Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 360 (2020) 112783
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A Total-Lagrangian Material Point Method for solid mechanics problems involving large deformations

Alban de Vaucorbeil[a,b,*], Vinh Phu Nguyen[c], Christopher R. Hutchinson[b]

[a] *Institute for Frontier Materials, Deakin University, Geelong, VIC, 3216, Australia*
[b] *Department of Materials Science and Engineering, Monash University, Clayton 3800, VIC, Australia*
[c] *Department of Civil Engineering, Monash University, Clayton 3800, VIC, Australia*

Received 16 September 2019; received in revised form 2 December 2019; accepted 10 December 2019
Available online xxxx

Abstract

The material point method (MPM) has found successful applications in many engineering problems involving large displacement, large deformation and contacts. The standard MPM formulation, which adopts piece-wise linear basis functions, suffers from the so-called cell-crossing instability, low order of convergence and numerical fracture. Modifications have been made to this standard MPM to mitigate these issues: B-spline MPM (BSMPM), the generalized interpolation material point (GIMP) and convected particle domain interpolation (CPDI) all decrease cell-crossing instabilities and increase the order of convergence, but only CPDI effectively suppresses numerical fracture. However, these methods, CPDI in particular, significantly increase the method's implementation and computational complexity. This paper presents a total Lagrangian MPM, dubbed TLMPM, that overcomes the issues of the conventional MPM while being more efficient and easier to implement than CPDI. The method is used for impact analyses of a cylinder bar made of steel and necking and fracture of cylinder alloy specimens. The numerical solutions are in satisfactory agreement with the experiment data. No numerical fracture occurred for simulations involving very large tensile deformation without special treatment such as done in the CPDI. Convergence analyses using the method of manufactured solutions show that the TLMPM is second-order accurate for problems of which boundary is axis-aligned. For the challenging generalized vortex problem, it also converges quadratically for relatively coarse meshes. Moreover, the model is able to simulate physically based fracture using continuum damage mechanics.

Keywords: Total-Lagrangian; Material Point Method; MPM; Large deformation; TLMPM

1. Introduction

Meshless methods (MMs) present many advantages for the simulation of solids subjected to large deformations over mesh based methods, e.g. finite element method (FEM) and finite difference method. The absence of a mesh enables these methods to deal with larger local distortion and deformations than mesh based methods [1]. However, these methods often suffer from tensile instabilities, rank-deficiency, low order of convergence and numerical fracture.

* Corresponding author at: Institute for Frontier Materials, Deakin University, Geelong, VIC, 3216, Australia.
E-mail address: alban.devaucorbeil@deakin.edu.au (A. de Vaucorbeil).

https://doi.org/10.1016/j.cma.2019.112783
0045-7825/

Numerical fracture is the fracture of a continuum body that occurs when the numerical approximations introduce separations between some of the particles forming the discretization of this body. This type of fracture is therefore artificial. It depends only on the details of algorithm used and thus is unable to match physical fracture, in most cases. Methods susceptible to numerical fracture therefore cannot accurately simulate engineering problems such as machining, wear, impacts, etc. where solids in large deformations experience damage and fracture. Numerical fracture does not occur in Total-Lagrangian (TL) particle based methods such as the reproducing kernel particle method (RKPM) [2] or TL Smooth Particle Hydrodynamics (TLSPH) [3], as long as the original configuration is not updated. RKPM is similar to using a moving-least-squares interpolant kernel [4] which is computationally intensive due to the required pointwise matrix inversions at each time step [5]. TLSPH, on the other hand, is impaired by a rank-deficiency problem that generate another type of instabilities [6,7]. Rank-deficiency can be overcome using stress points [6,8], but this solution was not been widely adopted by the SPH community probably due to the increase of computation time involved. Another solution to control rank-deficiency driven instabilities is to use the hourglass algorithm proposed by Ganzenmüller [7], which is, unfortunately, effective only in elasticity as it inhibits plastic flow. When plasticity is involved, one has to control these instabilities using a viscosity based correction force [7]. This solution is not satisfactory as it introduces a set of un-physical *ad hoc* parameters to be determined using trials-and-errors and do not work in all cases.

The Material Point Method is one of the latest developments in particle-in-cell (PIC) methods. The first PIC technique was developed in the early 1950s by Francis Harlow [9] and was used primarily for applications in fluid mechanics. Early implementations suffered from excessive energy dissipation which was overcome in 1986 by Brackbill and Ruppel and they introduced FLIP — the Fluid Implicit Particle method [10]. The FLIP technique was modified and tailored for applications in solid mechanics by Sulsky and co-workers [11–13] and they named the method MPM (material point method). In the MPM the material is represented by a set of particles overlaying a background mesh that serves as a computational scratchpad. As such, it combines the advantages of both FEM and MMs. The MPM and its variants have been shown to be successful and robust in simulating a large number of complicated engineering problems that involve large deformation and contact such as machining [14] and forming processes [15], wear [16], fracture [17,18], impact [19,20], volumetric collapse (anticracks) in snow [21] as well as creating animations in the movie industry [22–24].

In the MPM, like in SPH, the material points (particles) carry all the field variables. However, unlike in SPH, they serve only as integration points, while the background mesh is used to solve the equations of motion. While SPH suffers from tensile instabilities, the MPM does not. In addition, since the number of particles is usually greater than that of the grid nodes, the MPM does not suffer from rank-deficiency either [19]. The MPM is also more CPU efficient. Indeed, Ma et al. [19] demonstrated that the CPU time per step used by both the MPM and SPH increases linearly with the increase of number of particles, but the rate of increase of the SPH is much higher than that of the MPM. However, in some simulations, the MPM suffers numerical noise especially in the stress or velocity field arising from the material points moving across the background mesh cell boundaries during deformation [19]. This so-called cell-crossing error can be mitigated using a variety of techniques such as high order B-splines basis functions [25], the generalized interpolation material point (GIMP) method [26] or the use of modified gradient of shape functions [27]. However, just like SPH, the MPM suffers from numerical fracture. This is however not the case for the Convected Particle Domain Interpolation (CPDI), the most accurate GIMP variant to date [28–30]. In the CPDI, the shape functions on the background grid are replaced with alternative shape functions defined on each particle's domain (volume attached to each particle). This domain is usually assumed to be a parallelogram [28], which edges have to be tracked. The CPDI adds a level of complexity to the MPM (e.g. complex parallel implementation [31]), solves some problems, while adding others. Another problem crippling the family of the MPM is the absence of convergence when large deformations occur [32,33]. To improve convergence, Gong [32] developed an improved MPM (iMPM) by using Moving Least Squares (MLS) shape functions to improve the velocity mapping and a one-point quadrature to reduce quadrature errors. Gong also used the same ideas to improve CPDI. Gong showed that both the iMPM and the iCPDI have a second order convergence, which is obtained at the expense of computation time. However, the iMPM fails when simulating the large deformation vibration of a cantilever beam as numerical instabilities appear.[1] Similar improved MPM or high-order MPM can be found in [34,35].

[1] This observation was made using our Matlab implementation of iMPM and since we cannot find solutions the results are not published.

Just as Total-Lagrangian SPH can be used to avoid numerical fracture, similarly a Total-Lagrangian Material Point Method (TLMPM) could be developed for the same purpose while being more CPU efficient than TLSPH or CPDI. The idea of the Total-Lagrangian MPM has been used before by Steffen et al. to help decouple space and time errors in the standard MPM [25] and also by Zhu et al. [36] for graphic animations. However its convergence and abilities to simulate the physics of solids subjected to small and large deformations were not discussed, nor were its abilities to model fracture.

In this contribution, we present a Total-Lagrangian Material Point Method and demonstrate for the first time both its convergence and abilities to simulate solids subjected to extremely large deformations. Convergence of the TLMPM is studied using the Method of Manufactured Solutions (MMS) for both large tension/compression and shear deformation problems, while the abilities to simulate solids undergoing large deformations is studied using well established problems such as the vibration of a compliant bar in a gravity field and the Taylor bar impact test. Also, we show using simulations of simple tensile tests that TLMPM paired with continuum damage mechanics is able to simulate ductile materials (such as steels) experiencing large deformations and eventually fracture. Our results are compared with results obtained with an Updated Lagrangian FEM (ULFEM) and with experimental data when the latter is available. Moreover, we investigate the performance of Bernstein basis functions, commonly used in isogeometric analysis [37,38] in the TLMPM and provide for the first time formula of the modified boundary cubic B-splines presented in the PhD thesis of Steffen [39].

Is the TLMPM identical to the Total Lagrangian FEM (TLFEM)? No, The TLMPM is a particle based method, which is fundamentally different from the TLFEM. Indeed, the TLMPM does not require a conforming mesh contrary to the TLFEM. The TLMPM algorithm is as different from TLFEM as the standard MPM (*i.e.* Updated Lagrangian MPM) is different from ULFEM. The absence of a conforming mesh allows the standard MPM to solve problems involving large deformations that the ULFEM cannot. Similarly, this absence allows the TLMPM to solve extremely large deformation mechanics problems that the TLFEM cannot.

2. Total Lagrangian MPM for solid mechanics

2.1. Governing equations

In a Lagrangian formulation, the independent variables are the material coordinates $\boldsymbol{X}$ in the reference (undeformed) configuration and time t. In the total Lagrangian formulation, the stress and strain are Lagrangian, i.e., they are defined with respect to the reference configuration (for example, the first Piola–Kirchhoff stress is employed) and spatial derivatives are computed with respect to the material coordinates. The corresponding weak form therefore involves integrals over the reference configuration. The material coordinates in the current (deformed) configuration $\boldsymbol{x}$ are linked to the reference coordinates $\boldsymbol{X}$ through the displacement $\boldsymbol{u}$ as:

$$\boldsymbol{x} = \boldsymbol{u} + \boldsymbol{X} \tag{1}$$

Therefore, the conservation equations in total Lagrangian are given by [40]

$$\rho = J\rho_0 \tag{2}$$

$$\frac{\partial \boldsymbol{v}}{\partial t} = \frac{1}{\rho_0}\nabla_0 \cdot \boldsymbol{P}^T + \boldsymbol{b} \tag{3}$$

where $\boldsymbol{v} = \frac{\partial \boldsymbol{u}}{\partial t}$ is the velocity field, ρ is the mass density, $\boldsymbol{P}$ is the first Piola–Kirchhoff stress tensor, $\boldsymbol{b}$ are the external forces, and ∇ is the gradient operator. The subscript 0 indicates that the corresponding quantity is evaluated in the reference configuration, while its absence signifies that the current configuration is used. J is the determinant of the deformation gradient $\boldsymbol{F}$:

$$\boldsymbol{F} = \frac{d\boldsymbol{x}}{d\boldsymbol{X}} = \frac{d\boldsymbol{u}}{d\boldsymbol{X}} + \boldsymbol{I} \tag{4}$$

We have skipped boundary and initial conditions as they are standard. Details can be found in e.g. [40]. Constitutive models used in this work are presented in Section 2.2.5.

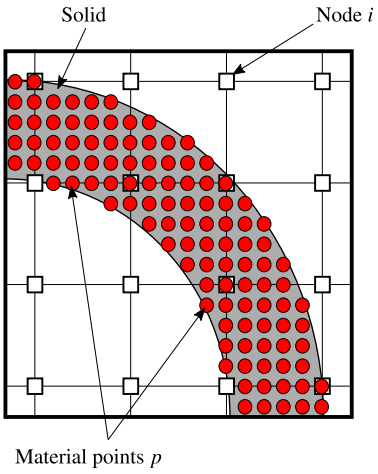


Fig. 1. Discretization: the space is discretized by a Cartesian grid (or background mesh) while solids are discretized using particles. The grid consists of cubes which are referred to as elements or cells interchangeably.

2.2. Total-Lagrangian Material Point Method

A general description of how the Total-Lagrangian Material Point Method (TLMPM) solves the above conservation equations and its difference from the standard MPM is given in this section. In TLMPM, just like in the MPM, the space occupied by the solid under consideration is discretized by a Cartesian grid and the solid itself by a set of material points also called particles as illustrated in Fig. 1 for a 2D problem. Therefore, the method involves two types of point-like objects: particles (material points) and grid nodes. In order to make the necessary distinction between them, in the following, the particles are referred to by the subscript 'p' and the grid nodes by 'i'.

In the standard MPM, the algorithm involves three steps, namely particle to node mapping, update of the nodal momenta, update of the particle positions and velocities, and reset of the background grid. In the first step, the particles' momentum as well as internal and external forces are extrapolated onto the nodes of the background mesh. In the second step Newton's law is applied to these nodes. In the third step, the velocity and position of particles are updated from the grid nodal acceleration and velocity respectively. Finally, the background grid is reset to its initial state, and the cycle is repeated. In this method, all the interpolations and derivations are made in the current, thus deformed state. This causes particles that were in the initial state, located at the integration points, to move away from them and quadrature errors to increase. Under large deformations, particles can also cross cells, which generates interpolation errors and instabilities [26]. Under even more severe deformations, particles that were located in the same cell in the initial configuration can eventually be separated by at least an empty cell, causing a loss of connection between them, leading to non-physical numerical fracture [28,41].

In our proposed TLMPM, the algorithm involves the same steps as the standard MPM, as shown in Fig. 2, however, all interpolations and derivations are performed in the undeformed reference configuration, taken as the initial state. Therefore, no matter what the deformation, the reference configuration being always the same, there is no cell crossing instability, and no numerical fracture. As far as quadrature is concerned, if the solid boundaries are aligned with the background grid, there will be no quadrature error. For solids with curved boundaries, there is certainly some quadrature error close to the boundaries. More specifically, the evolution of the mass, velocities, stress and deformation gradients during these steps are described below. Precisely, we follow the modified update stress last (MUSL) presented in [11] and a blended PIC/FLIP particle velocity update taken from [14].

2.2.1. Particle to node mapping

This is the initial step of the TLMPM algorithm. The mass m_i, velocities $\boldsymbol{v}_i^{t^n}$, internal forces $\boldsymbol{f}_i^{ext,t^n}$ and external forces $\boldsymbol{f}_i^{int,t^n}$, at time step t^n are computed at the grid nodes from particle quantities:

$$m_i = \sum_p \Phi_i(\boldsymbol{X}_p) m_p \qquad \text{mass} \tag{5}$$

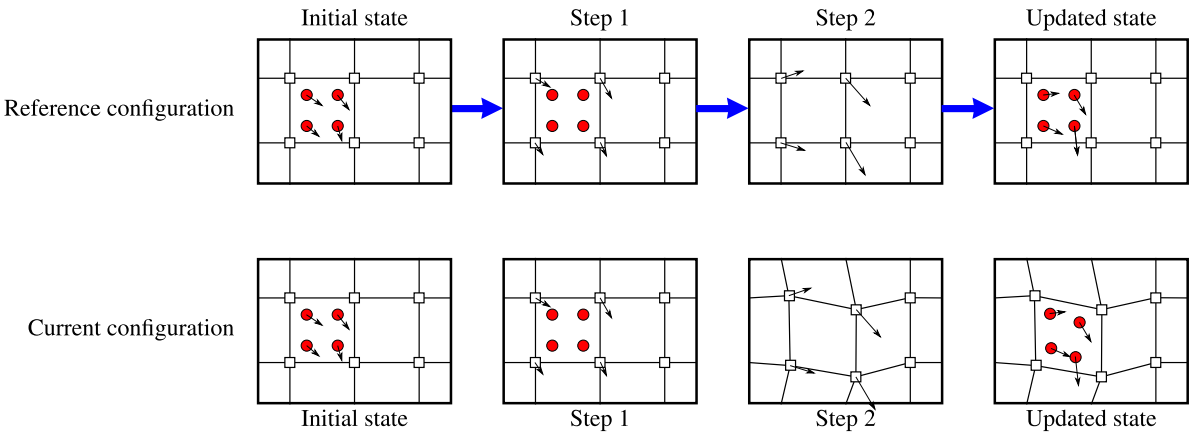


Fig. 2. Flow diagram of the Total-Lagrangian Material Point Method.

$$\boldsymbol{v}_i^{t^n} = \frac{1}{m_i}\sum_p \Phi_i(\boldsymbol{X}_p) m_p \boldsymbol{v}_p^{t^n} \qquad \text{momentum} \tag{6}$$

$$\boldsymbol{f}_i^{ext,t^n} = \sum_p \Phi_i(\boldsymbol{X}_p) m_p \boldsymbol{b}_p^{t^n} + m_I \boldsymbol{b}_i^{t^n} \qquad \text{external forces} \tag{7}$$

$$\boldsymbol{f}_i^{int,t^n} = \sum_p -V_p^0 \boldsymbol{P}_p^{t^n} \nabla_0 \Phi_i(\boldsymbol{X}_p) \qquad \text{internal forces} \tag{8}$$

where $\Phi_i(\boldsymbol{X}_p)$ are the shape function of node i evaluated at particle p, of which concrete forms will be described in Section 2.4, $\boldsymbol{b}_p^{t^n}$ are the external body forces applied at the centre of the particle p, $\boldsymbol{b}_i^{t^n}$ are the external body forces applied at the node i, V_p its volume, and $\boldsymbol{P}_p^{t^n}$ the first Piola–Kirchhoff stress tensor; $\nabla_0 \Phi_i(\boldsymbol{X}_p)$ denotes the gradient with respect to $\boldsymbol{X}$ of the shape functions. Note that, for simplicity, external forces due to tractions are omitted.

