Mixed Material Point Methods for Stiff Elastoplasticity

GILLES DAVIET, NVIDIA, France

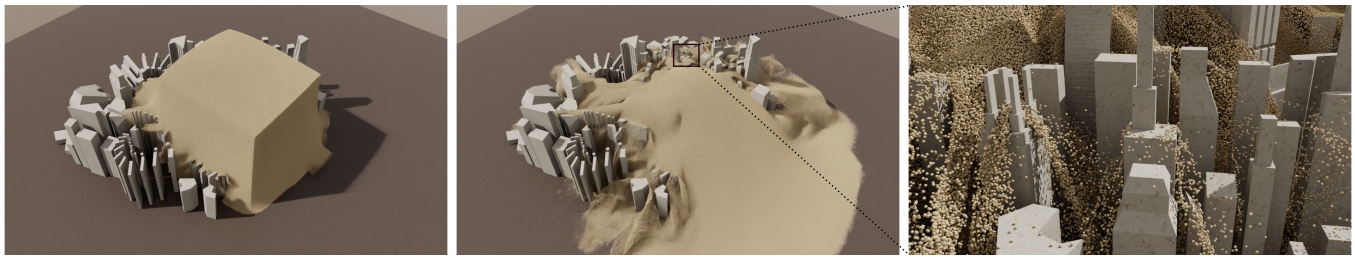


Fig. 1. A large cuboid of sand comprising 49M particles falls upon a model city, creating intricate patterns as it rushes through the narrow streets and skyline details. Thanks to the compact stencils of our mixed discretization, the simulation runs at 4s per frame on a single GPU.

We present a family of mixed Material Point Methods well suited for the CFL-rate simulation of stiff elastoviscoplastic materials, up to the incompressible limit. Our work builds upon the mixed discretization from Daviet and Bertails-Descoubes [2016a] and extends it to handle finite-strain viscoelasticity and more general flow rules, allowing the simulation of a much wider range of materials. Our implicit integration scheme leads to a well-posed, symmetric optimization problem with compact stencils for which we propose an efficient GPU solver. We demonstrate our method on a variety of examples ranging from granular materials and snow to elastic solids, including two-way coupling with rigid-body solvers.

CCS Concepts: • **Computing methodologies** → *Continuous models*; **Physical simulation**.

ACM Reference Format:
Gilles Daviet. 2026. Mixed Material Point Methods for Stiff Elastoplasticity. *ACM Trans. Graph.* 45, 4, Article 151 (July 2026), 19 pages. https://doi.org/10.1145/3811345

1 Introduction

The ever-growing amount of geometric data coming out of 3d reconstruction pipelines as unstructured point clouds such as Gaussian splats has sparked a renewed interest for mesh-free, particle-based simulation methods [Feng et al. 2024; Modi et al. 2024; Xie et al. 2023]. Among those, the Smoothed Particle Hydrodynamics [SPH; Gingold and Monaghan 1977; Koschier et al. 2022] and Material Point Method [MPM; Jiang et al. 2016; Sulsky et al. 1994] have long been popular in computer graphics and mechanical engineering for their ability to simulate a wide variety of materials under arbitrary topology changes.

However, those are not without limitations; SPH is known to be highly sensitive to the choice of radial kernel, with computational cost rising quickly with stencil size, while singularities become more likely with smaller ones [Westhofen et al. 2023]. MPM leverages a background grid in a Particle-In-Cell [PIC; Evans et al. 1957] fashion to mitigate this effect, and robust implementations have not only made their way into VFX production software, allowing for visually stunning simulations [Autodesk 2019; SideFX 2024], but have also been able to reproduce real-life experiments [Gaume et al. 2018; Rousseau et al. 2023]. Despite these achievements, MPM is usually associated with high computational cost; the main bottleneck being the use of interpolation stencils that span many particles, leading to large amounts of data having to be exchanged between the particles and the background grid at each time step or implicit iteration [Liu et al. 2025]. Compellingly, for fluid simulation Um et al. [2017] have observed PIC techniques to be computationally more efficient than SPH at the same perceptual quality. Indeed, in contrast to MPM, PIC methods for incompressible fluids typically leveraged mixed velocity–pressure discretizations — e.g, on a staggered "MAC" grid — in which the gradient and divergence operators have relatively small kernels, that do not vary with the local number of particles. Daviet and Bertails-Descoubes [2016b] proposed a mixed velocity–stress discretization for MPM granular simulations; however, their technique was limited to Drucker–Prager viscoplastic flows. In this work, we aim to generalize this approach to general elastoviscoplastic materials.

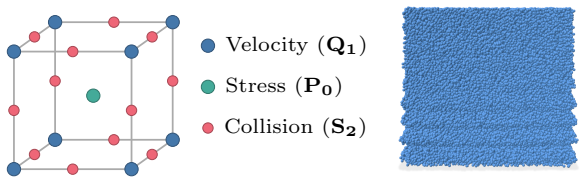
Our contributions are as follows: first, a holistic derivation of a family of mixed MPM discretizations from arbitrary elastic and dissipation potentials; second, a unified flow rule that allows interpolating between materials with widely different behavior, from Drucker–Prager granulars to brittle solids; and third, a GPU-friendly implicit solver for said discretization and flow rule, including two-way coupled frictional collisions with rigid bodies.

2 Related Work

A descendant from PIC [Evans et al. 1957], MPM [Sulsky et al. 1994] has been introduced to computer graphics by Stomakhin et al. [2013] for the simulation of snow, and was quickly extended to many more material types: foams [Ram et al. 2015; Yue et al. 2015], granulars [Daviet and Bertails-Descoubes 2016a; Dunatunga and Kamrin 2015; Klár et al. 2016], soft-bodies [Fei et al. 2018; Jiang et al. 2017, 2015], and combinations thereof. Thanks to its inherent ability to handle topology changes, granted from the updated Lagrangian formalism, cutting and fracture have been a particular focus of

Author's Contact Information: Gilles Daviet, gdaviet@nvidia.com, NVIDIA, Annecy, France.

This work is licensed under a Creative Commons Attribution 4.0 International License.
© 2026 Copyright held by the owner/author(s).
ACM 1557-7368/2026/7-ART151
https://doi.org/10.1145/3811345



(a) The $\mathbf{Q_1}$–$\mathbf{P_0}$–$\mathbf{S_2}$ mixed element. (b) $\mathbf{Q_1}$–$\mathbf{Q_1}$ instability.

Fig. 2. Our mixed MPM approach discretizes independently the velocity, stress, and collision fields. The $\mathbf{Q_1}$–$\mathbf{P_0}$–$\mathbf{S_2}$ mixed element (a) with trilinear velocities, piecewise-constant stresses, and quadratic serendipity collision nodes (illustrated in Figure 9) features a good performance–accuracy compromise and is used in most of our examples. In contrast, the $\mathbf{Q_1}$–$\mathbf{Q_1}$ element with collocated velocity and stress nodes used by Daviet and Bertails-Descoubes [2016a] is prone to parasitic oscillations, as evidenced here for an elastic cube resting on the ground (b).

attention for MPM [Fan et al. 2022; Hu et al. 2018; Wolper et al. 2020, 2019].

Implicit MPM. While many MPM implementations use explicit integration, those come with heavy timestep restrictions, inversely proportional to the material stiffness. A number of MPM methods have been proposed that leverage some degree of implicitness, starting from the snow simulation method of Stomakhin et al. [2013], or the hierarchical algorithm from [Wang et al. 2020a]. There, the elastic force equilibrium is solved implicitly, while plasticity, hardening, and advection are kept explicit. Other methods treat both elasticity and plasticity implicitly, but keep advection explicit; Klár et al. [2016] notes that plasticity breaks the symmetry of a Newton-based implicit solver. Closely related to our work, Fang et al. [2019] use an optimization-based viewpoint to solve elastoviscoplasticity implicitly, while Daviet and Bertails-Descoubes [2016a] proposed a Gauss–Seidel-based implicit solver for a viscoplastic model with critical-fraction-based hardening. Qu et al. [2023] and Yu et al. [2024] applied Gauss–Seidel to general elastoplastic models over Power Particles and SPH discretizations, respectively. Su et al. [2021] developed a second-order accurate implicit method for viscoelastic fluids. Recently, Zhao et al. [2026] leveraged sparse auto-differentiation to facilitate building implicit solvers.

Mixed MPM. MPM can be seen as a particular Finite Element Method (FEM), and like for other methods of this class, the choice of the grid basis functions significantly impacts the accuracy, stability, and computational performance of the solve. The use of Lagrangian particles to discretize the transport operators imposes supplemental restrictions. In particular, trilinear shape functions suffer from so-called *grid-crossing instabilities* [Bardenhagen et al. 2004; Zhang et al. 2011], as the force stemming from a given particle's strain switches direction when crossing a cell boundary. Quadratic and cubic splines do not suffer from this instability and as a result are largely employed; however, as their stencil covers many more cells and particles, the computational cost increases significantly. Several works have focused on reducing this cost, either from an algorithmic optimization point of view [Fei et al. 2021; Gao et al. 2018; Wang et al. 2020b] or by devising well-behaved basis functions with more

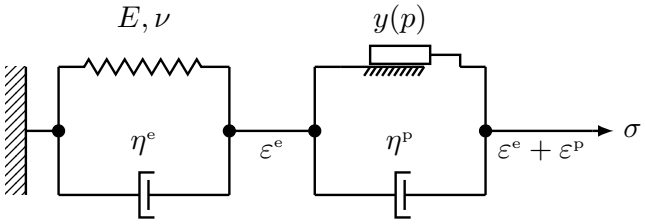


Fig. 3. Rheological diagram of our elastoviscoplastic constitutive model. An elastic potential (E, ν) with damping η^{e} opposes the elastic strain rate $\dot{\varepsilon}^{\mathrm{e}}$, while a viscoplastic flow rule with yield stress $y(p)$ and viscosity η^{p} oppose the plastic strain rate $\dot{\varepsilon}^{\mathrm{p}}$. The Cauchy stress is σ, such that $p = -\operatorname{Tr}\sigma/3$.

compact stencils [Liang et al. 2019; Liu et al. 2025]. Another avenue of research lies in discretizing not only the velocity field on the background grid, but also the strains and stresses; in FEM parlance, using mixed elements; see Figure 2a for an example. On top of broadening discretization options, mixed formulations can more robustly simulate materials near the incompressible limit by employing a compliance-based rather than stiffness-based formulation [Frâncu et al. 2021; Ruan et al. 2024]. Using mixed elements in MPM can be traced back to the Lagrangian Integration Point method of Moresi et al. [2003] for incompressible viscoelastic fluids. Like mixed FEM, mixed MPM is theoretically subject to an inf-sup stability criterion, and stabilized variants have been proposed for geomechanics and poroelasticity [Chandra et al. 2024; Iaconeta et al. 2019; Zhao and Choo 2020], though material heterogeneity complicates analysis. In computer graphics, Daviet and Bertails-Descoubes [2016a] employed a mixed MPM discretization for Drucker–Prager granular materials, using collocated velocity and stress nodes. However, this choice suffers from instabilities in more general settings, in particular for elastic solids (Figure 2b); our method can be seen as a generalization of theirs applicable to a broader class of constitutive models.

3 Constitutive model

Notations. For a 3d symmetric tensor τ, we note $\operatorname{Dev}\tau = \tau - \operatorname{Tr}\tau/3$ its deviatoric part. We define the norm $|\tau|^2 := \tau : \tau/2$ from the double contraction operator $\cdot : \cdot$ so that $|\operatorname{Dev}\tau|^2$ coincides with the second invariant of the deviatoric part of τ, often referred to as J_2 in mechanics. We write $\partial_{,x} f$ the partial derivative w.r.t. x of a function f, or indifferently its subdifferential when f is a nonsmooth convex real-valued function. Derivatives with respect to the time variable t may be abbreviated as $\dot{f}$. We write $\chi_{\mathcal{S}}$ the characteristic function of a closed convex set $\mathcal{S}$,

$$\chi_{\mathcal{S}}(\sigma) = \begin{cases} 0 & \text{if } \sigma \in \mathcal{S}, \\ +\infty & \text{otherwise.} \end{cases}$$

The subdifferential of the characteristic function is useful to write flow rules; indeed, $\partial_{,\sigma}\chi_{\mathcal{S}}(\sigma)$ coincides with the *normal cone* to $\mathcal{S}$ at σ, the set of elements $\dot{\varepsilon}$ such that $\sigma + \alpha\dot{\varepsilon}$, $\alpha > 0$ projects orthogonally back to σ on $\mathcal{S}$.

Material domain. As is traditional in updated Lagrangian settings, we consider a mapping $X^t : \hat{x} \mapsto x^t$ from a rest configuration

Table 1. Summary of parameters used in our model.

Symbol	Unit	Description
ρ	kg m^{-3}	Density
g	m s^{-2}	External acceleration
E	Pa	Young modulus
ν	-	Poisson ratio
η^e/η^p	Pa.s	Elastic/plastic viscosities
p_c	Pa	Compressive yield strength
β	-	Tensile yield ratio
μ	-	Friction coefficient
τ_c	Pa	Shear yield stress
θ	-	Dilatancy coefficient
ϕ_c	-	Critical fraction
ξ	-	Hardening factor
ζ_+/ζ_-	-	Softening/hardening rates

$\hat{x} \in \widehat{\Omega}^t$ to the current configuration at time t, $x^t \in \Omega^t$, and define the *deformation gradient* as $F^t := \partial_{,\hat{x}} X^t$. The velocity field is defined on Ω^t as $u^t := \dot{x}^t$, and we write $\nabla^t u := \partial_{,x^t} u$ its gradient with respect to the configuration at time t. The deformation gradient is then updated over a finite timestep Δ_t as $F^{t+\Delta_t} = F^t + \Delta_t \nabla^t u^{t+\Delta_t} F^t$. To lighten notations, when referring to the current (end-of-timestep) configuration, we will from now on drop the time superscript.

3.1 Strain rate decomposition

We decompose the velocity gradient ∇u as a symmetric strain rate tensor $\dot{\varepsilon}$ and a skew part ω, $\nabla u = \dot{\varepsilon} + \omega$. The symmetric strain rate is furthermore decomposed into *elastic* and *plastic* parts, $\dot{\varepsilon} := \dot{\varepsilon}^e + \dot{\varepsilon}^p$, such that the elastic part of the deformation gradient — the part that affects the elastic energy — is updated as $F^e = F^{e\,t} + \Delta_t \left(\dot{\varepsilon}^e + \omega\right) F^{e\,t}$. The remaining plastic flow rate $\dot{\varepsilon}^p$ is subject to a viscoplastic flow rule, as illustrated in Figure 3.

As remarked by Smith et al. [2019], many isotropic elastic energies can be written on invariants of S, the symmetric factor of the polar decomposition $F^e = RS$. At the first order in Δ_t, we may ignore the dependency of S on ω ($\mathbb{I} + \Delta_t \omega$ is a rotation up to $O(\Delta_t^2 \omega^2)$ terms), so that the end-of-timestep S can be conveniently approximated as a function of the symmetric elastic strain rate $\dot{\varepsilon}^e$ only,

$$S : \dot{\varepsilon}^e \mapsto S^t + \Delta_t \left(\partial_{,F^e} S : \left(\dot{\varepsilon}^e F^{e\,t}\right)\right). \tag{1}$$

3.2 Conservation equations

We now consider the incremental problem for a timestep Δ_t, which consists in the minimization of three potentials: a convex continuously differentiable kinetic potential $\mathcal{I}(u)$, a strongly convex elastic potential $\mathcal{E}_F(F^e)$ that we immediately rewrite using Eq. (1) as $\mathcal{E}(\dot{\varepsilon}^e)$, and in the case of an associated flow rule, a convex, lower semi-continuous plastic dissipation potential $\mathcal{P}(\dot{\varepsilon}^p)$ [Halphen and Son Nguyen 1975; Moreau 1970]. The velocity field is assumed to be at least square integrable with square-integrable derivatives, while the tensor fields are assumed at least square-integrable. The incremental problem reads

$$\min_{\dot{\varepsilon}^e + \dot{\varepsilon}^p = D(u)} \mathcal{I}(u) + \mathcal{E}(\dot{\varepsilon}^e) + \mathcal{P}(\dot{\varepsilon}^p), \tag{2}$$

with $D(u) := \frac{1}{2}\left(\nabla u + (\nabla u)^T\right) = \dot{\varepsilon}$. We note that these potentials may contain viscous terms, if desired.

We now leverage convex optimization tools to derive our continuous mixed formulation; concrete expressions for our final system will be given in the following sections. Introducing a symmetric, square integrable Lagrange multiplier tensor field σ, the constrained optimization problem (2) can be recast as a saddle point problem,

$$\max_{\sigma} \min_{\dot{\varepsilon}^e, \dot{\varepsilon}^p, u} \mathcal{I}(u) + \mathcal{E}(\dot{\varepsilon}^e) + \mathcal{P}(\dot{\varepsilon}^p) + \langle \sigma, D(u) - \dot{\varepsilon}^e - \dot{\varepsilon}^p \rangle .$$

Reordering, and recognizing the expression of the convex conjugate of a function f, $f^*(\sigma) := -\min_\tau \left(f(\tau) - \langle \sigma, \tau \rangle\right)$, problem (2) becomes

$$\min_u \max_\sigma \mathcal{I}(u) - \mathcal{E}^*(\sigma) - \mathcal{P}^*(\sigma) + \langle \sigma, D(u) \rangle ,$$

whose optimality conditions read

$$\partial_{,u} \mathcal{I} + \langle \sigma, D(\cdot) \rangle = 0 \tag{3}$$

$$D(u) - \partial_{,\sigma} \mathcal{E}^*(\sigma) \in \partial_{,\sigma} \mathcal{P}^*(\sigma). \tag{4}$$

Here, we have exploited the fact that the convex conjugate of a strongly convex function is smooth [Parikh and Boyd 2013], so that the subdifferential of $\mathcal{E}^*$ reduces to its gradient; in practice, this translates to the strain-stress mapping being locally invertible. We recognize Eq. (3) as the conservation of momentum, with σ being the Cauchy stress; indeed, for any velocity field v satisfying Neumann boundary conditions, $\langle \sigma, D(v) \rangle = \int_\Omega \sigma : D(v) = -\int_\Omega v^T \nabla \cdot \sigma$. In turn, Eq. (4) is the associated flow-rule; we can identify $\partial_{,\sigma} \mathcal{E}^*(\sigma)$ as $\dot{\varepsilon}^e$, so that $\dot{\varepsilon}^p \in \partial_{,\sigma} \mathcal{P}^*(\sigma)$. It would be usual to define $\mathcal{P}^*$ as the characteristic function $\chi_{\mathcal{S}}$ of the admissible stress set $\mathcal{S}$. Intuitively, the condition $\dot{\varepsilon}^p \in \partial_{,\sigma} \mathcal{P}^*(\sigma)$ means that either σ is in the interior of $\mathcal{S}$, and there is no plastic flow; or σ is on the boundary of $\mathcal{S}$, and the plastic flow rate $\dot{\varepsilon}^p$ is along the normal of $\mathcal{S}$ at that point (belongs to the normal cone if the boundary is nonsmooth).

