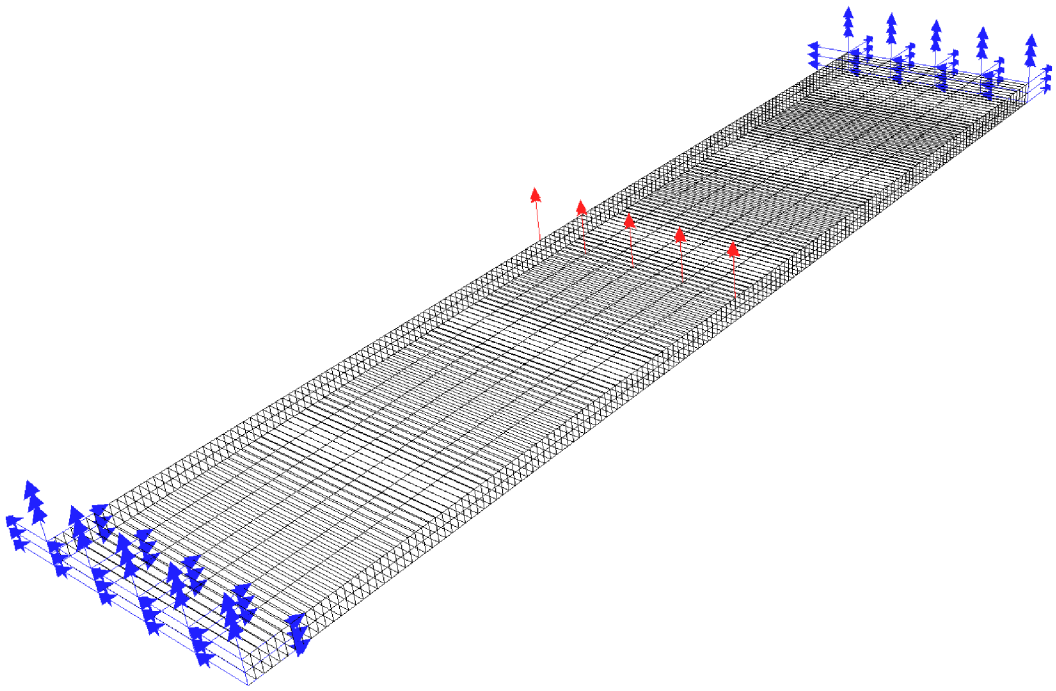

Figure 8. Clamped-clamped plate: finite element mesh (the blue arrows indicate the constrained dof and the red arrows indicate the uniform line loading).

733, 1964.
12. M. Kiehl. Parallel multiple shooting for the solution of initial value problems. *Parallel Computing*, 20(3):275–295, 1994.
13. A. Bellen and M. Zennaro. Parallel algorithms for initial-value problems for difference and differential equations. *J. Comput. Appl. Math.*, 25:341–350, 1989.
14. P. Chartier and B. Philippe. A parallel shooting technique for solving dissipative ODE's. *Computing*, 51(3-4):209–236, 1993.
15. J. Erhel, B. Philippe, and S. Rault. Parallélisation de calculs d'orbites. Rapport de recherche 3150, INRIA, 1997.
16. S. Rault. *Algorithmes Parallèles pour le Calcul d'Orbites*. PhD thesis, Université Rennes I, 1998.
17. P. Saha, J. Stadel, and S. Tremaine. A parallel integration method for solar system dynamics. *Astron. J.*, 114:409–415, 1997.
18. J. L. Lions, Y. Maday, and G. Turinici. Résolution d'EDP par un schéma en temps "pararéel". *C. R. Acad. Sci. Paris Ser. I Math.*, 332(7):661–668, 2001.
19. Y. Maday and G. Turinici. A parareal in time procedure for the control of partial differential equations. *C. R. Acad. Sci. Paris Ser. I Math.*, 335(4):387–392, 2002.
20. G. Bal and Y. Maday. A "parareal" time discretization for non-linear PDE's with application to the pricing of an American put. *Recent developments in domain decomposition methods (Zürich, 2001), Lect. Notes Comput. Sci. Eng.*, 23:189–202, 2002.
21. G. Bal. On the convergence and the stability of the parareal algorithm to solve partial differential equations. In R. Kornhuber, R.H.W. Hoppe, D.E. Keyes, J. Périaux, O. Pironneau, and J. Xu, editors, *Domain Decomposition Methods in Science and Engineering*, volume 40 of *Lect. Notes Comput. Sci. Eng.*, pages 425–432. Springer, 2004.
22. M. Gander and S. Vandewalle. Analysis of the parareal time-parallel time-integration method. Report TW 443, Katholieke Universiteit Leuven, 2005.
23. G. Bal and Q. Wu. Symplectic parareal. In Ulrich Langer, Marco Discacciati, David E. Keyes, Olof B.

