Event-driven integration of linear structural dynamics models under unilateral elastic constraints

— Includes correction of typos —

Alexandre Depouhon[a,b], Emmanuel Detournay[a], Vincent Denoël[b,†]

[a] Department of Civil Engineering, University of Minnesota, USA
[b] Department of Architecure, Geology, Environment and Constructions, Structural Engineering Division, University of Liège, Belgium
[†] Corresponding author: V. Denoël, +32 (0) 43662930, +32 (0) 472 666 890, v.denoel@ulg.ac.be, University of Liege, Department of Architecture, Geology, Environment and Constructions, Structural Engineering Division, Chemin des Chevreuils, 1, B52/3, 4000 Liege, Belgium

0. Nomenclature

Conventions apply throughout the paper.

- Vectors and matrices are denoted by bold lower and upper case characters, *e.g.*, $\mathbf{v} \in \mathbb{R}^m = \mathbb{R}^{m\times 1}$ and $\mathbf{M} \in \mathbb{R}^{m\times n}$.
- Indices i, j are used to denote component extraction from vectors and matrices
  - v_i refers to the i^{th} entry of vector $\mathbf{v}$;
  - $\mathbf{M}_i$ denotes the i^{th} column of matrix $\mathbf{M}$;
  - M_{ij} indicates the scalar entry of matrix $\mathbf{M}$ located at row i and column j.
- Indices k, n indicate indexation with respect to time, *i.e.*, $x_k := x(t_k)$, $x_n := x(t_n)$.
- Indices a, b refer to body identification in multibody systems.
- Calligraphic letters are used to denote compact integer sets. For instance, the set of indices 1 to q is referred to as $\mathcal{C} = \{1, \cdots, q\}$.
- We write the vertical concatenation of column vectors as $\mathbf{x} = [\mathbf{a}\,;\,\mathbf{b}]$.

1. Introduction

The development of methods for the numerical handling of unilateral constraints in a dynamic setting is of the highest interest for modeling today's complex engineering applications. Fruits of intense research activities in computational mechanics, several approaches have been proposed. Each of them aims at guaranteeing the quality/accuracy, the robustness and the stability of the integration procedure, in the most efficient manner. Two main families of methods are designed for the handling of unilateral constraints [1]: formulations that exactly enforce the non-interpenetration condition between contacting surfaces, *e.g.*, methods relying on Lagrange multipliers [2] or time-stepping schemes [3], and penalty-based methods that relax it by introducing a constitutive contact model relating the interpenetration to the contact force [4]. Covering both families, Doyen et al. [5] propose a review of several methods, as applied to the dynamic Signorini problem; that is, the 1-dimensional longitudinal impact of an elastic bar against a rigid wall.

This paper is concerned with the problem of handling unilateral elastic constraints in combination with linear dynamics problems (but for the contact interaction). Such constraints arise, for instance, from the use of a quadratic potential to penalize body interpenetration. They also show up in the modeling of percussive drilling—the main driver to this research—to represent the bit/rock interaction

law [6, 7, 8]. However, in the latter problem, their (de)activation is not solely driven by the normal gap (distance between the contact interfaces), but also by functions depending on the velocity field [8]. With that application in mind, event-driven integration imposes itself as the *ad hoc* integration procedure. Although it can be interpreted as a form of active set strategy for the handling of contact constraints, the use of event-driven schemes is nonetheless more to be found in the fields of non-smooth dynamics and hybrid (switched) systems rather than in computational contact mechanics [9, 10, 11].

With a growing interest for these disciplines over the last decades, substantial contributions have been made on the theoretical and numerical aspects, *e.g.*, the pioneering work of Filippov [12] on discontinuous ordinary differential equations (ODEs) or the extensive review tutorial on hybrid dynamical systems by Göbel et al. [13]. This growing research momentum has also led to the development of software simulation packages, such as INRIA's SICONOS simulation framework for the time integration of mechanical systems subject to unilateral contacts and Coulomb friction [14] or the Computational Continuation Core (COCO) for the continuation of non-smooth problems developed by Dankowicz and co-workers [15]. However, to the authors' knowledge, there are no dedicated algorithms or software packages for the time integration of linear structural dynamics problems under unilateral elastic constraints that can be easily extended to trigger functions that depend on the velocity field, which motivates the present research work.

Particular to non-smooth systems is the occurrence of events along the system trajectory. At these specific points, the trajectory loses smoothness and can even exhibit discontinuities in the presence of reset maps [16, Chap. 2], thereby yielding a piecewise smooth trajectory. Examples of such events are, for instance, the impacts of a bouncing ball [17], the collisions between granular particles [18], or the switches between sticking and sliding phases in systems subject to friction [19]. Event-driven integration schemes are integration procedures [9, 10, 11] that capture the exact instants (up to a numerical tolerance) at which these events take place. They enable a simple treatment of the sequential combination of continuous-time and discrete-time dynamics that arise in hybrid systems, such as switches between governing equations during integration. Although very accurate and versatile (any standard integration scheme with root-solving can be used), these procedures can, nevertheless, become time-consuming should the density of events be important along the system trajectory [17, 20], as is the case when constraint chatter takes place. They can even fail in case the system exhibits Zeno behavior, *i.e.*, an infinite number of events in finite time; the naming convention follows from Zeno's paradoxes [21]. Also, these procedures can be the subject of accuracy and robustness issues if the numerical tolerances with which event localization is performed are inadequately chosen [17].

The root-solving capability, or the capacity of an integration scheme to locate during the procedure the zeros of state- and time-dependent algebraic equations defining events, is a standard feature on most solver packages for systems of first-order ODEs, *e.g.*, [22]. Several approaches exist [23, 24, 25] but, mostly, advanced solvers exploit a continuous extension of the discrete solution to detect and localize events: a polynomial approximating with sufficient accuracy the continuous solution on the basis of the discrete one is constructed and used for event detection and localization [26, 27]. These prove best at balancing the responses to the two main challenges faced by root-solving [27]: robustness and efficiency. Procedures must indeed guarantee that events shall not be overlooked and that their localization can be performed with a reasonable computational effort. Nonetheless, to our best knowledge, no such continuous extensions exist for the schemes dedicated to the integration of the second-order equations of motion arising in mechanics, such as the well-known Newmark [28] and α-like methods [29, 30, 31]. Embedding these schemes in event-driven integration procedures therefore requires the extension of root-solving techniques.

The main purpose of this paper is the establishment of a numerical procedure for the time integration of linear dynamics problems under unilateral elastic constraints. That goal is reached by embedding structural one-step integration schemes in an event-driven procedure that relies on a continuous extension of the displacement field using cubic Hermite interpolation. In particular, the procedure can also serve for the numerical integration of contacting bodies via the penalty method (equivalent to stiff elastic unilateral constraint), as long as the underlying problem remains linear and elastic. As is demonstrated both analytically and numerically, a salient property of the proposed algorithm is the prevention of the typical integration instabilities that arise from the use of the penalty method for the handling of contact interactions. Indeed, the coupling of the penalty method with time integration schemes requires specific safeguards to prevent the energetic instabilities resulting from the non-vanishing work of the contact forces over a contact cycle (closure/opening), see [5, Fig. 6]. Examples of such remedies are the energy-preserving and energy-decaying procedures given by Armero and Petőcz [4] and Armero and Romero [32], respectively. These precautions are however unnecessary with the solution we propose. From the very nature of event-driven integration, the non-vanishing contact work over contact cycles is,

indeed, bounded by the accuracy with which root-solving is performed, so that the energy artificially introduced in the system can be controlled by a proper choice of the root-solving tolerances.

The paper is organized as follows. Section 2 introduces the model problem governing linear structural dynamics under unilateral elastic constraints. Section 3 elaborates on the concept of event-driven integration when coupled to one-step schemes for integrating equations of the model problem type. The core strategy for event detection and event localization is presented in detail. Two unconditionally stable one-step integration schemes for structural dynamics are briefly reviewed: the second-order generalized-α method [30] and the third-/fourth-order 2-level BoTr scheme that can be related to the time discontinuous Galerkin method [33]. Their coupling to the proposed root-solving strategy is studied in Section 4 by application to three examples, two of them having an analytical solution: (*i*) the elastic bouncing bar proposed by Doyen et al. [5], (*ii*) a simplified representation of Newton's cradle [34, Chap. 5] on the basis of the one-dimensional wave propagation equation and (*iii*) the pounding of two substructures of a building subject to an earthquake. The paper then concludes with a summary of the main results and an Appendix section providing details about a MATLAB [35] implementation of the algorithm, available online.

2. Linear structural dynamics and the unilateral elastic contact problem

The typical problem of (unconstrained) linear structural dynamics is generally described by the second-order vector equation of motion

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} + \mathbf{K}\mathbf{u} = \mathbf{f}, \qquad \mathbf{v} = \dot{\mathbf{u}}, \tag{1}$$

where $\mathbf{M}, \mathbf{C}, \mathbf{K} \in \mathbb{R}^{m\times m}$ denote the constant mass, damping and stiffness matrices, $\mathbf{u}, \mathbf{v} \in \mathbb{R}^m$ the displacement and velocity fields, and $\mathbf{f} \in \mathbb{R}^m$ the vector of external forces. All field variables are function of time $t \in \mathbb{R}^+$, the differentiation with respect to it being denoted by an overhead dot, $\dot{x} := \mathrm{d}x/\mathrm{d}t$. Positive integer $m := d \cdot n_{\mathrm{dof}}$ represents the number of degrees of freedom of the model n_{dof} times its dimensionality $d \in \{1, 2, 3\}$. Such models can typically be obtained from a spatial semi-discretization of solid or structural mechanics problems.

The model problem addressed in this paper combines equation (1) with unilateral elastic constraints, *i.e.*, unilateral springs. These define nodal contact forces that are proportional to the interpenetration experienced by two paired contact nodes. For a single contact interface, the magnitude of the contact force is given by

$$f^c := -\kappa\,[g]_-\,, \tag{2}$$

where the Macaulay brackets are defined as $[x]_- := \min([0\,;\,x])$ and g denotes the gap function. Figure 1 illustrates the definition of the gap function used to measure the interpenetration; it is related to the distance between the contact nodes. Starting from an initial configuration defined by position vectors $\vec{x}_1, \vec{x}_2$ that define the initial gap $\vec{g}_0 := \vec{x}_2 - \vec{x}_1$, the system evolves to the deformed configuration characterized by nodal displacements $\vec{u}_1, \vec{u}_2$ under external loading. Restricting our consideration to the linear geometrical setting (that is already hidden in the definition of the linear structural dynamics problem), the reference configuration is taken to be the initial one. The magnitude of the normal gap is then obtained by projection onto the unit vector defining the direction of the initial gap $\vec{e}_0 := \vec{g}_0 / \|\vec{g}_0\|$

$$g := \vec{e}_0 \cdot (\vec{g}_0 + \vec{u}_2 - \vec{u}_1)\,. \tag{3}$$

This approximation, that holds to first order, is illustrated in Figure 1. Expanding the dot product, the gap function can be reformulated as an affine transformation of the displacement field, so that, for the considered contact interface,

$$g = g_0 + \mathbf{w}^T\mathbf{u}, \tag{4}$$

with $g_0 := \|\vec{g}_0\|$. The vector $\mathbf{w} \in \mathbb{R}^m$ acts as a signed localization vector of the degrees of freedom active at the contact interface, with its entries weighted by the projection of $\vec{e}_0$ into the reference axis system. The distribution of the contact force on the nodal variables follows from the definition of the localization vector

$$\mathbf{f}^c = -\mathbf{w}\kappa\,[g]_-\,. \tag{5}$$

The extension to the case of q unilateral elastic constraints is straightforward by use of vector notation

$$\mathbf{g} = \mathbf{g}_0 + \mathbf{W}^T\mathbf{u}, \tag{6}$$

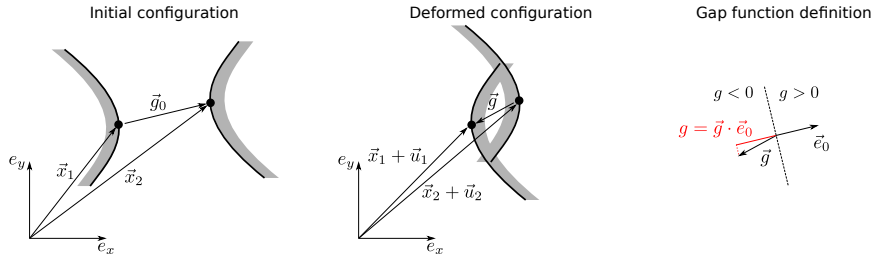

Figure 1: Definition of the gap function for two paired contact nodes by projection of the current gap on the initial gap direction $\vec{e}_0 := \vec{g}_0 / \|\vec{g}_0\|$.

where $\mathbf{g}, \mathbf{g}_0 \in \mathbb{R}^q$ are the vectors of normal gap functions and initial gaps, and $\mathbf{W} \in \mathbb{R}^{m \times q} : W_{ij} = \partial g_j / \partial u_i$ is the signed, weighted, localization matrix for the active contacts. The nodal contact forces then read

$$\mathbf{f}^c = -\mathbf{W}\boldsymbol{\kappa}\left[\mathbf{g}\right]_- , \tag{7}$$

with the Macaulay brackets applied componentwise to the entries of the normal gap vector; matrix $\boldsymbol{\kappa} \in \mathbb{R}^{q \times q}$ is diagonal with entries corresponding to the stiffness of the unilateral elastic constraints. The governing equations of the constrained problem thus read

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} + \mathbf{K}\mathbf{u} = \mathbf{f} - \mathbf{W}\boldsymbol{\kappa}\left[\mathbf{g}_0 + \mathbf{W}^T\mathbf{u}\right]_- , \qquad \mathbf{v} = \dot{\mathbf{u}}. \tag{8}$$

Given the elastic nature of the contact constraints, the nonlinearity introduced by the Macaulay brackets automatically vanishes by use of an active set strategy. Let $\mathcal{C} = \{1, 2, \ldots, q\}$ be the index set of all constraints; every entry of $\mathcal{C}$ thus identifies a contact interface. As negative gaps correspond to the interpenetration of the contacting surfaces, the subset $\mathcal{A}(t)$ of active constraints collects the constraint IDs corresponding to negative gap functions

$$\mathcal{A}(t) := \{i \in \mathcal{C} : g_i(t) \leq 0\}. \tag{9}$$

Following the definition of the Macaulay brackets, the contact forces associated with the active constraints read

$$\mathbf{f}_{\mathcal{A}} := -\sum_{i \in \mathcal{A}} \kappa_{ii}\mathbf{W}_i \left(g_{0,i} + \mathbf{W}_i^T\mathbf{u}\right); \tag{10}$$

inactive constraints do not contribute to the contact force, since the contact interface is open. The governing equation (8) becomes the combination of equation (9) and

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} + \left(\mathbf{K} + \mathbf{K}_{\mathcal{A}}\right)\mathbf{u} = \mathbf{f} + \mathbf{f}^0_{\mathcal{A}}, \qquad \dot{\mathbf{u}} = \mathbf{v}, \tag{11}$$

where

$$\mathbf{K}_{\mathcal{A}} := \sum_{i \in \mathcal{A}} \kappa_{ii}\mathbf{W}_i\mathbf{W}_i^T, \qquad \mathbf{f}^0_{\mathcal{A}} := -\sum_{i \in \mathcal{A}} \kappa_{ii}\mathbf{W}_i g_{0,i}. \tag{12}$$

The integration of the constrained equations of motion (8) is thus equivalent to marching equations (11) in time while, at the same time, tracking the evolution of the set of active constraints $\mathcal{A}(t)$ defined by equation (9).