Remark 2.1. Note that contrary to the standard MPM, here the shape functions are expressed in terms of the particle position in the reference configuration, namely $\boldsymbol{X}_p$. Therefore, they are invariant in time.

Remark 2.2. Note also that as the particles' mass m_p are invariant due to mass conservation, and that the shape functions are also invariant, so are the node mass m_i. Therefore Eq. (5) needs not be updated at every time step.

2.2.2. Update nodal momenta and fix Dirichlet nodes

At this stage, the velocities and forces are known for time step t^n on the grid. The grid nodes' velocity is then determined using an explicit forward-Euler scheme:

$$\tilde{\boldsymbol{v}}_i^{t^{n+1}} = \boldsymbol{v}_i^{t^n} + \frac{\Delta t^{t^n}}{m_i}\left(\boldsymbol{f}_i^{int,t^n} + \boldsymbol{f}_i^{ext,t^n}\right) \qquad \text{temporary momentum} \tag{9}$$

$$\tilde{\boldsymbol{v}}_i^{t^{n+1}} = \boldsymbol{0} \qquad \text{for fixed nodes i} \tag{10}$$

where Δt^{t^n} is the time step at time t^n.

2.2.3. Update particle positions and velocities

The nodal velocities for time step t^{n+1} being known, they are temporarily interpolated back onto the particles:

$$\tilde{\boldsymbol{v}}_p^{t^{n+1}} = \sum_i \Phi_i(\boldsymbol{X}_p) \tilde{\boldsymbol{v}}_i^{t^{n+1}} \qquad \text{temporary velocity} \tag{11}$$

$$\tilde{\boldsymbol{a}}_p^{t^{n+1}} = \sum_i \Phi_i(\boldsymbol{X}_p)\left(\tilde{\boldsymbol{v}}_i^{t^{n+1}} - \boldsymbol{v}_i^{t^n}\right)/\Delta t^{t^n} \qquad \text{temporary acceleration} \tag{12}$$

$$\boldsymbol{x}_p^{t^{n+1}} = \boldsymbol{x}_p^{t^n} + \Delta t^{t^n} \tilde{\boldsymbol{v}}_p^{t^{n+1}} \qquad \text{position} \tag{13}$$

The variables decorated by a tilde ˜ are temporary.

The final particle velocity can be determined following two different methods: the pure particle-in-cell (PIC) method, or the fluid implicit particle (FLIP) method. Using either method, the update of the particle velocities is performed as follows:

$$\text{PIC:} \quad \boldsymbol{v}_p^{t^{n+1}} = \tilde{\boldsymbol{v}}_p^{t^{n+1}} \tag{14}$$

$$\text{FLIP:} \quad \boldsymbol{v}_p^{t^{n+1}} = \boldsymbol{v}_p^{t^n} + \Delta t \tilde{\boldsymbol{a}}_p^{t^{n+1}} \tag{15}$$

Originally, the standard MPM uses FLIP [11] which significantly reduces the dissipation at the expense of stability [22]. On the other hand, PIC is stable but highly dissipative. Alternatively, a linear combination of both methods can be used [14]. Thus, the particle velocities are updated as:

$$\boldsymbol{v}_p^{t^{n+1}} = (1-\beta)\tilde{\boldsymbol{v}}_p^{t^{n+1}} + \beta(\boldsymbol{v}_p^{t^n} + \Delta t \tilde{\boldsymbol{a}}_p^{t^{n+1}}) \tag{16}$$

where β is a constant taking values from 0 (pure PIC) to 1 (pure FLIP). Typically, $\beta = 0.99$ is used [14], but this value is empirical. In the provided simulations in Section 3, $\beta = 0$ (pure PIC) and $\beta = 0.99$ will both be used and compared.

2.2.4. Deformation gradient

In this contribution, we adopt the modified-update-stress-last (MUSL) approach which consists in updating the stress at the end of the time step. This approach, compared to the original update-stress-first scheme developed by [42], which updates the stress at the beginning of the time step, gives a large gain in numerical stability and accuracy [43].

First, the updated particle velocities are re-approximated at the nodes:

$$\boldsymbol{v}_i^{t^{n+1}} = \frac{1}{m_i} \sum_p \Phi_i(\boldsymbol{X}_p) m_p \boldsymbol{v}_p^{t^{n+1}} \tag{17}$$

Then, the Lagrangian velocity gradient ($\dot{\boldsymbol{F}} = \frac{d\boldsymbol{v}}{d\boldsymbol{X}}$) is determined:

$$\dot{\boldsymbol{F}}_p^{t^{n+1}} = \sum_i \boldsymbol{v}_i^{t^{n+1}} \otimes \nabla_0 \Phi_i(\boldsymbol{X}_p) \tag{18}$$

Utilizing a forward-Euler method, one can write:

$$\frac{\boldsymbol{F}_p^{t^{n+1}} - \boldsymbol{F}_p^{t^n}}{\Delta t} = \dot{\boldsymbol{F}}_p^{t^{n+1}}, \quad \Rightarrow \boldsymbol{F}_p^{t^{n+1}} = \boldsymbol{F}_p^{t^n} + \Delta t \dot{\boldsymbol{F}}_p^{t^{n+1}} \tag{19}$$

which allows to calculate the density:

$$\rho_p^{t^{n+1}} = \rho_P^0 / \det \boldsymbol{F}_p^{t^{n+1}} \tag{20}$$

The Lagrangian velocity gradient ($\dot{\boldsymbol{F}}$), the deformation gradient ($\boldsymbol{F}$) and the current density ρ are necessary quantities for the determination of the stress.

Remark 2.3. Unlike in conventional MPM, the deformation matrix $\boldsymbol{F}$ is here determined using the Lagrangian velocity gradient $\dot{\boldsymbol{F}}$, and not the velocity gradient $\boldsymbol{L}$. However, they are both linked by:

$$\boldsymbol{L} = \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{X}} \cdot \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{x}} = \dot{\boldsymbol{F}} \boldsymbol{F}^{-1} \tag{21}$$

and substituting Eq. (21) into Eq. (19), one gets the equation commonly used to determine the deformation matrix at step t^{n+1}: $\boldsymbol{F}_p^{t^{n+1}} = (\boldsymbol{I} + \boldsymbol{L}_p^{t^{n+1}}) \boldsymbol{F}_p^{t^n}$.

Remark 2.4. Alternatively, and similar to TL FEM, the deformation gradient can be directly calculated from the grid's current position. In order to do this, an addition step needs to be performed. After the particle velocities are re-approximated at the nodes (Eq. (17)), their position is updated:

$$\boldsymbol{x}_i^{t^{n+1}} = \boldsymbol{x}_i^{t^n} + \Delta t \boldsymbol{v}_i^{t^{n+1}} \tag{22}$$

so that one can compute the nodal displacement $\boldsymbol{x}_i^{t^{n+1}} - \boldsymbol{X}_i$. Then, the deformation gradient can be computed as:

$$\boldsymbol{F}_p^{t^{n+1}} = \boldsymbol{I} + \sum_p \nabla_0 \Phi_i(\boldsymbol{X}_p)(\boldsymbol{x}_i^{t^{n+1}} - \boldsymbol{X}_i) \tag{23}$$

This formulation was compared to Eq. (19), and for the simulations presented in this contribution, no significant difference was observed.

2.2.5. Constitutive model and stress update

The constitutive model is independent of the numerical discretization of the partial differential equations and is not therefore an essential part of the Total-Lagrangian Material Point Method methodology. That is why, in most of the examples presented in this work, a simple compressible Neo-Hookean model is used. An elasto-plastic model is also adopted to simulate a ductile material all the way to rupture.

Neo-Hookean. Following this model, the first Piola–Kirchhoff stress tensor is given by:

$$\boldsymbol{P} = \mu(\boldsymbol{F} - \boldsymbol{F}^{-T}) + \lambda\,(\ln J)\,\boldsymbol{F}^{-T} \tag{24}$$

where μ and λ are the Lamé constants, J is the determinant of $\boldsymbol{F}$. We refer the readers to Bonet and Wood [44] for derivation details of this model.

Elasto-plastic materials. To model ductile materials, a hypoelastic-damage-plastic material model is adopted. It uses an equation of state (EOS) for the determination of the hydrostatic pressure, and an empirical model for the deviatoric stress tensor, respectively. Fracture is modelled using the classic continuum damage mechanics approach which scales linearly the stress tensor with a damage variable [45].

For these materials, the first Piola–Kirchhoff stress tensor is determined using the Cauchy stress:

$$\boldsymbol{P} = J\boldsymbol{\sigma}\,\boldsymbol{F}^{-T} \tag{25}$$

The symmetrical tensor $\boldsymbol{\sigma}$ is expressed as the sum of its isotropic part, *i.e.* the hydrostatic pressure (σ_m), and the traceless symmetric deviatoric stress $\boldsymbol{\sigma^d}$:

$$\boldsymbol{\sigma} = -\sigma_m \boldsymbol{I} + \boldsymbol{\sigma^d} \tag{26}$$

The hydrostatic pressure, on the one hand, is estimated using a modified Mie–Grüneisen EOS which was modified to account for damage (D) [46]:

$$\begin{cases} \sigma_m = \dfrac{\rho_0(1-D){c_0}^2(\eta-1)\left[\eta - \dfrac{\Gamma_0}{2}(\eta-1)\right]}{[\eta - S_\alpha(\eta-1)]^2}; \quad \eta = \dfrac{\rho(1-D)}{\rho_0} \text{ if } \sigma_m > 0 \\ \sigma_m = \dfrac{\rho_0 {c_0}^2(\eta-1)[\eta - \frac{\Gamma_0}{2}(\eta-1)]}{[\eta - S_\alpha(\eta-1)]^2}; \quad \eta = \dfrac{\rho}{\rho_0} \text{ otherwise} \end{cases} \tag{27}$$

where c_0 is the bulk speed of sound, Γ_0 the Grüneisen Gamma in the reference state. The deviatoric stress, on the other hand is determined using the Johnson–Cook model which is scaled with damage [47]. Thus, the relationship for the equivalent von Mises flow stress is:

$$\sigma_f(\varepsilon_p, \dot{\varepsilon}_p) = \left[A + B\left(\varepsilon_p\right)^n\right]\left[1 + C \ln \dot{\varepsilon}_p^*\right](1-D) \tag{28}$$

where ε_p is the equivalent plastic strain, ε_p^* is the normalized plastic strain rate, A the yield stress, B and n the strain hardening parameters, and C the strain rate parameters. The normalized plastic strain rate is given by:

$$\dot{\varepsilon}_p^* = \dot{\varepsilon}_p / \dot{\varepsilon}_0 \tag{29}$$

where $\dot{\varepsilon}_p$ and $\dot{\varepsilon}_0$ are the plastic strain rate, and the user-defined reference plastic strain rate.

The determination of the equivalent plastic strain is performed jointly with that of the deviatoric stress tensor according to the algorithm developed by Leroch et al. [48] and presented in Algorithm 1. In this case, the deviatoric stress is calculated incrementally from the strain rate $\boldsymbol{D} = \frac{1}{2}(\boldsymbol{L} + \boldsymbol{L}^T)$, where $\boldsymbol{L} = \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}} = \dot{\boldsymbol{F}}\boldsymbol{F}^{-1}$ is the velocity gradient. Due to non-objectivity (described by Bonet and Wood [44]), the stress rate is not invariant under rigid body rotation, making its integration non-trivial. This problem is overcome by eliminating the rigid body motion from

the strain rate, and thus from the stress rate. This is done by first performing a polar decomposing of $\boldsymbol{F}$ in rotation $\boldsymbol{R}$ and stretch $\boldsymbol{U}$ parts, *i.e.* $\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U}$, using the singular value decomposition. Then, the rigid body rotations are subtracted from the strain rate tensor $\boldsymbol{D}$ to obtain the un-rotated strain rate tensor $\boldsymbol{d} = \boldsymbol{R}^T\boldsymbol{D}\boldsymbol{R}$. Once the un-rotated stress rate is integrated into the un-rotated stress $\boldsymbol{\sigma}'$ using the Algorithm 1, the later is rotated back to the current configuration: $\boldsymbol{\sigma} = \boldsymbol{R}\boldsymbol{\sigma}'\boldsymbol{R}^T$.

Algorithm 1 Plasticity algorithm proposed by Leroch et al.

Inputs: ε_n^p (equivalent plastic strain), $\boldsymbol{\sigma}'^{\,d}_n$ (un-rotated deviatoric stress), $\boldsymbol{d}^d$, damage D_n
Outputs: ε_{n+1}^p (equivalent plastic strain), $\boldsymbol{\sigma}'^{\,d}_{n+1}$
Compute $G' = (1 - D^t)G$
$\boldsymbol{\sigma}'^{\,d}_{\text{trial}} = \boldsymbol{\sigma}'^{\,d}_n + 2G'\Delta t\boldsymbol{d}^d$ ▷ purely elastic stress deviator update
$\sigma'^{\,\text{eq}}_{\text{trial}} = \sqrt{\frac{3}{2}\boldsymbol{\sigma}'^{\,d}_{\text{trial}} : \boldsymbol{\sigma}'^{\,d}_{\text{trial}}}$ ▷ equivalent von Mises trial stress
$\sigma_f = \left[A + B\left(\varepsilon_n^p\right)^n\right]\left[1 + C\ln\dot{\varepsilon}_p^*\right](1 - D_n)$ ▷ JC flow stress
if $\sigma'^{\,\text{eq}}_{\text{trial}} < \sigma_f$ **then** ▷ yielding did not occur, purely elastic step
  $\boldsymbol{\sigma}'^{\,d}_{n+1} = \boldsymbol{\sigma}'^{\,d}_{\text{trial}}$ ▷ keep trial deviatoric stress
else ▷ yielding has occurred
  $\Delta\varepsilon_p = (\sigma'^{\,\text{eq}}_{\text{trial}} - \sigma_f)/(3G')$ ▷ compute the equivalent plastic strain increment
  $\varepsilon_{n+1}^p = \varepsilon_n^p + \Delta\varepsilon_p$ ▷ update the undamaged matrix plastic strain
  $\boldsymbol{\sigma}'^{\,d}_{n+1} = \frac{\sigma_f}{\sigma'^{\,\text{eq}}_{\text{trial}}}\boldsymbol{\sigma}'^{\,d}_{\text{trial}}$ ▷ scale deviatoric stress back to yield surface
end if

The amount of damage (D) in each particle is determined using the Johnson–Cook damage model, widely used for engineering applications [47]. It is a strain rate dependent phenomenological model based on the local accumulation of plastic strain. According to this model, damage initiates when the accumulated equivalent plastic strain reaches the equivalent strain at failure ε_f [49] (see Fig. 3):

$$D_{init} \coloneqq \sum \frac{\Delta\varepsilon_p}{\varepsilon_f} = 1 \tag{30}$$

where $\Delta\varepsilon_p$ is the equivalent plastic strain increment. The equivalent strain at failure given by Johnson–Cook's empirical equation:

$$\varepsilon_f = \left(D_1 + D_2\exp(D_3\sigma^*)\right)(1 + \dot{\varepsilon}_p^*)^{D_4} \tag{31}$$

and where $D_1, \ldots, D_4$ are four material constants, $\sigma^* = \sigma_m/\sigma_{eq}$ is the stress triaxiality, and $\sigma_{eq} = \sqrt{\frac{3}{2}\boldsymbol{\sigma}^d : \boldsymbol{\sigma}^d}$, the equivalent von Mises stress.