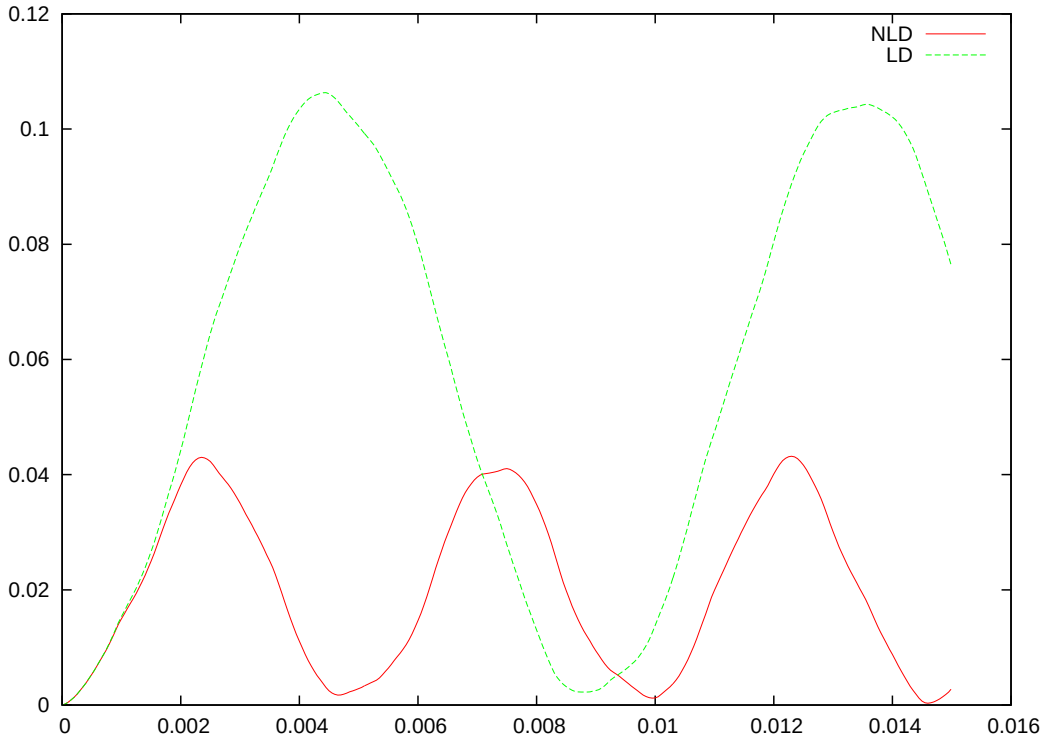

Figure 9. Clamped-clamped plate: comparison of the linear and nonlinear dynamic responses (tracked dof, $\Delta t = 5.0 \times 10^{-5}$ s).

Widlund, and Walter Zulehner, editors, *Domain Decomposition Methods in Science and Engineering XVII*, volume 60 of *Lect. Notes Comput. Sci. Eng.*, pages 401–408. Springer, 2008.

24. C. Farhat and M. Chandesris. Time-decomposed parallel time-integrators – part I: Theory and feasability studies for fluid, structure, and fluid-structure applications. *Int. J. Numer. Methods Eng.*, 58(9):1397–1434, 2003.
25. J. Cortial and C. Farhat. A time-parallel implicit methodology for the near-real-time solution of systems of linear oscillators. In L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Wanders, editors, *Real-Time PDE-Constrained Optimization*. Springer, 2006.
26. C. Farhat, J. Cortial, C. Dastillung, and H. Bavestrello. Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses. *Int. J. Numer. Methods Eng.*, 67(5):697–724, 2006.
27. A. Li, H. Waisman, and J. Fish. A space-time multiscale method for molecular dynamics simulation of biomolecules. *Int. J. Multiscale Comput. Eng.*, 4(5–6):791–803, 2006.