Non-associated flow rule. The above derivations assume an associated flow rule, which is not always desired. In the case of Drucker–Prager plasticity, the associated flow rule implies that $\dot{\varepsilon}^p$ and σ are always orthogonal, so that no energy is dissipated from plastic flow. Visually, associated flow rules may have undesirable effects on the material volume, as argued by Wolper et al. [2019]. Unfortunately, non-associated flow rules cannot be expressed as a pure minimization problem like Eq. (2), due to a supplemental coupling term between the stress and the plastic flow rate. However, de Saxcé and Feng [1998] show that it is sufficient to replace the plastic potential $\mathcal{P}$ with a bi-potential $\mathcal{B}$, and replace inclusion (4) with

$$D(u) - \partial_{,\sigma} \mathcal{E}^*(\sigma) = \dot{\varepsilon}^p \in \partial_{,\sigma} \mathcal{B}(\sigma, \dot{\varepsilon}^p). \tag{5}$$

In particular, the associated flow rule can be recovered by picking $\mathcal{B}(\sigma, \dot{\varepsilon}^p) = \mathcal{P}^*(\sigma) + \mathcal{P}(\dot{\varepsilon}^p)$. Note that in contrast to many MPM works writing the flow rule on e.g. the Hencky strain and Kirchoff stress pairs, Eq. (5) involves the Cauchy stress and rate of plastic deformation tensor pair. Our elastic potential can be expressed on any finite strain measure, though.

We will now strive to make this framework less abstract by picking concrete formulas for our (bi-)potentials. For reference, the full list of model parameters introduced in the remainder of this section is summarized in Table 1.

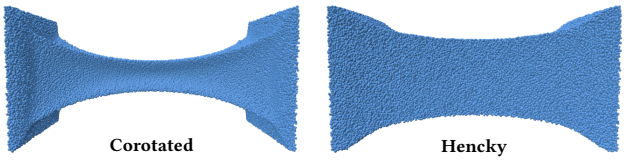


Fig. 4. The corotated elastic model (left) fails to conserve volume under large strains, here for an elastic cube with $\nu = 0.5$ stretched by 2×. In this scenario, the Hencky strain measure (right) performs much better.

3.3 Kinetic and elastic potentials

We express our kinetic and elastic potentials as integrals of energy densities, $\mathcal{I}(u) := \int_\Omega \iota(u)$, $\mathcal{E}(\dot{\epsilon}^e) := \int_\Omega \epsilon(\dot{\epsilon}^e)$. The former lumps inertial and external acceleration terms, in its simplest form, $\iota(u) := \rho/2\Delta_t(u - u^\star)^2 - \rho < u, g >$, where $u^\star$ is the forward-advected velocity and g is the gravity acceleration.

As mentioned in Section 3.1, we express the elastic energy density as a function of the symmetric $S = R^T F^e$, with RS the polar decomposition of the elastic deformation gradient F^e, i.e, $\epsilon(\dot{\epsilon}^e) := \mathcal{E}_S(S(\dot{\epsilon}^e))$ with S the affine function of $\dot{\epsilon}^e$ from Eq. (1). Though any convex hyperelastic potentials written on invariants of S could be considered, in our implementation we use corotated elasticity,

$$\epsilon_S(S) = \frac{E}{2(1+\nu)}\left((S-\mathbb{I}) : (S-\mathbb{I}) + \frac{\nu}{1-2\nu}(\mathrm{Tr}(S-\mathbb{I}))^2\right), \quad (6)$$

with E and ν the material's Young modulus and Poisson ratio. In our mixed formulation however, we are concerned with the convex conjugate of the elastic potential, $\mathcal{E}^*$. Under reasonable summability conditions, we can write the convex conjugate of the functional integral as the integral of the functional's convex conjugate [Rockafellar 1968], i.e, $\mathcal{E}^*(\sigma) = \int_\Omega \epsilon^*(\sigma)$. Expressing the convex conjugate for an affine map, we have

$$\epsilon^*(\sigma) = \epsilon_S \circ S^*(\sigma) = {\epsilon_S}^*\left(\left(\left(\partial_{,F^e S}\right)^{-T} : \frac{\sigma}{\Delta_t}\right)\left(F^{et}\right)^{-T}\right) - S^t : \left(\left(\partial_{,F^e S}\right)^{-T} : \frac{\sigma}{\Delta_t}\right)\left(F^{et}\right)^{-T},$$

where, for the co-rotated elastic model,

$${\epsilon_S}^*(\sigma) = \frac{1+\nu}{2E}\sigma : \sigma - \frac{\nu}{2E}(\mathrm{Tr}\,\sigma)^2 + \sigma : \mathbb{I}.$$

Damping. We can model elastic damping by adding a term to our potential operating on $\dot{S} = \left(S - S^t\right)/\Delta_t$. This amounts to replacing our corotated potential with

$$\epsilon_S(S) = \frac{E + \eta^e/\Delta_t}{2(1+\nu)}\left((S-S_\eta) : (S-S_\eta) + \frac{\nu}{1-2\nu}\left(\mathrm{Tr}(S-S_\eta)\right)^2\right),$$

with η^e the viscosity in Pa.s and $S_\eta := \left(E\Delta_t\mathbb{I} + \eta^e S^t\right)/(E\Delta_t + \eta^e)$. The conjugate potential becomes

$${\epsilon_S}^*(\sigma) = \frac{(1+\nu)\,\sigma : \sigma - \nu\,(\mathrm{Tr}\,\sigma)^2}{2\left(E + \eta^e/\Delta_t\right)} + \sigma : S_\eta.$$

Hencky strain measure. As we are using a compliance-based formulation, our approach allows setting the Poisson ratio to the $\nu = 0.5$ limit. However, for corotated elasticity, this does not mean that the volume will be actually conserved, as shown on Figure 4. Indeed, the volume term of the corotated model only enforces $\mathrm{Tr}\,(S - \mathbb{I}) = 0$. For better volume conservation under large strain, an alternative is to use the Hencky strain measure, replacing the $S - \mathbb{I}$ term in Eq. (6) with $\log S$:

$$\epsilon_S(S) = \frac{E}{2(1+\nu)}\left(\log S : \log S + \frac{\nu}{1-2\nu}(\mathrm{Tr}\log S)^2\right).$$

The volume term now enforces $\mathrm{Tr}\log S = 0$, i.e, $\det S = 1$. We write the corresponding conjugate potential in Appendix A.

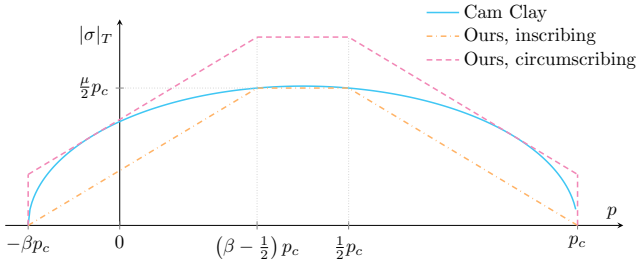


Fig. 5. Comparison of our polygonal yield curve with Cohesive Cam Clay (CCC) from Gaume et al. [2018], for the same friction coefficient μ, compressive strength p_c, and tensile yield ratio β. The circumscribing curve adds a non-zero shear yield stress τ_c so as to be tangent to CCC.

3.4 Unified flow rule

While our framework does not prescribe any particular rheology, in our mixed formulation the flow rule may be enforced at locations that do not coincide with the particles. Since material parameters are stored on particles, they will need to be interpolated to these off-particle locations; this motivates defining a unified flow rule parametrization that can smoothly interpolate between a wide variety of plastic behaviors. Defining such an unified flow rule can also be advantageous for material identification and phase change modeling [Ma et al. 2023; Su et al. 2023].

The Cohesive Cam Clay [Gaume et al. 2018] and Non-Associated Cam Clay [Wolper et al. 2019] are versatile, but approximate poorly the Drucker–Prager yield surface near $p = 0$, where they largely overestimate the friction angle; see Figure 6. Below, we propose an isotropic yield surface, polygonal in the $(p, |\mathrm{Dev}\,\sigma|)$ space, that is a generalization of the Drucker–Prager and Von Mises yield surfaces and can either inscribe or circumscribe the Cam Clay yield surface, as illustrated on Figure 5. We build both associated and non-associated flow rules for this yield surface, with a dilatancy coefficient interpolating smoothly in-between.

Like the Cohesive Cam Clay [Gaume et al. 2018], our yield surface is parametrized with the critical pressure p_c (possibly infinite), a tensile yield ratio β, and a friction coefficient μ. We add a possibly-zero tangential yield stress τ_c. The admissible stress region is then

$$\mathcal{S} := \{|\mathrm{Dev}\,\sigma| \le y(p),\ p = -\mathrm{Tr}\,\sigma/3 \in [-\beta p_c, p_c]\},$$
$$y : p \mapsto \tau_c + \mu \min\left(p + \beta p_c, p_c - p, \frac{p_c}{2}\right). \quad (7)$$

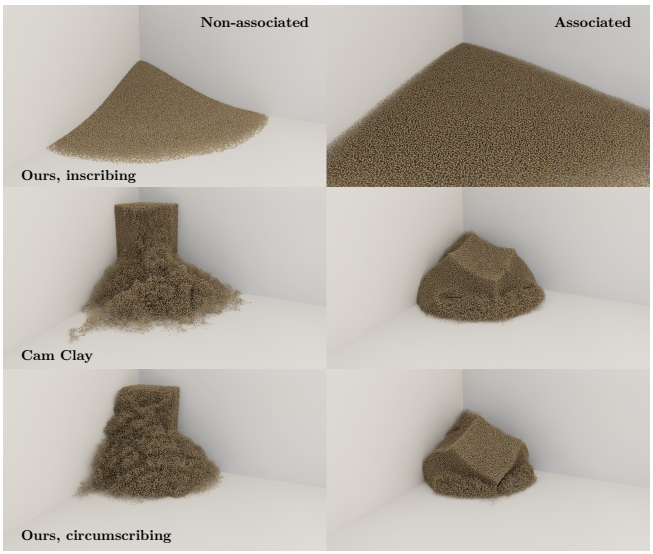


Fig. 6. Reproduction of the notched sand column collapse experiment from Zhu and Bridson [2005], using non-associated (left) and associated (right) flow rules, for different yield surfaces: ours (top); Cam Clay with similar parameters (middle); and a circumscribing variant of our yield surface, with higher friction coefficient μ and yield stress τ_c (bottom).

The associated flow rule is recovered by choosing $\mathcal{B}(\sigma, \dot{\varepsilon}^{\mathrm{p}}) := \chi_{\mathcal{S}}(\sigma) + \chi_{\mathcal{S}}{}^{*}(\dot{\varepsilon}^{\mathrm{p}})$, where[1]

$$\chi_{\mathcal{S}}{}^{*}(\dot{\varepsilon}^{\mathrm{p}}) = p_c \max \begin{pmatrix} \beta \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}, \\ \mu \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right| + \left(\beta - \frac{1}{2}\right) \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}, \\ -\operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}, \\ \mu \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right| - \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}/2 \end{pmatrix} + 2\tau_c \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right|,$$

while a fully non-associated flow rule [2] can be written [Daviet 2016]

$$\mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right) = \chi_{\mathcal{S}}(\sigma) + \max\left(\beta \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}, -\operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}\right) + 2y(p) \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right|.$$

More generally, we can smoothly interpolate between the fully associated and fully non-associated variants of our flow rule by introducing a *dilatancy* coefficient θ,

$$\begin{aligned} \mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right) =& \chi_{\mathcal{S}}(\sigma) + \chi_{\mathcal{S}}^{*,\theta}(\dot{\varepsilon}^{\mathrm{p}}) + 2(1-\theta) \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right| y(p), \\ \chi_{\mathcal{S}}^{*,\theta}(\dot{\varepsilon}^{\mathrm{p}}) :=& \chi_{\mathcal{S}}{}^{*}\left(\frac{\operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}}{3}\mathbb{I} + \theta \operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right). \end{aligned} \tag{8}$$

While Eq. (8) may look unwieldy at first, this expression as a bipotential provides a convenient way to evaluate the residual of the flow rule thanks to the analog of the Fenchel-Young inequality,

$$\sigma \in \partial_{,\dot{\varepsilon}^{\mathrm{p}}} \mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right) \iff \mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right) = \sigma : \dot{\varepsilon}^{\mathrm{p}} \iff \dot{\varepsilon}^{\mathrm{p}} \in \partial_{,\sigma} \mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right).$$

We provide a practical, semi-analytical algorithm for solving this inclusion in Section 5.2.

[1] The convex conjugate of our convex polygon $\mathcal{S}$ reduces to $\max_{\sigma \in \mathcal{S}} \sigma : \dot{\varepsilon}^{\mathrm{p}}$, a linear program for which the extremum will be reached at one of the four vertices with coordinates $(-\beta p_c, \tau_c)$, $(-\beta p_c + p_c/2, \tau_c + \mu p_c/2)$, $(p_c/2, \tau_c + \mu p_c/2)$, (p_c, τ_c) in $(p, |\operatorname{Dev} \sigma|)$ space.

[2] By design, the non-associated flow rule presented here still permits plastic volume changes when p is on the boundary of the admissible interval $[-\beta p_c, p_c]$.

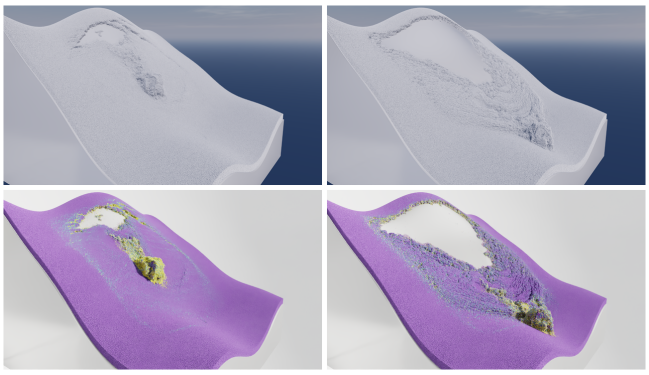


Fig. 7. An avalanche is triggered by a large snow cube hitting an inclined slope featuring cohesive and weak snow layers. The weak layer has a negative hardening rate ζ_-, so that a local compression leads to a long-range collapse. On the bottom row, hue indicates the hardening state J^{p} of the particles, highlighting dilatancy-induced weakening on the avalanche boundaries.

Viscosity. The flow-rule presented above is purely plastic, but we can augment it with another dissipation potential, for instance, $\mathcal{P}_\eta(\dot{\varepsilon}^{\mathrm{p}}) = 2\eta^{\mathrm{p}} \left|\operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}\right|^2$ for a Newtonian viscosity η^{p}. The associated flow rule case becomes

$$\sigma \in \partial_{,\dot{\varepsilon}^{\mathrm{p}}} \left(\chi_{\mathcal{S}}{}^{*} + \mathcal{P}_\eta\right)\left(\dot{\varepsilon}^{\mathrm{p}}\right),$$

which amounts to $\sigma - 2\eta^{\mathrm{p}} \operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}} \in \partial_{,\dot{\varepsilon}^{\mathrm{p}}} \chi_{\mathcal{S}}{}^{*}(\dot{\varepsilon}^{\mathrm{p}})$, i.e, the yield surface only limits the non-viscous part of the Cauchy stress. We can adapt our general dilatancy flow rule in a similar fashion,

$$\dot{\varepsilon}^{\mathrm{p}} \in \partial_{,\sigma} \mathcal{B}\left(\sigma - 2\eta^{\mathrm{p}} \operatorname{Dev} \dot{\varepsilon}^{\mathrm{p}}, \dot{\varepsilon}^{\mathrm{p}}\right).$$

Critical fraction. We may follow Daviet and Bertails-Descoubes [2016b]; Narain et al. [2010] and define a critical fraction ϕ_c such that the material opposes resistance to compression and shear only once this volume fraction is reached, a form of implicit hardening. In terms of mathematical modeling, this simply amounts to shifting $\dot{\varepsilon}^{\mathrm{p}}$ by a constant isotropic tensor, $\dot{\varepsilon}_c^{\mathrm{p}} := \max\left(0, \frac{\phi_c - \phi^t}{3\Delta_t}\right)\mathbb{I}$. As such, we will generally omit this term in the following derivations.