3. Event-driven scheme

The essence of event-driven integration is simple: (*i*) integrate the current set of governing equations from given initial conditions until an event occurs or the final integration time has been reached, (*ii*) if integration stopped due to the reaching of the final simulation time, exit the integration procedure, otherwise, update the set of active constraints and post-event state as required, and return to (*i*). Alterations to the set of active constraints at events typically lead to piecewise trajectories. Figure 2 illustrates such a trajectory as it would be computed from an event-driven integration procedure. Every transverse crossing of the system trajectory with the hypersurface defining an event gives rise to a state transition: the set of active constraints is updated and a new arc of smooth trajectory is initiated.

Although simple in appearance, the implementation of a robust event-driven integration scheme requires overcoming several challenges inherent to floating point arithmetic and the discrete-time representation of a (piecewise) continuous-time problem. For instance, the handling of grazing or that of

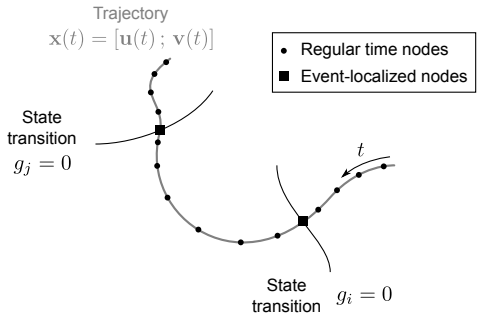

Figure 2: Piecewise smooth trajectory in state space.

Algorithm 1 *timeIntegration*

Inputs: initial conditions, model data, integration parameters, root-solving parameters
Outputs: state and constraint time histories

1 Initialize procedure (storage, active constraints at initial time, etc.)
2 Loop over time increments until final simulation time is reached
3 Compute provisional state $\mathbf{x}_{n+1}$ from $\mathbf{x}_n$ using $\mathcal{A}_n$
4 Detect event(s) that possibly occurred over $[t_n, t_{n+1}]$
5 if *events have been detected*
6 Localize earliest occurrence of event(s) from the right and its(their) ID by timestep reduction
7 Update constraint statuses
8 end if
9 Increment counter $n \leftarrow n + 1$
10 end Loop
11 return

simultaneous events in finite precision are non trivial tasks. A proper treatment of these issues and the likes requires the *ad hoc* combination of an integration procedure, a root-solving strategy and a decision module for the update of active constraints.

Algorithm 1 presents the main steps of the procedure we advocate. After the initialization of the required variables, the procedure enters the event-driven integration loop that is conducted until the final simulation time is reached. The computation of each time increment is the result of several key intermediate steps that are specifically addressed and detailed in the sequel of the paper, for the specifics of the model problem introduced in Section 2. First, the provisional state $\mathbf{x}_{n+1} := \mathbf{x}(t_{n+1})$ at t_{n+1} is computed by use of a one-step integration scheme, assuming that contact constraint statuses remain the same over the timestep, *i.e.*, $\mathcal{A}(t) = \mathcal{A}_n, t \in [t_n, t_{n+1}]$, with $\mathcal{A}_n := \mathcal{A}(t_n)$. Second, event detection is conducted to verify that no constraint status changes over the timestep. If any do(es), the event localization module is called, else it is bypassed. This module computes the system state at intermediate times $t_k \in [t_n, t_{n+1}]$ approximating the earliest occurrence of an event over the timestep. It returns a time t_k and a corresponding state $\mathbf{x}_k := \mathbf{x}(t_k)$ at which events may or may not take place. The effective timestep thus covers time interval $[t_n, t_k]$. If an event has occurred, by design of the procedure, it has in the nearby prior vicinity of t_k (implicit verification of the transversality condition); the update of constraint statuses is then performed, merely following from the event occurrence. The next increment is taken from t_k or t_{n+1} following the detection or not of an event.

A MATLAB implementation of the procedures presented in the paper is available online. See the Appendix for access details and file descriptions.

3.1. One-step time integration schemes

Any increment of the integration procedure starts with the computation of the provisional state $\mathbf{x}_{n+1}$ (Algorithm 1, line 3). Multistep schemes require several past data points to march the governing equations by one step in time. Were chattering to occur, this could prove cumbersome with successively computed data points possibly corresponding to different dynamics (constraint statuses). We therefore restrict our discussion to one-step schemes that only require data from a single previous time instant to step the solution in time.

When applied to a system of linear ODEs, these schemes give rise to the following update equation for the unknown state vector $\mathbf{x}_{n+1}$

$$\mathbf{H}_0\mathbf{x}_{n+1} = \mathbf{H}_1\mathbf{x}_n + \boldsymbol{\ell}_n^{n+1}. \tag{13}$$

The iteration matrices and the load vector are denoted by $\mathbf{H}_0, \mathbf{H}_1$ and $\boldsymbol{\ell}_n^{n+1}$ respectively; they depend on the model governing matrices $\mathbf{M}, \mathbf{C}, \mathbf{K}$, the external load vector $\mathbf{f}$, the integration timestep h and, possibly, algorithmic parameters related to the integration scheme. The state vector $\mathbf{x}$ contains the displacement and velocity variables plus any additional variables required by the scheme definition, *e.g.*, the acceleration for α-like schemes [29, 31].

As examples, we specialize equation (13) to the generalized-α [30] and the 2-level BoTr [33] schemes. The following update equations are respectively obtained where, besides the time indexation subscript n, all other indices pertain to the definition of the variable names

$$\begin{pmatrix} (1-\alpha_f)\mathbf{K} & (1-\alpha_f)\mathbf{C} & (1-\alpha_m)\mathbf{M} \\ \mathbf{I} & \mathbf{0} & -\beta h^2\mathbf{I} \\ \mathbf{0} & \mathbf{I} & -\gamma h\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{n+1} \\ \mathbf{v}_{n+1} \\ \dot{\mathbf{v}}_{n+1} \end{pmatrix} = \begin{pmatrix} -\alpha_f\mathbf{K} & -\alpha_f\mathbf{C} & -\alpha_m\mathbf{M} \\ \mathbf{I} & h\mathbf{I} & h^2(1/2-\beta)\mathbf{I} \\ \mathbf{0} & \mathbf{I} & h(1-\gamma)\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{u}_n \\ \mathbf{v}_n \\ \dot{\mathbf{v}}_n \end{pmatrix} + \begin{pmatrix} \mathbf{f}(t_{n+1-\alpha_f}) \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \tag{14}$$

$$\begin{pmatrix} \mathbf{C} + \frac{\eta+3}{6}h\mathbf{K} & \mathbf{M} - \frac{1+\eta}{12}h^2\mathbf{K} \\ \mathbf{M} - \frac{1+\eta}{12}h^2\mathbf{K} & -\frac{\eta+3}{6}h\mathbf{M} - \frac{1+\eta}{12}h^2\mathbf{C} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{n+1} \\ \mathbf{v}_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{C} + \frac{\eta-3}{6}h\mathbf{K} & \mathbf{M} - \frac{1-\eta}{12}h^2\mathbf{K} \\ \mathbf{M} - \frac{1-\eta}{12}h^2\mathbf{K} & -\frac{\eta-3}{6}h\mathbf{M} - \frac{1-\eta}{12}h^2\mathbf{C} \end{pmatrix} \begin{pmatrix} \mathbf{u}_n \\ \mathbf{v}_n \end{pmatrix} + \begin{pmatrix} \mathbf{F}_1 \\ \left(t_{n+1/2} - \frac{\eta}{6}h\right)\mathbf{F}_1 - \mathbf{F}_2 \end{pmatrix}. \tag{15}$$

For both schemes, the algorithmic parameters can be related to the spectral radius of the amplification matrix at infinite frequency ρ_∞. We respectively have

$$\alpha_m := \frac{2\rho_\infty - 1}{\rho_\infty + 1}, \quad \alpha_f := \frac{\rho_\infty}{\rho_\infty + 1}, \quad \beta := \frac{1}{4}\left(1 - \alpha_m + \alpha_f\right)^2, \quad \gamma := \frac{1}{2} - \alpha_m + \alpha_f, \tag{16}$$

and

$$\eta := \frac{1-\rho_\infty}{1+\rho_\infty}. \tag{17}$$

They are unconditionally stable, *i.e.,* the amplification matrix has spectral radius below one $\rho \leq 1$, conservative for $\rho_\infty = 1$ and dissipative when $\rho_\infty = [0, 1)$. Scheme (14) is second-order accurate for all algorithmic configurations whereas scheme (15) is third-order accurate in the dissipative setting and fourth order in the numerically conservative case. For completeness, additional definitions read

$$t_{n+1-\alpha_f} := (1-\alpha_f)t_{n+1} + \alpha_f t_n, \quad t_{n+1/2} := \frac{1}{2}\left(t_n + t_{n+1}\right), \tag{18}$$

$\mathbf{I} \in \mathbb{R}^{m\times m}$ and $\mathbf{0} \in \mathbb{R}^m, \mathbb{R}^{m\times m}$ refer to the identity matrix and the zero vector or matrix depending on the context. Integral actions $\mathbf{F}_1, \mathbf{F}_2 \in \mathbb{R}^m$ are fourth-order accurate approximations of the external force mean and first time-moment values by the Simpson-Cavalieri quadrature formula

$$\mathbf{F}_1 := \frac{h}{6}\left(\mathbf{f}(t_n) + 4\mathbf{f}(t_{n+1/2}) + \mathbf{f}(t_{n+1})\right), \quad \mathbf{F}_2 := \frac{h}{6}\left(t_n\mathbf{f}(t_n) + 4t_{n+1/2}\mathbf{f}(t_{n+1/2}) + t_{n+1}\mathbf{f}(t_{n+1})\right). \tag{19}$$

Figure 3 reproduces the spectral radius of the amplification matrix (left plot) and the relative period error (right plot) for both schemes as the reduced frequency $\Omega := \omega h$ is varied. The relative period error is fourth order and second order for the 2-level BoTr and generalized-α schemes, respectively.

Although the two presented examples are implicit integration schemes, nothing prevents the procedure to be coupled to explicit one-step schemes. Explicit schemes of the Runge-Kutta family fit into the format of equation (13) with $\mathbf{H}_0$ equal to the identify matrix [36]. These are, however, usually expensive to use with equations of motion, and schemes dedicated to structural dynamics will typically be preferred, in a multi-stage implementation exploiting a diagonal mass (lumped) matrix rather than the format of equation (13). An implementation of the Noh-Bathe procedure [37] is provided online, for the reader to explore the application examples; see details in the Appendix. Attention to the selection of the timestep must, nonetheless, be paid, as explicit schemes are typically only conditionally stable (whereas common implicit schemes are unconditionally stable). In particular, closed contact configurations associated with a high contact stiffness are likely to drive the critical timestep of the integration scheme. Further comparison of explicit schemes, as applied to impact dynamics, is given by Nsiampa et al. [38].

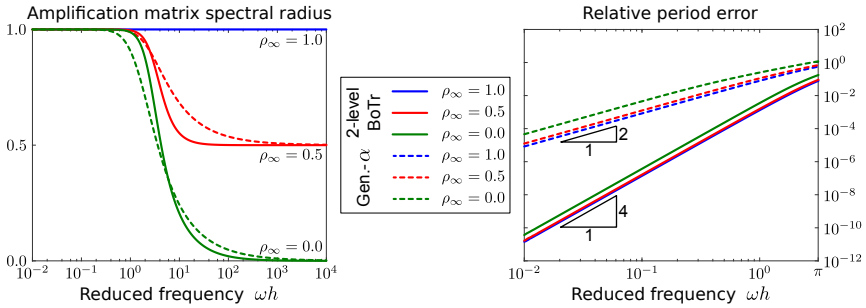

Figure 3: Spectral radius of the amplification matrix (left) and relative period error (right) for the generalized-α and 2-level BoTr integration schemes.

3.2. Root-solving procedure

Restricting our consideration to linear structural dynamics under unilateral elastic constraints, event detection and localization is equivalent to the detection and identification (Algorithm 1, lines 4 and 6) by the root-solving procedure of the zeros of the normal gap functions $\mathbf{g}(\mathbf{u}(t)) \in \mathbb{R}^q$ occurring over the current timestep, *i.e.*, $t \in [t_n, t_{n+1}]$. Further assuming that the only information available are the state vectors at the beginning and at the end of the timestep $\mathbf{x}_n, \mathbf{x}_{n+1}$, we resort to cubic Hermite interpolation to provide a continuous representation of the normal gap functions on the basis of their values and that of their derivatives at time instants t_n, t_{n+1}, $\mathbf{g}_n := \mathbf{g}(t_n), \mathbf{g}_{n+1} := \mathbf{g}(t_{n+1})$ and $\dot{\mathbf{g}}_n := \dot{\mathbf{g}}(t_n), \dot{\mathbf{g}}_{n+1} := \dot{\mathbf{g}}(t_{n+1})$. As normal gap functions depend on the displacement field only, their time derivative are functions of the velocity field and can be computed from the state vector provided by the integration procedure, without additional computations.

Introducing the dimensionless time $\tau(t) = (t - t_n)/h$ that has unit parent domain over the timestep, $\tau(t) : [t_n, t_{n+1}] \to [0, 1]$, the interpolated normal gap functions read

$$\tilde{\mathbf{g}}(\tau) := \mathbf{g}_n + \tau h \dot{\mathbf{g}}_n + \tau^2 \left(-3\mathbf{g}_n - 2h\dot{\mathbf{g}}_n + 3\mathbf{g}_{n+1} - h\dot{\mathbf{g}}_{n+1}\right) + \tau^3 \left(2\mathbf{g}_n + h\dot{\mathbf{g}}_n - 2\mathbf{g}_{n+1} + h\dot{\mathbf{g}}_{n+1}\right). \quad (20)$$

This continuous extension of the normal gap functions is the backbone of the event detection and event localization as both procedures are based on the computation of its zeros. It is also the source of possible numerical complications such as numerical grazing. The tilde notation has been introduced to underscore the fact that $\tilde{\mathbf{g}}(\tau)$ is an approximation to the time-continuous normal gap functions $\mathbf{g}(\tau)$.

Sturm sequences [39] and the likes based on Vincent's theorem [40], which associate the presence of polynomial roots in arbitrary intervals to sign changes in specific sequences, are well known means for the detection of roots of univariate polynomials in selected intervals. They are commonly implemented in Computer Algebra Systems (CAS) in combination with exact arithmetic. However, when combined to floating-point arithmetic, these become more difficult to use not to say unreliable when badly scaled polynomials are to be assessed; see for instance the specific iterative treatment proposed by Suzuki and Sasaki [41] to regularize Sturm sequences in such cases. By design of the proposed root-solving algorithm, there will be numerous situations where badly scaled and degree degenerate polynomials will be expected. A simple example is when the timestep h becomes very small as compared to the characteristic timescale of the oscillations (dictated by the external loading and the system maximum eigenfrequency), for the system trajectory is almost linear in that setting. Alternative techniques must thus be exploited.