As this model only describes damage initiation, in order to have a complete model of the fracture phenomenon, a damage evolution model is required. Here, for the sake of simplicity, it was assumed that the damage variable is given by:

$$D = \begin{cases} 0 & \text{when } 0 \le D_{init} < 1 \\ 10\,(D_{init} - 1) & \text{when } D_{init} \ge 1 \end{cases} \tag{32}$$

Even if this assumption affects the details of the damage propagation, it does not change the fundamentals of the implementation on which we focus here.

2.3. Time step

The time integration scheme used here being explicit, the time increment Δt must satisfy

$$\Delta t \le \frac{h}{c} \tag{33}$$

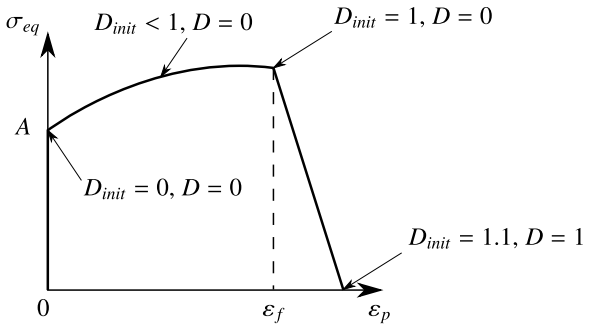


Fig. 3. Schematic of a typical equivalent stress–plastic strain curve showing the evolution of both the damage initiation variable D_{init} and the damage variable D. Three points are highlighted: the yield stress A, the point at which damage initiates $D_{init} = 1$, and the point of total failure $D = 1$.

where h is the background mesh cell size and

$$c = \sqrt{\frac{K + 4\mu/3}{\rho}} = \sqrt{J\frac{K + 4\mu/3}{\rho_0}} \tag{34}$$

where K and μ are the bulk and shear modulus, respectively. The final time step corresponds to the minimum time step for all particles. As shown in Eq. (34) by the presence of J, the time step is restricted by the amount of deformation which is a common feature of Lagrangian methods [32]. As c varies in time, we actually adopt adaptive time steps.

2.4. Shape functions (weighting functions)

In the MPM, mapping is done with the use of shape functions or weighting functions. In the original MPM, the shape functions used are linear and identical to the hat shape functions used in FEM. These functions are the simplest form of shape functions, however their use has been shown to generate errors when particles cross the background grid cells [50]. To mitigate these errors, Steffen and coworkers proposed to use cubic B-splines instead. They showed that for simple problems, their use improves the spatial convergence properties of the method [25,50]. These shape functions are effective at decreasing the so-called cell crossing instability, however, they are complex to use as each spans multiple cells. Moreover, distinction between nodes depending on their distance to the boundaries has to be made in order to ensure exact partition of unity.

In the TLMPM, cell crossing is not an issue as in the reference configuration particles do not move with respect to the background mesh. Therefore, the use of cubic splines can be only justified by a higher rate of convergence than that of the hat functions. But high order shape functions developed for FEM, such as Bernstein polynomials, can also be used. In this contribution, hat functions, cubic B-splines and Bernstein polynomials of degree 2 are used as shape functions in order to study their effect on the accuracy and convergence rate of the TLMPM. In the following parts of this section, these three shapes functions are detailed. Our contributions include the first use of Bernstein basis in MPM and explicit formula for the modified boundary cubic splines.

2.4.1. Hat functions

Hat functions, being linear, are the simplest type of shape functions. For a given grid node i, they are defined as:

$$\begin{cases} \Phi_i(\boldsymbol{X}_p) &= \left|1 - \frac{X_i - X_p}{h}\right| \times \left|1 - \frac{Y_i - Y_p}{h}\right| \text{ in 2D} \\ \Phi_i(\boldsymbol{X}_p) &= \left|1 - \frac{X_i - X_p}{h}\right| \times \left|1 - \frac{Y_i - Y_p}{h}\right| \times \left|1 - \frac{Z_i - Z_p}{h}\right| \text{ in 3D} \end{cases} \tag{35}$$

where h is the background grid cell size. An illustration of the hat shape functions is given in Fig. 4 for a series of three elements in 1D. Note that for simplicity, a grid of squares (2D) or cubes (3D) is used and thus the cell size is identical in all directions. We refer to the MPM that adopts these hat functions as the standard MPM in this text.

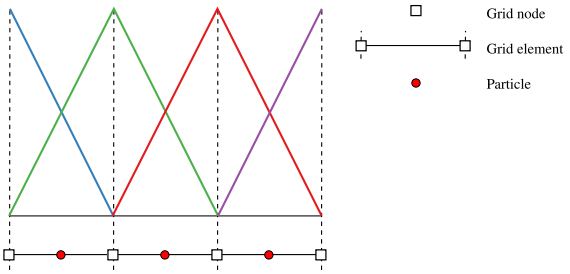


Fig. 4. Hat shape functions for a series of three elements in 1D.

In the following, when using hat shape functions, one-point quadrature is used with each particle located at the Gauss point located in the middle of the element (see Fig. 4).

2.4.2. Cubic B-splines

There are different ways to construct the B-splines basis functions, namely through a recurrence or a convolution concept. The latter was used by Steffen et al. [25,50] to relate them to GIMP weighting functions. However, herein, we use B-splines constructed using a recursive formula for their generality. These recursive B-splines, which have been extensively used in isogeometric analysis that unifies finite element analysis and CAD (computer aided design), see e.g. [37,51], are also used in [52]. Herein, we adopt the so-called modified boundary B-spline basis functions, used in the doctoral thesis of [39]. We provide explicit formula for all cubic B-splines where [39] only presented them for internal B-splines. Thus, they are more efficient than recursive B-splines.

Given a knot vector $\Xi^1 = \{\xi_1, \xi_2, \ldots, \xi_{n+k+1}\}$, which is defined as an ordered set of increasing parameter values, the associated set of B-spline basis functions $\{N_{j,k}\}_{j=1}^{n}$ are defined recursively by the Cox-de-Boor formula [53], starting with the zeroth order basis function ($k = 0$):

$$N_{j,0}(\xi) = \begin{cases} 1 & \text{if } \xi_j \leq \xi < \xi_{j+1} \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

and for a polynomial order $k \geq 1$

$$N_{j,k}(\xi) = \frac{\xi - \xi_j}{\xi_{j+k} - \xi_j} N_{j,k-1}(\xi) + \frac{\xi_{j+k+1} - \xi}{\xi_{j+k+1} - \xi_{j+1}} N_{j+1,k-1}(\xi) \tag{37}$$

in which fractions of the form $0/0$ are defined as zero.

High order B-spline basis functions are C^{k-1} not C^0 as high order Lagrange polynomial basis, the connectivity of elements is, therefore, different from standard finite elements. Elements are defined as non-zero knot spans. Note that the B-splines functions are not interpolatory except at the boundaries when open knots[2] are used. Open knots facilSTATE the imposition of Dirichlet boundary conditions.

Fig. 5 shows the one-dimensional cubic ($k = 3$) B-spline basis functions for a uniform knot vector $\Xi = \{0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5\}$. The knot vector is made of 5 knot spans ($n = 5$) and 8 control points ($n + k$). Thus, there are 8 basis functions. In order to use the knot spans as elements, the same way Steffen et al. did [25,50], one more shape function than knot spans is required (*i.e.* in this example, 6 shape functions). Therefore, there are two more basis functions than the number of shape functions required. The right number of shape function is obtained by combining the two basis functions (on each side) that do not peak at the junction between two elements as to obtain the one dimensional shape functions ($S_{i,\zeta}$, where ζ corresponds to any axis x, y or z) plotted in Fig. 6. By doing this, the partition of unity is respected and all elements have the same size. Similar to Steffen et al. [25,50], the two dimensional and three-dimensional shape functions are obtained as the product of the different one-dimensional

[2] Open knots are those where the first and last knots are repeated $p + 1$ times.

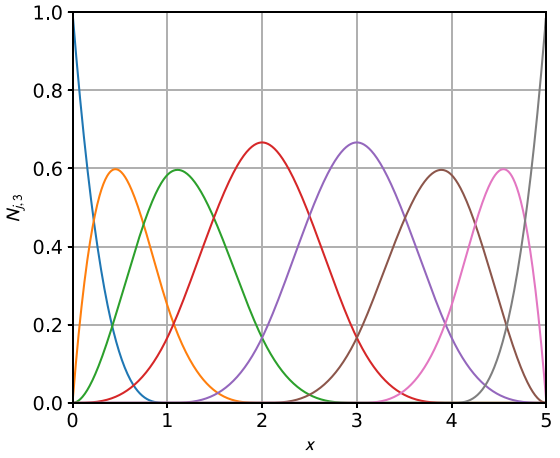


Fig. 5. One dimensional cubic ($k = 3$) B-spline basis functions on an open uniform knot $\Xi = \{0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5\}$. There are 8 nodes (control points in CAD terminology) and 5 elements (or knot spans in CAD). It can be seen from the figure, at any point there are 4 ($= k + 1$) non-zero basis functions. Therefore each element has 4 nodes. The first element's connectivity is $[1, 2, 3, 4]$ *i.e.* particles locate in this element contribute to nodes 1, 2, 3 and 4. The second element's connectivity is $[2, 3, 4, 5]$ and so on [37].

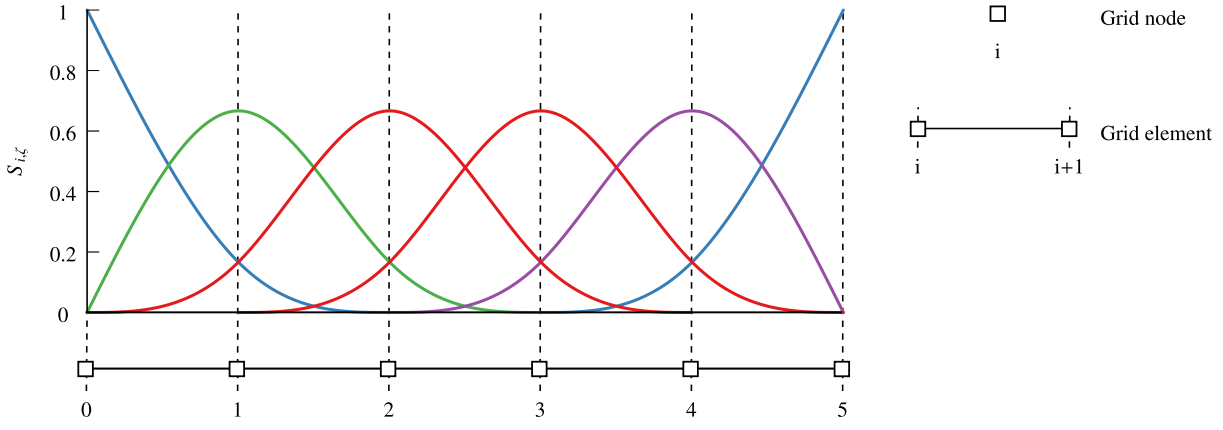


Fig. 6. Cubic B-spline shape functions for a series of five elements in 1D. Note that there are now just 6 basis functions.

shape functions as:

$$\begin{cases} \Phi_i(\boldsymbol{X}_p) & = S_{i,x}\left(\frac{X_p - X_i}{h}\right) \times S_{i,y}\left(\frac{Y_p - Y_i}{h}\right) \text{ in 2D} \\ \Phi_i(\boldsymbol{X}_p) & = S_{i,x}\left(\frac{X_p - X_i}{h}\right) \times S_{i,y}\left(\frac{Y_p - Y_i}{h}\right) \times S_{i,z}\left(\frac{Z_p - Z_i}{h}\right) \text{ in 3D} \end{cases} \tag{38}$$

Because of the presence of boundaries, there are four different types of shape functions $S_{i,\zeta}$ which differ by the position of node i with respect to the boundaries. They are represented in Fig. 6 by different colours and their expressions are:

- Shape functions of **type 1** (blue in Fig. 6): the particle i is located at the boundary, *i.e.* $\zeta_i = \zeta_B$, and have the following form:

$$S^1_{i,\zeta}(r) = \begin{cases} \frac{1}{6}r^3 + r^2 + 2r + \frac{4}{3}, & -2 \le r \le -1 \\ -\frac{1}{6}r^3 + r + 1, & -1 \le r \le 0 \\ \frac{1}{6}r^3 - r + 1, & 0 \le r \le 1 \\ -\frac{1}{6}r^3 + r^2 - 2r + \frac{4}{3}, & 1 \le r \le 2, \end{cases} \tag{39}$$

  where $r = (\zeta_p - \zeta_i)/h$.
- Shape functions of **type 2**: the particle i is located on the right side of the closest boundary one cell away from it, *i.e.* $\zeta_i = \zeta_B + h$, and have the following form:

$$S^2_{i,\zeta}(r) = \begin{cases} -\frac{1}{3}r^3 - r^2 + \frac{2}{3}, & -1 \le r \le 0 \\ \frac{1}{2}r^3 - r^2 + \frac{2}{3}, & 0 \le r \le 1 \\ -\frac{1}{6}r^3 + r^2 - 2r + \frac{4}{3}, & 1 \le r \le 2 \end{cases} \tag{40}$$

- Shape functions of **type 3**: the particle i is located at least two cells away from any boundary, *i.e.* $\zeta_i \ge \zeta_B + 2h$, and have the following form:

$$S^3_{i,\zeta}(r) = \begin{cases} \frac{1}{6}r^3 + r^2 + 2r + \frac{4}{3}, & -2 \le r \le -1 \\ -\frac{1}{2}r^3 - r^2 + \frac{2}{3}, & -1 \le r \le 0 \\ \frac{1}{2}r^3 - r^2 + \frac{2}{3}, & 0 \le r \le 1 \\ -\frac{1}{6}r^3 + r^2 - 2r + \frac{4}{3}, & 1 \le r \le 2 \end{cases} \tag{41}$$

- Shape functions of **type 4**: the particle i is located on the left side of the closest boundary, one cell away from it, *i.e.* $\zeta_i = \zeta_B - h$, and have the following form:

$$S^4_{i,\zeta}(r) = \begin{cases} \frac{1}{6}r^3 + r^2 + 2r + \frac{4}{3}, & -2 \le r \le -1 \\ -\frac{1}{2}r^3 - r^2 + \frac{2}{3}, & -1 \le r \le 0 \\ \frac{1}{3}r^3 - r^2 + \frac{2}{3}, & 0 \le r \le 1 \end{cases} \tag{42}$$

These four types of one-dimensional shape functions $S_{i,\zeta}$ translate into $4^2 = 16$ two dimensional shape functions Φ_i, and into $4^3 = 64$ three dimensional shape functions Φ_i: each node i having a different type along the respective axes x, y, and z. Unless otherwise stated, in the following, when cubic B-splines are used, each background cell is populated by 2, 4 and 8 material points in 1D, 2D and 3D, respectively. These particles will be located at positions defined by $\xi_1 = 0.2113$ and $\xi_2 = 0.7887$ which correspond to the Gauss quadrature points.

Remark 2.5. Note that we have redefined the B-splines functions to have exactly $n+1$ nodes (and basis functions) for a mesh of n cells (1D). This is just a matter of implementation as the B-splines grid is now exactly the same as the grid that uses hat functions. Other implementation of B-splines in MPM usually follow the isogeometric analysis one, see e.g. [52].

2.4.3. Bernstein polynomials

Bernstein polynomials form a basis for the Bézier elements used in isogeometric analysis [54]. These polynomials are used in CAD to construct the so-called Bézier curves/surfaces [53]. The univariate Bernstein basis functions of order k are defined over the biunit interval [0, 1] as:

$$B_{i,k}(\xi) = \binom{k}{i} \xi^i \, (1-\xi)^{k-i} \tag{43}$$

where the binomial coefficient $\binom{k}{i} = \frac{k!}{i!(k-i)!}$ for $1 \le i \le k+1$. Bernstein polynomials of degree 2, used in this paper, are plotted in Fig. 7. These polynomials form a partition of unity: $\sum_{i=1}^{k} B_{i,k}(\xi) = 1$.

