https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

Matter (v1): An open-source MPM solver for granular matter

Lars Blatny[1,2,3] and Johan Gaume[1,2,3]

[1]Institute for Geotechnical Engineering, ETH Zürich, CH-8093 Zürich, Switzerland
[2]WSL Institute for Snow and Avalanche Research SLF, CH-7260 Davos Dorf, Switzerland
[3]Climate Change, Extremes, and Natural Hazards in Alpine Regions Research Center CERC, CH-7260 Davos Dorf, Switzerland

Correspondence: Lars Blatny (lars.blatny@slf.ch)

Abstract. Simulating the mechanics and flow of granular media requires numerical methods that can handle extreme deformations, along with accurate constitutive models capable of capturing phenomena such as elasticity, shear banding, viscous behavior, compressibility, intergranular attractive forces and rate-dependent friction. In pursuit of this, this article introduces *Matter*, a Material Point Method (MPM) solver equipped with a range of models to describe dry and cohesive granular media. Rooted in a finite strain elastoplastic framework, this software features Drucker-Prager models, overstress models, critical state mechanics models as well as the $\mu(I)$-rheology. This includes the recently proposed "critical state $\mu(I)$-rheology model" for cohesive and compressible flows. Moreover, *Matter* provides a simple way of dealing with complex terrains and introduces a novel material-induced frictional boundary condition. Implemented in C++ with few required dependencies and parallelized on shared memory, it represents a lightweight yet computationally efficient option for laptops and desktops.

1 Introduction

Sand, snow, powders and other granular materials typically undergo large deformations, display changes in topology and even experience phase transitions between solid- and fluid-like behavior. Therefore, simulating the dynamics of these media as continua can be a challenging endeavor. The well-known Finite Element Method (FEM) can suffer significantly from mesh distortion issues when the material domain is sufficiently deformed. In order to overcome this issue, the Arbitrary Lagrangian-Eulerian (ALE) FEM was proposed already in the early 1980s (Donea et al., 1982), however, also with this scheme the mesh can become too distorted when very large or unpredictable deformations occur. Alternatively, fully meshless Lagrangian methods naturally circumvent mesh distortion challenges. Prominently, the Smoothed-Particle Hydrodynamics (SPH) method was originally developed for problems in astrophysics (Lucy, 1977; Gingold and Monaghan, 1977) and later extended to free surface flow problems (Monaghan, 1994) and solid mechanics problems (Libersky et al., 1993). Among other challenges, this method is burdened with keeping track of the neighboring discretization points, which can be computationally expensive, and boundary conditions often pose challenges (Raymond et al., 2018; Vacondio et al., 2021). Other examples of meshfree schemes include, e.g., the Element-Free Galerkin method (EFG) (Belytschko et al., 1994) and the Reproducing Kernel Particle Method (RKPM) (Liu et al., 1995).

As a hybrid Eulerian-Lagrangian method, the Material Point Method (MPM) has since its original formulation in the 1990s established itself as a popular numerical method capable of simulating materials undergoing very large deformations and topology changes. While the original formulation of MPM is often attributed to Sulsky et al. (1994), it may also be viewed as an extension of Particle-In-Cell (PIC) methods (Harlow, 1964), in particular the FLuid-Implicit-Particle method (FLIP) (Brackbill et al., 1988), to solid mechanics problems, especially problems involving history-dependent materials. Simply put, MPM tracks Lagrangian (material) points on which material properties such as mass, deformation gradient and velocity are stored, while also relying on an Eulerian grid which facilitates solving the weak form of the momentum balance equation in a FEM-like manner. As such, it offers the advantage of completely avoiding mesh distortion issues in a conceptually simple approach. While its similarity to FEM is undoubtedly an advantage, it does not inherit all the well-established convergence and stability properties that have been developed over FEM's long history. Nevertheless, earlier criticism related to cell-crossing instabilities and numerical dissipation in MPM has been resolved by its community (e.g., Steffen et al., 2008; Jiang et al., 2017; Fei et al., 2021). Methods similar in concept to MPM, incorporating the hybrid idea, include the Particle Finite Element Method (PFEM) (Idelsohn et al., 2004) and the Finite Element Method with Lagrangian Integration Points (FEMLIP) (Moresi et al., 2003). In the former, originally conceived for fluid problems and later applied to solids (Oliver et al., 2007), mesh points move in space like Lagrangian particles and remeshing occurs every time the distortion becomes significant (Cremonesi et al., 2020).