Hardening. Outside of this special case of a critical fraction, we handle hardening in an explicit fashion. The p_c and τ_c parameters are multiplied by a coefficient h following the scaling law from Gaume et al. [2018], $h(J^{\mathrm{p}}) = \sinh -\xi \min(0, \log J^{\mathrm{p}})$. As advocated by Gaume et al. [2018]; Wolper et al. [2019] and leveraged in Figure 7, it is useful to allow J^{p} to deviate from the true volume plastic strain, $\det F^p$, by introducing tunable softening and hardening rate parameters ζ_+ and ζ_-,

$$J^{\mathrm{p}} = \begin{cases} e^{\zeta_+ \Delta_t \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}} J^{\mathrm{p}\,t} & \text{if } \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}} \geq 0, \\ e^{\zeta_- \Delta_t \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}} J^{\mathrm{p}\,t} & \text{if } \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}} < 0. \end{cases} \tag{9}$$

4 Discretization

Now that all terms in our continuous model are properly defined, let us proceed and discretize our constitutive equations (3, 5) using the Material Point Method (MPM).

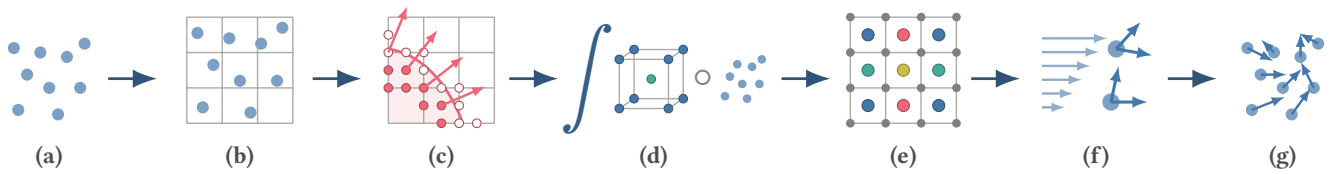


Fig. 8. Overview of a single timestep. We **(a)** start from particle data, which includes positions x_q, velocities v_q optionally augmented with APIC affine information, elastic deformation gradient F^{e}_q, plastic hardening J^{p}_q, and material parameters. We **(b)** build a dynamic sparse grid around the particles, and (c) rasterize colliders' properties onto it (Section 4.4). We pick mixed basis functions (Section 4.2) and **(d)** assemble the discrete system through numerical integration at particles of the variational formulation (10–11). We **(e)** solve the discrete system and flow rules via alternating impulse and stress iterations (Section 5.2). We use the solution to **(f)** update particle elastic and plastic strains, and **(g)** advect particles. For nonlinear materials, steps **(d–f)** may be repeated.

4.1 Particle-based integration

In MPM, the material domain is represented by the volume fraction field ϕ, itself discretized as a sum of Dirac measures located at the particles, $\int_\Omega f = \int \phi f = \sum_q V_q f(x_q)$, where V_q and x_q denote the volume and position of the particle. This allows writing the weak form of our conservation equations as, for all velocity test functions v and symmetric tensor test functions τ of sufficient regularity,

$$a(u,v) + b(v,\sigma) = 0 \qquad \forall v, \tag{10}$$
$$b(u,\tau) - c(\sigma,\tau) = m(\dot{\varepsilon}^{\mathrm{p}},\tau) \qquad \forall \tau, \tag{11}$$
$$\dot{\varepsilon}^{\mathrm{p}} \in \partial_\sigma \mathcal{B}(\sigma,\dot{\varepsilon}^{\mathrm{p}}), \tag{12}$$

where

$$a(u,v) = \int \phi \left\langle \partial_{,u}\iota(u), v\right\rangle = \sum_q V_q \left\langle \partial_{,u}\iota(u)(x_q^t), v(x_q^t)\right\rangle$$
$$b(u,\tau) = \int \phi D(u) : \tau = \sum_q V_q D(u)(x_q^t) : \tau(x_q^t)$$
$$c(\sigma,\tau) = \int \phi \partial_{,\sigma}\epsilon^*(\sigma) : \tau = \sum_q V_q \partial_{,\sigma}{\epsilon_q}^*(\sigma(x_q^t)) : \tau(x_q^t)$$
$$m(\tau,\sigma) = \int \phi \tau : \sigma = \sum_q V_q \sigma(x_q^t) : \tau(x_q^t).$$

For completeness, the necessary derivatives of our kinetic and elastic potentials are given in Appendix A.

Though a, b and m are either bilinear or affine forms, for hyperelastic materials c is nonlinear. In practice, for our co-rotated material model, we find that linearizing c once around the begin-of-timestep elastic deformation gradient $F^{\mathrm{e}t}$ works well; for materials with more drastic nonlinearities, one could perform successive linearizations in a Sequential Quadratic Programming fashion. In any case, the nonlinearity coming from the flow rule will be tackled implicitly by our solver.

At the end of the timestep, the elastic and plastic strain rates $\dot{\varepsilon}^{\mathrm{e}}$ and $\dot{\varepsilon}^{\mathrm{p}}$ are recovered by solving $m(\dot{\varepsilon}^{\mathrm{e}},\tau) = c(\sigma,\tau)\forall\tau$ and from the weak strain rate decomposition, Eq. (11). Then u and $\dot{\varepsilon}^{\mathrm{e}}$ are used to update the particle positions, velocity and deformation gradient as

$$v_q = u(x_q^t),$$
$$F^{\mathrm{e}}_q = \left(\mathbb{I} + \Delta_t \dot{\varepsilon}^{\mathrm{e}}(x_q^t) + \Delta_t \omega_u(x_q^t)\right) F^{\mathrm{e}t}_q, \quad \omega_u := \frac{1}{2}\left(\nabla u - (\nabla u)^T\right)$$
$$x_q = x_q^t + \Delta_t v_q.$$

Finally, we also update the hardening variables from $\dot{\varepsilon}^{\mathrm{p}}$ according to Eq. (9).

As standard with PIC methods, we assemble a lumped version of the mass matrix corresponding to the bilinear form a, and choose a particle-to-grid velocity transfer scheme that compensates the resulting loss of momentum; here we use APIC [Jiang et al. 2015]. An affine approximation of the velocity field is stored at each particle and used to reconstruct the forward-advected grid velocity $u^\star$ at the next timestep. The full timestep pipeline is summarized in Figure 8.

Table 2. Shape functions used for our velocity, stress and impulse spaces.

Name	Description	Nodes
$\mathbf{P_0}$	Piecewise constant	1 per voxel
$\mathbf{Q_1}$	Trilinear, continuous	1 per vertex
$\mathbf{dQ_1}$	Trilinear, discontinuous	8 per voxel
$\mathbf{dP_1}$	Linear, discontinuous	4 per voxel
$\mathbf{B_2}$, $\mathbf{B_3}$	Quadratic and cubic splines	1 per vertex
$\mathbf{S_2}$	Quadratic serendipity	1 per vertex, 1 per edge
Part.	Particle-based (Dirac)	1 per particle

4.2 Basis functions

To complete the discretization of our conservation equations, it remains to define finite spaces for our vector and symmetric tensor fields. At each timestep, we define a sparse, regular background grid with voxel size Δ_x around the particles. Then, we have several options to define basis functions for our fields, with nodes located either on the background grid or on the particles, depending on regularity requirements. We list some of those possibilities in Table 2 and illustrate them in Figure 9, and discuss them below.

Velocities. The velocity field needs to have square integrable derivatives, which requires at least continuity. Quadratic ($\mathbf{B_2}$) or cubic ($\mathbf{B_3}$) splines, which furthermore boast smooth gradients, are

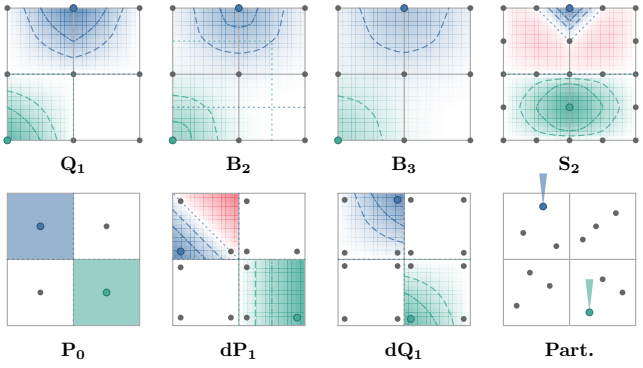


Fig. 9. 2D illustration of shape functions from Table 2. For two selected nodes, we show the basis function value as a color gradient, with isolines at 0.5 (dashed), 0.25, and 0 (dotted). Negative regions are shown in red.

traditionally used in MPM. Linear splines ($\mathbf{Q_1}$), though computationally efficient, are generally avoided due to being prone to grid-crossing instabilities [Jiang et al. 2016; Zhang et al. 2011]. However, the method from Daviet and Bertails-Descoubes [2016b] using $\mathbf{Q_1}$ velocity shape functions demonstrated quantitatively accurate results when compared against physical experiments [Rousseau et al. 2023]. This could be attributed to a few reasons: the use of a continuous ($\mathbf{Q_1}$) stress field; the unilaterally incompressible material, so that particles do not carry an elastic deformation gradient; the proportionality of normal and deviatoric stresses in the Drucker–Prager yield surface. In our experiments, we find that while $\mathbf{Q_1}$ velocities may lead to more noisy stresses (Figure 10), they still produce visually appealing results at a much reduced computational cost compared to wider kernels. We thus use $\mathbf{Q_1}$ velocities in most of our examples.

Stresses and strains. Stresses and strain fields, on the other hand, need only be square integrable; discontinuities are acceptable, which expand our available options. Daviet and Bertails-Descoubes [2016b] used continuous $\mathbf{Q_1}$ strains, chosen for consistency between the void fraction $\phi_c - \phi^t$ discretization and the flow rule enforcement nodes. However, they note that this choice leads to artifacts in the presence of cohesion ($\beta > 0$), which we also observe for elastic solids (Figure 2b). Instead, we use discontinuous elements; on top of particle-based stresses and strains (**Part.**), as traditionally used in MPM, we explore piecewise-constant ($\mathbf{P_0}$), piecewise-linear ($\mathbf{dP_1}$) and piecewise-trilinear ($\mathbf{dQ_1}$). Most of our examples are using the $\mathbf{Q_1}$–$\mathbf{P_0}$ or $\mathbf{Q_1}$–$\mathbf{dP_1}$ pairs, the latter being preferred for large strains that cannot be adequately represented with $\mathbf{P_0}$, as in the cantilever experiment from Figure 11.

4.3 Discrete flow rule

Linear and bilinear forms on symmetric tensor spaces are mapped to $\mathbb{R}^6$ using the isomorphism from Daviet and Bertails-Descoubes [2016a], a variant of the Voigt mapping which has the property of preserving norms and orthogonality for the symmetric tensor double contraction halved product $\frac{1}{2}\cdot:\cdot$ and the usual Euclidean dot product on $\mathbb{R}^6$. In particular, the symmetric tensor invariants map as $\sqrt{6}\widehat{\tau}_\mathrm{N} = \mathrm{Tr}\,\tau$ and $\|\widehat{\tau}_\mathrm{T}\| = |\mathrm{Dev}\,\tau|$, with $\widehat{\tau}_\mathrm{N} \in \mathbb{R}$ and $\widehat{\tau}_\mathrm{T} \in \mathbb{R}^5$ referred to as the *normal* and *tangential* parts of the vector.

With $\widehat{\sigma}$ (resp. $\widehat{\dot{\varepsilon}^\mathrm{p}}$) the image in $\mathbb{R}^6$ of σ (resp. $\dot{\varepsilon}^\mathrm{p}$) through this isomorphism, we can rewrite our bipotential from Eq. (8) as [3]

$$
\begin{aligned}
\widehat{\mathcal{B}}\left(\widehat{\sigma}, \widehat{\dot{\varepsilon}^\mathrm{p}}\right) &:= \chi_{\widehat{\mathcal{S}}}(\widehat{\sigma}) + \chi^{*,\theta}_{\widehat{\mathcal{S}}}(\widehat{\dot{\varepsilon}^\mathrm{p}}) + (1-\theta)\left\|\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{T}\right\| \hat{y}(\widehat{\sigma}_\mathrm{N}), \\
\chi^{*,\theta}_{\widehat{\mathcal{S}}}(\widehat{\dot{\varepsilon}^\mathrm{p}}) &:= \widehat{p_c} \max \begin{pmatrix} \beta \widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{N}, \\ \frac{\hat{\mu}\theta}{2}\left\|\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{T}\right\| + \left(\beta - \frac{1}{2}\right)\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{N}, \\ -\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{N}, \\ \frac{\hat{\mu}\theta}{2}\left\|\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{T}\right\| - \frac{1}{2}\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{N} \end{pmatrix} \\
&\quad + \theta\tau_c \left\|\widehat{\dot{\varepsilon}^\mathrm{p}}_\mathrm{T}\right\|, \\
\widehat{\mathcal{S}} &:= \left\{\|\widehat{\sigma}_\mathrm{T}\| \le \hat{y}(\widehat{\sigma}_\mathrm{N}), -\widehat{\sigma}_\mathrm{N} \in \left[-\beta\widehat{p_c}, \widehat{p_c}\right]\right\}, \\
\hat{y} &: \widehat{\sigma}_\mathrm{N} \mapsto \tau_c + \hat{\mu}\min\left(\beta\widehat{p_c} - \widehat{\sigma}_\mathrm{N}, \widehat{p_c} + \widehat{\sigma}_\mathrm{N}, \widehat{p_c}/2\right).
\end{aligned}
\tag{13}
$$

with the rescaled parameters $\widehat{p_c} := \sqrt{3/2}p_c$ and $\hat{\mu} := \sqrt{2/3}\mu$. We can then express our flow rule on $\mathbb{R}^6$ equivalently as

$$
\widehat{\dot{\varepsilon}^\mathrm{p}} \in \partial_{\widehat{\sigma}}\widehat{\mathcal{B}}\left(\widehat{\sigma}, \widehat{\dot{\varepsilon}^\mathrm{p}}\right) \iff \widehat{\mathcal{B}}\left(\widehat{\sigma}, \widehat{\dot{\varepsilon}^\mathrm{p}}\right) = \widehat{\sigma}^T\widehat{\dot{\varepsilon}^\mathrm{p}} \iff \widehat{\sigma} \in \partial_{\widehat{\dot{\varepsilon}^\mathrm{p}}}\widehat{\mathcal{B}}\left(\widehat{\sigma}, \widehat{\dot{\varepsilon}^\mathrm{p}}\right).
$$

To lighten notations, we will drop the $\widehat{\text{hat}}$ on the vector-related quantities from now on.

Constraint nodes. As it is not possible to ensure that the non-smooth flow rule (5) holds everywhere using a discrete basis space, we resort to enforcing it at a finite number of points.

Assembling the matrices and vectors corresponding to the linearization of the forms a, b, c and m from Eqs. (10–11) as $A\mathbf{u} - \mathbf{f}$, $B\mathbf{u}$, $C\sigma + \mathbf{c}$ and $M\dot{\varepsilon}^\mathrm{p}$, respectively, the discrete system can be written as

$$
\begin{aligned}
A\mathbf{u} + B^T\sigma &= \mathbf{f} \\
B\mathbf{u} - C\sigma - \mathbf{c} &= M\dot{\varepsilon}^\mathrm{p}.
\end{aligned}
$$

The flow rule is written at a set of nodes $(\widetilde{\sigma_i}, \widetilde{\dot{\varepsilon}^\mathrm{p}})$ derived from the function space nodes $(\sigma_j, \dot{\varepsilon}^\mathrm{p}_j)$ through a to-be-chosen linear operator E, as $\widetilde{\dot{\varepsilon}^\mathrm{p}} = E\dot{\varepsilon}^\mathrm{p}$ and $\widetilde{\sigma} = E\sigma$. More compactly, this writes

$$
\underbrace{E^{-1}ME}_{\Lambda}\widetilde{\dot{\varepsilon}^\mathrm{p}} = -\underbrace{E^{-1}(BA^{-1}B^T + C)E}_{W}\widetilde{\sigma} + E^{-1}\left(BA^{-1}\mathbf{f} - \mathbf{c}\right)
$$

To preserve the symmetry of the system, we want Λ and W to commute. For full elements and discontinuous shape functions, Daviet and Bertails-Descoubes [2016a] recommend enforcing the flow rule at the Gauss–Legendre quadrature points; then Λ becomes a diagonal matrix, with constant coefficient per node, and as the flow rule is invariant to a positive scaling on $\dot{\varepsilon}^\mathrm{p}$, it can be eliminated altogether. Similarly, we shall aim to make Λ diagonal. For $\mathbf{P_0}$ and **Part.** stress basis functions, M is already diagonal; we pick $E = \mathbb{I}$ and enforce the flow rule per-element and per-particle, respectively. For discontinuous strain basis functions, M is block-diagonal, with symmetric, positive semi-definite blocks of size 4×4 for $\mathbf{dP_1}$, 8×8 for $\mathbf{dQ_1}$. It is natural and efficient to compute an eigenvalue decomposition $M = P\Lambda P^T$, and choose $E = P$. However, doing this naively will not work; our flow-rule is oriented — e.g, in Drucker–Prager case, enforcing positive divergence — while the orientation

[3] The bipotential from Eq. (8) was written using the dot product $\cdot:\cdot$, while the euclidean $\mathbb{R}^6$ dot product is isometric to $\cdot:\cdot/2$, leading to some scaling factor changes.