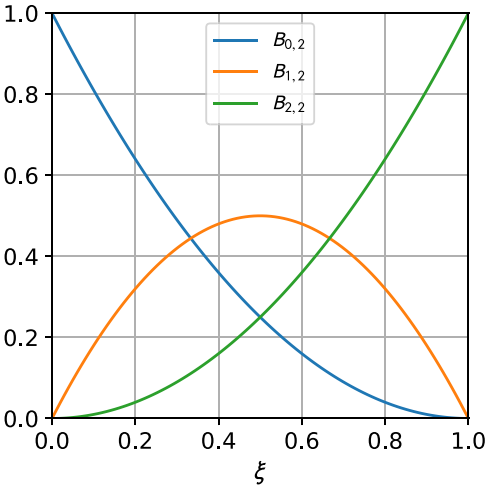


Fig. 7. Bernstein basis polynomials of degree 2.

Each cell of the background mesh is made of 3 (in 1D), 9 (in 2D), or 27 (in 3D) nodes. As some of these nodes are common to different cells, we express the shape functions as a function of the normalized distance between a particle p and a node i, *i.e.* $(\boldsymbol{X}_i - \boldsymbol{X}_p)/h$:

$$\Phi_i(\boldsymbol{X}_p) = S_{i,x}\left(\frac{X_i - X_p}{h}\right) \times S_{i,y}\left(\frac{Y_i - Y_p}{h}\right) \times S_{i,z}\left(\frac{Z_i - Z_p}{h}\right) \tag{44}$$

where $S_{i,\zeta}$ are the shape functions along the axis ζ (*i.e.* x, y or z). The shape function depends on the position of the nodes in a cell: if it is located on an edge or the cell centre along the axis i, they take two different forms: if the node i is located on an edge of a mesh element along the axis ζ:

$$S_{i,\zeta}(r) = B_{0,2}(|r|) = \begin{cases} (1-|r|)^2 & \text{if } -1 \leq r \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{45}$$

otherwise, *i.e.* if the node i is located at the centre (or inside) an element along the axis ζ:

$$S_{i,\zeta}(r) = B_{1,2}\left(|r| + \frac{1}{2}\right) = \begin{cases} \frac{1}{2} - 2r^2 & \text{if } -1/2 \leq r \leq 1/2 \\ 0 & \text{otherwise} \end{cases} \tag{46}$$

Therefore, in 1D and for a solid discretized using three elements of length h, the different shape functions are as shown in Fig. 8.

In this work, unless otherwise stated, when using Bernstein shape functions, each background cell is populated by 3, 9 and 27 material points in 1D, 2D and 3D, respectively. The particles will be located at positions defined by $\xi_{1,2 \text{ and } 3} = 0.1127$, 0.5, and 0.8873. These positions correspond to the Gauss quadrature points.

3. Simulations

A number of examples which demonstrate the capabilities of the Total-Lagrangian Material Point Method for solid mechanics problems are presented in this section. The selected cases are:

1. Vibration of a vertical bar subjected to gravity (Section 3.1)
2. Convergence of TLMPM using the method of manufactured solutions (Section 3.2) including:
   (a) tension and compression (Section 3.2.1),
   (b) rotation and shear (Section 3.2.2).
3. The Taylor bar impact test (Section 3.3),
4. Tensile test of a smooth elasto-plastic specimen (Section 3.4).

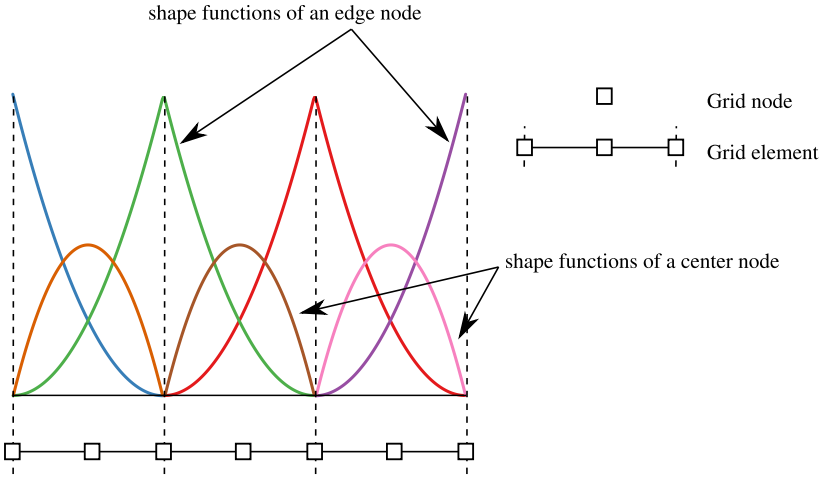


Fig. 8. Bernstein quadratic shape functions for a series of three elements in 1D. Note that Bernstein functions are still C^0.

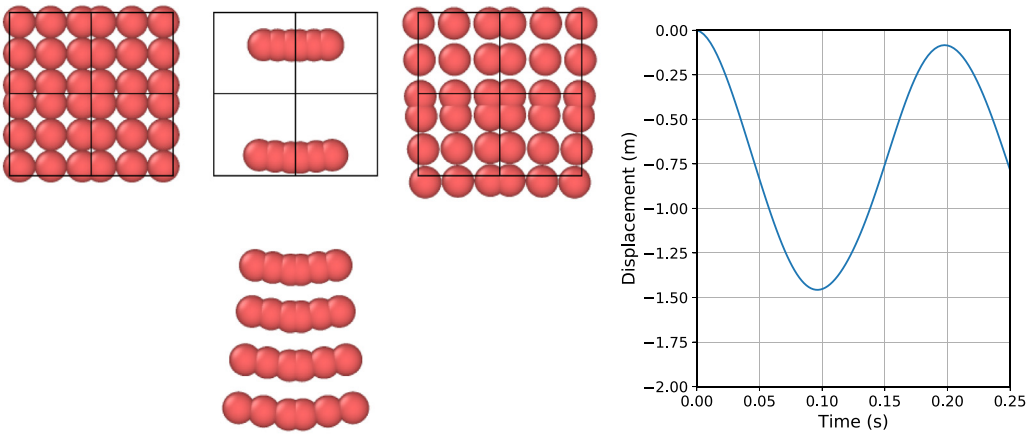
The first example is used to demonstrate the ability of the TLMPM to simulate large deformations. For this example, the code's predictions are verified against an updated Lagrangian FEM. Then, the convergence is analysed using the method of manufactured solutions in both tension and compression, and in rotation and shear with the "generalized vortex problem". Next, the predictive capabilities are validated experimentally using the Taylor impact problem. Finally, preliminary results for the simulation of failure are shown using the simulation of an elasto-plastic tensile specimen stretched up to failure The later example uses a physically based model of fracture and demonstrates the ability of the method to simulate the type of fracture important in problems of wear, impact, fatigue, etc.

3.1. Vertical bar in gravity field

To test the stability of different variants of the MPM, Sadeghirad et al. [28] proposed to use the problem of the vibration of a vertical bar under its own weight with very large deformations. They showed that the simulations of this test using the standard MPM exhibits both instabilities and numerical fracture, while using CPDI they were stable and did not fracture. The same test is applied here to show that the TLMPM is stable and does not exhibit numerical fracture, contrary to the original MPM.

First, to show that TLMPM is stable and does not suffer from numerical fracture, we performed the same test as Sadeghirad and coworkers [28] in 2D with a coarse grid. The solid is a 1 m $\times$ 1 m square modelled using the compressible Neo-Hookean model introduced in Section 2.2.5 with a Young's modulus, Poisson's ratio, and initial density of $E = 1 \times 10^6$ Pa, $\nu = 0.3$, and $\rho_0 = 1050$ kg/m^3, respectively. The upper edge of the bar has roller boundary conditions while the others are traction free. The simulation time is $T = 0.25$ s. Very large deformations are obtained using a gravity field of magnitude $g = 1000$ m/s^2 applied suddenly at $t = 0$ s. For this simulation only, we use linear shapes functions with 9 particle per background grid cell which are of a coarse size: 0.5. Moreover $\beta = 0.99$. The results of this simulation are presented in Fig. 9. The vertical displacement profile of the bottom left particle (see Fig. 9(d)) does not show any instabilities and is qualitatively similar to what was obtained using CPDI by Sadeghirad et al. [28]. From the deformation snapshots, one can see that even though the particles leave the domain made by the grid (see Fig. 9(b)), the bar does bounce back up (see Fig. 9(c)) as no fracture occurs. Note that in TLMPM the background grid has to cover only the whole solid in the reference configuration, while in all the other variants of MPM, CPDI included, the background grid needs to big large enough to cover the whole deformed configuration.

In order to study the qualitative results of the TLMPM simulations the same test was also performed in 3D, using the different shape functions presented in Section 2.4 and with different values of β. The bar is now a cube whose edges have a length of 1 m. The upper face of the bar has roller boundary conditions while the others are traction free as shown in Fig. 10. The material used is the same as in the 2D simulation. The simulation time is also $T = 0.25$ s and again, a gravity field of magnitude $g = 1000$ m/s^2 is applied suddenly at $t = 0$ s. However,



(a) $t = 0$ s (b) $t = 0.95$ s (c) $t = 0.20$ s (d) Vertical displacement

Fig. 9. 2D vertical bar simulations with coarse background grid'. (a), (b) and (c) are snapshots of the bar featuring the background grid. (c) is the vertical displacement profile of the bottom left corner. Note that the grid remains undeformed and that having particles leaving the grid does not mean that fracture has occurred. The apparent size of the particles is independent of their volume.

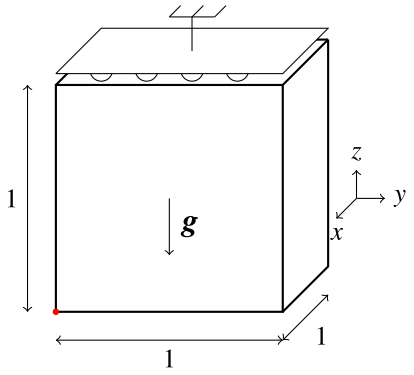
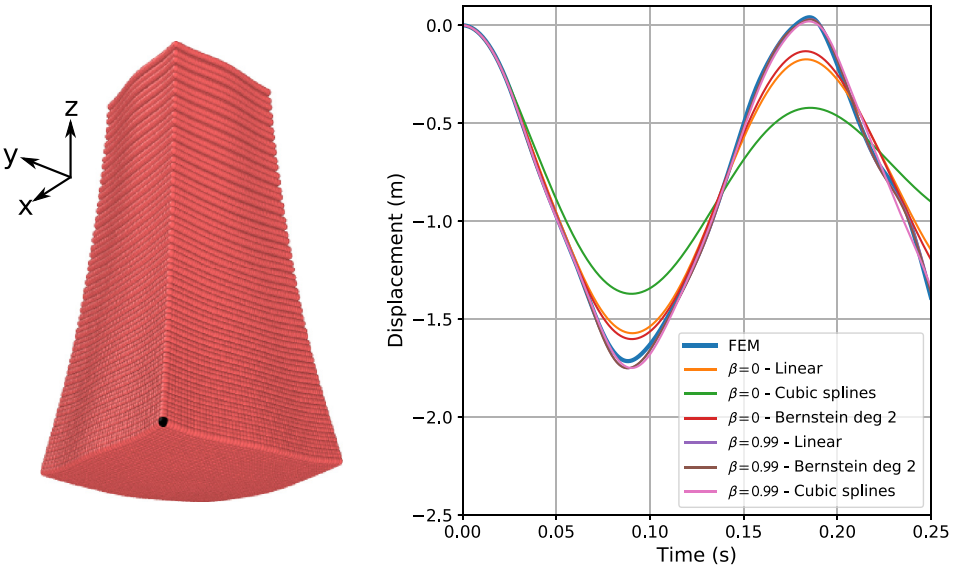


Fig. 10. Geometry of the vertical bar in gravity field example.

we adopt a regular background grid of cell size $1/30$ m and each cell is occupied by either 1, 8 or 27 particles depending if the linear, cubic B-splines or Bernstein shape functions are used. They are positioned within the cells according to what is stated in Section 2.4. Dirichlet boundary conditions are enforced by setting the vertical velocity of all nodes of the background mesh that coincide with the upper face to zero.

Fig. 11 shows the evolution of the displacement of one of the bottom corners (represented in red in Fig. 10) with time as obtained using either $\beta = 0$ (pure PIC), or $\beta = 0.99$. Also plotted is the displacement evolution of the same corner obtained using FEM.[3] It can be first seen that, in all simulated cases and similarly to the 2D case, the TLMPM is stable in 3D and does not exhibit numerical fracture. Second, as expected, when using $\beta = 0$, a significant amount of energy is dissipated, resulting in an artificial damping of the particles displacements as attested by the decrease of the amplitude of the oscillations with time. Finally, for $\beta = 0.99$, good agreement is obtained between the TLMPM and FEM, no matter what shape functions are used. Nonetheless, better agreement is seen when using linear and Bernstein polynomial shape functions. It is surprising that the results obtained using cubic B-splines shape functions are further away from that of FEM compared to those obtained using linear shape functions. Indeed, because of their higher order, one would expect the B-splines to generate less error than the hat functions, for the same cell size.

[3] FEM solutions are obtained using an in-house code adopting the updated Lagrangian FEM described in [40].



(a) (b)

Fig. 11. Vertical bar in gravity field example: (a) Deformed shape at $t = 0.19$ s using $\beta = 0.99$ with cubic B-splines and (b) Vertical displacement of the corner represented by the black particle (in (a)) obtained with FEM, TLMPM using different mix of PIC and FLIP velocity update schemes and either linear, cubic splines and quadratic Bernstein functions. Using $\beta = 0.99$, the results using linear or Bernstein shape functions are nearly identical.

In the conventional MPM, the shapes functions are function of the particles positions in the current configuration ($\boldsymbol{x}_p$) and therefore need to be evaluated at every time step. Moreover, in CPDI, the shapes functions are also function of the position of the edges of the particles' domain which have thus to be tracked. In contrast, in TLMPM shape functions are only function of the particles positions in the reference configuration ($\boldsymbol{X}_p$). Therefore, they only need to be evaluated once. This generates a substantial computational speed gain compared to conventional MPM, and an even greater compared to CPDI, while reducing implementation complexity.

3.2. Convergence using the method of manufactured solutions

Code verification has received attention in recent decades as expensive projects rely more and more heavily on numerical simulations using non linear codes. Assessing the convergence of such codes is a difficult task which led to the development of the method of manufactured solutions (MMS) as it has the advantage of being able to test codes with nonlinearities for which exact solutions will never be known [55]. This method is now an accepted standard for code verification.

In the MMS, the solution (here the displacement field) is assumed a priori, *i.e.* is manufactured. Given the assumed prescribed displacements, the constitutive model can be evaluated to determine the corresponding stress field. Therefore, the divergence of both the stress and acceleration can be evaluated, and the required body forces to achieve these solutions are analytically determined from the equations of motion. Note that boundary and initial conditions are also determined from the manufactured solutions. This allows verification of nonlinear codes by running them with the computed external force and demonstrating that the assumed solution is recovered [42].

A suite of code verification tests for solid mechanics is presented by Kamojjala et al. in [56] for rate-independent constitutive models. A test more specific to MPM, highlighting its difficulties to simulate problems involving rotation and shear was presented by Brannon et al. [57]. Here we use both the axis-aligned displacement in a unit square presented by Kamojjala et al. (but extended to 3D), as well as the generalized vortex problem presented by Brannon et al. to evaluate the convergence of TLMPM.