There exist several open-source implementations of MPM. Among the first publicly available projects are *Uintah MPM* (Davison De St. Germain et al., 2000), *Nairn MPM* (Nairn, 2015), *Anura3D* (Anura3D, 2017), *MPM3D-F90* (Zhang et al., 2017), as well as *MPM-GIMP* (Wallstedt, 2011) which is an implementation of the algorithm presented in Bardenhagen and Kober (2004). More recent projects include *CB-Geo* (Kumar et al., 2019), *Karamelo* (de Vaucorbeil et al., 2021) as well as the Matlab learning environment *AMPLE* (Coombs and Augarde, 2020). The efficient implementation *fMPMM* in Matlab (Wyser et al., 2020) and the implementation by Iaconeta et al. (2019) in *Kratos* (Dadvand et al., 2010) should also be mentioned here. Recently, the computer graphics community has seen a number of MPM implementations in the general framework of the programming language *Taichi* (Hu et al., 2019), including MLS-MPM (Hu et al., 2018a) and ASFLIP (Fei et al., 2021). The communities have also recently been enriched with several open-source MPM implementations for GPU architectures. One of the earlier is *GMPM* (Gao et al., 2018) for single-GPU and *Claymore* (Wang et al., 2020) for multi-GPU architectures. In the context of granular flow, a sparse-memory-encoding GPU framework was recently described by Chen et al. (2025).

Each of the above-mentioned open-source codes offers its own advantages and are typically designed to address specific types of problems, be it in the graphics community or in the engineering communities. In the context of modeling granular matter, however, the existing codes display a limited availability of constitutive laws and rheologies to capture the breadth of such materials which may display elastic behavior, undergo plastic deformation and/or act as a viscous fluid depending on the conditions. Under small/slow deformations, granular media act as elastoplastic solids and appropriate constitutive models must accurately capture their internal frictional behavior and transition into a critical state (Radjai et al., 2017). Under large/fast deformations, granular media behave as a pressure-dependent viscoplastic fluid, and it is generally accepted that dense granular flows follow the so-called $\mu(I)$-rheology (GDR MiDi, 2004; Jop et al., 2006). Even more complicated, cohesive granular media present additional complexities due to the presence of intergranular forces that influence both their mechanical and flow properties. A primary example of a (natural) cohesive granular material is snow, which can display a varying degree of intergranular forces depending on its environment. The cohesion may substantially alter the dynamics of a snow avalanche. Additionally, snow is an example of a highly compressible granular medium that can undergo very large variations in density, e.g., during the process of an avalanche. The study of the flow of cohesive granular media has seen a significant boost in recent years, e.g., with new models or proposed alterations to existing rheologies (Rognon et al., 2006, 2008; Radjai and Richefeu, 2009; Khamseh et al., 2015; Berger et al., 2015; Badetti et al., 2018; Vo et al., 2020; Abramian et al., 2020; Mandal et al., 2020, 2021; Macaulay and Rognon, 2021; Ku et al., 2023; Blatny et al., 2024).

This article introduces *Matter* (Blatny and Gaume, 2025), an open-source MPM solver encompassing a wide range of models to capture the complex and diverse behavior of (cohesive and compressible) granular media. It is intended for users studying fundamental granular mechanics as well as those interested in large-scale practical applications. First, the general numerical framework is outlined, presenting the governing equations, their discretized form and the finite strain formulation on which the software is based. Second, the algorithm is detailed, and the details of the implementations are discussed and validated. Third, the various constitutive models are summarized, and finally, a few use cases are demonstrated.