To avoid any robustness issue, we opt for the computation of the roots of each normal gap function Hermite interpolant, although it has a significant cost. We base our approach on two eigenvalue companion problems: one that requires the scaling of the polynomial by the leading-order coefficient and that is thus sensitive to degree degeneracy, and one that is based on the barycentric representation proposed by Corless et al. [42], less sensitive to degree degeneracy but more expensive. Cardano's analytical formulas are avoided due to their sensitivity to degree degeneracy and, in floating-point arithmetic, to error propagation in the computation of the intermediate coefficients defining the polynomial roots [43].

Considering a sequential approach for the handling of events (loop over all event functions, $\forall i \in \mathcal{C}$), the coefficients of the polynomial interpolant approximating the i^{th} gap function are easily obtained from equation (20) by the replacement $\mathbf{g} \leftarrow g_i$. The cubic approximant $P(\tau)$ and its normalized version $P_0(\tau)$ read

$$P(\tau) := \tilde{g}_i(\tau) = a_3\tau^3 + a_2\tau^2 + a_1\tau + a_0, \qquad P_0(\tau) := \frac{1}{a_3}P(\tau) = \tau^3 + c_2\tau^2 + c_1\tau + c_0. \quad (21)$$

Algorithm 2 *eventDetection* - Exploded view of Algorithm 1, line 4.

Inputs: $h, \mathbf{g}_n, \dot{\mathbf{g}}_n, \mathbf{g}_{n+1}, \dot{\mathbf{g}}_{n+1}$, `degTol`
Outputs: $\mathcal{I}_{\text{evt}}, \mathbf{h}_{\text{evt}}, \mathrm{d}\tilde{g}_{\mathcal{I}_{\text{evt}}}/\mathrm{d}t$

$\mathcal{I}_{\text{evt}} = [\ ]$
$\mathbf{h}_{\text{evt}} = [\ ]$
for $i \in \mathcal{C}$
  Form polynomial coefficients $a_j, j = 0, 1, 2, 3$ by setting $\mathbf{g} \leftarrow g_i$ in Eq. (20)
  if $a_3 \geq$ `degTol` % No degree degeneracy
    Normalize polynomial coefficients by a_3
    $\mathbf{s} = \lambda(\boldsymbol{\Upsilon})$ % See Eq. (22)
  else % Assume degree degeneracy
    $\mathbf{s} = \lambda(\boldsymbol{\Upsilon}_0, \boldsymbol{\Upsilon}_1)$ % See Eq. (23)
  end if
  if $\exists j : s_j \in [0, 1]$ % Store roots that are within [0,1]
    $\mathcal{I}_{\text{evt}} \leftarrow [\mathcal{I}_{\text{evt}}\ ;\ j]$, $\mathbf{h}_{\text{evt}} \leftarrow [\mathbf{h}_{\text{evt}}\ ;\ hs_j]$
  end if
end for
Sort output according to occurrence time of event(s)
Compute time derivative of $\tilde{g}_{\mathcal{I}_{\text{evt}}}$ at crossing points
return

On the one hand, if we define the companion matrix

$$\boldsymbol{\Upsilon} := \begin{pmatrix} -c_2 & -c_1 & -c_0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \tag{22}$$

it readily appears that the characteristic polynomial of $\boldsymbol{\Upsilon}$ is equal to the normalized approximant $P_0(\tau)$, *i.e.*, $P_0(\tau) \equiv |\tau\mathbf{I} - \boldsymbol{\Upsilon}|$. Thus, event times are approximated by the eigenvalues of the companion matrix $\mathbf{s} = \lambda(\boldsymbol{\Upsilon})$, which can be numerically computed using any appropriate library. On the other hand, it can be shown [42] that, upon definition of matrices

$$\boldsymbol{\Upsilon}_0 := \begin{pmatrix} 0 & & & & P(0) \\ 1 & 0 & & & \dot{P}(0) \\ & & 1 & & P(1) \\ & & 1 & 1 & \dot{P}(1) \\ -2 & -1 & 2 & -1 & 0 \end{pmatrix}, \quad \boldsymbol{\Upsilon}_1 := \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 0 \end{pmatrix}, \quad \dot{P}(\tau) = \frac{\mathrm{d}P}{\mathrm{d}\tau}, \tag{23}$$

the roots of the cubic approximant are given by the generalized eigenvalues $\mathbf{s} = \lambda(\boldsymbol{\Upsilon}_0, \boldsymbol{\Upsilon}_1) = \{\tau \in \mathbb{C} : |\tau\boldsymbol{\Upsilon}_1 - \boldsymbol{\Upsilon}_0| = 0\}$; less the two spurious eigenvalues at infinity that result from the companion problem formulation that are to be discarded. This alternative formulation enables the handling of degenerate polynomials, for which $a_3 \ll 1$. Similarly, these eigenvalues can be obtained using any appropriate library.

Algorithm 2 summarizes the event-detection procedure; it is written in pseudo-code inspired from the MATLAB syntax [35]. For each gap function ($\forall i \in \mathcal{C}$), the coefficients of the cubic approximant are formed and its roots are computed using the former companion problem if degree degeneracy is not expected (`degTol` is an input parameter used to assess degree degeneracy) and the latter one in the opposite situation. Should any root be located in the parent domain, the event indices as well as the root magnitudes (provisional event times) are stored concurrently to the already computed results. As a last step, the outputs are sorted according to the forecast event times and the time derivatives of the event functions are computed at the crossing points, for use in the event localization procedure. In case no event is detected, the output variables return empty arrays, as denoted by [ ].

As $\tilde{\mathbf{g}}$ is an approximation to $\mathbf{g}$, their zeros do not necessarily coincide. In case an event is detected, it must be accurately localized, in accordance with the problem original dynamics, through an iterative procedure that must prove robust to numerical accumulation and numerical grazing; these are two artifacts that result from event localization up to a given tolerance instead of an exact localization and from the use of an approximate representation of the normal gap function. They can, nevertheless, be prevented by two algorithmic ingredients: (i) the driving of the iterative process beyond the event

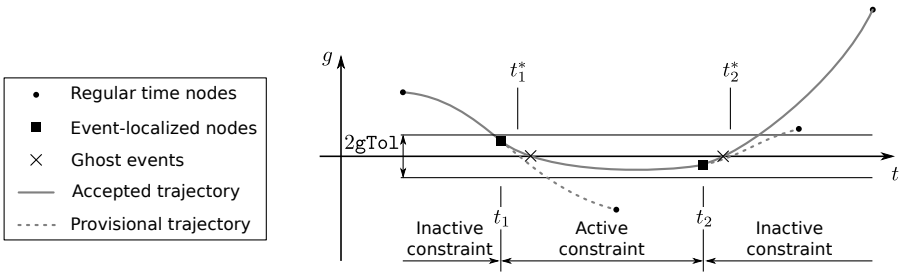

Figure 4: Illustration of numerical accumulation.

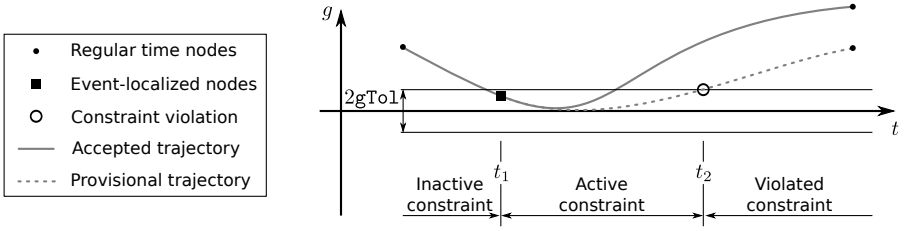

Figure 5: Illustration of numerical grazing and constraint violation.

occurrence, as suggested by Birta et al. [44], and (*ii*) the acceptance of all computed points prior to the event occurrence. We illustrate the two situations, then propose a strategy that avoids them.

Let us first illustrate the issue of numerical accumulation, also known as discontinuity sticking in the literature [26, 45]. For this, we consider the single gap function and the associated constraint status evolution displayed in Figure 4. A first zero takes place at time t_1^* where the provisional trajectory crosses the $g = 0$ boundary/axis. Assuming that iterative localization is performed until convergence, deemed by criterion $|g(t)| < \texttt{gTol}$, it may well be that the root verifies $t_1 < t_1^*$, *i.e.*, event occurrence is localized before the actual zero of the normal gap function. Given the negative gap velocity ($\dot{g}(t_1) < 0$) and the positive residual after convergence ($0 < g(t_1) < \texttt{gTol}$), the trajectory is expected to effectively cross the boundary $g = 0$ shortly after t_1. A similar setting is observed at $t_2 \neq t_2^*$, for which the converged configuration is $\dot{g}(t_2) > 0, -\texttt{gTol} < g(t_2) < 0$, yielding a possible crossing shortly after t_2. These events (depicted by crosses in Figure 4) are, however, to be discarded for they are ghost representations of the events localized at t_1 and t_2. Not rejecting them could well lead to a failure of the integration procedure due to an infinite (numerical) accumulation of events should the event-localization procedure always converge to gap residuals of the same sign (positive at t_1, negative at t_2).

The situation of numerical grazing is illustrated in Figure 5. At time t_1, an event is localized with positive gap residual. The negative crossing velocity suggests that the constraint status should be switched from inactive (open contact) to active (closed contact). However, due to the loading circumstances and the problem dynamics, the trajectory actually never crosses the constraint hyperplane $g = 0$, which results in the constraint violation at time t_2 and erroneous computations.

To circumvent the issues of numerical accumulation and grazing, the event localization procedure detailed in Algorithm 3 is proposed. It is based on the prediction of the earliest occurrence of an event returned by the cubic approximant $t_n + h_{\mathcal{I}_{\mathrm{evt}_1}}$, shifted by a function of a relaxation parameter $\eta_k \in (0, 1]$ that monotonically decreases from 1 to 0 as the number of iterations tends to `maxIter` (the maximum number of allowed iterations), gap velocity at the detected event $\mathrm{d}\tilde{g}_{\mathcal{I}_{\mathrm{evt}_1}}/\mathrm{d}t$

$$t_k = t_n + h_{\mathcal{I}_{\mathrm{evt}_1}}(1 + \sigma_k), \tag{24}$$

that serves to compute the state and the normal gap functions from data at t_n. The purpose of the shift function is to ensure the crossing of the constraint hyperplane ($g = 0$). From the definition of iterative time t_k, the timestep interval is then divided in three non-overlapping subintervals,

$$[t_n, t_{n+1}] = I_1 \cup I_2 \cup I_3 := [t_n, t_k - \texttt{aTol}) \cup [t_k - \texttt{aTol}, t_k) \cup [t_k, t_{n+1}], \tag{25}$$

where parameter $\texttt{aTol} := \min([\texttt{tTol}\,;\, \texttt{gTol}/\max(|\dot{\mathbf{g}}_{\mathcal{I}_{\mathrm{evt}}}|)])$ is a tolerance introduced to increase the accuracy of event localization in near-grazing situations; $\texttt{gTol}, \texttt{tTol}$ are parameters of the root-solving procedure. Computation of the earliest event occurrence is then performed by use of Algorithm 2. If it is located in I_1, an additional iteration is requested. If it belongs to I_2 and all events have a gap

Algorithm 3 *eventLocalization* - Exploded view of Algorithm 1, line 6.

Inputs: $\mathcal{I}_{\text{evt}}, \mathbf{h}_{\text{evt}}, \mathcal{I}_{\text{evt}}^{\text{old}}$, `gTol`, `tTol` all data required to form iteration matrices
Outputs: $\mathcal{I}_{\text{evt}}, t_k, \mathbf{x}_k, h_{\text{next}}$

$k = 1$
$t_k = t_n + h_{\mathcal{I}_{\text{evt}_1}}(1 + \sigma_k)$
while `true`
  Compute state at t_k
  Compute gap functions at t_k
  Check occurrence of events on time interval $[t_n, t_k) = I_1 \cup I_2$ using *eventDetection*
  if *no event occurs in* $[t_n, t_k)$
    Check if any gap function is equal to 0
    if *there is none*
      return $\mathcal{I}_{\text{evt}} = [\ ]$, h_{next} based on occurrence on I_3, using *eventDetection*,
        or the regular timestep if no event (numerical accumulation)
    else
      return $\mathcal{I}_{\text{evt}}$ corresponding to 0 normal gap functions
    end if
  else % At least one event takes place in $[t_n, t_k)$
    if *all events are in* I_2 *and verify* $|g_k + \texttt{gTol} \cdot \text{sign}(g_n)| < \texttt{gTol}$
      return time and state at t_k, and their indices
    else % Proceed with an additional iteration
      $k = k + 1$
      $t_k = t_n + h_{\mathcal{I}_{\text{evt}_1}}(1 + \sigma_k)$
    end if
  end if
  Increment iteration counter
  if $k >$ `kMax` % Maximum number of iterations reached
    return `error`
  end if
end while

function with magnitude verifying $|g_k + \texttt{gTol} \cdot \text{sign}(g_n)| < \texttt{gTol}$, their IDs are stored in integer subset $\mathcal{I}_{\text{evt}}$; multiple events are then automatically handled. The procedure exits the iterative loop and returns $t_k, \mathbf{x}_k$ and $\mathcal{I}_{\text{evt}}$. Otherwise, an additional iteration is requested. If the earliest event is located in I_3, the current state at t_k is accepted as is, even though no events have been localized. The procedure exits the iterative loop and returns $t_k, \mathbf{x}_k$ and $\mathcal{I}_{\text{evt}} = [\ ]$. In that case, the next timestep is taken with $h = 2h_{\mathcal{I}_{\text{evt}_1}}$ to ensure that the next increment slightly overpasses the supposed instant of event occurrence if an event is detected in I_3; otherwise, the regular timestep is used (this corresponds to near-grazing). The robust handling of numerical grazing is thus automatically achieved; that is, when the cubic approximant of the gap function predicts the existence of an event when, actually, the state trajectory never crosses nor touches the event hyperplane, *i.e.*, $\exists i \in \mathcal{C}, \tau_* \in [0,1] : \tilde{g}_i(\tau_*) = 0$ but $\forall i \in \mathcal{C}, \tau_* \in [0,1] : g_i(\tau_*) \neq 0$; see Figure 5.

3.3. Decision module for constraint (de)activation

Conceptually, the change of constraint status is simple (Algorithm 1, line 8); it is driven by the transverse nature of the trajectory and constraint hyperplane intersection. A negative normal gap velocity corresponds to the closure of the contact interface and a positive to its opening. These drivers to constraint switches are, however, implicitly buried in the the event-localization module. Indeed, as localized events necessarily verify the transversality condition, constraint switches merely follow from the occurrence of an event and the dichotomous nature of the contact status. A contact interface closed prior to event occurrence will be thus be opened and vice versa.