3.2.1. Axis-aligned displacement in a unit cube

This MMS test involves only tension and compression as the displacement field is assumed to be:

$$\boldsymbol{u}(\boldsymbol{X},t) = G\sin(\pi\boldsymbol{X})\sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \boldsymbol{\phi}\right) \tag{47}$$

where G is the maximum amplitude of displacement, E denotes Young's modulus, and $\boldsymbol{\phi} = [0, \pi, \pi]$, an arbitrary phase angle. $\boldsymbol{X}$ denotes the material coordinates in the reference configuration. The solid is a unit cube occupying the domain delimited by $0 \leq X_1 \leq 1$, $0 \leq X_2 \leq 1$, and $0 \leq X_3 \leq 1$. One period of oscillation is considered, *i.e.* the time domain is $0 \leq t \leq T$, with $T = \frac{2\pi}{\sqrt{E/\rho_0}\pi}$. From the prescribed displacements, and using the compressible Neo-Hookean model introduced in Section 2.2.5, the necessary body forces applied on the nodes are obtained as (see Appendix A.1 for derivation details):

$$\boldsymbol{b}(\boldsymbol{X},t) = \begin{bmatrix} \frac{\pi^2 u_1}{\rho_0}\left(\frac{\lambda}{F_{11}^2}\left(1-\ln(F_{11}F_{22}F_{33})\right)+\mu\left(1+\frac{1}{F_{11}^2}\right)-E\right) \\ \frac{\pi^2 u_2}{\rho_0}\left(\frac{\lambda}{F_{22}^2}\left(1-\ln(F_{11}F_{22}F_{33})\right)+\mu\left(1+\frac{1}{F_{22}^2}\right)-E\right) \\ \frac{\pi^2 u_3}{\rho_0}\left(\frac{\lambda}{F_{33}^2}\left(1-\ln(F_{11}F_{22}F_{33})\right)+\mu\left(1+\frac{1}{F_{33}^2}\right)-E\right) \end{bmatrix}. \tag{48}$$

with deformation gradient matrix given as:

$$F_{ij}(\boldsymbol{X},t) = \delta_{ij} + \frac{\partial u_i}{\partial X_j} = \left(1 + \pi G\cos(\pi X_i)\sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_i\right)\right)\delta_{ij} \tag{49}$$

We also impose the following boundary conditions, obtained using Eq. (47), onto the nodes of the background mesh:

$$\boldsymbol{v}(X_i = 0, t) = \boldsymbol{v}(X_i = 1, t) = 0 \tag{50}$$

as well as the following initial conditions:

$$\boldsymbol{v}(\boldsymbol{X},0) = \pi\sqrt{\frac{E}{\rho_0}}G\sin(\pi\boldsymbol{X})\cos(\boldsymbol{\phi}) \tag{51}$$

$$\boldsymbol{P}(\boldsymbol{X},0) = 0 \tag{52}$$

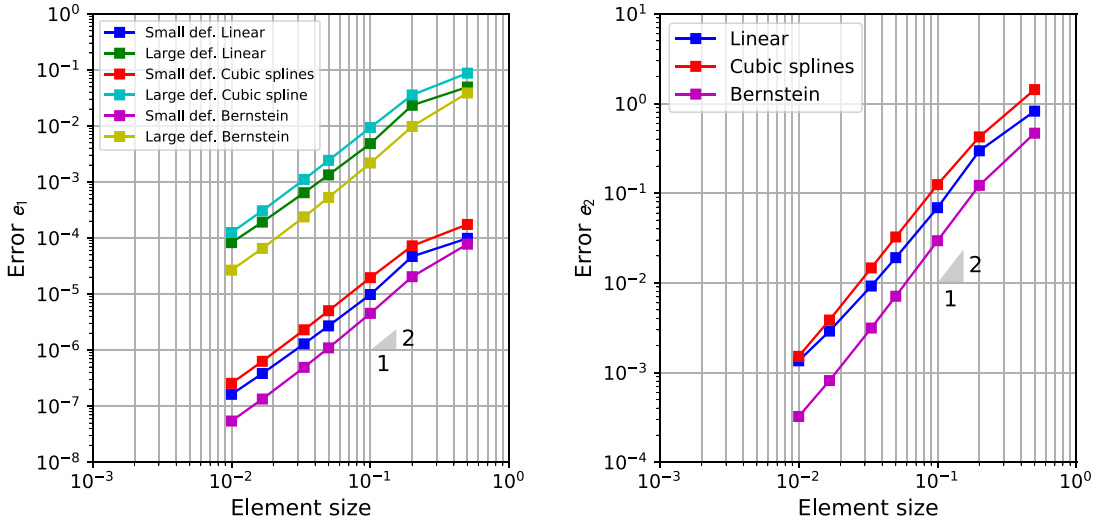
In order to assess the accuracy of the numerical solution, one needs a measure of the error made compared to the analytical solution (represented by Eq. (47), in this case). However, what measure of the error to be adopted is confusing as different authors used different definitions. Therefore, in this work, we used two different errors, the first is not-normalized while the second is. The derivation of the first error is based on the "distance" measure between the numerical and analytical solution and integrated in space. This measure is a function of time, and at time step t^n it is defined as:

$$e(t^n) = \sqrt{\frac{\sum_{p=0}^{N_p} V_p^0 \left|\left|\boldsymbol{u}_p^h(t^n) - \boldsymbol{u}(\boldsymbol{X}_p, t^n)\right|\right|^2}{V_{tot}^0}} \tag{53}$$

where $\boldsymbol{u}_p^h(t^n)$ is the numerical displacement at particle p, N_p the total number of particles in the solid and V_{tot}^0 its total initial volume. This error function is similar to that used by Brannon et al. [57]. However, to not have to compare the error at every time step, the first error measure is taken as the overall maximum of the error function:

$$e_1 = \max_n(e(t^n)) \tag{54}$$

This measure of error is not normalized and has the dimension of a length. It is therefore dependent on the maximum amplitude of the displacement. Normalizing the error function can be done by dividing Eq. (53) by $\sqrt{\sum_{p=0}^{N_p} V_p^0 \left|\left|\boldsymbol{u}(X_p, t^n)\right|\right|^2 / V_{tot}^0}$. However, this term can become null at different time step resulting in a non-defined



(a) Non-normalised error plot

(b) Normalised error plot

Fig. 12. Plot of the displacement error as a function of the background mesh refinement for small and large deformation obtained using TLMPM. Note that as e_2 is normalized, and since we did not see any difference between small and large deformations, this distinction is not made in (b).

error function. Therefore, the solution adopted here first integrates in time the square of the error function before normalizing it. This leads to the following expression of our second error measure:

$$e_2 = \sqrt{\frac{\sum_{t^n=0}^{T}\sum_{p=0}^{N_p} V_p^0 \left|\left|\boldsymbol{u}_p^h(t^n) - \boldsymbol{u}(\boldsymbol{X}_p, t^n)\right|\right|^2}{\sum_{t^n=0}^{T}\sum_{p=0}^{N_p} V_p^0 \left|\left|\boldsymbol{u}(\boldsymbol{X}_p, t^n)\right|\right|^2}} \tag{55}$$

This MMS verification test is performed using the following material parameters: $E = 10^7$ Pa, $\nu = 0.3$, and $\rho_0 = 1000$ kg/m^3. Both small and large displacements are tested by setting the maximum amplitude displacement $G = 10^{-4}$ and $G = 0.05$, respectively. In both cases, the test is done using linear, cubic splines, and Bernstein shape function as described in Section 2.4. The same time step of $0.2h/c$, where h is the background grid element size, was used. We believe that this small time step eliminates any error due to time discretization. And note that herein we focus on spatial convergence only. For all the tested cases, the error as a function of the background grid cell size made by the TLMPM using a combination of PIC and FLIP with $\beta = 0.99$ for the velocity update is plotted in Fig. 12. It can be seen that contrary to the standard MPM which does not converge (see [28]), the TLMPM converges for all the different shape functions tested. In particular, its convergence is quadratic when using either cubic splines or Bernstein shape functions and quasi quadratic when using linear shape functions. The trends are similar for either error measure e_1 or e_2 used.

Remark 3.1. Note that as e_1 is not normalized and is therefore proportional to G, the absolute error is higher for large than for small deformations, while as e_2 is normalized, and no difference was seen between these two cases, Fig. 12(b) does not distinguish between large and small deformations.

3.2.2. Generalized vortex problem

The "generalized vortex problem" is an example of MMS involving simple shear with superimposed rotation which is a complicated problem to simulate using MPM and its advanced (improved) variants [57,58]. It is thus a good example to show the potential of TLMPM. This problem features a 2D ring which is locally subjected to a

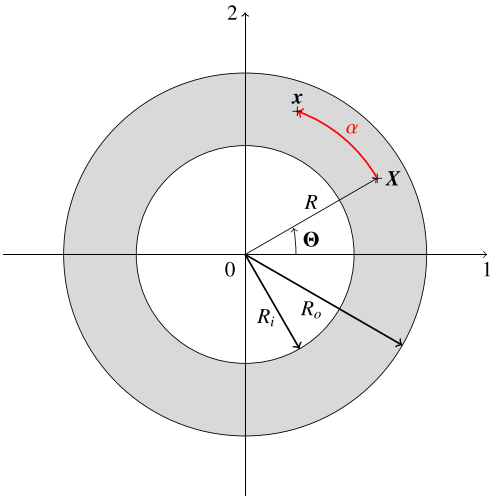


Fig. 13. Schematic of the problem domain of the generalized vortex example.

pure circular motion. The local displacement is purely angular and varies with the radial coordinate. Therefore, the material is only subjected to shear and circular motion.

The ring centre is at the origin ($x = y = 0$) and its inner and outer radii are R_i and R_o, respectively (see Fig. 13). The current position $\boldsymbol{x}$ of any given point of the ring is thus given as:

$$\boldsymbol{x} = \boldsymbol{Q} \cdot \boldsymbol{X} \tag{56}$$

with $\boldsymbol{Q}$ being a standard rotation matrix in 2D:

$$\boldsymbol{Q} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}, \tag{57}$$

where α is the rotation angle which varies only with time and radial coordinate R ($R = \sqrt{X^2 + Y^2}$). It is given as:

$$\alpha(t, R) = g(t)h(R) \tag{58}$$

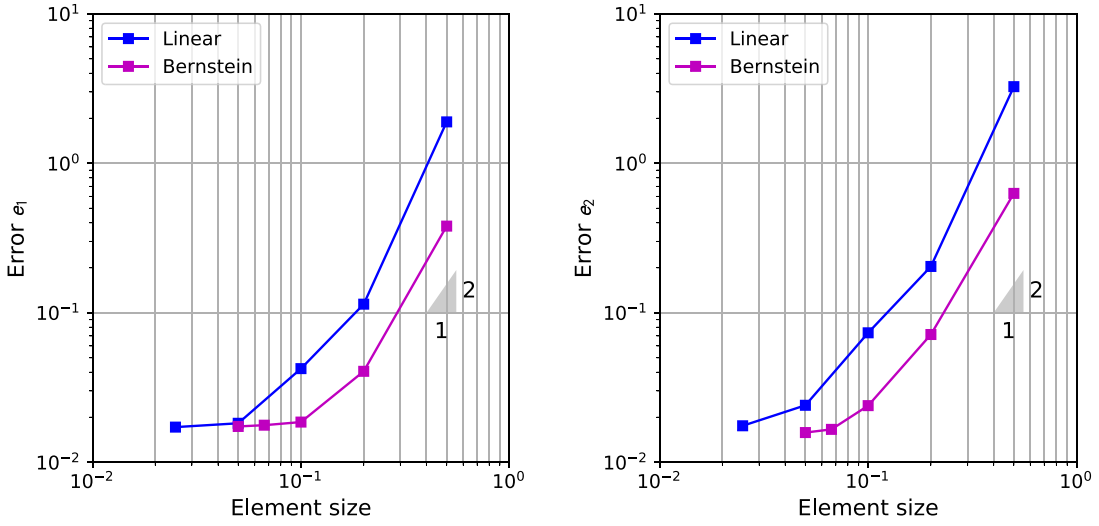
where $g(t)$ controls the amplitude of the deformation with time and $h(R)$ controls the relative radial variation of the rotation; $g(t)$ is taken as periodic, and $h(R)$ such that the outer and inner radii do not move: $h(R_i) = h(R_o) = 0$. Zero traction on the boundaries is insured by having: $h'(R_i) = h'(R_o) = 0$ as well. Following Brannon et al. [57], they are given by:

$$h(R) = 1 - 8\left(\frac{R - \bar{R}}{R_i - R_0}\right)^2 + 16\left(\frac{R - \bar{R}}{R_i - R_0}\right)^4, \quad g(t) = G\sin\left(\frac{\pi t}{T}\right) \tag{59}$$

with $\bar{R} = (R_0 + R_i)/2$.

After deriving the deformation matrix and solving the momentum equation for a Neo-Hookean solid, the body forces necessary to obtain this vortex motion are expressed as (see Appendix A.2 for detailed derivation):

$$\begin{aligned} \boldsymbol{b}_1(R, t) &= \boldsymbol{b}_R(R, t)\cos\Theta - \boldsymbol{b}_\Theta(R, t)\sin\Theta \\ \boldsymbol{b}_2(R, t) &= \boldsymbol{b}_\Theta(R, t)\cos\Theta + \boldsymbol{b}_R(R, t)\sin\Theta \end{aligned} \tag{60}$$



(a) Non-normalised error plot

(b) Normalised error plot

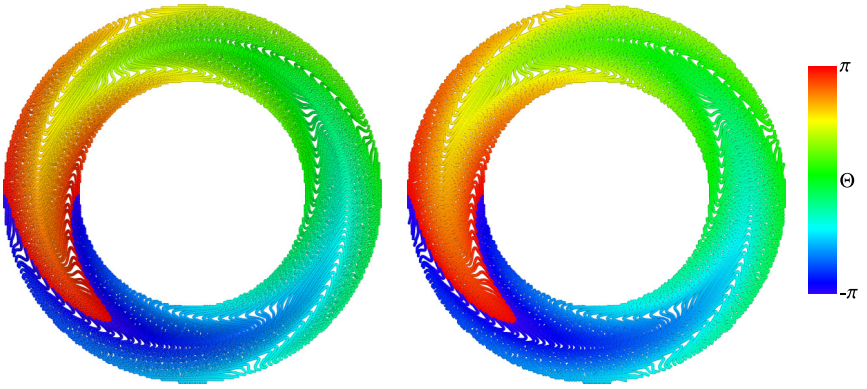
Fig. 14. Plot of the displacement error as a function of the background mesh refinement obtained using TLMPM. Note that e_2 is normalized.

where:

$$\begin{aligned}
\boldsymbol{b}_R(R,t) &= \left[\frac{\mu}{\rho_0}\left(3g(t)h'(R) + Rg(t)h''(R)\right) - Rg''(t)h(R)\right]\sin\alpha \\
&+ \left[\frac{\mu}{\rho_0}R(g(t)h'(R))^2 - R(g'(t)h(R))^2\right]\cos\alpha \\
\boldsymbol{b}_\Theta(R,t) &= \left[-\frac{\mu}{\rho_0}\left(3g(t)h'(R) + Rg(t)h''(R)\right) + Rg''(t)h(R)\right]\cos\alpha \\
&+ \left[\frac{\mu}{\rho_0}R(g(t)h'(R))^2 + R(g'(t)h(R))^2\right]\sin\alpha
\end{aligned} \tag{61}$$

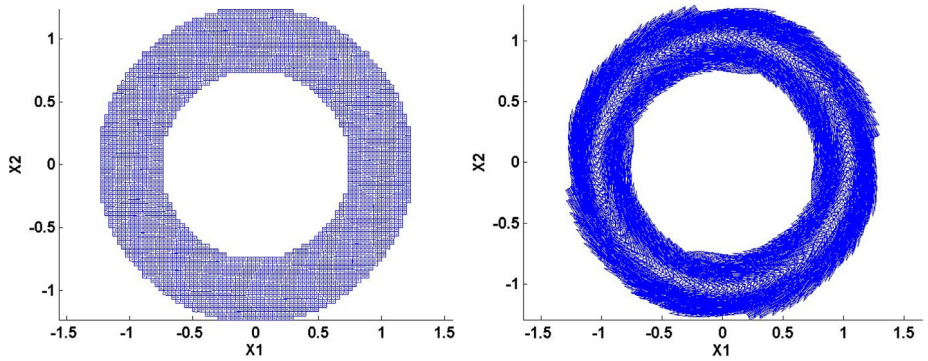
Similarly to Brannon et al. [57], the inner and outer radii are taken as 0.75 m and 1.25 m, respectively. Material parameters are taken as: $E = 10^3$ Pa, $\rho_0 = 1000$ kg/m^3, $\nu = 0.3$, and the maximum displacement amplitude $G = 1$. Depending on the shape function used, the material points (particles) are positioned according to what is stated in Section 2.4. If a particle is located outside of the area delimited by the inner and outer radius, it is removed. The particles' velocities are updated using a mix of PIC and FLIP with the mixing factor $\beta = 0.99$. Moreover, the velocity of all the nodes outside of the ring is set to zero, and the body forces (Eq. (60)) are applied directly on the nodes.