2 Theory and discretization

In this section, MPM is introduced as a numerical continuum scheme for approximating solutions to the momentum and mass conservation laws subject to constitutive models.

2.1 Governing balance laws

The conservation of mass and linear momentum are given by

$$\frac{D\rho(\boldsymbol{x},t)}{Dt} + \rho(\boldsymbol{x},t)\nabla \cdot \boldsymbol{v}(\boldsymbol{x},t) = 0 \tag{1}$$

and

$$\rho(\boldsymbol{x},t)\frac{D\boldsymbol{v}(\boldsymbol{x},t)}{Dt} = \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{x},t) + \rho(\boldsymbol{x},t)\boldsymbol{g}, \tag{2}$$

respectively, where ρ is the mass density, $\boldsymbol{v}$ is the velocity, $\boldsymbol{\sigma}$ is the symmetric Cauchy stress tensor and D/Dt denotes the material time derivative. In Equation (2) it was assumed that the only external body force present is the one resulting from the gravitational acceleration $\boldsymbol{g}$.

2.2 Discretization

Similarly to FEM, the numerical discretization of MPM starts with the weak form of the governing partial differential equations introduced above. While information is stored on the material points which may move in space, the momentum conservation equation is solved on a stationary mesh which usually takes the form of a regular and uniform grid, see Figure 1.

The notation used in this and the next sections can be summarized as follows:

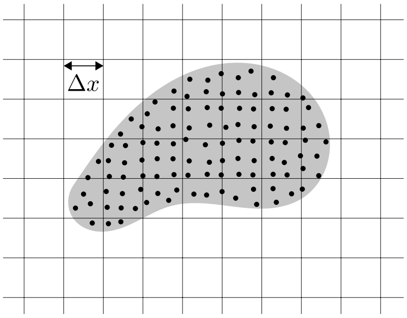

Figure 1. A body discretized into $N_{\text{particles}}$ particles with N_{nodes} background grid nodes.

- The domain of the body is described by a finite number $N_{\text{particles}}$ particles and a finite number N_{nodes} grid nodes.
- Subscript p means the quantity is associated with particle $p \in [1,...,N_{\text{particles}}]$.
- Subscript i means the quantity is associated with grid node $i \in [1,...,N_{\text{nodes}}]$.
- Superscript n means the quantity is evaluated at time t^n associated with time step $n \in \mathbb{N}$.

In order to interpolate between particles and grid nodes, interpolation functions $N_i(\boldsymbol{x})$ which are strictly non-negative are considered such that $\boldsymbol{x}_p = \sum_i \boldsymbol{x}_i N_i(\boldsymbol{x}_p) \; \forall \; p$. Conservation of mass is automatically fulfilled as the total mass of the system $\sum_p m_p$ is the sum of all particle masses which remain constant in time. The volume V_p is, however, not necessarily constant in time.

In the framework of the standard (i.e., updated Lagrangian) MPM the weak form of the momentum conservation equation must be discretized on the grid in the Eulerian configuration. Following closely the derivation in Jiang et al. (2016); Klár (2016); Pradhana (2017), the spatial and temporal discretization of the momentum equation on the Eulerian grid nodes i assuming a time step size Δt can be derived (neglecting the boundary term) as

$$\frac{m_i^n \boldsymbol{v}_i^{n+1} - m_i^n \boldsymbol{v}_i^n}{\Delta t} = -\sum_p V_p^n \boldsymbol{\sigma}_p^n \nabla N_{\boldsymbol{i}}(\boldsymbol{x}_p^n) + m_i^n \boldsymbol{g} \tag{3}$$

where now the left hand side represents the change in momentum while the right hand side represents the internal and external forces. The following remarks should be made:

- This is an *updated Lagrangian MPM*, which by far is the most popular variant of MPM. This variant can be thought of as the equivalent to the updated Lagrangian FEM discretization, where the quadrature points are now the material points which can move in space and the quadrature weights are now the particle volumes. For this reason, MPM is sometimes referred to as a variant of FEM (de Vaucorbeil et al., 2020b; Nguyen et al., 2023). A *total Lagrangian* variant of MPM has been proposed by de Vaucorbeil et al. (2020a) in analogy to the total Lagrangian FEM.
- An explicit time discretization is assumed as the variables on the right-hand side of Equation (3) are evaluated at time t^n. While implicit formulations of MPM also exist, explicit MPM remains the most popular in the literature. Although

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

implicit schemes are computationally more expensive than explicit integration (due to the need of solving a nonlinear system of equations at each time step), larger time steps are allowed. This is in contrast to explicit schemes where the time step size must be severely restricted (Sołowski et al., 2021).

- The weak form above is derived with mass lumping. Mass lumping is a computational simplification also commonly adopted in FEM schemes, however, it results in energy dissipation (Love and Sulsky, 2006; de Vaucorbeil et al., 2020b).

The reader is referred to, e.g., de Vaucorbeil et al. (2020b); Sołowski et al. (2021); Nguyen et al. (2023) for a more detailed overview of current MPM schemes.

2.3 Finite strain elastoplasticity

In the standard finite strain framework of a multiplicative decomposition of the deformation gradient $\boldsymbol{F} = \boldsymbol{F}^E \boldsymbol{F}^P$, *Matter* adopts Hencky's elasticity model which gives a linear relation between the Kirchhoff stress $\boldsymbol{\tau} = \det(\boldsymbol{F})\boldsymbol{\sigma}$ and the Hencky strain $\boldsymbol{\varepsilon}^E$,

$$\boldsymbol{\tau} = \lambda(\operatorname{tr}\boldsymbol{\varepsilon}^E)\boldsymbol{I} + 2G\boldsymbol{\varepsilon}^E, \tag{4}$$

where λ and G are the two Lamé parameters. The superscript on the Hencky strain serves as a reminder that only the elastic part of the strain promotes stresses. In the context of this isotropic and hyperelastic model, it can be shown that the Kirchhoff stress $\boldsymbol{\tau}$ and the Hencky strain $\boldsymbol{\varepsilon}^E$ constitute a work-conjugate pair. This, as well as other admirable properties, in particular in relation to hypoelastic models, are further presented and derived in Bruhns et al. (1999); Xiao and Chen (2002).

A yield surface $y(\boldsymbol{\tau}, \ldots) = 0$, defined through the yield function y, determines the onset of permanent, plastic, deformations. Not considering overstress models (Section 7.4), the yield surface defines admissible states $y \leq 0$ where plastic deformations are restricted to $y = 0$.

The velocity gradient $\boldsymbol{l}$ is assumed additively decomposed as $\boldsymbol{l} = \boldsymbol{l}^E + \boldsymbol{l}^P$. The plastic part of the velocity gradient is determined through the flow rule as

$$\boldsymbol{l}^P = \dot{\gamma}\frac{\partial g}{\partial \boldsymbol{\tau}}, \tag{5}$$

i.e., with a magnitude determined by the plastic multiplier $\dot{\gamma}$ and a direction provided through the plastic potential function $g = g(\boldsymbol{\sigma})$. The Kuhn-Tucker conditions must be fulfilled,

$$\dot{\gamma} \geq 0\,, \quad y \leq 0\,, \quad \dot{\gamma}y = 0, \tag{6}$$

although in the context of overstress models (Section 7.4) these restrictions may be relaxed.