In addition to the update of constraint statuses, this module can also handle the choice of the timestep size h. Faithful to the practice of structural dynamics simulation, we opt for two values for the timestep, a larger one to be used whenever all contact interfaces are open, and a smaller one to be used when at least one interface is closed. These can, for instance, be computed on the basis of the highest eigenfrequency of the structural dynamics model, so as to avoid aliasing.

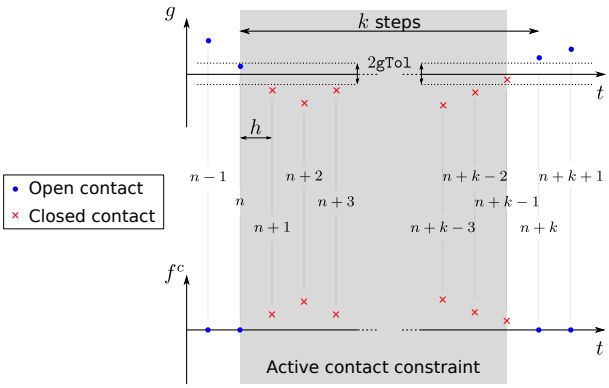

Figure 6: Schematic representation of a k-step persistent contact sequence.

3.4. Scheme stability from an energetic perspective

A common issue arising from the use of the penalty method in the definition of the contact forces is the introduction of spurious energy in the system through the non-vanishing work of the contact forces over contact closure/opening cycles. This energy can lead to the instability of the time integration procedure and the blow up of the solution [5].

Remedies have been provided, notably under the form of the energy-momentum conserving algorithm (EMCA) [4] and the energy-dissipative momentum conserving (EDMC) procedure [32]. These second-order accurate procedures, based on the midpoint integration scheme, enforce an algorithmic definition of the contact force so as to ensure a zero net work of the contact force over a closure/opening cycle. This definition guarantees energetic stability but does require solving a nonlinear problem at each increment, contrary to the linear solver required for the proposed event-driven procedure. This apparent drawback is nonetheless counterbalanced by the fact that a constant timestep can be used, a positive point in settings where constraint chatter occurs, or in the presence of numerous contact interfaces. However, they are limited to the handling of unilateral elastic constraints governed by normal gap functions and are not suitable for the end application of percussive drilling that we have in mind.

To evaluate the amount of spurious energy introduced in the system by the contact forces during the event-driven integration, let us consider a single interface and a persistent contact sequence of k steps, as depicted in Figure 6. The work increment realized by the contact force f^c over a timestep is given by

$$\Delta W^c_{n\to n+1} := \frac{f^c_n + f^c_{n+1}}{2}\left(g_{n+1} - g_n\right) = -\frac{\kappa}{2}\left([g_{n+1}]_- + [g_n]_-\right)\left(g_{n+1} - g_n\right). \tag{26}$$

Accordingly, the work variation over the k-step sequence is given by

$$\Delta W^c_{n\to n+k} = -\frac{\kappa}{2}\sum_{j=0}^{k-1}\left([g_{n+j+1}]_- + [g_{n+j}]_-\right)\left(g_{n+j+1} - g_{n+j}\right) = -\frac{\kappa}{2}\left(g_{n+k}g_{n+k-1} - g_{n+1}g_n\right). \tag{27}$$

It only vanishes under the condition that the gap function verifies $(g_{n+k}g_{n+k-1} - g_{n+1}g_n) = 0$, a condition that is, in practice, never achieved. Positive or negative variations of the system mechanical energy are thus expected for each closure/opening cycle of a contact interface. They are the reason for a possible blow up of the solution.

However, in the frame of the event-driven scheme proposed in this paper, the magnitude of the gap function at the closure and opening of the contact interface is upper bounded by the tolerance of the event-localization procedure `gTol`. The variation of the contact force work is thus bounded as follows

$$\begin{aligned}\left|\Delta W^c_{n\to n+k}\right| &\leq \frac{\kappa}{2}\left(|g_{n+k}g_{n+k-1}| + |g_{n+1}g_n|\right), \\ &\leq \texttt{gTol}\kappa\left(g_{n+k} + g_n\right).\end{aligned} \tag{28}$$

Usage of a sufficiently strict tolerance `gTol` for the event-localization procedure therefore guarantees the stability of the scheme with respect to the contact definition, provided the number of contact closure/opening cycles remains finite and conditions for the stability of the integration scheme between events are met. Further, this bound can be exploited to provide an order of magnitude for `gTol`. Given orders of magnitude for the tolerated contact work over a contact cycle (it is a measure of the tolerated

energy drift over a contact cycle and must be chosen in balance with the level of mechanical energy in the system), for the gap function around contact status change $\mathcal{O}(g)$ and the contact stiffness, we have

$$\mathcal{O}(\texttt{gTol}) = \frac{1}{\kappa}\frac{\mathcal{O}(\Delta W^c_{n\to n+k})}{\mathcal{O}(g)}. \tag{29}$$

Stringent tolerances will therefore be required whenever contacts are stiff and energy drift is to be tightly controlled, though the estimate is very conservative, as is demonstrated next.

4. Application examples

This section proposes three application examples that illustrate the accuracy, stability and robustness of the proposed integration procedure. Problem complexity is gradually increased. The first example is that of an elastic bouncing bar in an acceleration field. This 1-dimensional problem features a single unilateral elastic constraint. The second example is a variation on the problem of impacts in Newton's cradle. Though it features several constraints, the system is not expected to experience the simultaneous occurrence of events at distinct contact interfaces, through the choice of specific initial conditions and gaps. The third and last example is that of the pounding between two substructures of a building when subjected to a horizontal ground acceleration mimicking an earthquake. Several contact interfaces are defined and simultaneous contact events occur.

The presented results have been obtained with the MATLAB software [35]. Although it does not offer the performance levels of equivalent codes in compiled languages, *e.g.*, C/C++ or FORTRAN, the versatility of the environment makes it ideal to test concepts and validate algorithms. MATLAB scripts to reproduce the simulations proposed next are available on the bibliographic repository of the Université de Liège (`orbi.ulg.ac.be`). See details in the Appendix.

4.1. Bouncing bar

As a first application of the proposed algorithm, we consider the bouncing bar benchmark proposed by Doyen et al. [5]. It consists of an elastic bar of constant properties and section, subject to a constant acceleration along its axis and whose motion is constrained by a rigid wall at one of its ends. Upon proper adjustment of the problem parameters, the bar motion follows a periodic sequence of persistent contact and free flight phases. The problem definition as well as the periodic response are shown in Figure 7. The reader is referred to their paper for more details on the problem definition and solution.

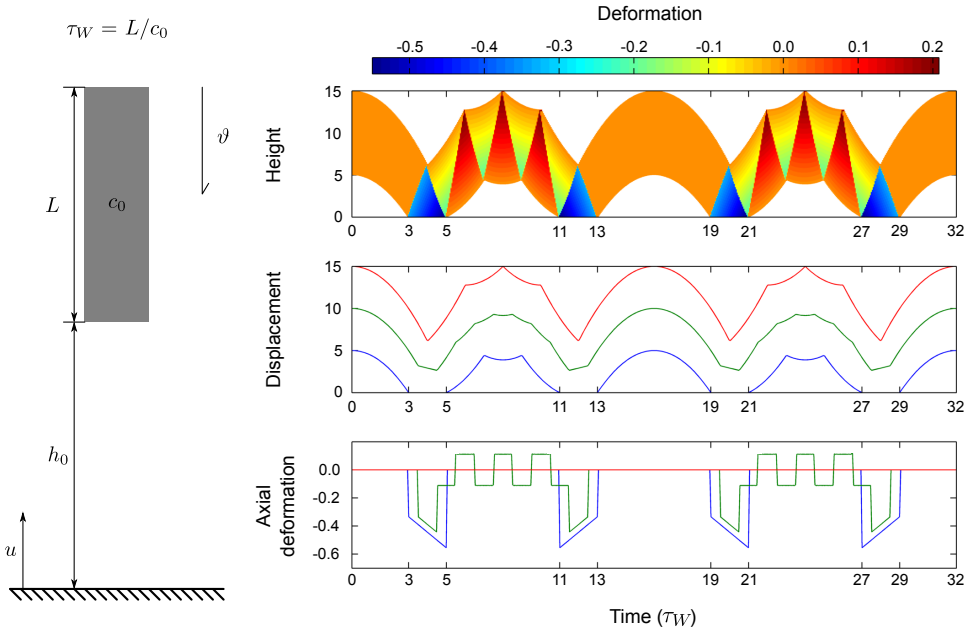

Figure 7: Analytical solution to the bouncing bar benchmark – The periodic motion comprises four phases: (i) non-oscillatory free flight phases represented by parabolas ($t/\tau_W \in [0,3] \cup [13,19] \cup [29,32]$), (ii) compressive contact phases ($t/\tau_W \in [3,5] \cup [19,21]$), (iii) oscillatory free flight phases ($t/\tau_W \in [5,11] \cup [21,27]$), and (iv) expansive contact phases ($t/\tau_W \in [11,13] \cup [27,29]$). Parameters: $(h_0, L, c_0, \vartheta) = (5, 10, 30, 10)$.

The governing equations for this example are given by

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{K}\mathbf{u} + \kappa\mathbf{w}\left[\mathbf{w}^T\mathbf{u}\right]_- = -\mathbf{M}\mathbf{1}\vartheta, \qquad \mathbf{v} = \dot{\mathbf{u}}, \tag{30}$$

with initial conditions $\mathbf{u}_0 = h_0, \mathbf{v}_0 = \mathbf{0}$. Column vector $\mathbf{1} \in \mathbb{R}^m$ consists of unit entries only. The global mass and stiffness matrices are obtained by assemblage of the elementary matrices relative to linear rod elements

$$\mathbf{M}_e = \frac{L_e}{12}\begin{pmatrix} 5 & 1 \\ 1 & 5 \end{pmatrix}, \qquad \mathbf{K}_e = \frac{c_0^2}{L_e}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}, \tag{31}$$

with element length L_e. The average of the lumped and consistent mass matrices is used, as it is known to reduce the error associated with the spatial discretization [29, p. 446]. This example problem having a single contact interface ($n = 1$), the vector of event functions degenerates into a scalar normal gap function

$$g = \mathbf{w}^T\mathbf{u}, \tag{32}$$

with $\mathbf{w} = [1\,;\,0\,;\,\cdots\,;\,0\,;\,0]$, following the numbering convention that bottom and top nodes have number 1 and m respectively.

Numerical integration of the governing equations using the proposed algorithm and the 2-level BoTr scheme with $\rho_\infty = \{1.0, 0.5, 0.0\}$ yields the results displayed in Figure 8. Two integration timesteps are used, one for free flight phases and one for persistent contact ones. They are computed so that there are at least 2.5 computed points per lowest eigenperiod of the system. Other simulation parameters are given in the figure caption. The analytical solution is represented as well, for comparison. Plot (a) displays the time evolution of the displacement field at the bar extremities and at its center of gravity. After two periods of motion, the effects of numerical dispersion become important, with significant alterations to the free flight phase ($t/\tau_W \in [29, 32]$). This limited fidelity translates the difficulties of the continuous Galerkin method (without resorting to more advanced formulations or adaptive meshing) to accurately capture the response of non-smooth problems with strong strain or velocity discontinuities, as is the case in the present example (discontinuity of the velocity and stress fields at the wave front). Plot (b) shows the evolution of the computed contact force at the bar/wall interface. The system experiences important chattering when conservative simulation is used, leading to an inaccurate representation of the contact force as it is computed at the end of the timestep when the gap magnitude is close to 0^+. Dissipative configurations do perform better. The initial peak force at contact closure is rapidly damped; the force then converges toward the analytical solution. Plots (c)-(d) pertain to system energy quantities. Plot (c) represents the normalized balance of the system total energy (kinetic+strain+gravity), constant for the continuous problem. Plot (d) shows the error on the system mechanical energy (kinetic+strain) with respect to that of the continuous problem. The energetic stability of the event-driven integration scheme is confirmed. In particular, for the conservative setting of time integration ($\rho_\infty = 1.0$), the total energy remains constant up to a negligible drift related to `gTol`, see Section 3.4. The dissipative character of the integration procedure for $\rho_\infty \in \{0.5, 0.0\}$ also clearly appears once the bar starts to vibrate. Even though initial levels of mechanical energy error are higher for dissipative configurations than for the conservative one, all configurations tend to significant peak error levels after two periods. The influence of numerical dispersion (that varies with ρ_∞) is visible in the error drops related to the periodic nature of motion, as they take place at distinct instants for the three simulations. Plots (e)-(f) illustrate the evolution of the bar total linear momentum $p_n := \mathbf{1}^T\mathbf{M}\mathbf{v}_n$ and that of its error. Trends similar to these observed with respect to the system energy are observed. When $\rho_\infty = 1.0$, chattering is responsible for the increased error levels during persistent contact. Nevertheless, during free flight phases, lower error levels are achieved than with dissipative configurations of the integration scheme. Thus, even though chattering prevents an accurate representation of the contact force, it does not significantly degrade the solution quality in regards to the accuracy of the displacement and velocity fields. Table 1 reports the simulation performances for the three numerical settings of Figure 8. It is seen that the conservative simulation requires much more computational effort than the dissipative ones. This is mainly caused by the larger number of increments engendered by the constraint chatter. Both performance and accuracy require the introduction of some numerical damping.