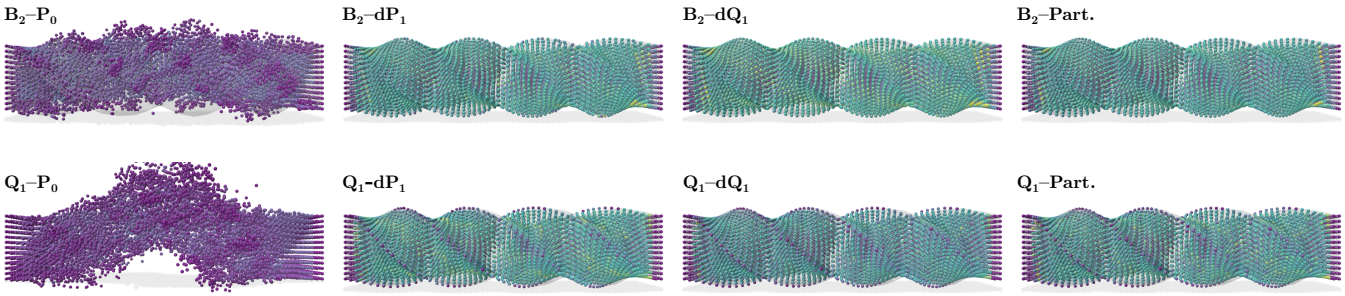


Fig. 10. A 5m×1m×1m elastic beam is subject to a 360 degrees twist of one of its two clamped ends. The beam is discretized with MPM using a voxel size $\Delta_x = 25$cm, and various pairs of velocity-stress shape function. Hue indicates the deviatoric stress magnitude interpolated back on particles. The corresponding FEM ground-truth equilibrium shape is visualized in gray.

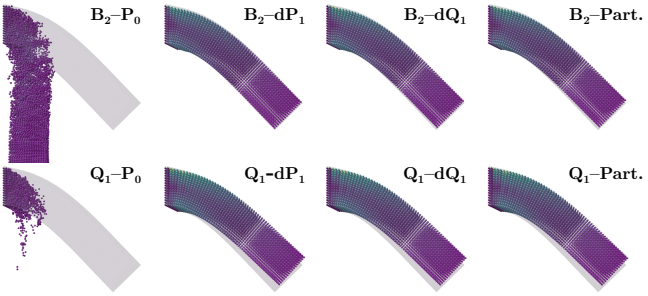


Fig. 11. A 1m×25cm×25cm elastic beam clamped at one end sags under gravity. The beam is discretized with MPM using a voxel size $\Delta_x = 5$cm, and various pairs of velocity-stress shape function. Hue indicates the deviatoric stress magnitude interpolated back on particles. The corresponding FEM ground-truth equilibrium shape is visualized in gray.

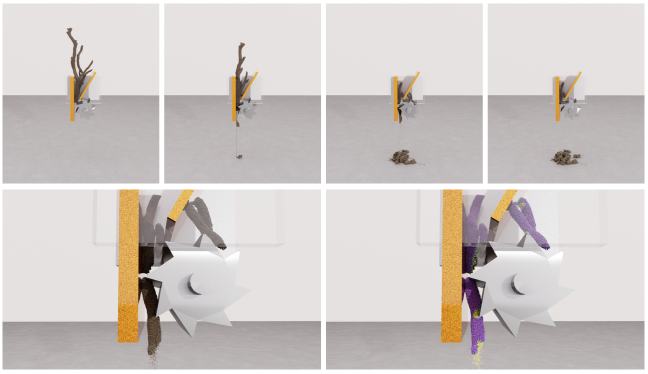


Fig. 12. Subgrid frictional contact enforcement is necessary to account for the fine interactions between the wood shredder teeth and the branches.

of the eigenvectors of P is arbitrary. For consistency, we make sure to pick the vectors of E such that they have a positive dot product with the vector associated to the linear form $\langle 1, \cdot \rangle$. That way, enforcing the divergence to be positive at our nodes $\operatorname{Tr} \widetilde{\dot{\varepsilon}^{\mathrm{p}}}_i \geq 0$ ensures that the volume-averaged divergence over the particles, $\langle 1, \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}} \rangle = \sum_q V_q \operatorname{Tr} \dot{\varepsilon}^{\mathrm{p}}(x_q)$, will be positive as well.

4.4 Collisions

Up until now we have discussed the behavior of our material in isolation; however, most interesting scenarios will involve interactions with other objects, either static, kinematically animated, or fully dynamic.

Frictional contact constraints . We handle collisions against external objects discretely, by enforcing the Signorini–Coulomb conditions at a finite number of points. Choosing the grid vertices, i.e, $\mathbf{Q_1}$ nodes as those points is the simplest, but insufficient for sharp regions where the collider normal changes quickly, as for the shredder teeth in Figure 12. We can also collocate collision constraints with the particles (**Part.**); this is most accurate, but may converge more slowly due to redundancies in the collision constraints. For trilinear velocities, we find that picking the nodes of the quadratic serendipity shape functions ($\mathbf{S_2}$), i.e, the nodes of the grids and the middle point of each edge, yields good accuracy with great performance, each contact interpolation stencil involving at most two velocity nodes.

At each collision node $\mathbf{x}$, we rasterize the collider velocity $\mathbf{v}$, the collider normal $\mathbf{n}$, the collider mass $\mathbf{m}$ associated to the point (∞ in the case of a kinematic collider), and the local friction coefficient μ. The relative collision velocity is computed as $\mathbf{w} := Gu - \mathbf{v}$, with the interpolation Jacobian $G : u \mapsto u(\mathbf{x})$. We then define an impulse $\mathbf{r}$ that should satisfy the Signorini–Coulomb conditions with $\mathbf{w}$, stated compactly as [de Saxcé and Feng 1998]

$$\mathbf{w} + \mu \|\mathbf{w}_{\mathrm{T}}\| \mathbf{n} \in -\partial_{,\mathbf{r}} \chi_{\mathcal{K}_\mu}(\mathbf{r}), \; \mathcal{K}_\mu := \{\|\mathbf{r}_{\mathrm{T}}\| \leq \mu \mathbf{r}_{\mathrm{N}}\} . \tag{14}$$

Two-way rigid-body coupling. The above is sufficient for static or kinematically animated boundary conditions. However, for stable two-way coupling, we need to account for the motion of the collider under the constraint impulse $\mathbf{r}$. For collisions against rigid bodies, we assemble the Jacobian J mapping the impulse at each point to forces and torques at the center of mass of the colliding body, and the block-diagonal matrix I combining the per-body mass and inertia matrices. We then define a *rigidity* Delassus operator as $R = J^T I^{-1} J$,

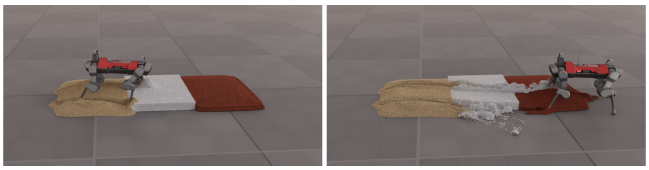


Fig. 13. An ANYmal quadruped walks over a complex terrain with sand, snow and clay according to a pretrained locomotion policy. Two-way coupling between the articulated rigid body solver and MPM terrain forces the robot to change its stride as it progresses through the various materials.

which maps an impulse $\mathbf{r}$ at collision points to a rigid velocity delta at those same points, taking into account the propagation of the impulses over the rigid body. For collision nodes that do not belong to a rigid body, the rigidity operator boils down to the inverse node mass $\mathrm{diag}(\mathbf{m})^{-1}$.

For independent rigid bodies, this approach yields implicit (strong) coupling; we simply pass the final forces and torques to the rigid body solver. For articulated or actuated assemblies as in Figure 13, exact strong coupling would require using the full rigid-body stiffness matrix in lieu of the block-diagonal inertia I, which would significantly increase the computational cost and implementation complexity. In practice, we find that using a block-diagonal approximation lumping the inter-body coupling terms yields a good stability–performance ratio; see Algorithm 6 for the full coupled step. In the limit where one picks $I = \mathrm{diag}(+\infty)$, we recover the "weak" coupling scheme, i.e, co-simulation.

Algorithm 1: Outline of numerical solve for system (15)

```
// Apply impulse and stress warmstart
u ← u + A^{-1} (G^T r + B^T σ);
v ← v − RΔr;
repeat
    Perform impulse solver iteration (Algorithm 2) ;
    Perform stress solver iteration (Algorithm 3) ;
until increment below threshold or iterations exceeded;
```

5 Numerical solve

Our incremental problem is now fully discretized and reads, combining constitutive equations with collision constraints,

$$\begin{aligned} A\mathbf{u} + B^T\sigma &= \mathbf{f} + G^T\mathbf{r} \\ \Lambda\dot{\varepsilon}^{\mathrm{p}} &= B\mathbf{u} - C\sigma - \mathbf{c} \\ \mathbf{w} &= G\mathbf{u} - \mathbf{v} + R\mathbf{r} \\ \dot{\varepsilon}^{\mathrm{p}} &\in \partial_{\sigma}\mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right) \\ \mathbf{w} + \mu\left\|\mathbf{w}_{\mathrm{T}}\right\|\mathbf{n} &\in -\partial_{\mathbf{r}}\chi_{\mathcal{K}_{\mu}}(\mathbf{r}). \end{aligned} \tag{15}$$

with A and Λ diagonal with constant coefficient per node.

The discrete system (15) is actually fully determined by two unknowns, the stress σ and the collision impulses $\mathbf{r}$. We proceed to solve the system in an iterative fashion, alternating between iterations updating $\mathbf{r}$, and iterations updating σ; in both cases, this results in an update of the velocity as $\Delta\mathbf{u} = -A^{-1}B^T\Delta\sigma$ and $\Delta\mathbf{u} = A^{-1}G^T\Delta\mathbf{r}$, respectively. We repeat these alternating solves until a fixed point, or a maximum number of iterations, is reached. Our full solver is outlined in Algorithm 1.

Algorithm 2: Jacobi impulse solver

```
// Once at solver initialization
foreach collision node i in parallel do
    /* Delassus operator diagonal with mass splitting:
       divide mass of velocity node j by number of
       non-zeros in row j of G                          */
    w_i ← 1/m_i + Σ_j G_ij^2 nnz(G_j.)/A_jj ;
end
// At each iteration
w ← Gu − v ;
foreach collision node i in parallel do
    // Isotropic local contact solve [Daviet 2020]
    w̃ ← w_i − w_i r_i ;
    if w̃_N < 0 then
        if ‖w̃_T‖ ≤ −μ w̃_N then // Sticking
            w̃ ← 0 ;
        else // Sliding
            w̃_T ← w̃_T (1 + μ w̃_N/‖w̃_T‖) ;
            w̃_N ← 0 ;
        end
    Δr_i ← (w̃ − w_i) /w_i
end
r ← r + Δr ;
// Update material and collider velocities
u ← u + A^{-1} G^T Δr ;
v ← v − RΔr;
```

5.1 Impulse solve

We leverage operator splitting for the impulse solve, using the diagonal approximation $\mathrm{diag}(\mathbf{m})$ instead of the full rigidity operator R for the implicit update,

$$\mathbf{w}^{k+1} = G\mathbf{u} - \mathbf{v}^{k+1} + \mathrm{diag}(\mathbf{m})\mathbf{r} + \left(R - \mathrm{diag}(\mathbf{m})\right)\mathbf{r}^k.$$

The Signorini–Coulomb conditions are solved using a Jacobi iteration with mass-splitting [Daviet 2023; Tonge et al. 2012]. Since the interpolation Jacobian G, and the mass matrices A and $\mathrm{diag}(\mathbf{m})$ only contain scalar blocks (scaled 3×3 identity matrices), we can use the isotropic local contact solver from Daviet [2020]. The precise iteration is listed in Algorithm 2. We note that for $\mathbf{Q_1}$ collision nodes, G is the identity matrix, meaning that all contacts are independent and the Jacobi iteration is exact.

5.2 Stress solve

For the stress solve, performing a Jacobi iteration with mass splitting is also an option. However, while impulses live only on the collision boundary, stresses need to propagate much farther, possibly through the whole material domain for solids. As such, the faster convergence from a Gauss–Seidel iteration is generally appealing. We use a grid-based coloring scheme so that independent nodes can be solved in parallel while maintaining determinism. If determinism is not a requirement, atomic-based versions would also be possible

Algorithm 3: Colored Gauss–Seidel stress solver

```
// Once at solver initialization
Data: v ← 10^{-6} Δ_x^3
foreach stress node i in parallel do
    // Delassus operator diagonal
    W ← C_ii + Σ_j B_ij B_ij^T / A_jj + v𝕀 ;
    Q_i, D_i ← eigenvalue decomposition of W_T ;
end
// At each iteration
foreach color c do
    foreach stress node i of color c in parallel do
        ė_i^p ← Q_i^T ( Σ_j B_ij u_j − Σ_j C_ij σ_j − c_i ) ;
        b_i ← ė_i^p + D_i Q_i^T σ_i ;
        D_η ← η^p D_iT / Λ_i + 1 ;
        ẽ^p ← result of Algorithm 4 at D_η^{-1} D_i, D_η^{-1} b_i ;
        Δσ_i ← Λ_i Q_i D_i^{-1} ( ė_i^p − ẽ^p ) ;
        σ_i ← σ_i + Δσ_i ;
        u_j ← u_j − A_jj^{-1} B_ij^T Δσ_i   ∀ j ;
    end
end
```

and possibly faster; we have not explored those. For $\mathbf{Q_1}$ velocities and discontinuous stresses, we need 8 colors (all strain nodes belonging to the same voxel are solved sequentially). For $\mathbf{B_2}$ or $\mathbf{B_3}$ velocities, we need 64 colors, so that colored Gauss–Seidel becomes less competitive to Jacobi — unless the grid is large enough for each color to saturate the GPU by itself.

At each iteration $k+1$ of the stress solve and for each node i, the trial plastic strain $\dot{\varepsilon}_i^{\mathrm{p}}$ and stress σ_i are implicitly related through

$$\Lambda_i \dot{\varepsilon}_i^{\mathrm{p}} = -W_i \sigma_i + b_i^k, \quad b_i^k := \sum_j B_{ij} u_j^k - c_i - \sum_j C_{ij} \sigma_j^k + W_i \sigma_i^k,$$

with b_i^k a constant offset depending on the previous iterate. The most natural is to pick W_i as the diagonal block of the Delassus operator; however, the choice of W_i affects the convergence, but not the final result of the iterative Jacobi or Gauss-Seidel, and alternatives can be considered [Erleben 2017]. Unlike for the impulse solve, diagonal blocks of the stress Delassus operator may be arbitrary 6×6 matrices. While a local solver working with arbitrary W_i could be conceived, it would likely require expensive 6×6 matrix operations, similar to the Fischer–Burmeister solver from Daviet and Bertails-Descoubes [2016b] for the special case of the Drucker–Prager rheology. On the other hand, picking W_i as a scalar matrix in order to use an isotropic local solver will negatively impact convergence. Here, we choose a middle-ground; we discard the normal–tangential coupling mode from the Delassus diagonal block, and perform an eigenvalue decomposition to diagonalize the 5×5 tangential block. Since our flow rule is isotropic in the tangential direction, we can perform the solve in this diagonalized tangential basis. Note that this decomposition is performed just once in the solver initialization; see Algorithm 3.

Concretely, we start with $\widehat{W} = \mathrm{diag}\left(BA^{-1}B^T + C\right) + v\mathbb{I}$, the diagonal block of the matrix to which we add a small amount $v \sim 10^{-6}$ of proximal regularization to avoid numerical difficulties. We then perform the symmetric eigendecomposition of the 5×5 bottom-left tangential part as $\widehat{W}_{\mathrm{TT}} = Q_{\mathrm{TT}} D_{\mathrm{T}} Q_{\mathrm{TT}}^T$. We complete the normal components of D as $D_{\mathrm{N}} = \widehat{W}_{\mathrm{N}}$, $Q_{\mathrm{N}} = 1$, and set the 1×5 top-right Q_{NT} and 5×1 bottom-left Q_{TN} parts to zero. With $\widehat{\dot{\varepsilon}}^{\mathrm{p}} = Q^T \dot{\varepsilon}^{\mathrm{p}}$, $\widehat{\sigma} = Q^T \sigma$, we can rewrite our local problem as

$$\Lambda_i \widehat{\dot{\varepsilon}}_i^{\mathrm{p}} = -D_i \widehat{\sigma}_i + Q^T b_i^k, \ \widehat{\dot{\varepsilon}}^{\mathrm{p}} \in \partial_{,\sigma} \mathcal{B}\left(\widehat{\sigma}, \widehat{\dot{\varepsilon}}^{\mathrm{p}}\right), \tag{16}$$

for which we derive an efficient semi-analytic solver in the next section. This new rotation may seem redundant with the one built in Section 4.3; however, the latter was acting over multiple nodes of a cell, with constant coefficients per node; while here Q rotates the tangential components *within* a single node.

We note that structurally, each iteration of our complementarity solver is similar to one of explicit MPM, with a gathering operation (computing b_i^k), a flow rule solve ("return mapping") and a scattering operation (updating the velocity values from the stress delta). However, here the stencils are usually smaller, and remain constant over the course of the solve (there is no "advection" step). We also perform a single 5×5 eigenvalue decomposition per timestep, instead of a 3×3 Singular Value Decomposition per particle in classical MPM return mapping implementations. Thanks to this structural similarity, it is likely that optimizations devised to accelerate explicit MPM [Wang et al. 2020b] could be applicable to our solver as well, though we have not explored this direction yet.

Solid materials. For solid materials, our problem is linear and we can use Preconditioned Conjugate Gradient (PCG). We use the block diagonal of the stress Delassus operator as our preconditioner, and evaluate the left-hand-side in a matrix-free fashion. Outside of solid materials, the PCG solver can also serve as initial guess for the complementarity Jacobi and Gauss-Seidel solvers.