Solving the finite strain elastoplastic problem can be accomplished following Simo (1992). An elastic predictor is calculated assuming no plasticity and corrected in case the yield condition was violated. Denoting the Hencky strain $\boldsymbol{\varepsilon}^{E,\text{trial}}$ predicted in an elastic step from time t^n, this "trial" state is corrected in the return mapping to the new state $\boldsymbol{\varepsilon}^{E,n+1}$ at time t^{n+1} as

$$\boldsymbol{\varepsilon}^{E,n+1} = \boldsymbol{\varepsilon}^{E,\text{trial}} - \boldsymbol{l}^P\Delta t \tag{7}$$

which is derived in detail in Blatny (2023). Consequently, the plastic Hencky strain occurring in a time step Δt can be quantified as $\Delta\boldsymbol{\varepsilon}^{P,n+1} = \boldsymbol{l}^P\Delta t$.

Specific elasto-viscoplastic constitutive models that define yield functions and flow rules for granular media are presented in Section 7.

3 Algorithm

Here, *Matter*'s algorithm is introduced in the context of finite strain elastoplasticity. Some key comments on the algorithm are provided below.

- The MPM community has distinguished schemes that update the particle stresses before or after the grid velocity updates, the former known as Update Stress First (USF) and the latter as Update Stress Last (USL). The algorithm described here considers USL as well as the Modified USL (MUSL) (Sulsky et al., 1995). Notably, USL schemes are by some claimed to be superior to USF in terms of stability and convergence (Wallstedt and Guilkey, 2008; de Vaucorbeil et al., 2020b).
- The time-integration used in the algorithm can be classified as a symplectic (or semi-explicit) Euler scheme since the position update relies on an updated velocity. This scheme is first-order accurate in time.
- In order to overcome the so-called cell-crossing issues typically associated with the original formulation of MPM, *Matter* relies on B-spline interpolation functions as proposed by Steffen et al. (2008). In particular, given a grid spacing Δx, the interpolation function between a grid node i and a particle p may be written

  $$N_i(\boldsymbol{x}_p) = \sum_{\alpha=0}^{d} N\left(\frac{x_{p\alpha} - x_{i\alpha}}{\Delta x}\right) \tag{8}$$

  where α denotes the component. Here, $N(u)$ could, e.g., be the quadratic B-spline

  $$N(u) = \begin{cases} \frac{3}{4} - |u|^2, & \text{if } |u| < \frac{1}{2} \\ \frac{1}{2}(\frac{3}{2} - |u|)^2, & \text{if } \frac{1}{2} \leq |u| < \frac{3}{2} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

  as illustrated in Figure 2. As such, particles are influenced by grid nodes up until a distance $1.5\Delta x$ away. Higher-order B-splines are also considered, in particular cubic B-splines, which require an increased local support. Other approaches to suppress cell-crossing instabilities include the Generalized Interpolation Material Point Method (GIMP) (Bardenhagen and Kober, 2004) and Moving Least Square MPM (MLS-MPM) (Hu et al., 2018b).

After distributing particles over the domain of the body, they are initiated with a mass m_p, an initial volume V_p^0, an initial velocity $\boldsymbol{v}_p^0 = \boldsymbol{0}$ (when initially at rest) and a deformation gradient $\boldsymbol{F}_p^0 = \boldsymbol{I}$ (when initially undeformed). Then, for each time step n of size Δt, the computational steps are outlined below, and a summary is presented in Algorithm 1.

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

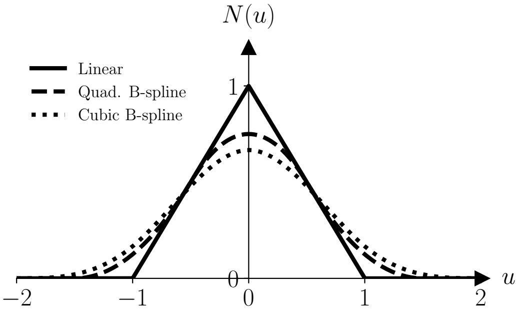

Figure 2. Interpolation functions.

Step 1: Initiate or adapt grid

A regular background Eulerian grid is adapted to cover the material domain by an excess given by the support of the chosen B-spline. Typically, the grid cell size Δx is chosen such that there are at least four and eight particles per grid cell in two and three spatial dimensions, respectively. In order to improve computational efficiency, the grid is expanded or contracted to only cover the material domain, keeping Δx constant with time.