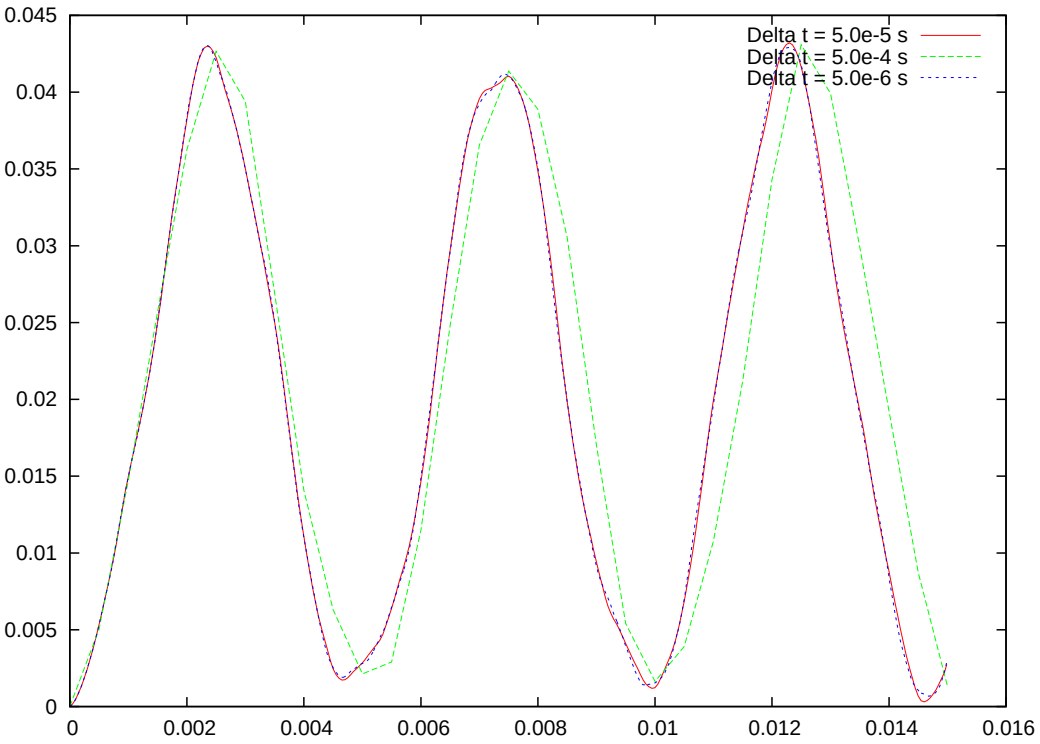

Figure 10. Clamped-clamped plate: convergence time-step of the chosen nonlinear (sequential) ITA (tracked dof).