5.3 Local flow rule solver

The last missing piece of our method is a way of solving the local problem (16), for each strain node and at each iteration of the complementarity (Gauss–Seidel or Jacobi) solver. To lighten notations, let's rewrite it as

$$\dot{\varepsilon}^{\mathrm{p}} = -D\sigma + b \in \partial_\sigma \mathcal{B}\left(\sigma, \dot{\varepsilon}^{\mathrm{p}}\right). \tag{17}$$

Viscosity. Rewriting the local problem (16) as (17), we have dropped Λ. As the flow-rule is invariant under a positive scaling of $\dot{\varepsilon}^{\mathrm{p}}$, this is generally without impact. However, one exception is the case of a non-zero viscosity η^{p}. There, introducing the supplemental stress $\widetilde{\sigma} = \sigma - \eta^{\mathrm{p}} \dot{\varepsilon}_{\mathrm{T}}^{\mathrm{p}}$, the local problem reads

$$\left(\mathbb{I} + \eta^{\mathrm{p}} D_{\mathrm{T}} / \Lambda\right) \Lambda \dot{\varepsilon}^{\mathrm{p}} = -D\widetilde{\sigma} + b, \ \dot{\varepsilon}^{\mathrm{p}} \in \partial_\sigma \mathcal{B}\left(\widetilde{\sigma}, \dot{\varepsilon}^{\mathrm{p}}\right).$$

As $\widetilde{D} := (\mathbb{I} + \eta^{\mathrm{p}} D_{\mathrm{T}} / \Lambda)$ is diagonal and positive definite, $\widetilde{D}^{-1} D$ remains diagonal. Left-multiplying D and b with $\widetilde{D}^{-1}$, the local problem falls back under the $\dot{\varepsilon}^{\mathrm{p}}$-scaling independent form (17).

Non-associated flow rule. To devise a solver for the local problem (17), let us first look at the simpler non-associated case, i.e, $\theta = 0$, where the normal and tangential equations can be fully decoupled; see Figure 14, left. The normal stress can be computed as $\sigma_{\mathrm{N}} = -\,\mathrm{clamp}\left(-b_{\mathrm{N}}/D_{\mathrm{N}}, -\beta p_c, p_c\right)$. Then for the tangential part, we have three cases:

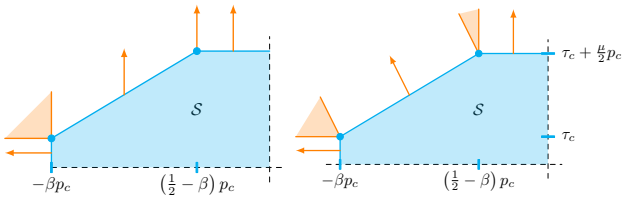


Fig. 14. Acceptable plastic strain rate directions in orange at different points on the boundary of our polygonal admissible stress set $\mathcal{S}$, for non-associated ($\theta = 0$; left) and associated ($\theta = 1$; right) flow rules.

(i) either $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}} = 0$, i.e, zero plastic shear, is a solution; this is the case if σ^0, $\sigma^0 := \left[\sigma_{\mathrm{N}}, D_{\mathrm{T}}^{-1} b_{\mathrm{T}}\right]$, is within the yield surface, i.e, $\left\|\sigma^0_{\mathrm{T}}\right\| \leq y(\sigma^0_{\mathrm{N}})$;
(ii) or $y(\sigma^0_{\mathrm{N}}) = 0$; in this case $\sigma_{\mathrm{T}} = 0$ is a solution;
(iii) or we have $\|\sigma_{\mathrm{T}}\| = y(\sigma^0_{\mathrm{N}})$, and there exists $\alpha > 0$, $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}} = \alpha\sigma_{\mathrm{T}}$ (the maximum dissipation principle). This can be recast as an orthogonal projection onto a 5d axis-aligned ellipsoid (Figure 15, left). We have $\|\sigma_{\mathrm{T}}\| = \left\|(D_{\mathrm{T}} + \alpha\mathbb{I})^{-1} b_{\mathrm{T}}\right\| = y(\sigma_{\mathrm{N}})$, which boils down to a one-dimensional root-finding problem on α, for which we give more details in the associated case below.

Note that this local solver does not make assumptions on y, so that it is applicable to any isotropic non-associated yield surface, such as NACC [Wolper et al. 2019].

Associated flow rule. In the associated or partially associated case, i.e, $\theta > 0$, case (i) is similar; if σ^0 lies in the admissible stress set, then we have a solution with $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}} = 0$. In other cases however, θ introduces an additional coupling between the normal and tangential components; see Figure 14, right. Indeed, changing the direction of σ_{T} changes the magnitude of $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}$, which may affect $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{N}}$ due to dilatancy, and in turn σ_{N} and $y(\sigma_{\mathrm{N}})$. To handle this additional coupling, we assume that the yield function y is piecewise linear, which is true for our flow rule. We note this constant slope $\mu^0 := -\partial_{,\sigma_{\mathrm{N}}} y\left(\sigma^0_{\mathrm{N}}\right)$; for our flow rule, we have $\mu^0 \in \{-\mu, 0, \mu\}$. Case (ii), when $y(\sigma^0_{\mathrm{N}}) = 0$, becomes more restrictive; $\dot{\varepsilon}^{\mathrm{p}} = (D_{\mathrm{N}}\sigma^0_{\mathrm{N}}, 0) + b$ now needs to belong to the second-order cone $\mathcal{K}_{\mu\theta}$ of aperture $\theta\mu^0$ for $\sigma_{\mathrm{T}} = 0$ to be a valid solution. Case (iii) now writes

$$\begin{cases} \alpha\sigma_{\mathrm{T}} = \dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}} = -D_{\mathrm{T}}\sigma_{\mathrm{T}} + b_{\mathrm{T}}, \\ \theta\mu^0 \left\|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}\right\| = \dot{\varepsilon}^{\mathrm{p}}_{\mathrm{N}} = -D_{\mathrm{N}}\sigma_{\mathrm{N}} + b_{\mathrm{N}}, \\ \|\sigma_{\mathrm{T}}\| = y(\sigma^0_{\mathrm{N}}) + \mu^0\left(\sigma^0_{\mathrm{N}} - \sigma_{\mathrm{N}}\right) \end{cases}$$

Combining the last three equations, we have

$$\|\sigma_{\mathrm{T}}\| = \underbrace{y(\sigma^0_{\mathrm{N}}) + \mu^0\left(\sigma^0_{\mathrm{N}} - D_{\mathrm{N}}^{-1} b_{\mathrm{N}}\right)}_{y^0} + \underbrace{\mu^0 D_{\mathrm{N}}^{-1}\theta\mu^0}_{\gamma} \left\|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}\right\|.$$

Finally, plugging-in $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}} = \alpha\sigma_{\mathrm{T}}$, we get

$$g(\alpha) := (1 - \gamma\alpha)\left\|(D_{\mathrm{T}} + \alpha\mathbb{I})^{-1} b_{\mathrm{T}}\right\| = y^0. \tag{18}$$

Once again, this is a one-dimensional root-finding problem on α; the only change compared to the non-associated case is the additional $(1 - \alpha\gamma)$ scaling term.

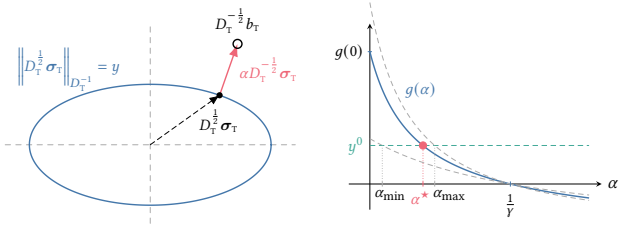


Fig. 15. Left: the non-associated flow rule (17) is equivalent to an orthogonal projection of $D_{\mathrm{T}}^{-1/2} b_{\mathrm{T}}$ onto the 5d ellipsoid $\left\|D_{\mathrm{T}}^{1/2}\sigma_{\mathrm{T}}\right\|_{D_{\mathrm{T}}^{-1}} = y$, which we solve by 1d root finding on strain–stress magnitude ratio α. Right: the monotonically decreasing function $g(\alpha)$ with upper and lower bounds ($\alpha_{\min}$, $\alpha_{\max}$), the target value y^0, and the solution $\alpha^\star$.

Now, the above relied on a linear approximation of y at σ^0_{N}, while in reality y is piecewise linear. It is possible that the result of the solve of Equation (18) updates σ_{N} in a way that it switches to another linear region. In that case, we simply re-apply the non-associated algorithm with σ^0_{N} at the boundary between the initial and final region. Indeed, our yield function alternates regions with $\partial_{,\sigma_{\mathrm{N}}} = 0$ and regions with $\partial_{,\sigma_{\mathrm{N}}} \neq 0$; the dilatancy coefficient only has effect when $\partial_{,\sigma_{\mathrm{N}}} \neq 0$, so the only possible transition is going from a non-zero dilatancy to a null dilatancy region, which will then necessarily be a fixed point for σ_{N}.

Solving for α. We can show (see Appendix B) that $g(\alpha)$ is monotonically decreasing from $g(0) = \left\|\left(D^{-1} b\right)_{\mathrm{T}}\right\|$ to $g(+\infty) = -\gamma\|b_{\mathrm{T}}\|$. We know that $y^0 \leq y(\sigma^0_{\mathrm{N}}) < g(0)$, otherwise case (i) would have been satisfied. Regarding the lower bound, $y^0 \leq g(+\infty)$ is possible when, and only when, $-\sigma^0_{\mathrm{N}} \in \{-\beta p_c, p_c\}$; in all other cases, $\sigma^0_{\mathrm{N}} = b_{\mathrm{N}}/D_{\mathrm{N}}$, so $y^0 = y(-\sigma_{\mathrm{N}}) \geq 0$. Assuming $y^0 \leq -\gamma\|b_{\mathrm{T}}\|$ holds, the solution to the non-associated flow rule is necessarily also a solution to the associated flow rule. Indeed, for such a solution we have $\|b_{\mathrm{T}}\| = \|(I + 1/\alpha D)\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}\| \geq \|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}\|$. It follows that

$$|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{N}}| \geq \left|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{N}} - sD_{\mathrm{N}}/\mu^0\right| = -\frac{y^0 D_{\mathrm{N}}}{|\mu^0|} \geq \frac{\gamma\|b_{\mathrm{T}}\| D_{\mathrm{N}}}{|\mu^0|} \geq |\mu^0|\theta\left\|\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{T}}\right\|,$$

meaning that $\dot{\varepsilon}^{\mathrm{p}}$ belongs to the second-order cone of aperture $\theta\mu^0$.

In the general case, we have $g(0) > y^0 > g(+\infty)$, and our root-finding problem admits a unique solution. We can furthermore leverage the inequality

$$\frac{\|b_{\mathrm{T}}\|}{D_{\max} + \alpha} \leq \left\|(D_{\mathrm{T}} + \alpha\mathbb{I})^{-1} b_{\mathrm{T}}\right\| \leq \frac{\|b_{\mathrm{T}}\|}{D_{\min} + \alpha},$$

with $D_{\min}$ (resp. $D_{\max}$) the minimum (resp. maximum) diagonal coefficient of D, as well as the analytical root of g at $\alpha = 1/\gamma$, to derive tighter bounds $[\alpha_{\min}, \alpha_{\max}]$ for the solution (see Figure 15, right). Note that for an isotropic local problem, i.e, when $D_{\min} = D_{\max}$, this gives an explicit expression for α. Finally, g is strictly convex for $\alpha \in (0, 1/\gamma)$; for $\alpha > 1/\gamma$, a non-convex region may exist for large $D_{\max}/D_{\min}$ ratios. In practice, we find that a one-dimensional Newton root finding algorithm, without linesearch, starting from $\alpha = \alpha_{\min}$, converges reliably. Our complete flow-rule solver is outlined in Algorithm 4.

Algorithm 4: Local solver for the flow rule (17)

Input: D, b, yield parameters p_c, β, θ, y
$\sigma^0 \leftarrow \left(-\text{clamp}\left(-b_N/D_N, -\beta p_c, p_c\right), D_T^{-1} b_T\right)$;
if $\left\|\sigma_T^0\right\| < y(\sigma^0)$ **then** // Zero plastic shear
  return $\left(-D_N\sigma_N^0 + b_N, 0\right)$;
$\mu^0 \leftarrow \partial_{,\sigma_N} y(\sigma_N^0)$;
$p_{\min}, p_{\max} \leftarrow$ bounds of local linear region of y at σ^0 ;
$y^0 \leftarrow y(\sigma^0) - \mu^0\left(\sigma_N^0 - b_N/D_N\right)$;
$\dot{\varepsilon}^p \leftarrow$ Result of Algorithm 5 for y^0, μ^0, θ;
// Check for change of linear region
$\sigma_N \leftarrow \left(b_N - \dot{\varepsilon}_N^p\right)/D_N$;
$\sigma_N^\star \leftarrow \text{clamp}\left(\sigma_N, -p_{\max}, p_{\min}\right)$;
if $\sigma_N^\star \neq \sigma_N$ **then** // Re-solve with non-associated
  $\dot{\varepsilon}^p \leftarrow$ Result of Algorithm 5 for $y^0 = y(\sigma_N^\star)$, $\mu^0 = 0$, $\theta = 0$;
  $\dot{\varepsilon}_N^p \leftarrow -D_N\sigma_N + b_N$;
end
return $\dot{\varepsilon}^p$;

Algorithm 5: Root-finding for the plastic strain rate

Input: D, b, yield stress y^0, derivative μ^0, dilatancy θ
Data: $\varepsilon \leftarrow 10^{-7}$
$\gamma \leftarrow \mu^0\theta\mu^0/D_N$;
$y^\star \leftarrow y^0 + \gamma\left\|b_T\right\|$;
if $y^\star \leq 0$ **then return** b_T; // Defer to non-associated
// Compute α bounds
$\alpha_+, \alpha_- \leftarrow \left(\left\|b_T\right\| - D_{\min} y^0\right)/y^\star, \left(\left\|b_T\right\| - D_{\max} y^0\right)/y^\star$;
$\alpha_0 \leftarrow 1/\gamma$;
if $y^0 \geq 0$ **then** $\alpha_{\min}, \alpha_{\max} \leftarrow \max\left(0, \alpha_-\right), \min\left(\alpha_0, \alpha_+\right)$ **else**
  $\alpha_{\min}, \alpha_{\max} \leftarrow \max\left(\alpha_0, \alpha_+\right), \alpha_-$;
// Root-finding on $g(\alpha) = (1-\gamma\alpha)\left\|\left(D_T + \alpha\mathbb{I}\right)^{-1} b_T\right\|$
$\alpha \leftarrow \alpha_{\min}$;
repeat
  $\delta\alpha \leftarrow -\left(g(\alpha) - y^0\right)/g'(\alpha)$;
  $\alpha \leftarrow \text{clamp}\left(\alpha + \delta\alpha, \alpha_{\min}, \alpha_{\max}\right)$
until $|\delta\alpha| < \varepsilon\alpha_{max}$;
$\dot{\varepsilon}_T^p \leftarrow \alpha\left(D_T + \alpha I\right)^{-1} b_T$;
$\dot{\varepsilon}_N^p \leftarrow \theta\mu^0\left\|\dot{\varepsilon}_T^p\right\|$;
return $\dot{\varepsilon}^p$;

6 Results

We have implemented our method using the NVIDIA Warp [Macklin 2022] language and made it available as part of the Newton physics engine [Newton Contributors 2025]. Integration of linear and bilinear forms of Eq. (10–11), particle-to-grid transfers and general interpolation are performed using the `warp.fem` submodule.

We used a workstation equipped with an Intel® Core™ i9-10980XE CPU and an NVIDIA RTX PRO 6000 Blackwell Max-Q GPU. We render the particle raw output of our simulation without any post-treatment; typical MPM post-processing steps could naturally be applied, such as rasterizing a volumetric density [Stomakhin et al. 2013], extracting mesh surfaces [SideFX 2024; Wolper et al. 2019] or explicitly tracking cracks [Fan et al. 2022].

Table 3. Convergence statistics for our local solvers. The bipotential residual is evaluated as $\left|\mathcal{B}\left(\sigma, \dot{\varepsilon}^p\right) - \sigma^T\dot{\varepsilon}^p\right| / \left(\|b\|\|D^{-1}b\|\right)$. The iteration count percentiles only consider local problems for which the root-finding algorithm is actually employed.

	mean	99%	max
Bipotential residual	$1.2\ 10^{-9}$	$4.0\ 10^{-8}$	$1.8\ 10^{-7}$
Root-finding iterations	6	12	19

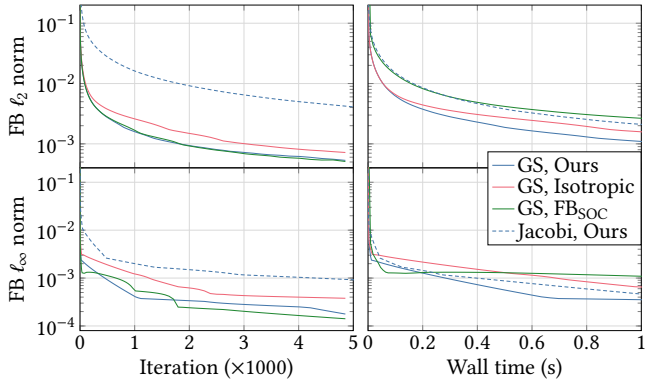


Fig. 16. ℓ_2 and ℓ_∞ residuals reached for different solvers as a function of iteration number and wall time for a Drucker–Prager sand column collapse. "GS, FB$_{SOC}$" is a Gauss–Seidel solver using the nonsmooth Newton on Second-Order-Cone Fischer–Burmeister from Daviet and Bertails-Descoubes [2016a] as the local solver. For consistency, all residuals are evaluated using FB$_{SOC}$.

Implementation details. Formally the matrix B contains a 6×3 dense block for each velocity–stress node pair with overlapping stencils; however, the storage can be compressed as a single 3d vector. Indeed, the bilinear operator $\sum_q D(u_i)(x_q) : \tau_j(x_q)$, is a simple function of the underlying shape function product $\sum_q \nabla N_i^u(x_q) N_j^\tau(x_q)$. This greatly reduces the amount of data that need to be read from global memory in the Gauss–Seidel iterations. Warmstarting of the stress field is handled by grid to particle and particle-to-grid transfers. Empirically, we find that this performs better than interpolating directly from the grid at the previous time step to the current one. We use the ℓ_2 and ℓ_∞ norms of the plastic strain delta as our termination criteria, as well as a maximum number of iterations, typically 10^{-3} and 250, respectively.