Step 2: Update size of time step

The time step is adapted for each time step following the Courant–Friedrichs–Lewy (CFL) condition,

$$\Delta t \leq C_{\text{cfl}} \frac{\Delta x}{\max\limits_{p} \left(||\boldsymbol{v}_p^{n+1}||_{L_2} \right)} \tag{10}$$

and is at the same time restricted through the elastic wave speed such that

$$\Delta t \leq C_{\text{elastic}} \frac{\Delta x}{\sqrt{E/\rho}} \tag{11}$$

where $C_{\text{cfl}} \leq 1$ and $C_{\text{elastic}} \leq 1$ are appropriate positive constants chosen to obtain a stable scheme.

Step 3: Transfer mass and velocity from particles to grid

The mass on a grid node is interpolated to

$$m_i^n = \sum_{p=1}^{N_{\text{particles}}} m_p N_i(\boldsymbol{x}_p^n). \tag{12}$$

In the original formulation of MPM (using a PIC or FLIP scheme as will be explained further in Step 6), the velocity is interpolated to the grid node as

$$\boldsymbol{v}_i^n = \frac{1}{m_i^n} \sum_{p=1}^{N_{\text{particles}}} m_p \boldsymbol{v}_p^n N_i(\boldsymbol{x}_p^n) \tag{13}$$

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

which preserves momentum. However, considering velocities as locally affine on each particle, transfer schemes that preserve affine velocity fields can be formulated where

$$\boldsymbol{v}_i^n = \frac{1}{m_i^n}\sum_{p=1}^{N_{\text{particles}}} m_p\left(\boldsymbol{v}_p^n + \boldsymbol{G}_p^n(\boldsymbol{x}_i^n - \boldsymbol{x}_p^n)\right)N_i(\boldsymbol{x}_p^n) \tag{14}$$

with $\boldsymbol{G}_p = \boldsymbol{B}_p\boldsymbol{D}_p^{-1}$ similar to a velocity gradient where $\boldsymbol{D}_p$ is similar to an inertia tensor[1]. It can be shown that $\boldsymbol{D}_p$ is approximately $\frac{1}{4}(\Delta x)^2\boldsymbol{I}$ for a uniform grid using quadratic B-splines (Jiang et al., 2015; Nguyen et al., 2023).

Step 4: Update grid velocity

Assuming an explicit Euler time integration, the grid velocity at the next time step is

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \frac{\Delta t}{m_i^n}\boldsymbol{f}_i^n \tag{15}$$

where the grid forces are

$$\boldsymbol{f}_i^n = -\sum_{p=1}^{N_{\text{particles}}} V_p^0\frac{\partial\Psi(\boldsymbol{F}^{E,n})}{\partial\boldsymbol{F}^E}(\boldsymbol{F}^{E,n})^T\nabla N_i(\boldsymbol{x}_p^n) + m_i^n\boldsymbol{g} \tag{16}$$

where V_p^0 is the initial particle volume.

Step 5: Apply boundary conditions

Appropriate boundary conditions are enforced on grid velocities $\boldsymbol{v}_{i\in D}^{n+1}$ where $i \in D$ denotes the set of grid points inside the domain D given by the level sets describing the objects/terrain. This is outlined in detail in Section 4.