Fig. 14 shows how the displacement errors e_1 and e_2 change according to the element size. It can be seen that at large element size, the convergence rate is quadratic, but decreases progressively as the cell size decreases. This is the sign of a competition between two errors: cell size related errors and mapping errors. As the cell size decreases, so does the error associated to it, while the mapping error which appeared negligible for large cell sizes becomes dominant. Thus the error plateaus. Note that there is another source of error — the mapping of particle momenta to the grid as shown in [33,59]. We did not implement those improvements to mitigate this error and leave it as a future work. However, the order of magnitude of this plateaus is so low that very good qualitative agreement exists between the deformed configurations at peak rotation angle as obtained with TLMPM using Bernstein shape functions and the analytical solution (see Fig. 15). This is to put in perspective with the results obtained by Brannon et al. using CPDI. Even though their simulation is stable, one can see in Fig. 16 that at peak rotation the boundaries of the domain as they obtained using CPDI are not as smooth and circular than what we obtained with TLMPM.



(a) analytical solution (b) TLMPM solution

Fig. 15. Deformed configuration at $t = 0.5$ s obtained (a) analytically and (b) with TLMPM using Bernstein polynomials of degree 2 shape functions and a cell size of 0.033.



(a) (b)

Fig. 16. Reproduction of (a) the undeformed problem of the generalized vortex example used by Brannon and coworkers and (b) its deformed configuration near the peak rotation angle obtained using CPDI [57].

Remark 3.2. As the current configuration moves away from the reference configuration, the ability of the deformation matrix to capture the solid's deformation decreases. As indicated by Leroch et al. [14], when dealing with extremely large material deformation, the use of a constant reference configuration could generate non-negligible errors. This phenomenon is common to all total Lagrangian schemes and could explain why the error plateaus as the element size decreases. In order to mitigate these errors, the reference configuration could be updated periodically similarly to what Leroch and coworkers have done with Total Lagrangian SPH [48]. This could have the detrimental effect of potentially generating numerical fractures, and increasing the quadrature error as the material points move away from the background grid's integration points. In spite of this limitation, the TLMPM was still found to be more accurate than other variants of MPM when extremely large deformations occur as shown by the results of the "generalized vortex problem". Therefore, this problem seems to be a second order effect.

3.3. Taylor bar test

The Taylor bar test is a well suited problem to evaluate the performance of elasto-plastic constitutive models and numerical codes when large deformations occur [60–63]. Here, this problem not only demonstrates the stability of TLMPM under large deformations, but also allows us to compare its predictions with experimental results published by Johnson and Holmquist [62]. This test has been studied using the MPM, see for instance [13,64]. Note that [64]

Table 1
Material parameters for the OFHC Copper material used in the simulation of the Taylor bar impact test.

Material parameters		Parameters for Johnson–Cook model		Parameters for EOS	
Density	8940 kg/m^3	A	65 MPa	c_0	3933 m/s
Young's modulus	115 GPa	B	356 MPa	S_α	1.5
Poisson's ratio	0.31	C	0.013	Γ_0	0
		n	0.37		

Table 2
Results of the Taylor bar impact study [13].

	Initial geometry	Final geometry (exp.)	Final geometry (TLMPM)	Final geometry (MPM)
Diameter	7.6 mm	13.5 mm	13.9 mm	14.6 mm
Bulge	7.6 mm	10.1 mm	9.40 mm	9.12 mm
Length	25.4 mm	16.2 mm	16.2 mm	18.3 mm

presented a staggered grid MPM (SGMP) — the latest effort to eliminate the cell-crossing noise in the ULMPM. It has been shown to be more accurate than the standard MPM with a slightly higher computational cost.

This test involves a cylinder hitting a fixed and rigid wall at a high velocity. It was used by Johnson and Holmquist to compare various constitutive models for both OFHC Copper and Armco Iron. In their work, they performed experiments to determine the material parameters for these two materials using different constitutive models. They also performed numerical simulations to compare the models.

The proposed TLMPM is tested against the predictions made by Sulsky et al. [13] using MPM. The bar is a cylinder of original length $L_0 = 25.4$ mm, original diameter $D_0 = 7.6$ mm, made of OFHC Copper. This material is modelled using the elasto-plastic constitutive model presented in Section 2.2.5, but without damage. The material parameters used are taken from Sulky's work and are listed in Table 1.

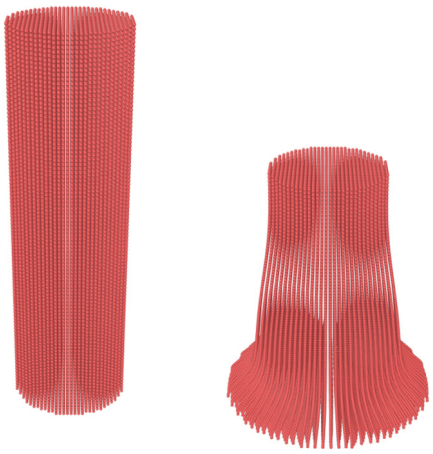
Fig. 17(a) shows the distribution of the material points in the initial undeformed cylinder seen in a plane perpendicular to its axis. The mesh cell size is $h = 0.25$ mm with 1 material point per element for a total of 74,052 points. Moreover, linear shape functions are used. The initial velocity is set at $v_0 = 190$ m/s. The presence of the wall is simulated by forcing the vertical velocity of the nodes coincident with its position to zero. Fig. 17(b) shows the geometry of the deformed cylinder at the end of the simulation. The predicted values by TLMPM for the final diameter, bulge and length are respectively $D_f = 13.9$ mm, $W_f = 9.40$ mm and $L_f = 16.2$ mm. The performance of TLMPM is compared to that of MPM using the error measure introduced by Johnson and Holmquist [62] and defined as:

$$\bar{\Delta} = \frac{1}{3}\left[\frac{|\Delta L|}{L_T} + \frac{|\Delta D|}{D_T} + \frac{|\Delta W|}{W_T}\right] \tag{62}$$

where L_T, D_T and W_T are the length, diameter and bulge measured experimentally and $\Delta L = L_f - L_T$, $\Delta D = D_f - D_T$ and $\Delta W = W_f - W_T$. The TLMPM simulations give $\bar{\Delta} = 0.03$ which is lower than that of MPM with $\bar{\Delta} = 0.1$. The results using TLMPM are therefore in better agreement with the experimental values than that obtained using MPM by Sulsky et al. as shown in Table 2.

3.4. Tensile test specimen experiencing necking and damage

The Total-Lagrangian Material Point Method is also a good method for the simulation of ductile materials experiencing large deformations, and eventually damage. This is important for the large range of problems such as wear, impact, fatigue and fracture. In this section, 3D simulations of smooth cylindrical tensile samples made of ductile Weldox steels all the way to failure are used to show that (a) the TLMPM is stable when material instabilities occur as experienced during necking, (b) it can simulate damage and fracture of ductile materials without requiring any algorithm changes, contrary to Total-Lagrangian Smooth Particle Hydrodynamics [65], and (c) it is ready to be used in engineering applications.



(a) (b)

Fig. 17. Taylor bar impact for an elasto-plastic OFHC copper cylinder given an initial downward velocity of 190 m/s. (a) Initial configuration of the material points. (b) Final configurations of the points.

Table 3
Material parameters for Weldox steels. ρ_0 is the reference bulk density, E Young modulus, ν Poisson's ratio, $c_0 = \sqrt{E/(2(1-2\nu)\rho_0)}$ is the bulk speed of sound, Γ_0 Grüneisen Gamma in the reference state.

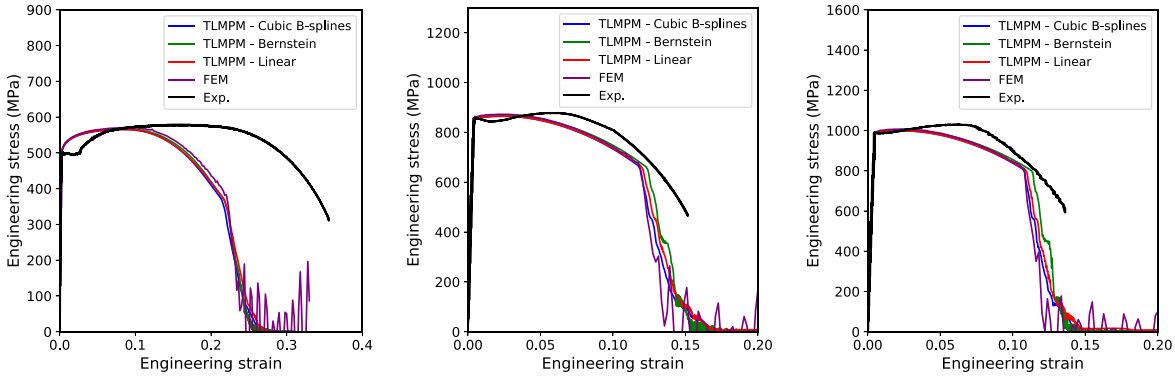
ρ_0 (kg/m^3)	E (GPa)	ν	c_0 (m/s)	S_α	Γ_0
7750	211	0.33	5166	1.5	0

Table 4
Material constants for the Johnson–Cook constitutive model and damage criterion as proposed by Dey et al. [49].

Material	Yield stress	Strain hardening			Damage			
	A (MPa)	B (MPa)	n	C	D_1	D_2	D_3	D_4
Weldox 460E	499	382	0.458	0	0.636	1.936	−2.969	−0.0140
Weldox 700E	859	329	0.579	0	0.361	4.768	−5.107	−0.0013
Weldox 900E	992	364	0.568	0	0.294	5.149	−5.583	0.0023

The tensile specimens are 3D smooth cylinders 30 mm in length and 6 mm in diameter made of three different Weldox steel alloys: W460E, W700E, and W900E. These materials were selected because material parameters for the widely used Johnson–Cook constitutive and damage laws were available in the literature. Moreover, experimental results for the simulated tensile tests have been published in the literature [49,66]. For these three alloys, their material parameters are directly taken from the literature and are listed in Tables 3 and 4. All simulations are supposed to be quasi-static. Therefore, the influence of the strain rate is not taken into account which is why $C = 0$ for all.

For each of the three materials, the TLMPM simulations are performed using each of the three different shape functions presented in Section 2.4: linear (hat functions), cubic B-splines, and Bernstein polynomials. For all these simulations, the specimens were discretized using a single particle for each cell of the background mesh, when they lie within the solid's limits. The nodes of the background mesh coincident with the top and bottom faces of the cylinders were subjected to a velocity $\pm v$ of the form $v_{max}(1 - e^{-t})$ is applied in the direction colinear to the each specimen's axis, with $v_{max} = 1$ mm/ms. Such a high velocity was chosen in order to decrease the total required time to be simulated and has no impact on the results as no strain rate effect was taken into account. The results of these simulations are presented in Fig. 18, alongside results from FEM simulations and experimental data published by Dey et al. [49]. One can see that the stress–strain curves as predicted by the TLMPM is in very good agreement with FEM. Also, as expected, after damage initiation a steady decline of stress is observed, similarly to



(a) W460E (b) W700E (c) W900E

Fig. 18. Comparison of the stress–strain curves obtained with the TLMPM, FEM and experimentally by Dey et al. [49].

what happens in FEM, but with greater stability. Finally, the numerical simulations are in reasonable agreement with the experimental data. However, they do not quantitatively predict the strain at failure. This was expected since all the materials parameters are directly taken from the literature and no calibration was carried out in order to obtain a better match (not the focus of this study).

Fig. 19 shows the typical evolution of both the equivalent stress and damage variable inside a tensile sample. It shows formation and evolution of necking in the specimen, as well as how the damage initiates and propagates inside the sample. These results prove that the simulation of ductile materials all the way to failure is possible using TLMPM.

4. Conclusions

Within the context of large deformation solid dynamics, we have presented a Total Lagrangian Material Point Method (TLMPM), which does not, as compared with the standard MPM, suffer from cell-crossing instabilities. Moreover, compared to other MPM variants quadrature errors in the TLMPM are minimal. The method has been demonstrated for impact analyses of a cylinder bar made of steel and necking and fracture of cylinder alloy specimens. The numerical solutions are in satisfactory agreement with the experiment data. No numerical fracture was observed for simulations involving very large tensile deformation without special treatment such as done in the convected particle domain interpolation (CPDI) methods. Convergence analyses using the method of manufactured solutions show that the TLMPM is second-order accurate for problems of which boundary is axis-aligned. For the challenging generalized vortex problem, it also converges quadratically for relatively coarse meshes.

In summary, compared with advanced MPM variants such as GIMP, CPDI and BSMPM (updated Lagrangian MPM adopting B-splines), the TLMPM is a more efficient and easier method to implement than the advanced MPM variants. Moreover, unlike the MPM, it does not suffer from cell-crossing instabilities, has a higher order of convergence, does not experience numerical fracture, and is able to simulate fracture using physically based models. However, contacts are not possible without special treatment. But this can be done using existing technologies developed for other particle methods.

Acknowledgements

The first and last authors gratefully acknowledge the financial support of the Australian Research Council (ARC) Training Centre in Alloy Innovation for Mining Efficiency (IC160100036). The second author (V.P. Nguyen) thanks the funding support from the Australian Research Council via DECRA project DE160100577.

Appendix. Method of manufactured solutions

This appendix provides the derivation of the body forces for the ‘axis-aligned displacement’ problem in Appendix A.1 and for the generalized vortex problem in Appendix A.2. The derivation given in [56] for the

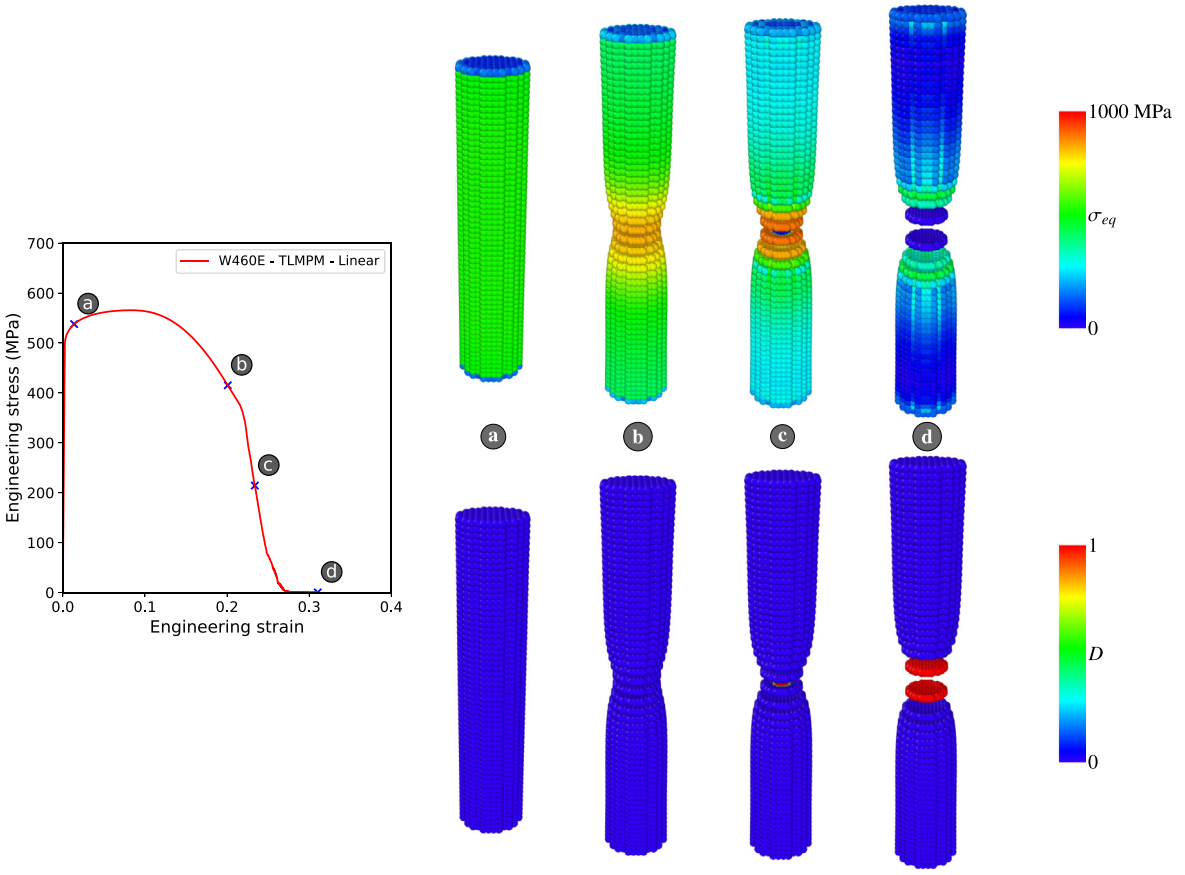


Fig. 19. Details of the evolution of the equivalent von Mises stress (σ_{eq}) and the damage (D) distribution during the tensile test of a W460E smooth cylinder using linear shape functions.

generalized vortex problem is extremely general and not easy to follow. Our derivation is made clearer, but is only valid in the case of Neo-Hookean materials.