6.1 Local solver validation

We start by validating that our local solver (Algorithm 4) actually solves our flow rule. For this, we generate 10^6 random local problems, with normally sampled D and uniformly sampled b and flow rule parameters. Table 3 shows that our algorithm was always able to compute solutions with low residual $\mathcal{B}\left(\sigma, \dot{\varepsilon}^p\right) - \sigma^T\dot{\varepsilon}^p$ in relatively few root-finding iterations.

Then we look at the numerical solve (Algorithm 1) as a whole. We save an instance of the discrete problem (15) and study the performance of different solvers on it. We used a problem from a Drucker–Prager sand column collapse, so we can compare to

Table 4. Average statistics and performance results for our more complex simulations. n_p: number of particles; n_u: number of velocity nodes; n_σ: number of strain nodes; n_{GS}: number of Gauss–Seidel iterations; ℓ_2: ℓ_2 residual $\times 10^{-3}$; FPS: animation frames per second; n_s: simulation substeps per animation frame; t_f: wall time for the computation of a full animation frame. The real-time ratio is thus $t_f \cdot$ FPS.

Name	n_p	n_u	n_σ	n_{GS}	ℓ_2	FPS	n_s	t_f
Rigid Ball	524k	28k	25k	239	1.3	60	4	0.5
Dam Break	679k	29k	99k	115	0.1	60	2	0.1
Viscous	224k	14k	44k	215	0.4	120	2	0.2
ANYmal	1.6M	1.8M	256k	92	1.7	50	4	0.7
Press	6.4M	327k	276k	247	2.1	240	4	0.8
Shredder	444k	37k	90k	221	3.3	120	8	1.4
Avalanche	14M	2.2M	1.9M	149	0.1	30	8	5.4
Sand city	49M	5.8M	5.1M	234	0.4	60	2	4.0

the Fischer–Burmeister solver from Daviet and Bertails-Descoubes [2016b]; our particular instance uses a $\mathbf{Q_1}$–$\mathbf{dP_1}$discretization and features 38k velocity nodes and 90k strain nodes. We show in Figure 16 the ℓ_2 and ℓ_∞ residual of the Second Order Cone Fischer–Burmeister merit function FB_{SOC} for a variety of solvers over iterations and wall time. Our anisotropic local solver with Gauss–Seidel yields consistently good performance. As this instance is too small to saturate our GPU for a single Gauss–Seidel color, we also see good performance of a Jacobi variant, though it lags in ℓ_2 norm minimization. The computational cost per iteration of the nonsmooth Newton from Daviet and Bertails-Descoubes [2016b] is too high to make up for the convergence gains from using full W_i diagonal blocks, compared to normal–tangential diagonalization. Finally, we derive an isotropic variant of our local solver that exploits the fact that $\alpha_{\min} = \alpha_{\max}$ when $D_{\min} = D_{\max}$, so that no root-finding is necessary. In this case, we use $D_{\text{iso}} = \text{diag}(D_{\max})$ (any lower value leads to divergence issues). As most local problems are isotropic — for inelastic materials anisotropy of D is a result of non-uniform particle distribution, so mostly on the boundary of the material — the isotropic local solver performs well on the ℓ_2 norm. However, the few anisotropic problems take longer to converge, penalizing the ℓ_∞ norm. As the cost per iteration is only slightly lower than the anisotropic version — the root-finding usually is not the bottleneck, the isotropic solver overall underperforms.

6.2 Model problems

We now verify the qualitative behavior of our method on various model problems; see the corresponding animations in the accompanying video. Additional physical parameters and performance results are listed in Tables 5 and 4.

Elasticity. We compare the behavior of our method using various mixed elements to ground-truth FEM results on $\mathbf{P_2}$ quadratic tetrahedrons. We setup two experiments: *beam twist* (Figure 10) and *cantilever* (Figure 11). *Beam twist* is a 5m×1m×1m elastic beam with Young modulus E =5MPa and Poisson ratio $\nu = 0.45$, clamped at both ends, subject to a 360-degrees rotation. *Cantilever* is a 1m×25cm×25cm beam with $E = 0.3$MPa and $\nu = 0.45$. Both have a

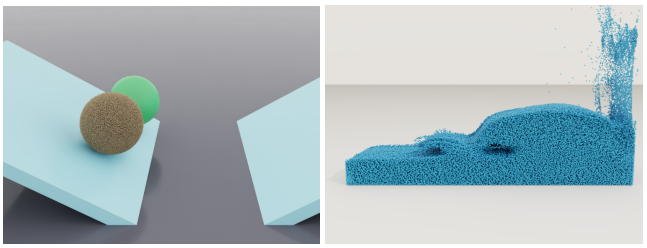


Fig. 17. Left: We discretize a rigid ball with MPM particles and let it roll off an inclined plane, comparing its motion to that of a reduced rigid body (in green) simulated with MuJoCo Warp [Google DeepMind and NVIDIA 2025]. Right: Dam break experiment for an inviscid fluid.

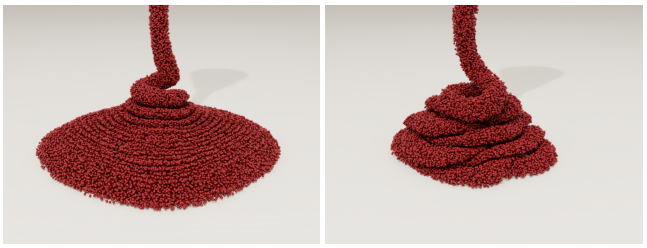


Fig. 18. Viscous fluids with $\eta^{\text{p}} = 50$Pa.s (left) and $\eta^{\text{p}} = 500$Pa.s (right) released from a funnel reservoir form typical coiling patterns.

density of $\rho = 1000$ kg.m^{-3} and are subject to standard earth gravity. We use a yield surface covering the whole stress space ($p_c = \infty$, $\beta = 1$, $\mu > 0$) so as to prevent any plastic flow. We use a Preconditioned Conjugate Gradient solver for the discrete problem with a low tolerance so as to reduce bias due to incomplete convergence.

We observe that the $\mathbf{P_0}$ strain elements break apart under large strains; this is expected, as the piecewise-constant discretization is not able to accurately represent the deformation. Other elements show good agreement with the ground truth, quadratic splines $\mathbf{B_2}$ velocities yielding higher accuracy than trilinear $\mathbf{Q_1}$, as well as smoother stress fields. Regarding the latter, elements using non-conforming linear stresses $\mathbf{dP_1}$ appear slightly noisier, with $\mathbf{dQ_1}$ the smoothest.

Rigid body. While the $\mathbf{P_0}$ element is unable to represent large strains, it is a good candidate for rigid or nearly-rigid bodies. In Figure 17 (left), we show a rigid MPM ball with $\mathbf{Q_1} - \mathbf{P_0}$ element racing against an actual rigid body simulated with Newton's MuJoCo Warp solver [Google DeepMind and NVIDIA 2025]. Our MPM ball loses momentum quicker, in part due to the boundary not being a perfect sphere, and is clearly not the most computationally efficient; however, we still observe the expected rolling behavior.

Incompressible fluids. While our mixed elements with trilinear velocities do not satisfy the theoretical inf-sup stability condition, they are still able to qualitatively reproduce the behavior of incompressible fluids by setting $E = \infty$, $p_c = \infty$, $\beta = 1$ and $\mu = 0$. Figure 17 (right) shows a dam break experiment for an inviscid fluid ($\eta^{\text{p}} = 0$), while Figure 18 demonstrates Newtonian fluid coiling effects [Fang et al. 2019; Larionov et al. 2017].

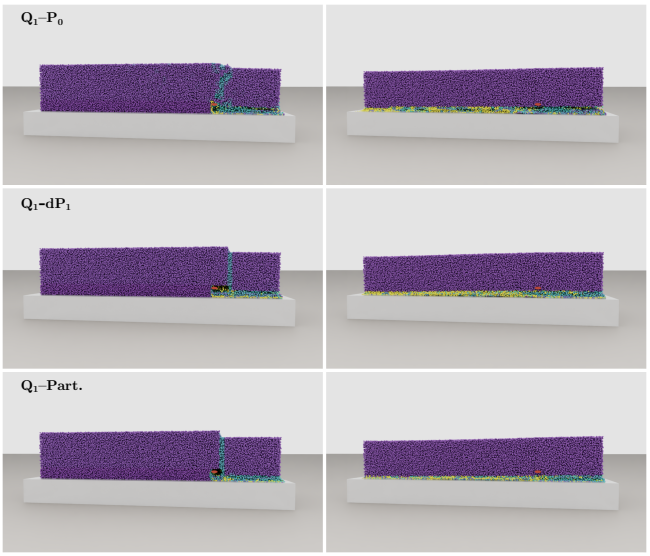


Fig. 19. We compare the behavior of $\mathbf{P_0}$, $\mathbf{dP_1}$ and **Part.** stress basis functions on an experiment inspired by the snow slab fracture tests from Gaume et al. [2018]. A dense 30cm-deep snow layer rests upon a 10cm-deep weak layer that is progressively weakened in the region marked by the red ellipse, mimicking the effect of a snow saw. Once a critical length (about 50cm) is reached, if the dense top slab is not cohesive enough, or if the weak layer is too strong (left), the slab will fracture locally, without further effect. However for different material parameters (right), the cohesive snow will propagate the load leading to a complete collapse of the weak layer. Hue indicates the hardening state J^p.

Notched sand column collapse. We reproduce the notched column collapse from Zhu and Bridson [2005] in Figure 6. We perform this experiment with cohesionless Drucker–Prager ($\beta = 0$, $\mu = 0.68$), the Cam Clay yield surface with the same parameters, and a version of our yield surface that circumscribes Cam Clay (using $\tau_c > 0$ and larger μ). We simulate both a non-associated flow rule ($\theta = 0$) with p_c=100MPa, and an associated flow rule with $\theta = 1$, p_c=10MPa. Even in this cohesionless configuration, Cam Clay cannot reproduce the free-flowing behavior of dry granular materials, while our yield surface, that generalizes Drucker–Prager, is able to. However, for materials for which it is desired, our unified yield surface parametrization is also able to reproduce the qualitative behavior of Cam Clay.

We also explored the use of different mixed elements on this experiment with non-associated Drucker–Prager; while the results are visually similar, the average per-frame time jumps from 106ms for $\mathbf{Q_1 - P_0}$ to 180ms for $\mathbf{Q_1 - dP_1}$, and 13.3s for $\mathbf{B_2 - dP_1}$, which is consistent with 8× more Gauss–Seidel colors and 8× larger stencils. $\mathbf{B_2 - Part.}$ with Jacobi performed at 3.1s per frame.

Snow fracture. For our next model problem, we look at hardening laws, and in particular, whether our stress discretization is able to capture the steep changes in material parameters due to damaged or consolidated material. We take inspiration from the experiments from Gaume et al. [2018] and model a dense snow slab resting on a weak layer, the latter using $\zeta_+ < 0$ so that compression will incur softening. We progressively weaken particles in a region visualized by the red ellipse in Figure 19, mimicking the effect of a snow saw moving from the right-end of the slab to the left — note that we're not using a real collider here. After a critical length is reached, depending on material parameters, either the slab fractures or the weak layer collapses entirely. We observe consistent behavior across $\mathbf{Q_1 - P_0}$, $\mathbf{Q_1 - dP_1}$ and $\mathbf{Q_1 - Part.}$ mixed elements.

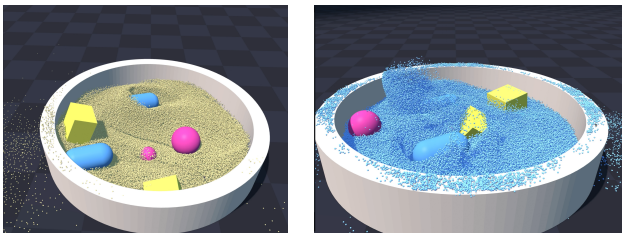


Fig. 20. Screenshots from our interactive sandbox application, where the user can freely interact with the rigid bodies in a two-way coupled simulation containing about 673k MPM particles of a granular material (left) or inviscid fluid (right).

Flow rule parameter exploration. Our flow rule has many parameters, and the influence of each of those may not be intuitive. We do not offer a comprehensive tuning guide to obtain a desired look, but propose in Figure 23 an exploration of the range of material behaviors that can be obtained. We let two cuboids of material drop onto a static collider, the left (red) one being fully rigid with $E = \infty$, the right (blue) one elastic with E = 10MPa.

We also use this setup to demonstrate the impact of the critical fraction ϕ_c in Figure 24. Using a large friction coefficient ($\mu = 2$) leads to the material flowing loosely, and gaining volume due to not being able to recompress, the flow rule with $p_c = \infty$ enforcing positive divergence of the flow. This could be alleviated by explicit hardening and a finite p_c, however the implicit hardening from the critical fraction formulation of Daviet and Bertails-Descoubes [2016b] is handy; here with $\phi_c = 0.25$, the volume gain is largely alleviated. However, we note that using larger ϕ_c with $\mathbf{P_0}$ stresses can lead to undesirable creneling artifacts.

6.3 Two-way coupling

We exercise the two-way coupling scheme from section 4.4.

Interactive sandbox. In Figure 20 we simulate a sand box filled with 673k MPM particles and 6 rigid bodies simulated with Newton's XPBD solver. We use either a non-associated Drucker–Prager flow rule with $\mu = 0.5$, or an inviscid one $\mu = 0$. The rigid bodies have varying densities of 500, 1000 and 2000kg.m^{-3}, demonstrating that buoyancy effects are properly recovered. The simulation runs at about 20FPS, or about 30FPS with particle rendering disabled, allowing live interaction with the rigid bodies and transitively with the granular material or fluid.

Locomotion policy. We demonstrate the effect of two-way coupled terrain on a locomotion policy in Figure 13. We simulate the ANYmal quadruped and the associated policy from the Newton physics engine with the MuJoCo Warp solver, instruct it to walk

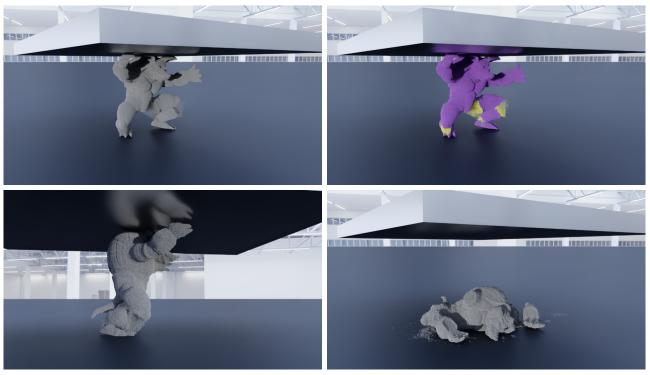


Fig. 21. A concrete Armadillo being crushed by an industrial press. On the top right, hue indicates the hardening state J^p of the particles.

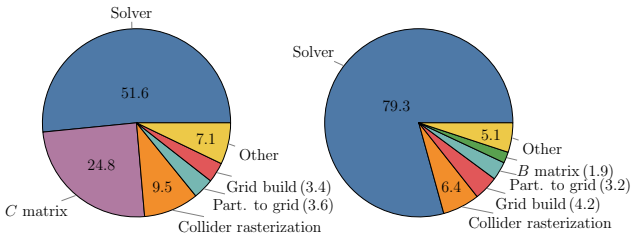


Fig. 22. Distribution of the computation time in the different parts of our method for the "Press" (left) and "Sand city" (right) examples.

forward through a terrain made of sand, snow, and Von Mises clay. Thanks to our implicit integration scheme, the MPM and rigid-body simulations can use the same timestep, which simplify velocity and impulse transfers. The complex terrain causes the robot to shorten its stride, as visible in the accompanying video. In contrast, a one-way coupled simulation leads to the robot applying unrealistic forces to the terrain.

6.4 Complex examples

Finally, we leverage our method to tackle more demanding scenarios.

Press. We crush a 1m-tall concrete Armadillo, discretized using $\mathbf{Q_1 - P_0}$ elements, with an industrial press. Figure 21 shows the cracks naturally appearing at the weak points of the model.

Shredder. Branches go through the wood shredder from Figure 12 — or get stuck, demonstrating elastoplasticity and frictional contact coupling. As the slenderness of the branches induces larger strains, we use a $\mathbf{dP_1}$ stress element, and keep $\mathbf{Q_1}$ for the velocity. We note that the use of an isotropic material is not the most adequate here; adapting the AnisoMPM model from Wolper et al. [2020], or the homogenization procedure of Chen et al. [2025] are interesting directions.

Avalanche. A snow cube hits a sine mountain side featuring the same structure as our slab experiment: a dense, cohesive top layer, and a weak bottom layer. While the ball rolls downhill, accumulating snow as it goes, the additional load triggers the collapse of the weak layer, and shear-induced damage of the slab anchors, eventually causing the whole slope to rush downhill with a characteristic crown crack pattern (Figure 7).

Sand city. A 5m×5.5m×6m sand cube comprising 49M particles crashes over the model city from Figure 1, forming intricate patterns as the granular material makes its way through the city's narrow streets and skyline. Thanks to the compactness of the $\mathbf{Q_1 - P_0}$ element, mean simulation time per 60FPS frame is just over 4s. We compare where most of the computational time is being spent for this example and the *Press* scenario in Figure 22.