Step 6: Transfer grid velocity to particles

Define the particle velocities

$$\boldsymbol{v}_p^{\text{PIC}} = \sum_{i\in\text{grid}}\boldsymbol{v}_i^{n+1}N_i(\boldsymbol{x}_p^n), \tag{17}$$

$$\boldsymbol{v}_p^{\text{FLIP}} = \boldsymbol{v}_p^n + \sum_{i\in\text{grid}}(\boldsymbol{v}_i^{n+1} - \boldsymbol{v}_i^n)N_i(\boldsymbol{x}_p^n). \tag{18}$$

In a PIC scheme, the particle velocity at $n+1$ is taken as

$$\boldsymbol{v}_p^{n+1} = \boldsymbol{v}_p^{\text{PIC}} \tag{19}$$

while in a FLIP scheme

$$\boldsymbol{v}_p^{n+1} = \boldsymbol{v}_p^{\text{FLIP}}. \tag{20}$$

[1] Inspired by rigid-body rotations, this is a generalization of $m_i\boldsymbol{v}_i = \sum_p m_p\left(\boldsymbol{v}_p + \boldsymbol{\omega}_p \times (\boldsymbol{x}_i - \boldsymbol{x}_p)\right)$ where $\boldsymbol{\omega}_p = \boldsymbol{K}_p^{-1}\boldsymbol{L}_p$ denotes the angular velocity, $\boldsymbol{K}_p$ the moment of inertia tensor and $\boldsymbol{L}_p$ the angular momentum.

A highly dissipative scheme, PIC does not preserve angular momentum in the grid-to-particle transfer, leading to rotational artifacts. The FLIP scheme partly improves angular momentum conservation and reduces dissipation by only transferring the velocity increments. However, this may promote a more pronounced "ringing instability" (Love and Sulsky, 2006). Stomakhin et al. (2013) suggested a weighted combination of the PIC and FLIP transfer schemes, creating a trade-off between reducing dissipation associated with PIC on the one hand and suppressing instabilities associated with FLIP on the other hand. In such a PIC-FLIP scheme, the particle velocity at $n+1$ is taken as

$$\boldsymbol{v}_p^{n+1} = C_{\text{transfer}} \boldsymbol{v}_p^{\text{FLIP}} + (1 - C_{\text{transfer}}) \boldsymbol{v}_p^{\text{PIC}} \tag{21}$$

$$= \sum_{i \in \text{grid}} \boldsymbol{v}_i^{n+1} N_i(\boldsymbol{x}_p^n) + C_{\text{transfer}} \left(\boldsymbol{v}_p^n - \sum_{i \in \text{grid}} \boldsymbol{v}_i^n N_i(\boldsymbol{x}_p^n) \right) \tag{22}$$

where $C_{\text{transfer}} \in [0,1]$ sets the PIC-FLIP combination ratio with lower values resulting in more dissipation.

Later, affine PIC (APIC) was proposed by Jiang et al. (2015), conserving angular momentum in both transfers, thus resolving rotations. With a stability similar to PIC without massive dissipation, the dissipation of APIC is comparable to that of FLIP. In an APIC scheme, the particle velocity at $n+1$ follows Equation (19) while the particle-to-grid transfer is accomplished through Equation (14). Therefore, an update of $\boldsymbol{B}_p$ is also necessary, in particular,

$$\boldsymbol{B}_p^{n+1} = \sum_i \boldsymbol{v}_i^{n+1} (\boldsymbol{x}_i^n - \boldsymbol{x}_p^n)^T N_i(\boldsymbol{x}_p^n). \tag{23}$$

Recently, Fei et al. (2021) introduced AFLIP which reduces dissipation further. A generalization of APIC, this was based on following an analogous approach to how PIC was combined with FLIP. In this new transfer scheme, the grid-to-particle transfer can be written exactly as Equation (21), and using Equation (23) to update $\boldsymbol{B}_p$. The parameter C_{transfer} now represents an APIC-AFLIP ratio, with $C_{\text{transfer}} = 0$ meaning pure APIC and $C_{\text{transfer}} = 1$ meaning pure AFLIP. A choice of C_{transfer} closer to unity significantly reduces dissipation compared to APIC.

Matter offers PIC, FLIP, APIC and AFLIP. In the context of USF MPM, these different transfer schemes have been critically compared in Duverger et al. (2024) to which the reader is referred for details. There exist also other recent transfer schemes, e.g., polynomial PIC (Fu et al., 2017), which is generalization of APIC using locally polynomial (as opposed to locally affine) approximations to the grid velocity field, and power PIC (Qu et al., 2022) based on the power particle idea of de Goes et al. (2015).