A.1. Derivation of the axis-aligned displacement in a unit cube body forces

The starting point is the prescribed displacements:

$$\boldsymbol{u}(\boldsymbol{X},t) = G\sin(\pi\boldsymbol{X})\sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \boldsymbol{\phi}\right) \tag{A.1}$$

The goal is to determine the body forces. They are calculated by inverting the momentum equation:

$$\rho_0\boldsymbol{a}(\boldsymbol{X},t) = \nabla_0\boldsymbol{P}(\boldsymbol{X},t) + \rho_0\boldsymbol{b}(\boldsymbol{X},t) \tag{A.2}$$

which depends on the acceleration $\boldsymbol{a}(\boldsymbol{X},t)$ and the divergence of the stress $\nabla_0\boldsymbol{P}(\boldsymbol{X},t)$.

First, let us determine the acceleration. This is done by taking the second derivative of the displacement, which gives:

$$\boldsymbol{a}(\boldsymbol{X},t) = -\pi^2\frac{E}{\rho_0}G\sin(\pi\boldsymbol{X})\sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \boldsymbol{\phi}\right) = -\pi^2\frac{E}{\rho_0}\boldsymbol{u} \tag{A.3}$$

Second, let us determine $\nabla_0 \boldsymbol{P}(\boldsymbol{X}, t)$. Owing to the use of the Neo-Hookean model, $\boldsymbol{P}$ depends only on the deformation matrix $\boldsymbol{F}$ which is calculated by taking the spatial derivative of the displacement with respect to the initial position:

$$\begin{aligned}\boldsymbol{F}(\boldsymbol{X}, t) &= \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} \\ &= \begin{bmatrix} 1 + \pi G \cos(\pi X_1) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_1\right) & 0 & 0 \\ 0 & 1 + \pi G \cos(\pi X_2) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_2\right) & 0 \\ 0 & 0 & 1 + \pi G \cos(\pi X_3) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_3\right) \end{bmatrix}\end{aligned} \tag{A.4}$$

Then, by taking the spatial derivative of $\boldsymbol{F}$, one gets:

$$\nabla_0 \boldsymbol{F}(\boldsymbol{X}, t) = \begin{bmatrix} -\pi^2 G \sin(\pi X_1) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_1\right) \\ -\pi^2 G \sin(\pi X_2) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_2\right) \\ -\pi^2 G \sin(\pi X_3) \sin\left(\sqrt{\frac{E}{\rho_0}}\pi t + \phi_3\right) \end{bmatrix} = -\pi^2 \boldsymbol{u} \tag{A.5}$$

Eventually, the divergence of the stress is given by, using Eq. (24)

$$\nabla_0 \boldsymbol{P}(\boldsymbol{X}, t) = \begin{bmatrix} \pi^2 u_1 \left(\frac{\lambda}{F_{11}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{11}^2}\right)\right) \\ \pi^2 u_2 \left(\frac{\lambda}{F_{22}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{22}^2}\right)\right) \\ \pi^2 u_3 \left(\frac{\lambda}{F_{33}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{33}^2}\right)\right) \end{bmatrix} \tag{A.6}$$

Finally, one can invert Eq. (A.2) by substituting for both the acceleration and the divergence of the stress using Eqs. (A.3) and (A.6). We can solve for the body force:

$$\boldsymbol{b}(\boldsymbol{X}, t) = \begin{bmatrix} \frac{\pi^2 u_1}{\rho_0} \left(\frac{\lambda}{F_{11}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{11}^2}\right) - E\right) \\ \frac{\pi^2 u_2}{\rho_0} \left(\frac{\lambda}{F_{22}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{22}^2}\right) - E\right) \\ \frac{\pi^2 u_3}{\rho_0} \left(\frac{\lambda}{F_{33}^2} \left(1 - \ln(F_{11} F_{22} F_{33})\right) + \mu \left(1 + \frac{1}{F_{33}^2}\right) - E\right) \end{bmatrix} \tag{A.7}$$

A.2. Derivation of the generalized vortex problem body forces

In the generalized vortex problem, the starting point is not the displacement, but the current position $\boldsymbol{x}$:

$$\boldsymbol{x} = \boldsymbol{Q} \cdot \boldsymbol{X} \quad \text{with} \quad \boldsymbol{Q} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \tag{A.8}$$

Before doing any derivation, let us note that for any scalar s the following identity is true:

$$\frac{d\boldsymbol{Q}}{ds} = \frac{d\boldsymbol{Q}}{d\alpha}\frac{d\alpha}{ds} = \boldsymbol{A}\,\boldsymbol{Q}\frac{d\alpha}{ds} \tag{A.9}$$

with:

$$\boldsymbol{A} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{A.10}$$

Similarly to the other MMS problem, the goal is to determine the body forces which are obtained by inverting the momentum equation:

$$\rho_0 \boldsymbol{a}(\boldsymbol{X}, t) = \nabla_0 \boldsymbol{P}(\boldsymbol{X}, t) + \rho_0 \boldsymbol{b}(\boldsymbol{X}, t) \tag{A.11}$$

which depends on the acceleration $\boldsymbol{a}(\boldsymbol{X}, t)$ and the divergence of the stress $\nabla_0 \boldsymbol{P}(\boldsymbol{X}, t)$.

Therefore, one first needs to take the second temporal derivative of the position to get the acceleration:

$$\begin{aligned}\boldsymbol{a}(r,t) = \frac{\partial \boldsymbol{v}}{\partial t} &= g''(t)h(R)\boldsymbol{A}\boldsymbol{x} + g'(t)h(R)\boldsymbol{A}\frac{\partial \boldsymbol{x}}{\partial t} \\ &= g''(t)h(R)\boldsymbol{A}\boldsymbol{x} + (g'(t)h(R)\boldsymbol{A})^2\boldsymbol{x} \\ &= g''(t)h(R)\boldsymbol{A}\boldsymbol{x} - (g'(t)h(R))^2\boldsymbol{x} \\ &= g''(t)h(R)\boldsymbol{A}\boldsymbol{Q}\boldsymbol{X} - (g'(t)h(R))^2\boldsymbol{Q}\boldsymbol{X}\end{aligned} \tag{A.12}$$

with $\boldsymbol{A}\boldsymbol{Q}\boldsymbol{X}$ given by:

$$\boldsymbol{A}\boldsymbol{Q}\boldsymbol{X} = R\begin{bmatrix}-\sin\alpha \\ \cos\alpha\end{bmatrix}. \tag{A.13}$$

Therefore, the final expression for the acceleration is:

$$\boldsymbol{a}(r,t) = R\begin{bmatrix}-g''(t)h(R)\sin\alpha - (g'(t)h(R))^2\cos\alpha \\ g''(t)h(R)\cos\alpha + (g'(t)h(R))^2\sin\alpha\end{bmatrix} \tag{A.14}$$

The acceleration being now known, let us derive the divergence of the stress. Since we are using a Neo-Hookean law the first step is to determine the deformation gradient $\boldsymbol{F}$. Its ij component F_{ij} is obtained by taking the derivative of Eq. (A.8) as follows:

$$\begin{aligned}\boldsymbol{F}_{ij} = \frac{dx_i}{dX_j} &= \frac{d}{dX_j}\left(\sum_k \boldsymbol{Q}_{ik}X_k\right) \\ &= \sum_k \frac{d\boldsymbol{Q}_{ik}}{dX_j}X_k + \sum_k \boldsymbol{Q}_{ik}\frac{dX_k}{dX_j} \\ &= \sum_k \frac{d\boldsymbol{Q}_{ik}}{d\alpha}\frac{d\alpha}{dX_j}X_k + \sum_k \boldsymbol{Q}_{ik}\delta_{kj} \\ &= \sum_k (\boldsymbol{A}\boldsymbol{Q})_{ik}\frac{d\alpha}{dX_j}X_k + \boldsymbol{Q}_{ij}\end{aligned} \tag{A.15}$$

Thus, recalling Eq. (A.13), one gets:

$$\begin{aligned}\boldsymbol{F} &= \boldsymbol{Q} + \boldsymbol{A}\boldsymbol{Q}\boldsymbol{X} \otimes \frac{d\alpha}{d\boldsymbol{X}} \\ &= \boldsymbol{Q} + \frac{d\alpha}{dR}\boldsymbol{A}\boldsymbol{Q}\boldsymbol{X} \otimes \boldsymbol{E}_R \\ &= \boldsymbol{Q} + Rg(t)h'(R)\boldsymbol{A}\boldsymbol{Q}\boldsymbol{E}_R \otimes \boldsymbol{E}_R \\ &= \boldsymbol{Q}(\boldsymbol{I} + Rg(t)h'(R)\boldsymbol{A}\boldsymbol{E}_R \otimes \boldsymbol{E}_R) \\ &= \boldsymbol{Q}(\boldsymbol{I} + Rg(t)h'(R)\boldsymbol{E}_\Theta \otimes \boldsymbol{E}_R)\end{aligned} \tag{A.16}$$

Substituting Eq. (A.8) into Eq. (A.16) yields:

$$\boldsymbol{F} = \begin{bmatrix}\cos\alpha - Rg(t)h'(R)\sin\alpha & -\sin\alpha \\ \sin\alpha + Rg(t)h'(R)\cos\alpha & \cos\alpha\end{bmatrix} \tag{A.17}$$

For Neo-Hookean materials, the first Piola–Kirchhoff stress tensor depends on J, the determinant of $\boldsymbol{F}$, which is a measure of the volume change. Now that $\boldsymbol{F}$ is known, let us derive J:

$$J = \det(\boldsymbol{F}) = \det(\boldsymbol{I} + Rg(t)h'(R)\boldsymbol{E}_\Theta \otimes \boldsymbol{E}_R) = 1 \tag{A.18}$$

Since $J = 1$, there is no change of volume. This was expected since the mode of deformation in this problem is pure shear. Therefore, recalling the Neo-Hookean model, the first Piola–Kirchhoff stress tensor is:

$$\boldsymbol{P} = \mu(\boldsymbol{F} - \boldsymbol{F}^{-T}) \tag{A.19}$$

Since

$$\boldsymbol{F}^{-T} = \frac{1}{F_{RR}F_{\Theta\Theta}}\begin{bmatrix} F_{\Theta\Theta} & -F_{\Theta R} \\ F_{R\Theta} & F_{RR} \end{bmatrix} = \begin{bmatrix} F_{\Theta\Theta} & -F_{\Theta R} \\ -F_{R\Theta} & F_{RR} \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha - Rg(t)h'(R)\cos\alpha \\ \sin\alpha & \cos\alpha - Rg(t)h'(R)\sin\alpha \end{bmatrix} \tag{A.20}$$

Eq. (A.19) becomes:

$$\boldsymbol{P} = \mu\begin{bmatrix} \cos\alpha - Rg(t)h'(R)\sin\alpha - \cos\alpha & -\sin\alpha + \sin\alpha + Rg(t)h'(R)\cos\alpha \\ \sin\alpha + Rg(t)h'(R)\cos\alpha - \sin\alpha & \cos\alpha - \cos\alpha + Rg(t)h'(R)\sin\alpha \end{bmatrix} \tag{A.21}$$

and yields after simplifications:

$$\boldsymbol{P} = \mu Rg(t)h'(R)\begin{bmatrix} -\sin\alpha & \cos\alpha \\ \cos\alpha & \sin\alpha \end{bmatrix} \tag{A.22}$$

The last step consists in deriving the divergence of the first Piola–Kirchhoff stress tensor. In polar coordinates, the divergence of $\boldsymbol{P}$ is expressed as follows:

$$\begin{aligned} \boldsymbol{\nabla}_0\boldsymbol{P} &= \mu\begin{bmatrix} \frac{\partial P_{RR}}{\partial R} + \frac{P_{RR}-P_{\Theta\Theta}}{R} \\ \frac{\partial P_{\Theta R}}{\partial R} + \frac{P_{R\Theta}+P_{\Theta R}}{R} \end{bmatrix} \\ &= \mu\begin{bmatrix} -g(t)h'(R)\sin\alpha - Rg(t)h''(R)\sin\alpha - R(g(t)h'(R))^2\cos\alpha - 2g(t)h'(R)\sin\alpha \\ g(t)h'(R)\cos\alpha + Rg(t)h''(R)\cos\alpha - R(g(t)h'(R))^2\sin\alpha + 2g(t)h'(R)\cos\alpha \end{bmatrix} \end{aligned} \tag{A.23}$$

After simplifications, Eq. (A.23) becomes:

$$\boldsymbol{\nabla}_0\boldsymbol{P} = \mu\begin{bmatrix} -\big(3g(t)h'(R) + Rg(t)h''(R)\big)\sin\alpha - R(g(t)h'(R))^2\cos\alpha \\ \big(3g(t)h'(R) + Rg(t)h''(R)\big)\cos\alpha - R(g(t)h'(R))^2\sin\alpha \end{bmatrix} \tag{A.24}$$

Substituting Eqs. (A.14) and (A.24) into Eq. (A.11), and solving it for the body forces $\boldsymbol{b}$ gives:

$$\begin{aligned} \boldsymbol{b}_1(R,t) &= \boldsymbol{b}_R(R,t)\cos\Theta - \boldsymbol{b}_\Theta(R,t)\sin\Theta \\ \boldsymbol{b}_2(R,t) &= \boldsymbol{b}_\Theta(R,t)\cos\Theta + \boldsymbol{b}_R(R,t)\sin\Theta \end{aligned} \tag{A.25}$$

where

$$\begin{aligned} \boldsymbol{b}_R(R,t) =& -Rg''(t)h(R)\sin\alpha - R(g'(t)h(R))^2\cos\alpha \\ &+ \frac{\mu}{\rho_0}\big(\big(3g(t)h'(R) + Rg(t)h''(R)\big)\sin\alpha + R(g(t)h'(R))^2\cos\alpha\big) \\ \boldsymbol{b}_R(R,t) =& \left[\frac{\mu}{\rho_0}\big(3g(t)h'(R) + Rg(t)h''(R)\big) - Rg''(t)h(R)\right]\sin\alpha \\ &+ \left[\frac{\mu}{\rho_0}R(g(t)h'(R))^2 - R(g'(t)h(R))^2\right]\cos\alpha \end{aligned} \tag{A.26}$$

and

$$\begin{aligned} \boldsymbol{b}_\Theta(R,t) =& Rg''(t)h(R)\cos\alpha + R(g'(t)h(R))^2\sin\alpha \\ &- \frac{\mu}{\rho_0}\left[\big(3g(t)h'(R) + Rg(t)h''(R)\big)\cos\alpha - R(g(t)h'(R))^2\sin\alpha\right] \\ \boldsymbol{b}_\Theta(R,t) =& \left[-\frac{\mu}{\rho_0}\big(3g(t)h'(R) + Rg(t)h''(R)\big) + Rg''(t)h(R)\right]\cos\alpha \\ &+ \left[\frac{\mu}{\rho_0}R(g(t)h'(R))^2 + R(g'(t)h(R))^2\right]\sin\alpha \end{aligned} \tag{A.27}$$

References

[1] T. Rabczuk, T. Belytschko, S. Xiao, Stable particle methods based on Lagrangian kernels, Comput. Methods Appl. Mech. Engrg. 193 (12) (2004) 1035–1063, http://dx.doi.org/10.1016/j.cma.2003.12.005.