In order to assess the influence of the tolerances of the event-localization procedure on the stability of the integration procedure and the accuracy of the solution, we have computed the relative energy drift of the system for several values of (`gTol`, `tTol`). Obviously, conservative integration was used, so that the drift can be assessed from the variation of the system total energy between times 0 and $2T = 32\tau_W$. The results are depicted in Figure 9, both in terms of magnitude and sign of the drift. Simulation parameters of Figure 8 have been used. As expected, the overall drift can be positive or negative as the

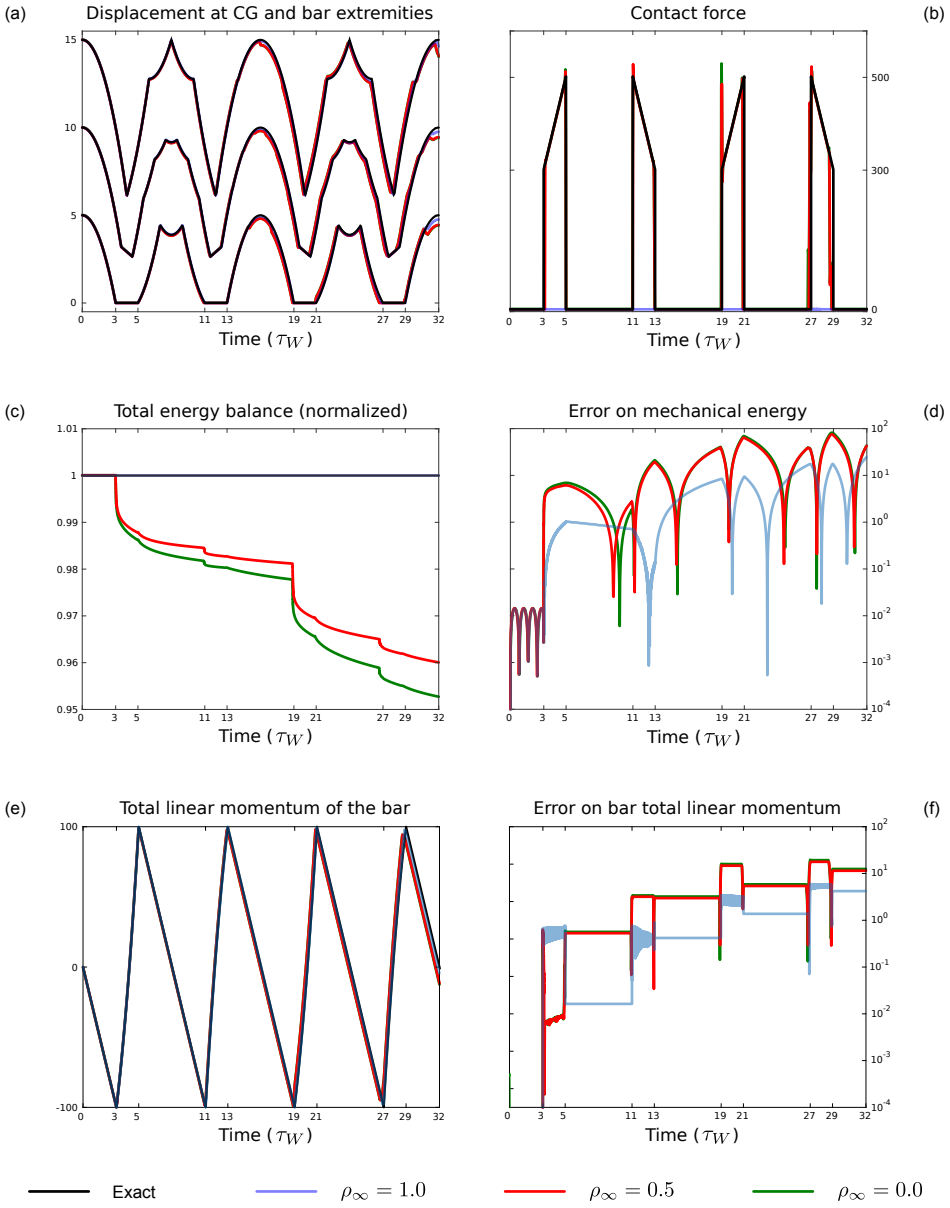

Figure 8: Numerical response of the elastic bouncing bar. Simulation parameters: 2-level BoTr scheme, $m = 101$ (100 elements), timestep h is computed so that there are 2.5 points per lowest eigenperiod of the model $(h, h_{\text{red}}) = (3.42 \cdot 10^{-3}, 3.77 \cdot 10^{-4})$, root-solving parameters $(\texttt{gTol}, \texttt{tTol}) = (10^{-8}, 10^{-8})$, contact stiffness $\kappa = 100 \cdot \max(\mathbf{K})$. Plots: (a) numerical dispersion of the traveling waves starts to significantly alter the computed displacement after two periods; (b) the higher the numerical dissipation, the faster chattering at the contact interface is damped and the numerical contact force converges toward the exact value, the contact force is improperly represented in the undamped case due to chattering; (c) the stability of the event-driven integration procedure is well confirmed by the boundedness of the system total energy, when conservative time integration is used, the drift is barely observable; (d) conservative integration performs better than dissipative integration with respect to the error on the system mechanical energy over the first period of motion, then error levels tend to become equivalent due notably to dispersion; (e) the bar total momentum is not significantly affected by the incorrect representation of the contact force in the numerically conservative setting; and (f) conservative integration performs slightly better than dissipative integration in the long term. The line colors refer to the amount of numerical dissipation generated by the integration scheme, as mentioned in the legend.

ρ_∞	1.0	0.5	0.0
# increments	11397	3633	3372
CPU time (normalized)	6.76	1.17	1.00

Table 1: Influence of the integration scheme numerical damping on the procedure performance.

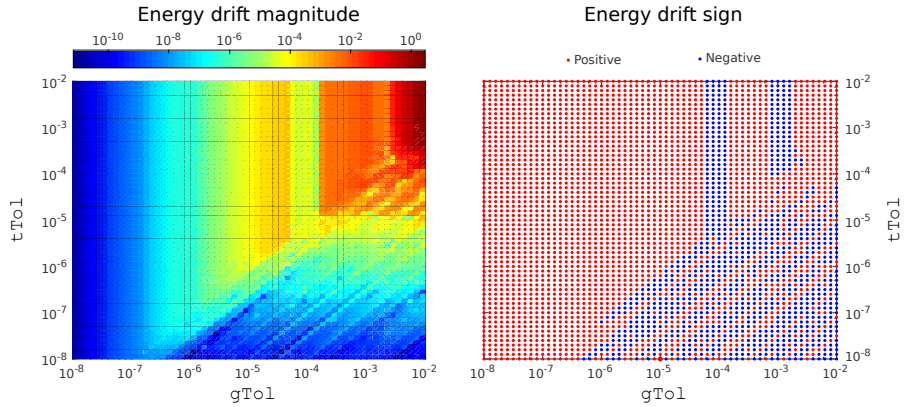

Figure 9: Tolerance influence on the relative energy drift of the system after two periods.

$\texttt{gTol} = \texttt{tTol}$	10^{-8}	10^{-6}	10^{-4}	10^{-2}
Relative energy drift $\Delta E/E_0$	$1.51 \cdot 10^{-11}$	$3.50 \cdot 10^{-7}$	$-4.80 \cdot 10^{-5}$	3.70
Number of contact cycles	975	921	889	656
$\mathcal{O}(\Delta W^c) = \kappa \cdot \texttt{gTol} \cdot \mathcal{O}(g)$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-3}$	$1.8 \cdot 10^{-1}$	$1.8 \cdot 10^{1}$
Worst case drift	$1.75 \cdot 10^{-2}$	$1.66 \cdot 10^{0}$	$1.60 \cdot 10^{2}$	$1.18 \cdot 10^{4}$

Table 2: Influence of the integration scheme numerical damping on the procedure performance.

net work of the contact forces over a contact cycle is not necessarily positive. For this specific problem, we observe that `gTol` is the single control parameter of the energy drift at low values, for the drift is virtually independent of `tTol`. For larger values of `gTol`, however, the role of `tTol` is more significant; the instability of the procedure also appears, with drifts up to several times the initial energy level for $\texttt{gTol} = \texttt{tTol} = 10^{-2}$; see the last column of Table 2.

Further investigation reveals that the `gTol` estimate provided by equation (29) is very conservative. Table 2 provides details about the overall relative energy drift and the number of contact cycles for a number of values of the tolerances `gTol`, `tTol`. *A posteriori*, having identified $\mathcal{O}(g) = 10^{-3}$ from the simulation results, we have calculated the order of magnitude of the energy drift per contact cycle $\mathcal{O}(\Delta W^c)$ and the worst case drift that could be expected by multiplying it by the number of contact cycles. The magnitudes of the worst case drift are several orders higher than those of the observed drift. The estimate (29) can thus be safely used to choose an order of magnitude for the event-localization tolerance `gTol`, once $\mathcal{O}(g)$ has been identified. Also, given the limited role of `tTol` at low `gTol`, both tolerances can be set equal in a first simulation run, for the sake of simplicity.

4.2. Newton's cradle

The next application example is inspired from the so-called Newton's cradle, a device that consists of collinear identical cable-suspended steel balls that, after initiation of a first impact, collide in a theoretically infinite manner through conservation of both momentum and energy. Far from the original model that comes with its lot of complexities due to the possibility of multiple impacts occurring simultaneously, see for instance [34], the one we consider is an academic simplification. Rather than working with spherical impactors, we consider B identical slender elastic bars with constant properties along their main axis. Figure 10 illustrates this conservative system for $B = 5$. Given the slenderness and the elasticity of the bars, we assume that their motion is ruled by the 1-dimensional wave equation. The problem is thus characterized by parameters (c_0, L, g_0, G_0, v_0), the wave propagation speed in the bars, the bar lengths, the initial gap between the bars, the initial gap between extreme bars and the rigid side walls, and the initial velocity of bar 1 (all other bars being initially at rest), respectively. The specific choice $g_0, G_0 > Lv_0/c_0$ ensures the sequential nature of persistent contact phases.

Denoting by $x \in [0, L]$ the abscissa along a bar and by $t \in \mathbb{R}^+$ the time variable, the motion of the b^{th} bar is ruled by

$$\frac{\partial_b v}{\partial t}(x,t) = c_0^2 \frac{\partial^2 {}_b u}{\partial x^2}(x,t), \qquad {}_b v(x,t) = \frac{\partial_b u}{\partial t}(x,t), \qquad b \in \{1, \cdots, B\}. \tag{33}$$

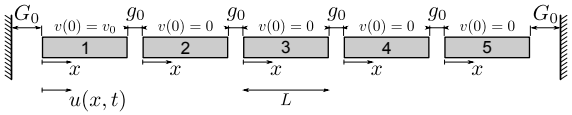

Figure 10: Simplified model of Newton's cradle with $B = 5$ identical bars of length L. Gaps between all bars (g_0) are taken identical and so is it for the gaps with the rigid side walls (G_0). They also verify the condition $g_0, G_0 > Lv_0/c_0$. Bar 1 has initial velocity v_0 while all other ones are at rest.

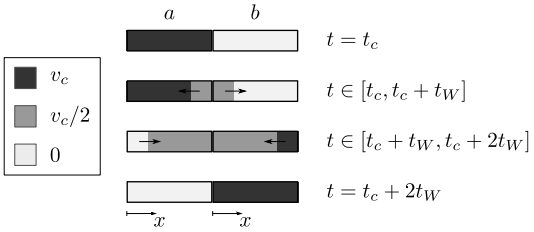

Figure 11: Velocity distribution during a phase of bar/bar contact. At $t = t_c$, the left bar impacts with uniform velocity v_c the right one that is at rest. The persistent contact phase lasts $2t_W = 2L/c_0$, that is the time taken for the wave front to travel forth and back the bars. After completion of the contact phase, both bars are free of vibrations and behave like rigid bodies; the impacted one has uniform velocity v_c and the impacting one is at rest.

The initial conditions read ${}_b u(x,0) = 0, b \in \{1,\dots,B\}$, ${}_1 v(x,0) = v_0$, ${}_b v(x,0) = 0, b \in \{2,\dots,B\}$. The boundary conditions depend on the bar contact configurations which evolve in time. Three configurations are to be considered: free flight, bar/bar contact, bar/wall contact.

During free flight, the bar is unconstrained and free to move axially. In the absence of external forces, the bar is thus at rest, in stationary motion or freely vibrating. Given the problem characteristics, the latter possibility can be discarded; see below.

During bar/bar contact, the conjugated ends of the contacting bars are constrained by force equilibrium and equality of the displacements (and velocities, obviously) at the contacting surfaces (condition of non-interpenetration), and unconstrained at their free ends. Given the identical nature of the bars and the specifics of the initial conditions, we can restrict our consideration to the case of the longitudinal impact of one bar a having uniform velocity v_c on an other one b that is at rest. This impact configuration is treated in details by Graff [46, Sec. 2.4] by use of d'Alembert's solution to the wave equation. Upon closure of the contact interface, a wave front propagates from the contact surface towards the free ends of the contacting bars. It propagates information as to the bar velocity which is discontinuous across it; see Figure 11. This propagation lasts for $2t_W = 2L/c_0$; that is, the time necessary for a round trip of the wave front inside the bars. During the contact phase, the velocity in each bar is given by

$$
\begin{aligned}
&{}_a v(x,t) = v_c \begin{cases} 1 & c_0(t-t_c) < L - x \\ \frac{1}{2} & c_0(t-t_c) \geq L - x \end{cases}, \quad {}_b v(x,t) = v_c \begin{cases} \frac{1}{2} & c_0(t-t_c) > x \\ 0 & c_0(t-t_c) \leq x \end{cases}, \qquad t \in [t_c, t_c + t_W]\,, \\
&{}_a v(x,t) = v_c \begin{cases} 0 & c_0(t-t_c) > L + x \\ \frac{1}{2} & c_0(t-t_c) \leq L + x \end{cases}, \quad {}_b v(x,t) = v_c \begin{cases} \frac{1}{2} & c_0(t-t_c) < 2L - x \\ 1 & c_0(t-t_c) \geq 2L - x \end{cases}, \quad t \in [t_c + t_W, t_c + 2t_W]\,.
\end{aligned}
\tag{34}
$$

After completion of the contact phase, the impacting bar is thus at rest and the impacted one has uniform velocity v_c, as a result of perfect momentum transfer. Both are free of vibrations and deformations. Also, each bar has achieved a displacement of $v_0 t_W$ over the duration of the contact phase.

During bar/wall contact, the situation is similar to that during bar/bar contact except that the impacting bar experiences a velocity reversal. The four phases of the contact sequence are depicted in Figure 12. At time $t = t_c + t_W$, the bar is in uniform compression state with strain $\partial_b u/\partial x = -v_c t_W/L$. The contact phase completes after $2t_W$. Over the duration of persistent contact, the velocity field is defined by

$$
\begin{aligned}
{}_b v(x,t) = \quad & v_c \begin{cases} 1 & c_0(t-t_c) < L - x \\ 0 & c_0(t-t_c) \geq L - x \end{cases}, \qquad t \in [t_c, t_c + t_W]\,, \\
{}_b v(x,t) = -& v_c \begin{cases} 0 & c_0(t-t_c) > L + x \\ 1 & c_0(t-t_c) \leq L + x \end{cases}, \qquad t \in [t_c + t_W, t_c + 2t_W]\,.
\end{aligned}
\tag{35}
$$

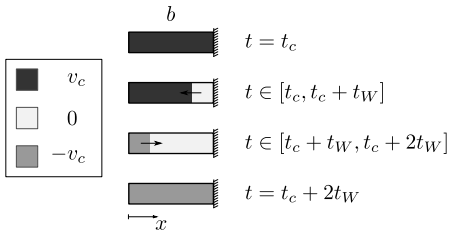

Figure 12: Velocity distribution during a phase of bar/wall contact. At $t = t_c$, the bar impacts a rigid wall with uniform velocity v_c. The persistent contact phase lasts $2t_W = 2L/c_0$; that is, the time taken for the wave front to travel forth and back the impacting bar. After completion of the contact phase, the impacting bar rebounds off the wall with uniform opposite velocity $-v_c$.

Given the chain nature of the system, the specificity of the initial conditions and the system behavior during interaction phases, the system global response corresponds to a sequence of free flight, bar/bar and bar/wall contact phases. Furthermore, motion will be periodic and simultaneous events are not expected for $g_0, G_0 > v_0 t_W$, for, in this case, each persistent contact phase completes before the next begins.