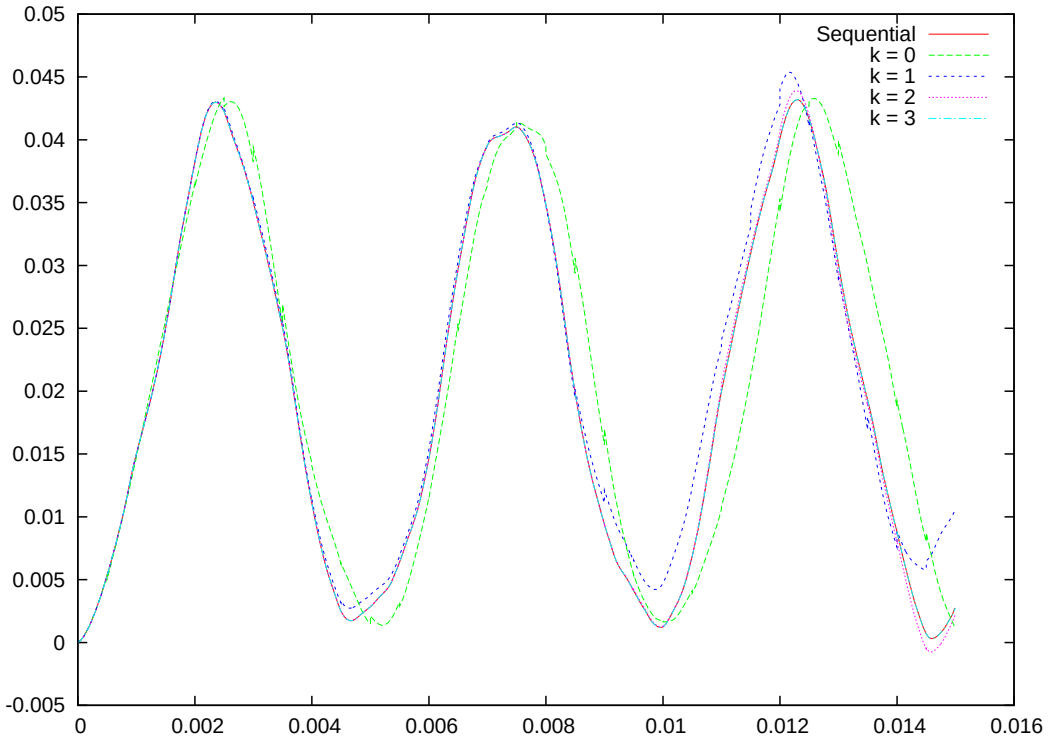

Figure 11. Clamped-clamped plate: convergence of the nonlinear PITA (tracked dof, $\Delta t = 5.0 \times 10^{-5}$ s).

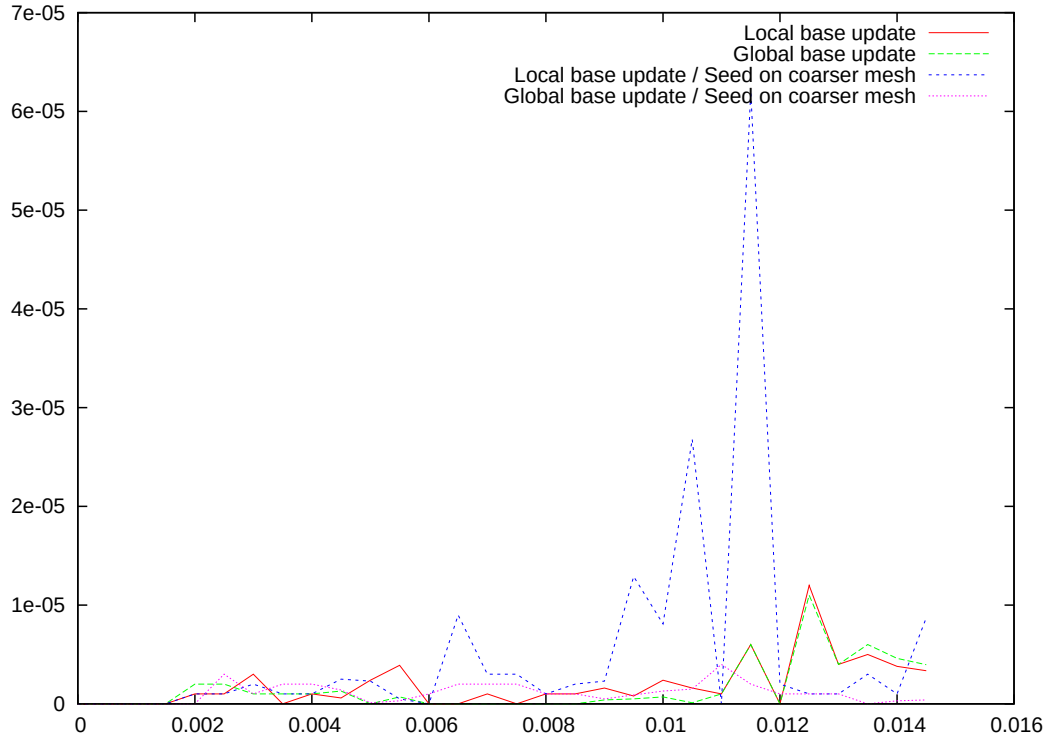

Figure 12. Clamped-clamped plate: magnitude of the jumps associated with the nonlinear PITA after four iterations using two different basis update strategies (tracked dof, $\Delta t = 5.0 \times 10^{-5}$ s)

.