7 Discussion

7.1 Limitations and future work

While our results show that visually appealing MPM simulations can be produced with trilinear velocities, the use of compact mixed element stencils remains a performance–accuracy compromise. Indeed, the stresses obtained using trilinear velocities exhibit larger spatial variations, which can impact whether plastic flow can happen or not. Our mixed-elements with trilinear velocities do not satisfy the inf-sup stability condition, and while our formulation allows simulating incompressible materials by setting $E = \infty$ or $\nu = 0.5$, the resulting numerical system may not be as well-behaved as when using specialized discretizations [Frâncu et al. 2021; Larionov et al. 2017]. Stabilization methods [Chandra et al. 2024; Iaconeta et al. 2019; Zhao and Choo 2020] could help counteract these effects. Another interesting endeavor would be to leverage the recent compact kernel from Liu et al. [2025]. As our material parameters are stored on particles while the flow rule may be solved on grid nodes, interpolation is necessary, which could blur the material boundaries. Different ways of parameterizing the yield surface, e.g, using absolute or relative quantities, lead to different interpolation results. That being said, we note that as velocities are always rasterized on grid, smoothing is unavoidable, especially with larger stencils.

The colored Gauss-Seidel or Jacobi algorithms that we use for our solver suffer from slow asymptotic convergence. While warm-starting stresses and frictional impulse help, we would like to investigate other strategies, like progressive blending of external forces in a small timestep fashion [Macklin et al. 2019], accelerated convergence through sampling [Lan et al. 2025], or developing multigrid solvers [Wang et al. 2020a]. Primal–dual formulations like AVBD [Giles et al. 2025] could also be of interest.

While our mixed discretization can be applied on arbitrary convex elastic potentials, we mainly used corotated elasticity in our examples. Investigating the use of hyperelastic and anisotropic materials, possibly to be solved within a Newton loop, constitutes future work. It would also be possible for our unified flow rule to better fit the Cam Clay yield surface by allowing the tuning of the pressure at which the maximum shear yield stress is reached; currently unconditionally set to $p_c/2$. While this would not change the polygonal structure of our surface, and thus our flow rule solver could be easily adapted, we decided against extending even more the parameter space to explore. We currently update the hardening parameters in an explicit fashion; integrating the hardening rule as a fully implicit part of the flow rule, similarly to the critical fraction, is something we would like to investigate. Similarly, our advection

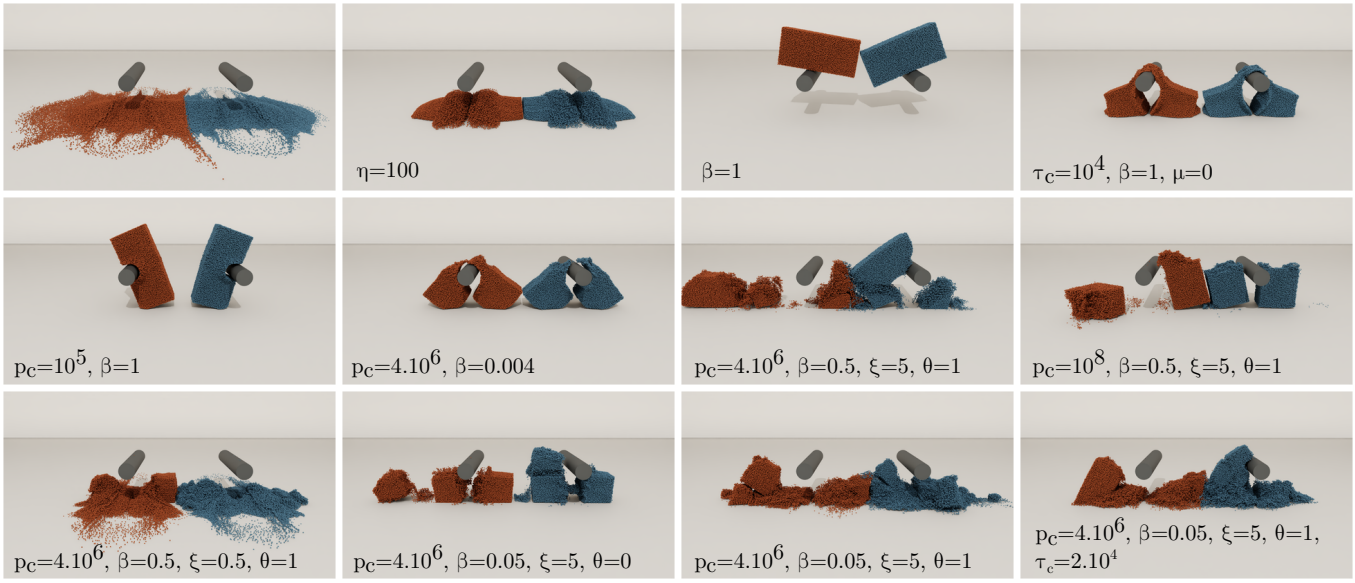


Fig. 23. To explore the range of materials that can be simulated with our method, we let two cuboids of material, one rigid ($E = \infty$, red) and one elastic (E = 10MPa, blue), drop onto cylinder colliders, and show the resulting end state. We only annotate parameters changed from the top-left Drucker–Prager material: $\mu = 0.68$, $p_c = \infty$, all other parameters at 0.

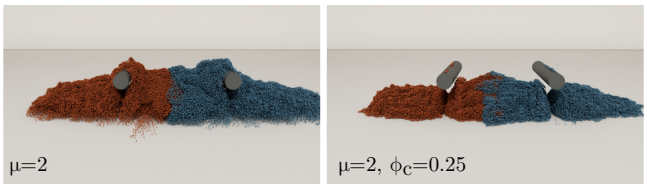


Fig. 24. Simulation of the double block drop from Figure 23 with zero (left) and non-zero (right) critical fraction ϕ_c. The critical fraction modeling allows the material to recompact after it has expanded.

step remains explicit, and as such remains subject to a CFL condition. The performance advantage of implicit over explicit time integration thus reduces as the grid resolution increases. Space-time adaptive methods [Fang et al. 2018] may be key to overcome this limitation.

7.2 Conclusion

We have presented a systematic way to derive mixed MPM discretizations from elastoviscoplastic potentials, as well as a practical algorithm for a unified and versatile flow rule. Our formulation permits the use of low order shape functions with small stencils, which allows our high-level implementation to offer state-of-the-art performance on both some large-scale and low-latency examples. Combined with implicit and two-way coupled treatment of frictional boundary conditions, this makes our method especially well suited for inclusion in robotics training environments. We finally believe this framework opens many opportunities to be explored, from the choice of mixed elements and constitutive model to numerical solver algorithms and low-level optimization of GPU kernels.

Acknowledgments

The author would like to thank the anonymous reviewers for the insightful discussion, as well as the Newton authors for providing the environment within which the presented simulations were made.

References

Autodesk. 2019. *Bifrost for Maya.* https://www.autodesk.com/products/maya/bifrost

Scott G Bardenhagen, Edward M Kober, et al. 2004. The generalized interpolation material point method. *Computer Modeling in Engineering and Sciences* 5, 6 (2004), 477–496.

Bodhinanda Chandra, Ryota Hashimoto, Ken Kamrin, and Kenichi Soga. 2024. Mixed material point method formulation, stabilization, and validation for a unified analysis of free-surface and seepage flow. *J. Comput. Phys.* 519 (2024), 113457. doi:10.1016/j.jcp.2024.113457

Yi-Lu Chen, Mickaël Ly, and Chris Wojtan. 2025. Numerical Homogenization of Sand from Grain-level Simulations. *ACM Trans. Graph.* 44, 6, Article 220 (dec 2025), 23 pages. doi:10.1145/3763344

Gilles Daviet. 2016. *Modeling and Simulating Complex Materials subject to Frictional Contact.* Theses. Université Grenoble Alpes. https://theses.hal.science/tel-01684673

Gilles Daviet. 2020. Simple and scalable frictional contacts for thin nodal objects. *ACM Trans. Graph.* 39, 4, Article 61 (Aug. 2020), 16 pages. doi:10.1145/3386569.3392439

Gilles Daviet. 2023. Interactive Hair Simulation on the GPU using ADMM. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23).* Association for Computing Machinery, New York, NY, USA, Article 24, 11 pages. doi:10.1145/3588432.3591551

Gilles Daviet and Florence Bertails-Descoubes. 2016a. Nonsmooth simulation of dense granular flows with pressure-dependent yield stress. *Journal of Non-Newtonian Fluid Mechanics* 234 (2016), 15–35. doi:10.1016/j.jnnfm.2016.04.006

Gilles Daviet and Florence Bertails-Descoubes. 2016b. A semi-implicit material point method for the continuum simulation of granular materials. *ACM Trans. Graph.* 35, 4, Article 102 (July 2016), 13 pages. doi:10.1145/2897824.2925877

Géry de Saxcé and Zhiqiang Feng. 1998. The bipotential method: A constructive approach to design the complete contact law with friction and improved numerical algorithms. *Mathematical and Computer Modelling* 28, 4-8 (Aug. 1998), 225–245. doi:10.1016/S0895-7177(98)00119-8

Sachith Dunatunga and Ken Kamrin. 2015. Continuum modelling and simulation of granular flows through their many phases. *Journal of Fluid Mechanics* 779 (2015), 483–513. doi:10.1017/jfm.2015.383

Kenny Erleben. 2017. Rigid body contact problems using proximal operators. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '17)*. Association for Computing Machinery, New York, NY, USA, Article 13, 12 pages. doi:10.1145/3099564.3099575

Martha W Evans, Francis H Harlow, and Eleazer Bromberg. 1957. *The particle-in-cell method for hydrodynamic calculations.* Technical Report.

Linxu Fan, Floyd M. Chitalu, and Taku Komura. 2022. Simulating Brittle Fracture with Material Points. *ACM Trans. Graph.* 41, 5, Article 177 (May 2022), 20 pages. doi:10.1145/3522573

Yu Fang, Yuanming Hu, Shi-Min Hu, and Chenfanfu Jiang. 2018. A Temporally Adaptive Material Point Method with Regional Time Stepping. *Computer Graphics Forum* 37, 8 (2018), 195–204. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13524 doi:10.1111/cgf.13524

Yu Fang, Minchen Li, Ming Gao, and Chenfanfu Jiang. 2019. Silly rubber: an implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids. *ACM Trans. Graph.* 38, 4, Article 118 (July 2019), 13 pages. doi:10.1145/3306346.3322968

Yun Fei, Yuhan Huang, and Ming Gao. 2021. Principles towards Real-Time Simulation of Material Point Method on Modern GPUs. *ArXiv* abs/2111.00699 (2021). https://api.semanticscholar.org/CorpusID:240354454

Yun (Raymond) Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2018. A multi-scale model for simulating liquid-fabric interactions. *ACM Trans. Graph.* 37, 4, Article 51 (July 2018), 16 pages. doi:10.1145/3197517.3201392

Yutao Feng, Xiang Feng, Yintong Shang, Ying Jiang, Chang Yu, Zeshun Zong, Tianjia Shao, Hongzhi Wu, Kun Zhou, Chenfanfu Jiang, and Yin Yang. 2024. Gaussian Splashing: Unified Particles for Versatile Motion Synthesis and Rendering. *arXiv preprint arXiv:2401.15318* (2024).

Mihai Frâncu, Arni Asgeirsson, Kenny Erleben, and Mads J. L. Rønnow. 2021. Locking-Proof Tetrahedra. *ACM Trans. Graph.* 40, 2, Article 12 (2021). doi:10.1145/3444949

Ming Gao, Xinlei Wang, Kui Wu, Andre Pradhana, Eftychios Sifakis, Cem Yuksel, and Chenfanfu Jiang. 2018. GPU optimization of material point methods. *ACM Trans. Graph.* 37, 6, Article 254 (Dec. 2018), 12 pages. doi:10.1145/3272127.3275044

J. Gaume, T. Gast, J. Teran, A. van Herwijnen, and C. Jiang. 2018. Dynamic anticrack propagation in snow. *Nature communications* 9, 1 (2018), 3047.

Chris Giles, Elie Diaz, and Cem Yuksel. 2025. Augmented Vertex Block Descent. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2025)* 44, 4 (08 2025), 12 pages. doi:10.1145/3731195

R. A. Gingold and J. J. Monaghan. 1977. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society* 181, 3 (12 1977), 375–389. arXiv:https://academic.oup.com/mnras/article-pdf/181/3/375/3104055/mnras181-0375.pdf doi:10.1093/mnras/181.3.375

Google DeepMind and NVIDIA. 2025. *MuJoCo Warp: A GPU-optimized version of the MuJoCo physics simulator.* https://github.com/google-deepmind/mujoco_warp

Bernard Halphen and Quoc Son Nguyen. 1975. Sur les matériaux standard généralisés. *Journal de Mécanique* 14, 1 (1975), 39–63. https://hal.science/hal-03600755

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 2018. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.* 37, 4, Article 150 (July 2018), 14 pages. doi:10.1145/3197517.3201293

Ilaria Iaconeta, Antonia Larese, Riccardo Rossi, and Eugenio Onate. 2019. A stabilized mixed implicit material point method for non-linear incompressible solid mechanics. *Computational Mechanics* 63, 6 (2019), 1243–1260.

Chenfanfu Jiang, Theodore Gast, and Joseph Teran. 2017. Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Trans. Graph.* 36, 4, Article 152 (July 2017), 14 pages. doi:10.1145/3072959.3073623

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM Trans. Graph.* 34, 4, Article 51 (July 2015), 10 pages. doi:10.1145/2766996

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. 2016. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 24, 52 pages. doi:10.1145/2897826.2927348

Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. 2016. Drucker-prager elastoplasticity for sand animation. *ACM Trans. Graph.* 35, 4, Article 103 (July 2016), 12 pages. doi:10.1145/2897824.2925906

Dan Koschier, Jan Bender, Barbara Solenthaler, and Matthias Teschner. 2022. A survey on SPH methods in computer graphics. In *Computer graphics forum*, Vol. 41. Wiley Online Library, 737–760.

Lei Lan, Zixuan Lu, Chun Yuan, Weiwei Xu, Hao Su, Huamin Wang, Chenfanfu Jiang, and Yin Yang. 2025. JGS2: Near Second-order Converging Jacobi/Gauss-Seidel for GPU Elastodynamics. *ACM Trans. Graph.* 44, 4, Article 44 (July 2025), 15 pages. doi:10.1145/3731183

Egor Larionov, Christopher Batty, and Robert Bridson. 2017. Variational stokes: a unified pressure-viscosity solver for accurate viscous liquids. *ACM Trans. Graph.* 36, 4, Article 101 (July 2017), 11 pages. doi:10.1145/3072959.3073628

Yong Liang, Xiong Zhang, and Yan Liu. 2019. An efficient staggered grid material point method. *Computer Methods in Applied Mechanics and Engineering* 352 (2019), 85–109. doi:10.1016/j.cma.2019.04.024

Michael Liu, Xinlei Wang, and Minchen Li. 2025. CK-MPM: A Compact-Kernel Material Point Method. *ACM Trans. Graph.* 44, 4, Article 152 (July 2025), 14 pages. doi:10.1145/3731155

Pingchuan Ma, Peter Yichen Chen, Bolei Deng, Joshua B. Tenenbaum, Tao Du, Chuang Gan, and Wojciech Matusik. 2023. Learning Neural Constitutive Laws from Motion Observations for Generalizable PDE Dynamics. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23, Vol. 202)*. PMLR, 23279–23300.

Miles Macklin. 2022. Warp: A High-performance Python Framework for GPU Simulation and Graphics. https://github.com/nvidia/warp. NVIDIA GPU Technology Conference (GTC).

Miles Macklin, Kier Storey, Michelle Lu, Pierre Terdiman, Nuttapong Chentanez, Stefan Jeschke, and Matthias Müller. 2019. Small steps in physics simulation. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '19)*. Association for Computing Machinery, New York, NY, USA, Article 2, 7 pages. doi:10.1145/3309486.3340247

Vismay Modi, Nicholas Sharp, Or Perel, David I. W. Levin, and Shinjiro Suead. 2024. Simplicits: Mesh-Free, Geometry-Agnostic, Elastic Simulation. *arXiv preprint* (2024).

Jean Jacques Moreau. 1970. Sur les lois de frottement, de plasticité et de viscosité. *Comptes rendus hebdomadaires des séances de l'Académie des sciences* 271 (1970), 608–611. https://hal.science/hal-01868140

L. Moresi, F. Dufour, and H.-B. Mühlhaus. 2003. A Lagrangian integration point finite element method for large deformation modeling of viscoelastic geomaterials. *J. Comput. Phys.* 184, 2 (2003), 476–497. doi:10.1016/S0021-9991(02)00031-1

Rahul Narain, Abhinav Golas, and Ming C. Lin. 2010. Free-flowing granular materials with two-way solid coupling. In *ACM SIGGRAPH Asia 2010 Papers* (Seoul, South Korea) *(SIGGRAPH ASIA '10)*. Association for Computing Machinery, New York, NY, USA, Article 173, 10 pages. doi:10.1145/1866158.1866195

Newton Contributors. 2025. *Newton: GPU-accelerated physics simulation for robotics, and simulation research.* Newton a Series of LF Projects, LLC. https://github.com/newton-physics/newton

Neal Parikh and Stephen P. Boyd. 2013. Proximal Algorithms. *Found. Trends Optim.* 1 (2013), 127–239. https://api.semanticscholar.org/CorpusID:51791656

Ziyin Qu, Minchen Li, Yin Yang, Chenfanfu Jiang, and Fernando De Goes. 2023. Power Plastics: A Hybrid Lagrangian/Eulerian Solver for Mesoscale Inelastic Flows. *ACM Trans. Graph.* 42, 6, Article 240 (2023). doi:10.1145/3618344

Daniel Ram, Theodore Gast, Chenfanfu Jiang, Craig Schroeder, Alexey Stomakhin, Joseph Teran, and Pirouz Kavehpour. 2015. A material point method for viscoelastic fluids, foams and sponges. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '15)*. Association for Computing Machinery, New York, NY, USA, 157–163. doi:10.1145/2786784.2786798

Ralph Rockafellar. 1968. Integrals which are convex functionals. *Pacific journal of mathematics* 24, 3 (1968), 525–539.