The computation of the system motion requires that of contact closure and opening times (event times). These can be established by composition of motion phases. Bars with $b \in \{2, \dots, B-1\}$ experience four contact phases with their immediate neighbors over a period of motion, two in the positive direction, two in the negative direction. These occur at times that, modulo the motion period, read

$$\begin{aligned} &b-1/b : t_{c,1} = (b-1)\frac{g_0}{v_0} + (b-2)t_W, && b/b+1 : t_{c,2} = t_{c,1} + \frac{g_0}{v_0} + t_W, \\ &b+1/b : t_{c,3} = t_{c,2} + 2\left(t_W + \frac{G_0}{v_0}\right) + 2\,(B-b-1)\left(t_W + \frac{g_0}{v_0}\right), && b/b-1 : t_{c,4} = t_{c,3} + \frac{g_0}{v_0} + t_W. \end{aligned} \tag{36}$$

Bars 1 and B experience only three contact phases over a period. These take place at times (again, modulo the motion period)

$$\begin{aligned} &1/2 : t_{c,1} = \frac{g_0}{v_0}, && B-1/B : t_{c,1} = (B-1)\frac{g_0}{v_0} + (B-2)t_W, \\ &2/1 : t_{c,2} = (2B-3)\frac{g_0}{v_0} + (2B-2)t_W + 2\frac{G_0}{v_0}, && B/\text{wall} : t_{c,2} = t_{c,1} + \frac{G_0}{v_0} + t_W, \\ &1/\text{wall} : t_{c,3} = t_{c,2} + 2t_W + \frac{g_0}{v_0} + \frac{G_0}{v_0}, && B/B-1 : t_{c,3} = t_{c,2} + \frac{G_0}{v_0} + 2t_W. \end{aligned} \tag{37}$$

The motion period is readily shown to be

$$T = 4\frac{G_0}{v_0} + 2(B-1)\frac{g_0}{v_0} + 2(B+1)t_W. \tag{38}$$

It represents the sum of the durations of free flight motion and wave propagation in the system. The displacement field merely follows from the time integration of the piecewise velocity field.

Figure 13 shows the evolution of the system motion over a period for $B = 5$. Time is displayed along the horizontal axis while the vertical axis represents the bar positions. Velocity information is superimposed using colors. Except when involved in a persistent contact phase (bar/bar or bar/wall), the bars have uniform velocity and zero deformation; they behave like rigid bodies. The reason for condition $g_0, G_0 > v_0 t_W$, to guarantee non simultaneous contact phases, is now evident.

An interesting perspective of the problem is that of the rigid body motion that can be associated with the spatial average of bar motions

$$\langle {}_b v \rangle (t) := \frac{1}{L}\int_0^L {}_b v(x,t)\, dx. \tag{39}$$

Given the piecewise constant velocity distribution in the bars during persistent contact, the average velocity of contacting bars is given by the linear envelope of their velocities prior to and after contact. Averaging equations (34) and (35) respectively yields

$$\langle {}_a v \rangle (t) = v_c\left(1 - \frac{t-t_c}{2t_W}\right), \qquad \langle {}_b v \rangle (t) = v_c\frac{t-t_c}{2t_W}, \qquad t \in [t_c, t_c + 2t_W], \tag{40}$$

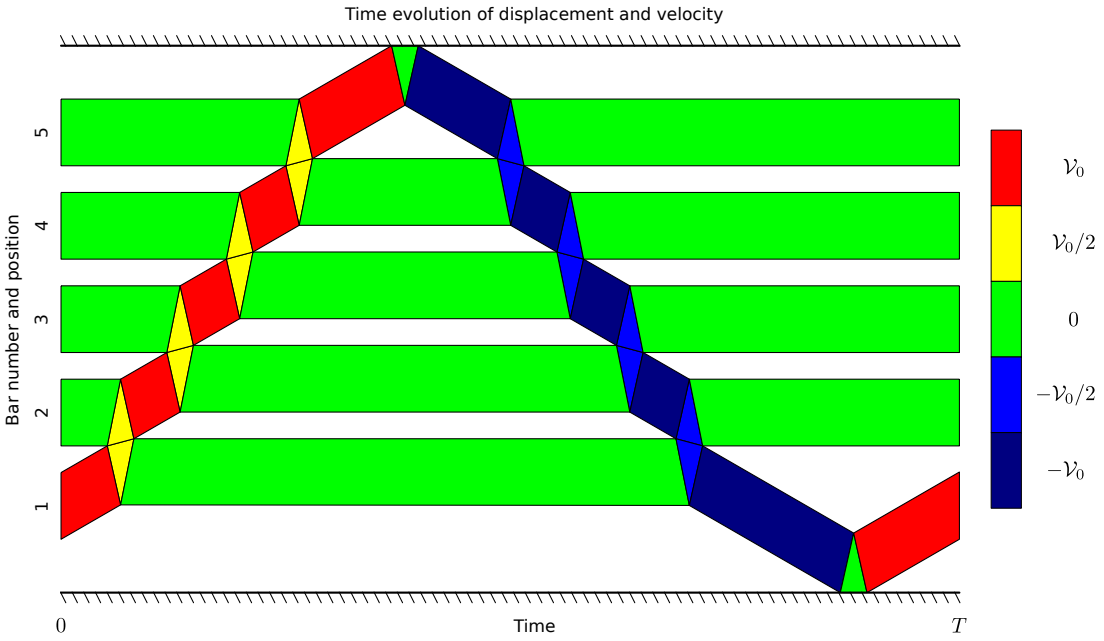

Figure 13: Motion of the system for $B = 5$ over one period T. Momentum propagates from initially moving bar 1 via the successive interaction phases, to eventually return to bar 1 that recovers its initial state given the system geometry and conservativeness, and the identical nature of the bars.

and

$$\langle {}_b v \rangle (t) = v_c \left(1 - \frac{t - t_c}{t_W} \right), \quad t \in [t_c, t_c + 2t_W], \tag{41}$$

with contact velocity $v_c = \pm v_0$ depending on the direction of motion at the initiation of contact. The average displacement follows from the time integration of the average velocity field. During contact phases, we respectively have

$$\langle {}_a u \rangle (t) = v_c (t - t_c) \left(1 - \frac{t - t_c}{4t_W} \right), \qquad \langle {}_b u \rangle (t) = v_c \frac{(t - t_c)^2}{4t_W}, \qquad t \in [t_c, t_c + 2t_W], \tag{42}$$

and

$$\langle {}_b u \rangle (t) = v_c (t - t_c) \left(1 - \frac{t - t_c}{2t_W} \right), \qquad t \in [t_c, t_c + 2t_W],$$

again with $v_c = \pm v_0$. During free flight phases, a single bar moves with uniform average velocity $\pm v_0$ while the others are at rest.

Figure 14 depicts the exact average motion of a system with $B = 5$ bars over a duration of two periods. It is to be viewed in parallel with Figure 13. The sequential nature of the system motion is well observed from the piecewise smooth velocity field, thereby confirming the absence of multiple contact.

The constrained equations of motion are a simplified version of (8), since there is no structural damping nor external forcing. They read

$$\mathbf{M}\dot{\mathbf{v}} + (\mathbf{K} + \mathbf{K}_{\mathcal{A}})\,\mathbf{u} = \mathbf{f}^0_{\mathcal{A}}, \qquad \mathbf{v} = \dot{\mathbf{u}}, \tag{43}$$

where, again, set $\mathcal{A}$ is given by (9) and the contact-related stiffness matrix and forcing term are defined in equation (10). Matrices $\mathbf{M}, \mathbf{K} \in \mathbb{R}^{Bm \times Bm}$ refer to the global mass and stiffness matrices of the multibody problem; that is, the B times diagonal repetition of ${}_b\mathbf{M}, {}_b\mathbf{K} \in \mathbb{R}^{m \times m}$, the assembled matrices for a single bar on the basis of the elementary matrices given in (31). The initial conditions are defined as

$${}_b\mathbf{u}_0 = \mathbf{0}, \quad b = 1, \ldots, B, \qquad {}_1\mathbf{v}_0 = \mathbf{1} v_0, \qquad {}_b\mathbf{v}_0 = \mathbf{0}, \quad b = 2, \ldots, B, \tag{44}$$

where vector $\mathbf{1} \in \mathbb{R}^m$ has m unit entries, and the problem has $B + 1$ contact interfaces, requiring $\mathbf{W} \in \mathbb{R}^{Bm \times (B+1)}, \mathbf{g}_0 \in \mathbb{R}^{B+1}$.

In the setting of the discrete problem, a counterpart to equation (39) can be obtained by projection of the velocity field of bar b into its mass matrix

$$\langle {}_b \tilde{v} \rangle (t_n) = \frac{1}{{}_b m} \mathbf{1}^T {}_b\mathbf{M}\, {}_b\mathbf{v}_n \tag{45}$$

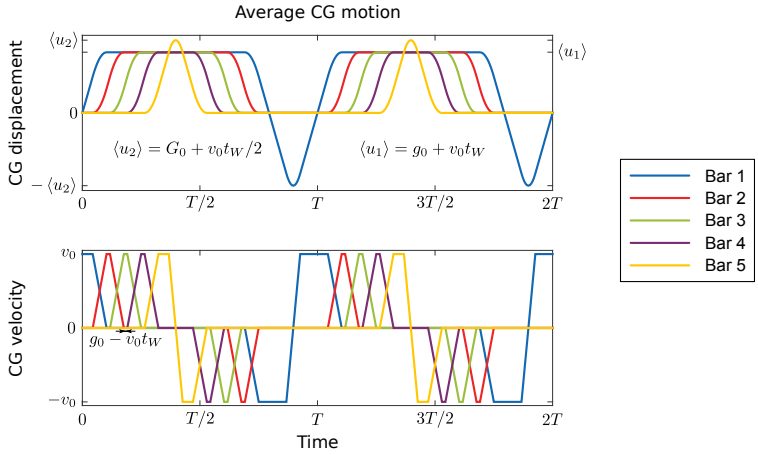

Figure 14: Average motion of the bar centers of gravity over two periods, for $B = 5$. Scaling corresponds to $(v_0, G_0, g_0, L, c_0) = (0.05, 0.125, 0.075, 1, 1)$.

and normalization by its mass ${}_b m = \sum_{k,l} {}_b M_{kl}$. Vector $\mathbf{1}^T$ plays the role of a summation operator. Except for bodies in contact with a rigid side wall, the averaging operation is equivalent to extracting the rigid body component of the bar motion, for the eigenmode associated with rigid body motion is parallel to $\mathbf{1}$. During persistent contact, the bodies colliding the side walls cannot experience rigid body motion, and the operation is a mere averaging projection weighted by the mass matrix entries. The same formula applies to compute the average displacement field at the center of gravity by substitution of the velocity field with the displacement field, $\mathbf{v}_n \leftarrow \mathbf{u}_n$.

To compare the numerical response with the analytical one, we introduce the following dimensionless error norm

$$E(t) := \frac{1}{B} \sum_{b=1}^{B} \left(\frac{|\langle {}_b\tilde{u} \rangle - \langle {}_b u \rangle|}{\sqrt{g_0 G_0}} + \frac{|\langle {}_b\tilde{v} \rangle - \langle {}_b v \rangle|}{v_0} \right), \tag{46}$$

where field variables with an overhead tilde refer to the finite element response and others to the analytical solution. Figure 15 shows the simulation results obtained with the two integration schemes presented in Section 3.1. Identical timestepping and root-solving parameters have been used for both simulations. The timestep was chosen so that there are at least 5 computed point per lowest eigenfrequency of the system, whatever the contact configuration, and $(\texttt{gTol}, \texttt{tTol}) = (10^{-8}, 10^{-8})$. Three configurations have been considered for the integration schemes, $\rho_\infty \in \{0, 0.5, 1.0\}$ and the contact stiffness was set to 100 times the largest component of the stiffness matrix. The model parameters are identical to those used in Figure 14, namely $(v_0, G_0, g_0, L, c_0) = (0.05, 0.125, 0.075, 1, 1)$. The 2-level BoTr scheme produces significantly more accurate results than the generalized-α one, at equivalent spectral radius at infinite frequency. Simulation performances are reported in Table 3. Except for the conservative setting, the generalized-α scheme requires less computational effort than the 2-level BoTr scheme. This is to be credited to the higher effective damping of the scheme that reduces chattering during persistent contact and the total number of increments to completion of the computation. In the conservative setting, the opposite score is observed. Even though the total number of increments is of the same order for both schemes, the 2-level BoTr scheme is about 70% faster than the 3-level generalized-α scheme. This results from the higher cost of event localization with the 3-level scheme than with the 2-level scheme.

ρ_∞	1.0	0.5	0.0	1.0	0.5	0.0
Increment number	59330	41089	22993	58338	22841	14943
CPU time (norm.)	9.50	7.59	3.00	13.60	3.63	1.00
Integration scheme	2-level BoTr			Generalized-α		

Table 3: Influence of the integration scheme and the numerical damping on the procedure performance.

Selection of the integration scheme and its numerical setting should thus be conducted in accordance with the simulation purpose. In light of the presented results, the generalized-α scheme with $\rho_\infty = 0$ is likely to be a good candidate for the fast computation of coarse results, whereas the 2-level BoTr scheme with some dissipation $\rho_\infty \in [0.5, 1)$ will be preferred when more accurate responses are desired.

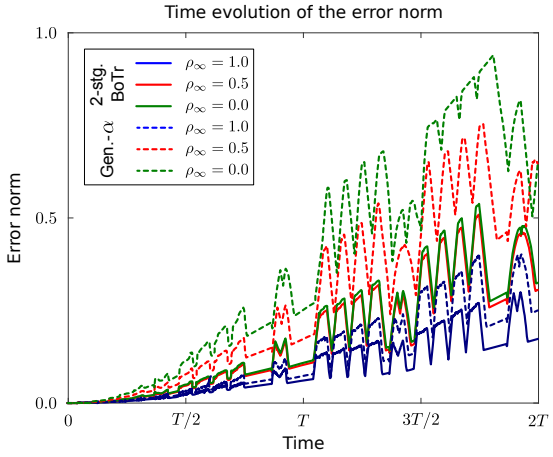

Figure 15: Evolution of the numerical error, according to equation (46), for $B = 5$ and model parameters $(v_0, G_0, g_0, L, c_0) = (0.05, 0.125, 0.075, 1, 1)$. Each bar is modeled using 100 linear finite elements, the root-solving tolerances are set to $(\texttt{gTol}, \texttt{tTol}) = (10^{-8}, 10^{-8})$ and the timestep h is chosen such that there are at least 5 computed points per lowest eigenperiod of the system, whatever its contact configuration, yielding $(h, h_{\text{red}}) = (5.13 \cdot 10^{-3}, 4.01 \cdot 10^{-4})$. The 2-level BoTr scheme achieves significantly lower error levels than the generalized-α scheme, for equivalent ρ_∞.

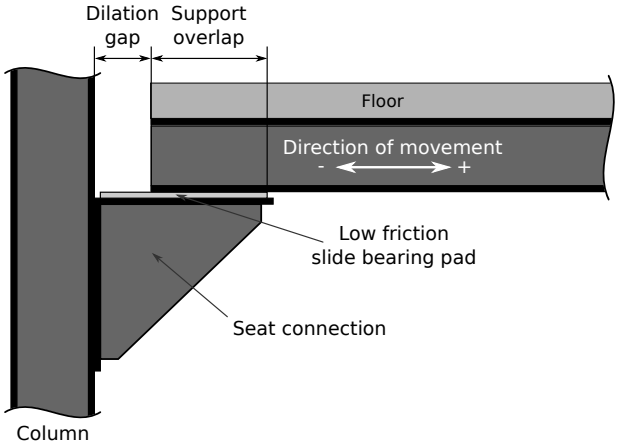

Figure 16: Slip surface joint used in steel and concrete structures to prevent excessive thermal stressing.