[2] J.-S. Chen, C. Pan, C.-T. Wu, W.K. Liu, Reproducing Kernel Particle Methods for large deformation analysis of non-linear structures, Comput. Methods Appl. Mech. Engrg. 139 (1–4) (1996) 195–227, http://dx.doi.org/10.1016/s0045-7825(96)01083-3.

[3] J. Bonet, S. Kulasegaram, Remarks on tension instability of Eulerian and Lagrangian corrected smooth particle hydrodynamics (CSPH) methods, Internat. J. Numer. Methods Engrg. 52 (11) (2001) 1203–1220.

[4] V.P. Nguyen, T. Rabczuk, S. Bordas, M. Duflot, Meshless methods: A review and computer implementation aspects, Math. Comput. Simulation 79 (3) (2008) 763–813, http://dx.doi.org/10.1016/j.matcom.2008.01.003.
[5] S. Li, W.K. Liu, Meshfree and particle methods and their applications, Appl. Mech. Rev. 55 (1) (2002) 1, http://dx.doi.org/10.1115/1.1431547.
[6] T. Belytschko, Y. Guo, W.K. Liu, S.P. Xiao, A unified stability analysis of meshless particle methods, Internat. J. Numer. Methods Engrg. 48 (9) (2000) 1359–1400.
[7] G.C. Ganzenmüller, An hourglass control algorithm for Lagrangian Smooth Particle Hydrodynamics, Comput. Methods Appl. Mech. Engrg. 286 (Suppl. C) (2015) 87–106, http://dx.doi.org/10.1016/j.cma.2014.12.005.
[8] P. Randles, L. Libersky, Smoothed Particle Hydrodynamics: Some recent improvements and applications, Comput. Methods Appl. Mech. Engrg. 139 (1) (1996) 375–408, http://dx.doi.org/10.1016/S0045-7825(96)01090-0.
[9] F. Harlow, The particle-in-cell computing method for fluid dynamics, Methods Comput. Phys. 3 (1964) 319–343.
[10] J. Brackbill, H. Ruppel, FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions, J. Comput. Phys. 65 (2) (1986) 314–343.
[11] D. Sulsky, Z. Chen, H. Schreyer, A particle method for history-dependent materials, Comput. Methods Appl. Mech. Engrg. 118 (1–2) (1994) 179–196, http://dx.doi.org/10.1016/0045-7825(94)90112-0.
[12] D. Sulsky, S.-J. Zhou, H.L. Schreyer, Application of a particle-in-cell method to solid mechanics, Comput. Phys. Comm. 87 (1–2) (1995) 236–252, http://dx.doi.org/10.1016/0010-4655(94)00170-7.
[13] D. Sulsky, H.L. Schreyer, Axisymmetric form of the material point method with applications to upsetting and Taylor impact problems, Comput. Methods Appl. Mech. Engrg. 139 (1–4) (1996) 409–429, http://dx.doi.org/10.1016/s0045-7825(96)01091-2.
[14] S. Leroch, S. Eder, G. Ganzenmüller, L. Murillo, M.R. Ripoll, Development and validation of a meshless 3D material point method for simulating the micro-milling process, J. Mater Process. Technol. 262 (2018) 449–458, http://dx.doi.org/10.1016/j.jmatprotec.2018.07.013.
[15] T. Fagan, V. Lemiale, J. Nairn, Y. Ahuja, R. Ibrahim, Y. Estrin, Detailed thermal and material flow analyses of friction stir forming using a three-dimensional particle based model, J. Mater Process. Technol. 231 (2016) 422–430, http://dx.doi.org/10.1016/j.jmatprotec.2016.01.009.
[16] T. Mishra, G.C. Ganzenmüller, M. de Rooij, M. Shisode, J. Hazrati, D.J. Schipper, Modelling of ploughing in a single-asperity sliding contact using material point method, Wear 418–419 (2019) 180–190, http://dx.doi.org/10.1016/j.wear.2018.11.020.
[17] B.L. Boyce, S.L.B. Kramer, H.E. Fang, T.E. Cordova, M.K. Neilsen, K. Dion, A.K. Kaczmarowski, E. Karasz, L. Xue, A.J. Gross, A. Ghahremaninezhad, K. Ravi-Chandar, S.-P. Lin, S.-W. Chi, J.S. Chen, E. Yreux, M. Rüter, D. Qian, Z. Zhou, S. Bhamare, D.T. O'Connor, S. Tang, K.I. Elkhodary, J. Zhao, J.D. Hochhalter, A.R. Cerrone, A.R. Ingraffea, P.A. Wawrzynek, B.J. Carter, J.M. Emery, M.G. Veilleux, P. Yang, Y. Gan, X. Zhang, Z. Chen, E. Madenci, B. Kilic, T. Zhang, E. Fang, P. Liu, J. Lua, K. Nahshon, M. Miraglia, J. Cruce, R. DeFrese, E.T. Moyer, S. Brinckmann, L. Quinkert, K. Pack, M. Luo, T. Wierzbicki, The Sandia Fracture Challenge: blind round robin predictions of ductile tearing, Int. J. Fract. 186 (1) (2014) 5–68, http://dx.doi.org/10.1007/s10704-013-9904-6.
[18] P. Yang, Y. Gan, X. Zhang, Z. Chen, W. Qi, P. Liu, Improved decohesion modeling with the material point method for simulating crack evolution, Int. J. Fract. 186 (1) (2014) 177–184, http://dx.doi.org/10.1007/s10704-013-9925-1.
[19] S. Ma, X. Zhang, X. Qiu, Comparison study of MPM and SPH in modeling hypervelocity impact problems, Int. J. Impact Eng. 36 (2) (2009) 272–282, http://dx.doi.org/10.1016/j.ijimpeng.2008.07.001.
[20] S. Sinaie, T.D. Ngo, V.P. Nguyen, T. Rabczukc, Validation of the material point method for the simulation of thin-walled tubes under lateral compression, Thin-Walled Struct. 130 (2018) 32–46.
[21] J. Gaume, T. Gast, J. Teran, A. van Herwijnen, C. Jiang, Dynamic anticrack propagation in snow, Nature Commun. 9 (1) (2018) http://dx.doi.org/10.1038/s41467-018-05181-w.
[22] C. Jiang, C. Schroeder, A. Selle, J. Teran, A. Stomakhin, The affine particle-in-cell method, ACM Trans. Graph. 34 (4) (2015) 51:1–51:10, http://dx.doi.org/10.1145/2766996.
[23] M. Gao, A.P. Tampubolon, C. Jiang, E. Sifakis, An adaptive generalized interpolation material point method for simulating elastoplastic materials, ACM Trans. Graph. 36 (6) (2017) 223:1–223:12, http://dx.doi.org/10.1145/3130800.3130879.
[24] Y. Hu, Y. Fang, Z. Ge, Z. Qu, Y. Zhu, A. Pradhana, C. Jiang, A moving least squares material point method with displacement discontinuity and two-way rigid body coupling, ACM Trans. Graph. 37 (4) (2018) 150:1–150:14, http://dx.doi.org/10.1145/3197517.3201293.
[25] M. Steffen, R.M. Kirby, M. Berzins, Decoupling and balancing of space and time errors in the material point method (MPM), Internat. J. Numer. Methods Engrg. 82 (10) (2010) 1207–1243, http://dx.doi.org/10.1002/nme.2787, http://arxiv.org/abs/https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.2787.
[26] S. Bardenhagen, E. Kober, The generalized interpolation material point method, CMES Comput. Model. Eng. Sci. 5 (6) (2004) 477–495.
[27] D.Z. Zhang, X. Ma, P.T. Giguere, Material point method enhanced by modified gradient of shape function, J. Comput. Phys. 230 (16) (2011) 6379–6398, http://dx.doi.org/10.1016/j.jcp.2011.04.032.
[28] A. Sadeghirad, R.M. Brannon, J. Burghardt, A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations, Internat. J. Numer. Methods Engrg. 86 (12) (2011) 1435–1456.
[29] A. Sadeghirad, R. Brannon, J. Guilkey, Second-order convected particle domain interpolation (CPDI2) with enrichment for weak discontinuities at material interfaces, Internat. J. Numer. Methods Engrg. 95 (11) (2013) 928–952.
[30] V.P. Nguyen, C.T. Nguyen, T. Rabczuk, S. Natarajan, On a family of convected particle domain interpolations in the material point method, Finite Elem. Anal. Des. 126 (2017) 50–64.
[31] M.A. Homel, R.M. Brannon, J. Guilkey, Controlling the onset of numerical fracture in parallelized implementations of the material point method (MPM) with convective particle domain interpolation (CPDI) domain scaling, Internat. J. Numer. Methods Engrg. 107 (1) (2016) 31–48.

[32] M. Gong, Improving the Material Point Method (Ph.D thesis), The University of New Mexico, 2015.
[33] D. Sulsky, M. Gong, Improving the material-point method, in: Innovative Numerical Approaches for Multi-Field and Multi-Scale Problems, Springer, 2016, pp. 217–240.
[34] R. Tielen, E. Wobbes, M. Möller, L. Beuth, A high order material point method, Procedia Eng. 175 (2017) 265–272.
[35] E. Wobbes, M. Möller, V. Galavi, C. Vuik, Conservative Taylor least squares reconstruction with application to material point methods, Internat. J. Numer. Methods Engrg. 117 (3) (2019) 271–290.
[36] F. Zhu, J. Zhao, S. Li, Y. Tang, G. Wang, Dynamically enriched MPM for invertible elasticity, Comput. Graph. Forum 36 (6) (2016) 381–392, http://dx.doi.org/10.1111/cgf.12987.
[37] T. Hughes, J. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (39–41) (2005) 4135–4195.
[38] V.P. Nguyen, H. Nguyen-Xuan, High order B-splines based finite elements for the delamination analysis of laminated composites, Compos. Struct. 102 (2013) 261–275.
[39] M. Steffen, Analysis-guided improvements of the Material Point Method (MPM) (Ph.D thesis), University of Utah, 2009.
[40] T. Belytschko, W.K. Liu, B. Moran, Nonlinear Finite Elements for Continua and Structures, John Wiley & Sons, Chichester, England, 2000.
[41] S. Ma, X. Zhang, Y. Lian, X. Zhou, Simulation of high explosive explosion using adaptive material point method, CMES Comput. Model. Eng. Sci. (2009).
[42] P. Wallstedt, J. Guilkey, An evaluation of explicit time integration schemes for use with the generalized interpolation material point method, J. Comput. Phys. 227 (22) (2008) 9628–9642.
[43] J.A. Nairn, Material point method calculations with explicit cracks, Tech Sci. Press CMES 4 (6) (2003) 649–663.
[44] J. Bonet, R.D. Wood, Nonlinear Continuum Mechanics for Finite Element Analysis, Cambridge University Press, 1997.
[45] J. Lemaitre, A continuous damage mechanics model for ductile fracture, J. Eng. Mater. Technol. 107 (1) (1985) 83–89, http://dx.doi.org/10.1115/1.3225775.
[46] M.L. Wilkins, Computer Simulation of Dynamic Phenomena, Springer Berlin Heidelberg, 1999, URL https://www.ebook.de/de/product/6699393/mark_l_wilkins_computer_simulation_of_dynamic_phenomena.html.
[47] G.R. Johnson, W.H. Cook, Fracture characteristics of three metals subjected to various strains, strain rates, temperatures and pressures, Eng. Fract. Mech. 21 (1) (1985) 31–48, http://dx.doi.org/10.1016/0013-7944(85)90052-9.
[48] S. Leroch, M. Varga, S. Eder, A. Vernes, M.R. Ripoll, G. Ganzenmüller, Smooth particle hydrodynamics simulation of damage induced by a spherical indenter scratching a viscoplastic material, Int. J. Solids Struct. 81 (Suppl. C) (2016) 188–202, http://dx.doi.org/10.1016/j.ijsolstr.2015.11.025.
[49] S. Dey, T. Børvik, O. Hopperstad, M. Langseth, On the influence of fracture criterion in projectile impact of steel plates, Comput. Mater. Sci. 38 (1) (2006) 176–191, http://dx.doi.org/10.1016/j.commatsci.2006.02.003.
[50] M. Steffen, R.M. Kirby, M. Berzins, Analysis and reduction of quadrature errors in the material point method (MPM), Internat. J. Numer. Methods Engrg. 76 (6) (2008) 922–948, http://dx.doi.org/10.1002/nme.2360, http://arxiv.org/abs/https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.2360.
[51] V.P. Nguyen, C. Anitescu, S.P. Bordas, T. Rabczuk, Isogeometric analysis: An overview and computer implementation aspects, Math. Comput. Simulation 117 (2015) 89–116.
[52] Y. Gan, Z. Sun, Z. Chen, X. Zhang, Y. Liu, Enhancement of the material point method using B-spline basis functions, Internat. J. Numer. Methods Engrg. 113 (3) (2018) 411–431.
[53] L. Piegl, W. Tiller, The NURBS Book, Springer-Verlag Berlin Heidelberg, 1995.
[54] M.J. Borden, M.A. Scott, J.A. Evans, T.J.R. Hughes, Isogeometric finite element data structures based on Bézier extraction of NURBS, Internat. J. Numer. Methods Engrg. 87 (1–5) (2010) 15–47, http://dx.doi.org/10.1002/nme.2968.
[55] P.M. Knupp, K. Salari, in: K.H. Rosen (Ed.), Verification of Computer Codes in Computational Science and Engineering, Chapman & Hall/CRC Press, Boca Raton, FL, 2003.
[56] K. Kamojjala, R. Brannon, A. Sadeghirad, J. Guilkey, Verification tests in solid mechanics, Eng. Comput. 31 (2) (2015) 193–213, http://dx.doi.org/10.1007/s00366-013-0342-x.
[57] R. Brannon, K. Kamojjala, A. Sadeghirad, Establishing credibility of particle methods through verification testing, Part.-Based Methods II - Fundam. Appl. (2011) 685–696.
[58] L. Wang, W. Coombs, C. Augarde, M. Cortis, T. Charlton, M. Brown, J. Knappett, A. Brennan, C. Davidson, D. Richards, et al., On the use of domain-based material point methods for problems involving large distortion, Comput. Methods Appl. Mech. Engrg. 355 (2019) 1003–1025.
[59] P.C. Wallstedt, J.E. Guilkey, Improved velocity projection for the material point method, Comput. Model. Eng. Sci. 19 (3) (2007) 223–232.
[60] M.L. Wilkins, M.W. Guinan, Impact of cylinders on a rigid boundary, J. Appl. Phys. 44 (3) (1973) 1200–1206, http://dx.doi.org/10.1063/1.1662328.
[61] W.W. Predebon, C.E. Anderson, J.D. Walker, Inclusion of evolutionary damage measures in Eulerian wavecodes, Comput. Mech. 7 (4) (1991) 221–236, http://dx.doi.org/10.1007/BF00370037.
[62] G.R. Johnson, T.J. Holmquist, Evaluation of cylinder-impact test data for constitutive model constants, J. Appl. Phys. 64 (8) (1988) 3901–3910, http://dx.doi.org/10.1063/1.341344, http://arxiv.org/abs/https://doi.org/10.1063/1.341344.
[63] T.J. Holmquist, G.R. Johnson, Determination of constants and comparison of results for various constitutive models, J. Phys. IV France 01 (1991) C3–853–C3–860, http://dx.doi.org/10.1051/jp4:19913119.
[64] Y. Liang, X. Zhang, Y. Liu, An efficient staggered grid material point method, Comput. Methods Appl. Mech. Engrg. 352 (2019) 85–109.

[65] A. de Vaucorbeil, C.R. Hutchinson, A new Total-Lagrangian Smooth Particle Hydrodynamics approximation for the simulation of damage and fracture of ductile materials, Int. J. Numer. Methods Eng. (2019) in press.

[66] S. Dey, T. Børvik, O. Hopperstad, J. Leinum, M. Langseth, The effect of target strength on the perforation of steel plates using three different projectile nose shapes, Int. J. Impact Eng. 30 (8) (2004) 1005–1038, http://dx.doi.org/10.1016/j.ijimpeng.2004.06.004.