Gauthier Rousseau, Thibaut Métivet, Hugo Rousseau, Gilles Daviet, and Florence Bertails-Descoubes. 2023. Revisiting the role of friction coefficients in granular collapses: confrontation of 3-D non-smooth simulations with experiments. *Journal of Fluid Mechanics* 975 (2023), A14. doi:10.1017/jfm.2023.835

Liangwang Ruan, Bin Wang, Tiantian Liu, and Baoquan Chen. 2024. MiNNIE: a Mixed Multigrid Method for Real-time Simulation of Nonlinear Near-Incompressible Elastics. *ACM Trans. Graph.* 43, 6, Article 258 (2024). doi:10.1145/3687758

SideFX. 2024. *Houdini MPM.* https://www.sidefx.com/tutorials/mpm-h21-masterclass/

Breannan Smith, Fernando De Goes, and Theodore Kim. 2019. Analytic Eigensystems for Isotropic Distortion Energies. *ACM Trans. Graph.* 38, 1, Article 3 (Feb. 2019), 15 pages. doi:10.1145/3241041

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. 2013. A material point method for snow simulation. *ACM Trans. Graph.* 32, 4, Article 102 (July 2013), 10 pages. doi:10.1145/2461912.2461948

Haozhe Su, Xuan Li, Tao Xue, Chenfanfu Jiang, and Mridul Aanjaneya. 2023. A Generalized Constitutive Model for Versatile MPM Simulation and Inverse Learning with Differentiable Physics. *Proc. ACM Comput. Graph. Interact. Tech.* 6, 3, Article 49 (Aug. 2023), 20 pages. doi:10.1145/3606925

Haozhe Su, Tao Xue, Chengguizi Han, Chenfanfu Jiang, and Mridul Aanjaneya. 2021. A Unified Second-Order Accurate in Time MPM Formulation for Simulating Viscoelastic Liquids with Phase Change. *ACM Trans. Graph.* 40, 4, Article 119 (Aug. 2021), 18 pages.

Deborah Sulsky, Zhen Chen, and Howard L Schreyer. 1994. A particle method for history-dependent materials. *Computer methods in applied mechanics and engineering* 118, 1-2 (1994), 179–196.

Richard Tonge, Feodor Benevolenski, and Andrey Voroshilov. 2012. Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph.* 31, 4, Article 105 (July 2012), 8 pages. doi:10.1145/2185520.2185601

Algorithm 6: Two-way coupled substep for articulated rigid bodies

Input: Previous impulses $\mathbf{r}_j$ at positions $\mathbf{x}_j$ acting on body b_j, body poses q_b and velocities $\dot{q}_b$, MPM particle state
Input: Block-diagonal approximation of rigid body Hessian I_b
`// Accumulate all MPM impulses acting on body b`
$\left(\mathbf{f}_b^{\text{lin}}, \mathbf{f}_b^{\text{ang}}\right) \leftarrow \sum_{j, b_j = b} \left(\mathbf{r}_j, \left(\mathbf{x}_j - \text{com}_b\right) \times \mathbf{r}_j\right) / \Delta_t$;
`// Step external rigid-body solver`
$\left(q_b^{\text{new}}, \dot{q}_b^{\text{new}}\right) \leftarrow \text{RigidBodyStep}\left(q_b, \dot{q}_b, \mathbf{f}_b, \Delta_t\right)$;
`// Subtract MPM forces to get prediction velocity`
$\dot{q}_b^{\text{pred}} \leftarrow \dot{q}_b^{\text{new}} - \Delta_t \, I_b^{-1} \mathbf{f}_b$;
`// Update MPM collider state and solve`
Perform MPM solve (Algorithm 1) assuming independent rigid bodies with inertia I_b, positions q_b and velocities $\dot{q}_b^{\text{pred}}$;
Collect new impulses $\mathbf{r}_j$ at positions $\mathbf{x}_j$; set $q_b \leftarrow q_b^{\text{new}}$, $\dot{q}_b \leftarrow \dot{q}_b^{\text{new}}$;

Kiwon Um, Xiangyu Hu, and Nils Thuerey. 2017. Perceptual evaluation of liquid simulation methods. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.

Xinlei Wang, Minchen Li, Yu Fang, Xinxin Zhang, Ming Gao, Min Tang, Danny M. Kaufman, and Chenfanfu Jiang. 2020a. Hierarchical Optimization Time Integration for CFL-Rate MPM Stepping. *ACM Trans. Graph.* 39, 3, Article 21 (April 2020), 16 pages. doi:10.1145/3386760

Xinlei Wang, Yuxing Qiu, Stuart R. Slattery, Yu Fang, Minchen Li, Song-Chun Zhu, Yixin Zhu, Min Tang, Dinesh Manocha, and Chenfanfu Jiang. 2020b. A massively parallel and scalable multi-GPU material point method. *ACM Trans. Graph.* 39, 4, Article 30 (Aug. 2020), 15 pages. doi:10.1145/3386569.3392442

Lukas Westhofen, Stefan Jeske, and Jan Bender. 2023. A Comparison of Linear Consistent Correction Methods for First-Order SPH Derivatives. *Proceedings of the ACM on Computer Graphics and Interactive Techniques (SCA)* 6, 3, Article 48 (aug 2023), 20 pages. doi:10.1145/3606933

Joshuah Wolper, Yunuo Chen, Minchen Li, Yu Fang, Ziyin Qu, Jiecong Lu, Meggie Cheng, and Chenfanfu Jiang. 2020. AnisoMPM: animating anisotropic damage mechanics. *ACM Trans. Graph.* 39, 4, Article 37 (Aug. 2020), 16 pages. doi:10.1145/3386569.3392428

Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. 2019. CD-MPM: continuum damage material point methods for dynamic fracture animation. *ACM Trans. Graph.* 38, 4, Article 119 (July 2019), 15 pages. doi:10.1145/3306346.3322949

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. 2023. PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics. *arXiv preprint arXiv:2311.12198* (2023).

Chang Yu, Xuan Li, Lei Lan, Yin Yang, and Chenfanfu Jiang. 2024. XPBI: Position-Based Dynamics with Smoothing Kernels Handles Continuum Inelasticity. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SA '24)*. Association for Computing Machinery, New York, NY, USA, Article 65, 12 pages. doi:10.1145/3680528.3687577

Yonghao Yue, Breannan Smith, Christopher Batty, Changxi Zheng, and Eitan Grinspun. 2015. Continuum Foam: A Material Point Method for Shear-Dependent Flows. *ACM Trans. Graph.* 34, 5, Article 160 (Nov. 2015), 20 pages. doi:10.1145/2751541

Duan Z. Zhang, Xia Ma, and Paul T. Giguere. 2011. Material point method enhanced by modified gradient of shape function. *J. Comput. Phys.* 230, 16 (2011), 6379–6398. doi:10.1016/j.jcp.2011.04.032

Yidong Zhao and Jinhyun Choo. 2020. Stabilized material point methods for coupled large deformation and fluid flow in porous materials. *Computer Methods in Applied Mechanics and Engineering* 362 (2020), 112742. doi:10.1016/j.cma.2019.112742

Yidong Zhao, Xuan Li, Chenfanfu Jiang, and Jinhyun Choo. 2026. GeoWarp: An automatically differentiable and GPU-accelerated implicit MPM framework for geomechanics based on NVIDIA Warp. *Advances in Engineering Software* 212 (2026), 104072. doi:10.1016/j.advengsoft.2025.104072

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (July 2005), 965–972. doi:10.1145/1073204.1073298

A Potential derivatives

Here we give concrete expressions for the derivatives of the potential densities necessary to assemble the linear and bilinear forms from System (10–11).

Inertial and kinetic potential. Our quadratic kinetic potential is

$$\mathcal{I}(u) := \int_\Omega \iota(u), \quad \iota(u) := \rho/2\Delta_t (u - u^\star)^2 - \rho < u, g > .$$

Its Gateaux derivatives give rise to the matrix A associated to the bilinear form

$$a(u,v) = \int_\Omega u^T \partial^2_{,uu} \iota v = \sum_q V_q \frac{\rho_q}{\Delta_t} u(x_q)^T v(x_q),$$

and to the force vector $\mathbf{f}$ associated to the linear form

$$f(v) = -\int_\Omega \partial_{,u} \iota v = \sum_q V_q \rho_q \left(\frac{u_q^\star}{\Delta_t} + g\right) v(x_q).$$

Elastic potential. The convex conjugate of our co-rotated elastic potential linearized around $F^{\text{e}} = F^{\text{e}t}$ is

$$\begin{aligned}
\mathcal{E}^*(\sigma) &= \int_\Omega \epsilon^*(\sigma), \\
\epsilon^*(\sigma) &= {\epsilon_S}^*\left(\Xi^t : \sigma\right) - S^t : \Xi^t : \sigma, \\
\Xi^t : \sigma &:= \left(\left(\partial_{,F^{\text{e}}} S\right)^{-T} : \frac{\sigma}{\Delta_t}\right)\left(F^{\text{e}t}\right)^{-T}, \\
{\epsilon_S}^*(\sigma) &= \frac{1+\nu}{2E}\sigma : \sigma - \frac{\nu}{2E}\left(\operatorname{Tr}\sigma\right)^2 + \sigma : \mathbb{I}.
\end{aligned}$$

Its gradient $\partial_{,\sigma}\epsilon^*$ expresses an affine stress to strain rate map,

$$\begin{aligned}
\partial_{,\sigma}\epsilon^*(\sigma) &= \left(\partial_{,\sigma}{\epsilon_S}^*\left(\Xi^t : \sigma\right) - S^t\right) : \Xi^t, \\
\partial_{,\sigma}{\epsilon_S}^*(\sigma) &= \frac{1+\nu}{E}\sigma - \frac{\nu}{E}\operatorname{Tr}\sigma\mathbb{I} + \mathbb{I}.
\end{aligned}$$

This yields the compliance matrix C associated to the bilinear form

$$\begin{aligned}
c(\sigma,\tau) &= \int_\Omega \sigma : \partial^2_{,\sigma\sigma}\epsilon^* : \tau \\
&= \sum_q V_q \sigma(x_q)^T : \partial^2_{,\sigma\sigma}\epsilon^*_q : \tau(x_q) \\
&= \sum_q V_q \left(\Xi^t_q : \sigma(x_q)\right)^T : \partial^2_{,\sigma\sigma}{\epsilon_S}^*_q : \left(\Xi^t_q : \tau(x_q)\right),
\end{aligned}$$

with

$$\sigma : \partial^2_{,\sigma\sigma}{\epsilon_{S_q}}^* : \tau = \frac{1+\nu_q}{E_q}\sigma : \tau - \frac{\nu_q}{E_q}\operatorname{Tr}\sigma \operatorname{Tr}\tau \quad \forall \sigma, \tau,$$

and the elastic strain rate vector corresponding to the affine part

$$\begin{aligned}
c^t(\tau) &= \int_\Omega \partial_{,\sigma}\epsilon^*(0) : \tau = \sum_q V_q \partial_{,\sigma}\epsilon^*_q(0) : \tau(x_q) \\
&= \sum_q V_q \left(\mathbb{I} - S^t_q\right) : \Xi^t_q : \tau(x_q).
\end{aligned}$$

Hencky strain. For the Hencky strain measure, the elastic potential is $\epsilon_{\log}(\dot{\epsilon}^{\text{e}}) := \epsilon_{\log S}(\log S)$,

$$\epsilon_{\log S}(\tau) := \frac{E}{2\left(1+\nu\right)}\left(\tau : \tau + \frac{\nu}{1-2\nu}\left(\operatorname{Tr}\tau\right)^2\right),$$

whose convex conjugate is

$$\epsilon_{\log S}{}^*(\sigma) = \frac{1+\nu}{2E}\sigma : \sigma - \frac{\nu}{2E}\left(\operatorname{Tr}\sigma\right)^2.$$

Table 5. Physical parameters for our simulations, in SI units. For simulations using multiple materials, they are listed over as many lines.

Name	Element	Δ_x	PPC	ρ	g	E	ν	η^e	μ	p_c	β	τ_c	η^p	θ	(ξ, ζ_+, ζ_-)
Rolling	$\mathbf{Q_1}$-$\mathbf{P_0}$-$\mathbf{S_2}$	0.03	3^3	1600	9.8	∞			0.68	∞	1	0	0	0	
Dam Break	$\mathbf{Q_1}$-$\mathbf{dP_1}$-$\mathbf{S_2}$	0.05	3^3	1000	9.8	∞			0.0	∞	1	0	0	0	
Viscous	$\mathbf{Q_1}$-$\mathbf{dP_1}$-$\mathbf{S_2}$	0.005	3^3	1000	9.8	∞			0.0	∞	1	0	50/	0	
…high													500		
Sandbox	$\mathbf{Q_1}$-$\mathbf{P_0}$-$\mathbf{S_2}$	0.05	3^3	1600/	9.8	∞			0.5/	∞	0	0	0	0	
…water				1000					0						
ANYmal	$\mathbf{Q_1}$-$\mathbf{P_0}$-$\mathbf{S_2}$	0.02	3^3	1600/	9.8	∞			0.48/	∞/	0/	0/	0	0/	$(0, 0, 0)$/
…snow				250/					0.3/	$2M$/	0.05/	0/		1/	$(1, 1, 1)$/
…mud				1500					0.0	∞	1	200		0.1	$(0, 0, 0)$
Press	$\mathbf{Q_1}$-$\mathbf{P_0}$-**Part.**	0.01	3^3	2500	9.8	40G	0.2	0.4G	0.5	10G	0.025	0	0	1	$(10, 1, 1)$
Shredder	$\mathbf{Q_1}$-$\mathbf{dP_1}$-**Part.**	0.005	3^3	900	9.8	1G	0.2	0.5G	0.5	500M	0.5	0	0	1	$(10, 0, 1)$
Avalanche	$\mathbf{Q_1}$-$\mathbf{P_0}$-$\mathbf{Q_1}$	0.05	3^3	400	9.8	14M/	0.3	0	0.5	10M/	0.05/	0	0	1	$(25, 1, 1)$ /
…weak layer						1M				1.5M	0.2				$(0.2, -200, 200)$
City	$\mathbf{Q_1}$-$\mathbf{P_0}$-$\mathbf{S_2}$	0.03	2^3	1600	9.8	∞			0.68	∞	0	0	0	0	

We once again linearize around the current elastic deformation gradient $F^{e\,t}$ as $\log S = \log S^t + \partial_{,\hat{\varepsilon}^e} \log S : \hat{\varepsilon}^e$, so that the convex conjugate of $\epsilon_{\log}$ is given by

$$\epsilon_{\log}{}^*(\sigma) = \epsilon_{\log S}{}^* \left(\Xi^t_{\log} : \sigma \right) - \log S^t : \Xi^t_{\log} : \sigma,$$
$$\Xi^t_{\log} : \sigma := \left(\partial_{,\hat{\varepsilon}^e} \log S \right)^{-T} : \sigma,$$

which falls back under the form of the co-rotated potential. The fourth-order tensor $\Xi_{\log}$ is not trivial to express exactly however; in practice, we use the approximation $\Xi_{\log} : \sigma \approx R^T \sigma R / \Delta_t$.

Elastic parameter interpolation. While in the continuous limit we can write our constitutive model on either the elastic potential or its convex conjugate, in the discrete setting this can give rise to different behaviors. Consider two particles, one perfectly rigid, one perfectly compliant, in single cell with piecewise-constant ($\mathbf{P_0}$) stress discretization. Discretizing the elastic potential as a sum over particles, we get an infinitely stiff material; discretizing the compliance potential as a sum over particles, we get a fully compliant material. Postulating that the former behavior is more desirable, we replace E_q in the forms above with $E(x_q)$, the value sampled at the particle positions from a continuous interpolant.

B Properties of the flow rule root-finding

We look at the properties of the function $g(\alpha)$ on which we need to perform root finding,

$$g(\alpha) := (1 - \gamma\alpha)\, s(\alpha), \quad s(\alpha) := \left\| (D_{\mathrm{T}} + \alpha \mathbb{I})^{-1} b_{\mathrm{T}} \right\|$$

with $\alpha > 0$, D diagonal positive definite, and $\gamma \geq 0$.

Non-associated case. We first look at the $\gamma = 0$ case, i.e, $g = s$. Let

$$S_0(\alpha) := s(\alpha)^2 = \sum_i \frac{b_i^2}{(D_i + \alpha)^2},$$
$$S_1(\alpha) := \sum_i \frac{b_i^2}{(D_i + \alpha)^3}, \qquad S_2(\alpha) := \sum_i \frac{b_i^2}{(D_i + \alpha)^4}.$$

We have $S_0' = -2S_1$, and $S_1' = -3S_2$. Let us now look at the variations of $s = \sqrt{S_0}$. We have $s' = -S_1/\sqrt{S_0}$, hence s is monotonically decreasing. Then

$$s'' = \frac{3S_2\sqrt{S_0} - 2S_1^2/\sqrt{S_0}}{S_0} = \frac{3S_2S_0 - 2S_1^2}{S_0^{3/2}}.$$

$S_1^2 \leq S_0 S_2$ from the Cauchy–Schwartz theorem, hence $s'' \geq 0$ and our function is strictly convex.

Associated case. The case $\gamma > 0$ is more complex, and in that case g can actually be locally non convex for $\alpha \geq 1/\gamma$ and an anisotropic D. Here we just look at the first variation of g to show monotonicity. We have $g' = -\left((1 - \gamma\alpha)S_1 + \gamma S_0\right)/\sqrt{S_0}$, and

$$(1 - \gamma\alpha)S_1 + \gamma S_0 = \sum_i \frac{b_i^2}{(D_i + \alpha)^2} \left(\frac{1 - \gamma\alpha}{D_i + \alpha} + \gamma \right)$$
$$= \sum_i \frac{b_i^2}{(D_i + \alpha)^2} \left(\frac{1 + \gamma D_i}{D_i + \alpha} \right) \geq 0$$

Hence g is monotonically decreasing on our root-finding interval.