4.3. Earthquake engineering

The third considered application belongs to the field of earthquake engineering. It is that of a building subject to horizontal ground acceleration. The building is assumed to comprise two main substructures connected by slip surface joints that provide the necessary dilation freedom to withstand differential movement due to thermal changes without engendering excessive stressing of the structure. Figure 16 illustrates the typical geometry of the joints used in buildings. They allow floor movement in a given direction through slipping on a low friction pad. However, should this movement be too large, structural pounding (negative differential movement larger than dilation gap) or floor collapse (positive differential movement larger than support overlap) could occur. Both situations prove detrimental to the structural integrity of the building. Their risk of occurrence should thus be evaluated as part of the building design process whenever seismic activity is expected at the construction site.

For simplicity, we consider a 2-dimensional representation of a building in this example. There is nonetheless no restriction for extension to a 3-dimensional model. Figure 17 details the geometry of the building and its first eigenmodes and eigenfrequencies. Euler beam elements with 2 nodes (cubic Hermite interpolation) [29, 31] have been used to model the structural features of the building. These elements have three degrees of freedom per node, namely the displacements along and transverse to the beam segment and the in-plane rotation at the beam extremity. After appropriate rotation, these can be matched to the global axes of the 2-dimensional problem that are defined by the horizontal, vertical and out-of-plane directions; see the definitions of u_h, u_v, u_z in Figure 17. Given the loading nature (excitation through building foundation), horizontal displacements are defined relatively to the ground. Both substructures feature the same structural elements (reinforced concrete columns and floor

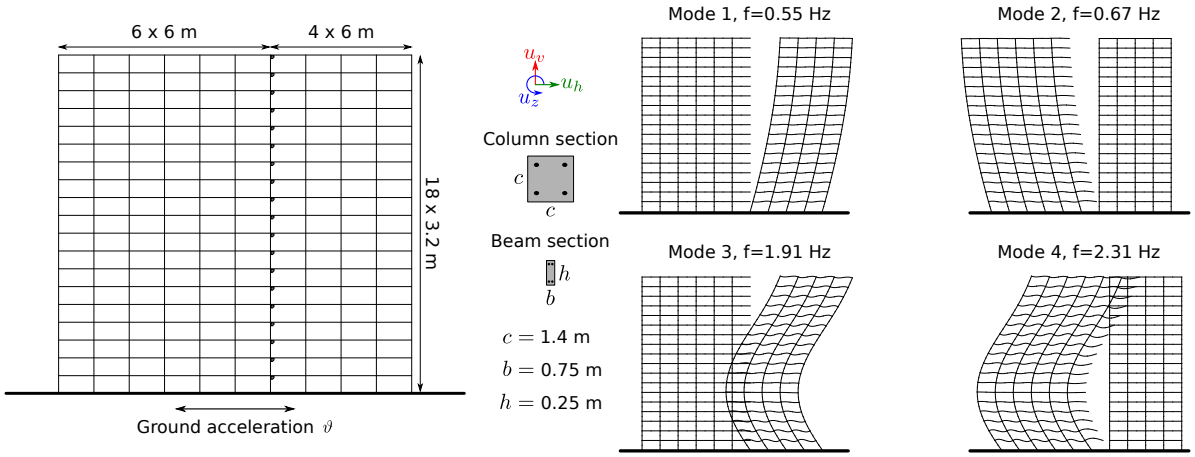

Figure 17: 18-story building with 10 horizontal spans. The left and right substructures are free to move horizontally at slip surface joints ($\bullet$). The unequal width of the building substructures results in different vibratory responses for each subsystem that can cause structural pounding or collapse under ground excitation.

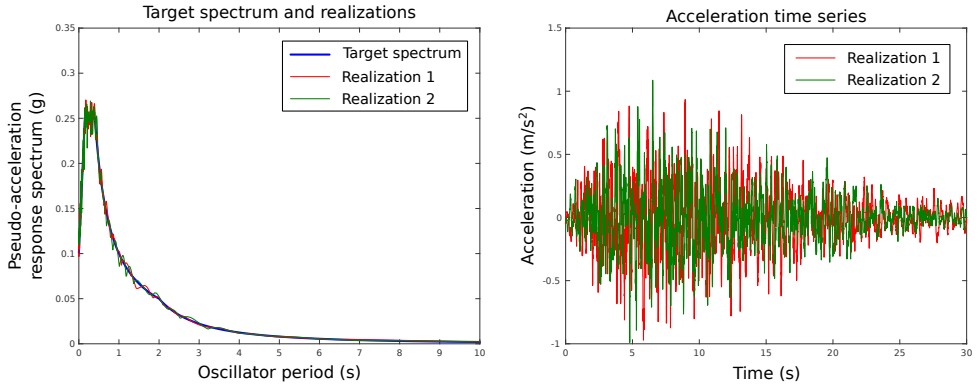

Figure 18: Ground pseudo-acceleration response spectrum and two realizations (left), following Eurocode 8 [48] (type 1, subsoil A, PGA $= 0.1g$), and acceleration time series corresponding to the two realizations given in the spectrum plot.

beams, see cross sections in Figure 17) and are subject to the same nominal static load. The building is 18-story tall; however, its left substructure is fifty percent wider than its right one. Consequent to this width difference is a discrepancy between the substructure eigenresponses and the possibility of structural pounding or floor collapse under ground excitation, through differential motion.

The equations of motion governing the problem read

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} + \mathbf{K}\mathbf{u} = \mathbf{M}\mathbf{1}_h\varrho - \boldsymbol{\kappa}\mathbf{W}\left[\mathbf{g}_0 + \mathbf{W}^T\mathbf{u}\right]_-, \qquad \mathbf{v} = \dot{\mathbf{u}}. \tag{47}$$

Boundary conditions are imposed by elimination of the translational degrees of freedom connected to the ground, yielding $m = 1775$ for the simulations subsequently presented. Rayleigh damping [31, 47] is added, with five percent of critical damping on the first two eigenmodes. The effect of gravity is neglected before the importance of the building response to ground motion. Initial conditions are therefore set to rest conditions, $\mathbf{u}_0 = \mathbf{v}_0 = \mathbf{0}$. Ground acceleration is denoted by ϱ while vector $\mathbf{1}_h \in \mathbb{R}^m$ contains zero entries but at rows corresponding to horizontal displacement degrees of freedom u_h where it has unit entries. The accelerograms defining the ground acceleration have been numerically generated by filtering of a random process so as to target a given pseudo-acceleration response spectrum. This spectrum follows the recommendations of Eurocode 8 [48] (type 1, subsoil A, PGA $= 0.1g$). Figure 18 shows the target spectrum (blue line) and the response spectra of two realizations (red and green lines); time series are also displayed for the two realizations. Gap functions are defined on horizontal degrees of freedom that coincide with slip surface joints in a number equal to the floor number, $q = 18$. Relative displacements can be used for their definition, as the foundations of both substructures experience the same horizontal motion. Vertical displacements are neglected in the definition of the gap functions, for the building response is predominantly horizontal.

For the purpose of this example, the initial gap vector is considered as a design parameter of the

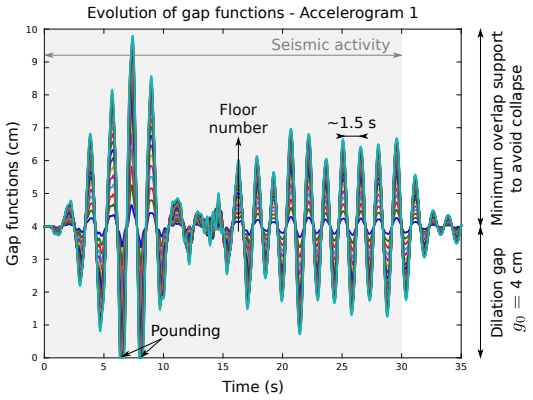

Figure 19: Typical evolution of the gap functions during seismic activity. Each line corresponds to a floor number; the larger the gap opening, the higher floor the result relates to. The dissipative 2-level BoTr scheme was used for the simulations, with $\rho_\infty = 0.5$ and at least 2.5 points per lowest eigenperiod. The contact stiffness was set to 100 times the maximum entry of the global stiffness matrix.

building, notwithstanding thermal dilation considerations. Its influence on the occurrence of pounding and on the magnitude of positive differential movement is assessed.

Figure 19 displays the evolution of all gap functions over the duration of the quake, for realization 1 depicted in Figure 18. The 2-level BoTr scheme with $\rho_\infty = 0.5$ has been used with at least 2.5 computed points per minimum eigenperiod of the system and the contact stiffness set to 100 times the largest entry of the stiffness matrix. During free flight phases, the time distance between peaks is of the same order of magnitude as the eigenperiods of the first two modes, see Figure 17. Also, during positive peaks, the differential movements at slip surface joints scale with the elevation of the floor they respectively relate to, with a maximum magnitude at the top floor. Both arguments confirm that the building response is dominated by the first two eigenmodes, whose deformation envelope is similar to the first mode of a fixed-free beam. Such low frequency response is typical for buildings subject to seismic activity. Consequently, structural pounding also typically appears first at the roof level. As a result of the complex external excitation and the finite accuracy of the event-localization procedure, simulations have revealed the simultaneous occurrence of events. These have confirmed the robustness of the proposed procedure to such algorithmic complications.

Figure 20 represents the maximum positive differential movement over all 18 gap functions and over the duration of the seismic activity for several values of the initial gap opening using the same simulation parameters as for Figure 19. Simulations were performed for 10 realizations of the ground acceleration. Results show that the occurrence of structural pounding reduces the maximal positive differential movement at the slip surface joint, which saturates at about $g_0 = 8$ cm, the initial clearance for which pounding is prevented. This result is nothing but a consequence of the inertial nature of the substructure motions. Vibration growth in a mechanical structure is not instantaneous in the presence of inertia; it builds up in finite time. However, the presence of unilateral constraints alters this growth process which results in a reduction of the maximal differential movement. Accordingly, if the slip surface joints are designed so that they can damp the effects of pounding and prevent excessive damage that would endanger the building structural integrity, limited dilation clearances and support overlaps can be used. If pounding is to be avoided at all costs, clearances and overlaps should then be chosen large enough so that differential movement in both directions can be borne by the joint. Alternatively, other technologies specifically designed to withstand seismic activity can be used.

5. Conclusions

This paper presents a procedure dedicated to the time integration of problems of linear structural dynamics subject to unilateral elastic constraints. The retained approach is that of event-driven integration, since the end application the procedure will be applied to—percussive drilling, not covered in this paper, however—requires the trigger of elastic constraints from arbitrary functions of the state (displacement and velocity). To that end, the contact problem is reformulated as a hybrid model by use of an active set strategy, *i.e.*, contact is either active or inactive; the occurrence of contact openings and closures must be tracked by the integration procedure through root solving. For the purpose of event detection and event localization (the two root solving steps), cubic Hermite interpolation is used to provide

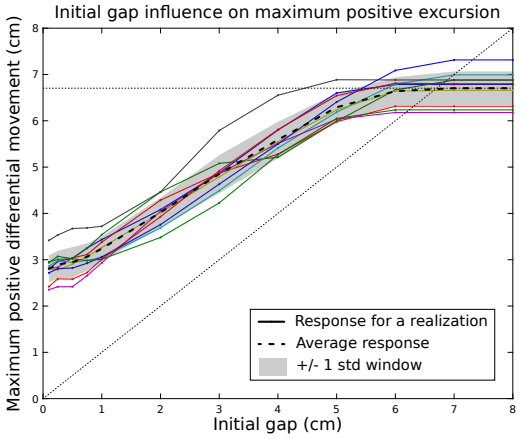

Figure 20: Influence of the initial gap on the maximum maximorum positive excursion attained during earthquake simulation, *i.e.*, the maximum positive differential displacement value over all gap functions over the entire duration of the quake.

a continuous extension of the displacement field that defines gap openings in combination with a specific iterative event-localization strategy that localizes events after they have occurred. Although presented in the linear framework, these two essential elements could be included in a more generic solver addressing nonlinear structural problems. Given the use of Hermite interpolation for the continuous extension, the proposed integration procedure is best combined with one-step integration schemes as only information at the timestep extremities are exploited to construct the continuous extension. Important algorithmic steps are presented in the paper and a MATLAB implementation is provided online (see Appendix). In particular, the handling of numerical accumulation and numerical grazing are addressed in detail.

By design, the event-driven integration procedure offers several advantages. First, it naturally overcomes the energetic instabilities arising from unilateral elastic constraints; as demonstrated, the use of stringent tolerances for event localization permits to control the energy drift ensuing from the closures and openings of contact interfaces. Second, it exploits the piecewise linear character of the governing equations. As such, no nonlinear solver, *e.g.,* based on Newton-Raphson iteration, is required to march the equations of motion in time. Additionally, in virtue of the first statement, global stability of the procedure is guaranteed provided the embedded integration scheme is exploited in its stability region. Unconditionally stable implicit schemes, such as the 2-level BoTr or the generalized-α schemes that we briefly recall in the paper, are therefore preferably used, even though the proposed algorithm can be used with conditionally stable explicit ones as well. Third, although the proposed continuous extension covers the displacement field only, it can readily be extended to the velocity field, as the equation of motion directly provides the acceleration field required to form the Hermite interpolants to the velocity field (provided the mass matrix is positive definite). This addition enables the adaption of the event-driven procedure to handle arbitrary event functions involving the velocity field, as is the case in percussive drilling. Thus, not only is the proposed procedure adapted to the simulation of structural dynamics in presence of unilateral elastic constraints but also is it suitable for the simulation of hybrid linear systems governed by second-order ODEs.

The method also presents a couple of downsides. Unless specific safeguards are implemented, it fails at simulating system trajectories exhibiting Zeno behavior. Additionally, the computational burden significantly increases with the number of contact interfaces. Indeed, event-detection must be performed for each contact interface, at each timestep; this requires solving a low-dimension eigenvalue companion problem which can prove expensive, as the use of floating point arithmetic prevents a robust utilization of the conventional Sturm sequences and the likes for the detection of polynomial roots in a specific interval. Also, as demonstrated in the proposed application examples, the efficiency of the method is significantly reduced in the presence of constraint chatter. This situation, that arises in the presence of persistent contact and a stiff constraint, is, nevertheless, seriously improved by the use of numerically dissipative time integration schemes, as they provide the necessary damping to tame the chattering oscillations. For these two reasons, it is likely that large scale problems with numerous unilateral elastic contact constraints will be more efficiently handled with other approaches dedicated to contact dynamics, *e.g.*, the EMCA and EDMC schemes [4, 32].

In the sequel of the algorithmic developments, the paper proposes three application examples that

span the whole range of event handling complexity. The elastic bouncing bar benchmark, a 1-dimensional wave propagation problem, features a single contact interface and the occurrence of persistent contact. Assessment of numerical damping influence reveals that it dampens chattering and increases the accuracy of the predicted contact force. Also, the analysis of the energetic drift of the system shows that the estimate provided for the choice of the event-localization tolerance is very conservative. The next example is inspired from Newton's cradle. Featuring multiple contact interfaces, the problem is, however, designed such that simultaneous events should not take place. Elements of its analytical solution are provided so that it can be used as a benchmark for other simulation solutions. A performance comparison between the two reviewed integration schemes shows that, in the dissipative setting, the generalized-α scheme is less accurate than the 2-level BoTr scheme but nonetheless computationally cheaper even though it is a 3-level scheme. Coarse solutions are thus preferably computed with the former scheme whereas the latter will be used for increased accuracy. The third illustration belongs to the field of earthquake engineering. The presented procedure is employed to simulate the response of a fictitious building to an earthquake. Multiple contact interfaces are defined and simultaneous events do occur. Completion of the simulations shows the robustness of the proposed computation strategy (as do previous examples). It also illustrates the influence of unilateral elastic constraints on differential movements of building substructures.

6. Acknowledgements

The first author expresses his thanks to the Graduate School of the University of Minnesota, for the partial support of this research through a Doctoral Dissertation Fellowship.

References

[1] N. Kikuchi, J. T. Oden, Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods, Society for Industrial and Applied Mathematics, 1981.
[2] P. Wriggers, Computational contact mechanics, John Wiley and Sons, Ltd, 2006.
[3] V. Acary, Projected event-capturing time-stepping schemes for nonsmooth mechanical systems with unilateral contact and coulomb's friction, Computer Methods in Applied Mechanics and Engineering 256 (0) (2013) 224–250. doi:10.1016/j.cma.2012.12.012.
[4] F. Armero, E. Petőcz, Formulation and analysis of conserving algorithms for frictionless dynamic contact/impact problems, Computer Methods in Applied Mechanics and Engineering 158 (3-4) (1998) 269–300. doi:10.1016/S0045-7825(97)00256-9.
[5] D. Doyen, A. Ern, S. Piperno, Time-integration schemes for the finite element dynamic signorini problem, SIAM Journal on Scientific Computing 33 (1) (2011) 223–249. doi:10.1137/100791440.
[6] B. Lundberg, Energy transfer in percussive rock destruction-II: Supplement on hammer drilling, International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts 10 (5) (1973) 401–419. doi:10.1016/0148-9062(73)90025-9.
[7] L. Karlsson, B. Lundberg, K. Sundin, Experimental study of a percussive process for rock fragmentation, International Journal for Rock Mechanics and Mineral Science-Geomechanics Abstracts 26 (1) (1989) 45–50. doi:10.1016/0148-9062(89)90524-X.
[8] A. Depouhon, V. Denoël, E. Detournay, A drifting impact oscillator with periodic impulsive loading: Application to percussive drilling, Physica D: Nonlinear Phenomena 258 (0) (2013) 1–10. doi:10.1016/j.physd.2013.04.012.
[9] R. I. Leine, H. Nijmeijer, Dynamics and Bifurcations of Non-smooth Mechanical systems, Springer-Verlag, 2004.
[10] V. Acary, B. Brogliato, Numerical Methods for Nonsmooth Dynamical Systems: Applications in Mechanics and Electronics, Springer-Verlag, 2008.
[11] L. Dieci, L. Lopez, A survey of numerical methods for ivps of odes with discontinuous right-hand side, Journal of Computational and Applied Mathematics 236 (16) (2012) 3967–3991.
[12] A. F. Filippov, Differential Equations with Discontinuous Righthand Sides, Mathematics and its Applications, Kluwer Academic Publishers, 1988.
[13] R. Goebel, R. G. Sanfelice, A. R. Teel, Hybrid dynamical systems, Control Systems, IEEE 29 (2) (2009) 28–93. doi:10.1109/MCS.2008.931718.
[14] SICONOS: INRIA open-source software platform, http://siconos.gforge.inria.fr.
[15] H. Dankowicz, F. Schilder, Recipes for Continuation, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2013.
[16] M. di Bernardo, C. J. Budd, A. R. Champneys, P. Kowalczyk, Piecewise-smooth dynamical systems: theory and applications, Vol. 163 of Applied mathematical science, Springer-Verlag, London, London, 2008.
[17] T. Schindler, V. Acary, Timestepping schemes for nonsmooth dynamics based on discontinuous galerkin methods: Definition and outlook, Mathematics and Computers in Simulation 95 (0) (2014) 180–199. doi:10.1016/j.matcom.2012.04.012.
[18] N.-S. Nguyen, B. Brogliato, Shock dynamics in granular chains: Numerical simulations and comparison with experimental tests, Granular Matter 14 (3) (2012) 341–362. doi:10.1007/s10035-012-0338-z.
[19] B. Besselink, N. van de Wouw, H. Nijmeijer, A Semi-Analytical Study of Stick-Slip Oscillations in Drilling Systems, Journal of Computational and Nonlinear Dynamics 6 (2). doi:10.1115/1.4002386.
[20] F. Pfeiffer, M. Foerg, H. Ulbrich, Numerical aspects of non-smooth multibody dynamics, Computer Methods in Applied Mechanics and Engineering 195 (50-51) (2006) 6891–6908. doi:10.1016/j.cma.2005.08.012.

[21] N. Huggett, Zeno's Paradoxes, The Stanford Encyclopedia of Philosophy (Winter 2010 Edition). URL http://plato.stanford.edu/archives/win2010/entries/paradox-zeno/
[22] L. F. Shampine, M. W. Reichelt, The MATLAB ODE Suite, Siam Journal on Scientific Computing 18. doi:10.1137/S1064827594276424.
[23] M. B. Carver, Efficient integration over discontinuities in ordinary differential equation simulations, Mathematics and Computers in Simulation 20 (1978) 190–196. doi:10.1016/0378-4754(78)90068-X.
[24] D. Ellison, Efficient automatic integration of ordinary differential equations with discontinuities, Mathematics and Computers in Simulation 23 (1981) 12–20. doi:10.1016/0378-4754(81)90003-3.
[25] J. M. Esposito, V. Kumar, A state event detection algorithm for numerically simulating hybrid systems with model singularities, ACM Transactions on Modeling and Computer Simulation 17 (1). doi:10.1145/1189756.1189757.
[26] T. Park, P. I. Barton, State Event Location in Differential-Algebraic Models, ACM Transactions on Modeling and Computer Simulation 6 (2) (1996) 137–165. doi:10.1145/232807.232809.
[27] L. F. Shampine, I. Gladwell, R. W. Brankin, Reliable Solution of Special Event Location-Problems for ODEs, ACM Transactions on Mathematical Software 17 (1) (1991) 11–25. doi:10.1145/103147.103149.
[28] N. M. Newmark, A method of computation for structural dynamics, Journal of Engineering Mechanics, ASCE 85 (EM3) (1959) 67–94.
[29] T. J. R. Hughes, The finite element method: linear static and dynamic finite element analysis, Dover, 1987.
[30] J. Chung, G. M. Hulbert, Time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method, Journal of Applied Mechanics, Transactions ASME 60 (2) (1993) 371–375. doi:10.1115/1.2900803.
[31] M. Géradin, D. Rixen, Mechanical Vibrations: Theory and Application to Structural Dynamics, Wiley, 1997.
[32] F. Armero, I. Romero, On the formulation of high-frequency dissipative time-stepping algorithms for nonlinear dynamics. part ii: second-order methods, Computer Methods in Applied Mechanics and Engineering 190 (51-52) (2001) 6783–6824. doi:10.1016/S0045-7825(01)00233-X.
[33] A. Depouhon, E. Detournay, V. Denoël, Accuracy of one-step integration schemes for damped/forced linear structural dynamics, International Journal for Numerical Methods in EngineeringIn press.
[34] M. Payr, An Experimental and Theoretical Study of Perfect Multiple Contact Collisions in Bodies, Ph.D. thesis, ETH Zürich (2008).
[35] MATLAB R2013a, The MathWorks, Inc., Natick, Massachusetts, United States.
[36] A. Quarteroni, R. Sacco, F. Saleri, Numerical mathematics, volume 37, Springer-Verlag, 2007.
[37] G. Noh, K.-J. Bathe, An explicit time integration scheme for the analysis of wave propagations, Computers & Structures 129 (0) (2013) 178 – 193. doi:10.1016/j.compstruc.2013.06.007.
[38] N. Nsiampa, J.-P. Ponthot, L. Noels, Comparative study of numerical explicit schemes for impact problems, International Journal of Impact Engineering 35 (12) (2008) 1688–1694. doi:10.1016/j.ijimpeng.2008.07.003.
[39] J. C. F. Sturm, Mémoire sur la résolution des équations numériques, Bulletin des Sciences de Férussac 11 (1829) 419–425.
[40] F. Boulier, Systèmes polynomiaux : que signifie "résoudre"?, Online lecture notes – Consulted on 14-oct-13 (2010). URL http://www.lifl.fr/~boulier/polycopies/resoudre.pdf
[41] M. Suzuki, T. Sasaki, On sturm sequence with floating-point number coefficients, RIMS Kokyuroku 685 (1989) 1–14, http://hdl.handle.net/2433/101222.
[42] R. M. Corless, A. Shakoori, D. A. Aruliah, L. Gonzalez-Vega, Barycentric Hermite Interpolants for Event Location in Initial-Value Problems, Journal of Numerical Analysis, Industrial and Applied Mathematics 1 (2007) 1–14.
[43] J. Kopp, Efficient numerical diagonalization of hermitian 3x3 matrices, International Journal of Modern Physics C 19 (2008) 523–548. doi:10.1142/S0129183108012303.
[44] L. G. Birta, T. I. Oren, D. L. Kettenis, A robust procedure for discontinuity handling in continuous system simulation, Trans. Soc. Comput. Simul. Int. 2 (3) (1985) 189–205.
[45] V. Bahl, A. A. Linninger, Modeling of continuous-discrete processes, in: M. D. Benedetto, A. Sangiovanni-Vincentelli (Eds.), Hybrid Systems: Computation and Control, Vol. 2034 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001, pp. 387–402. doi:10.1007/3-540-45351-2_32.
[46] K. F. Graff, Wave motion in elastic solids, Dover, 1975.
[47] R. Clough, J. Penzien, Dynamics of structures, McGraw-Hill, 1993.
[48] EN1998 – EUROCODE 8: Design of structures for earthquake resistance (2004).

MATLAB routines

The MATLAB routines that have been used to generate the results of Section 4 are available at `http://hdl.handle.net/2268/159502`. The archive file contains seven files, five of which being MATLAB functions. The remaining two are model data for the earthquake engineering example. As much as possible, the variable names in the code follow those used in the paper. The m-files are commented in such a way that variable descriptions are self-sufficient. Defensive coding was not enforced. Generic errors will be generated if input variables are incorrectly defined or issues with the file I/O arise.

`timeIntegration.m & timeIntegration_NoBa13.m`

These files contain an implementation of the event-driven integration procedure described in the paper, the first for three implicit schemes and the latter with the explicit scheme proposed by Noh-Bathe [37]. Both functions take the same four input variables and outputs one.

```
fHeader = timeIntegration(modelParam,integParam,rsParam,header)
fHeader = timeIntegration_NoBa13(modelParam,integParam,rsParam,header)
```

`modelParam` Data structure that contains all parameters related to model definition: `K`, `C`, `M`, `W`, `u0`, `v0`, `g0`, `k_con`, `fHdle`. Particular to the implementation is the definition of the external forcing $\mathbf{f}$ that is done via the function handle `fHdle`. Variable `k_con` contains the diagonal elements of $\boldsymbol{\kappa}$. Should all entries of `k_con` and `g0` be identical, a single scalar value can be specified; it will automatically be duplicated over the number of gap functions q.

`integParam` Vector that contains the parameters relative to the time integration scheme (in indexing order): `schemeID`, `rho`, `hCp`, `hRed`, `nRedH`, `hOut`, `tf`. Three implicit integration schemes are available: trapezoidal method (`schemeID` = 0), 2-level BoTr (`schemeID` = 1), generalized-α (`schemeID` = 2); and the Noh-Bathe explicit scheme (`schemeID` = 3). Timesteps for free flight and active constraint phases are input as `hCp`, `hRed`. Integer `nRedH` specifies the number of increments taken in a free flight phase required for `hCp` to be used and factorization of matrices $\mathbf{H}_0, \mathbf{H}_1$ to be performed (only for implicit schemes). Timestep `hOut` defines the distance between two output points outside event localization (data is output at each localized event) and `tf` is the final simulation time.

`rsParam` Vector with the root-solving parameters `gTol`, `maxIter`, `tTol` (in indexing order). `maxIter` defines the maximum number of iterations accepted during the iterative event-localization procedure.

`header` String that defines the problem name. It serves as a basis for the naming of the results data files.

`fHeader` Prefix string of the results data files. It is identical to `header` but with blank spaces replaced by underscores.

All computational outputs are sent to data files on the disk. A log file and seven history files are created. The former (`fHeader.log`) contains information about localized events and computation performance, while the latter record data history (`fHeader_XXX.his`, with `XXX` $\in$ {`time`, `displacements`, `velocities`, `flags`, `gaps`, `nrg`, `ctcForces`). Their loading requires the knowledge of the model dimension m and the number of contact interfaces q.

The contact forces being computed at the end of the timestep, their reported values might be meaningless in case chattering occurs (with conservative schemes). Nevertheless, this non-sense does not affect the validity of the computed state variables (nodal displacements and velocities).

`bouncingBar.m`

Driver file for the problem of Section 4.1. It defines the model as well as the integration parameters and passes them to `timeIntegration.m` or `timeIntegration_NoBa13.m` according to parameter `schemeID`. Upon completion of the integration procedure, results are loaded and output to the main workspace under the form of a data structure. The evolution of the positions of the bar extremities and center of gravity are plotted. The function prototype reads

```
results = bouncingBar
```

`newtonCradle.m`

This is the main procedure for the problem of Section 4.2. It defines the model and the integration parameters, and passes them to `timeIntegration.m` or `timeIntegration_NoBa13.m` according to parameter `schemeID`. Upon completion of the integration procedure, results are loaded and averaging of the bar positions and velocities is conducted. These are plotted against the analytical response before the function returns the results to the main workspace under the form of a data structure. The function takes no argument

```
results = newtonCradle
```

`quakeSimulation.m`

Driver file for the problem of Section 4.3. It loads the model definition from `buildingModel.mat` and the accelerogram realizations from `ACCEL.ACG`, and defines the integration parameters, then calls `timeIntegration.m` or `timeIntegration_NoBa13.m`. Rayleigh damping is discarded if the explicit scheme is selected as it puts too stringent constraints on the stable timestep of the integration scheme. Upon completion of the integration procedure, results are loaded. The evolution of the $q = 18$ gap functions is plotted before the function returns the results to the main workspace under the form of a data structure. The function prototype reads

```
results = quakeSimulation
```