Step 7: Re-interpolate grid velocities

When using MUSL (Sulsky et al., 1995) instead of USL, we must revisit Step 3 by interpolating $\boldsymbol{v}_p^{n+1}$ from Step 4 to the grid, thus overwriting $\boldsymbol{v}_i^{n+1}$. This is an optional step that may enhance the stability of the simulation.

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

Step 8: Compute trial strain

In an "elastic predictor - plastic corrector" scheme, the deformation gradient is first updated in a trial step, assuming the body deforms purely elastically. In a forward Euler step, the trial step can be written as

$$\boldsymbol{F}_p^{E,\text{trial}} = \boldsymbol{F}_p^{E,n} + \Delta t\, \nabla \boldsymbol{v}^{n+1}(\boldsymbol{x}_p)\, \boldsymbol{F}_p^{E,n} \tag{24}$$

$$= \left(\boldsymbol{I} + \Delta t \sum_{i \in \text{grid}} \boldsymbol{v}_i^{n+1} \left(\nabla N_i(\boldsymbol{x}_p^n)\right)^T \right) \boldsymbol{F}_p^{E,n}. \tag{25}$$

The singular value decomposition of the trial elastic deformation gradient is

$$\boldsymbol{F}_p^{E,\text{trial}} = \boldsymbol{U}_p^{E,\text{trial}} \boldsymbol{\Sigma}_p^{E,\text{trial}} \left(\boldsymbol{V}_p^{E,\text{trial}}\right)^T. \tag{26}$$

Assume an elastic model based on Hencky strain is used such that the trial elastic Hencky strain is given by

$$\boldsymbol{\varepsilon}_p^{E,\text{trial}} = \ln \boldsymbol{\Sigma}_p^{E,\text{trial}} \tag{27}$$

which from a computational perspective can be represented as a d-dimensional vector, where $d = 2$ or 3 denotes the number of spatial dimensions to be simulated.

Step 9: Check yield criterion and return mapping

The Kirchhoff stress $\boldsymbol{\tau}\left(\boldsymbol{\varepsilon}_p^{E,\text{trial}}\right)$ is calculated based on Hencky's elasticity model, and if the yield criterion is satisfied, then

$$\boldsymbol{F}_p^{E,n+1} = \boldsymbol{F}_p^{E,\text{trial}} \quad \text{if} \quad y\left(\boldsymbol{\tau}\left(\boldsymbol{\varepsilon}_p^{E,\text{trial}}\right)\right) \leq 0. \tag{28}$$

On the other hand, if the yield criterion is not satisfied, plastic deformations occurred, and a projection to the yield surface must be made according to the plastic flow rule as

$$\boldsymbol{\varepsilon}_p^{E,n+1} = \boldsymbol{\varepsilon}_p^{E,\text{trial}} - \Delta t \dot{\gamma}_p^n \frac{\partial g\left(\boldsymbol{\tau}\left(\boldsymbol{\varepsilon}_p^{E,n+1}\right)\right)}{\partial \boldsymbol{\tau}}. \tag{29}$$

The elastic deformation gradient at $n+1$ is then obtained as

$$\boldsymbol{F}_p^{E,n+1} = \boldsymbol{U}_p^{E,\text{trial}} e^{\boldsymbol{\varepsilon}_p^{E,n+1}} \left(\boldsymbol{V}_p^{E,\text{trial}}\right)^T \tag{30}$$

Step 10: Update particle positions

The particle positions are updated according to

$$\boldsymbol{x}_p^{n+1} = \boldsymbol{x}_p^n + \Delta t \boldsymbol{v}_p^{\text{PIC}} \tag{31}$$

regardless of the choice of transfer scheme, whether that is PIC, FLIP, APIC or AFLIP.

Algorithm 1 Symplectic B-spline (M)USL (A)PIC/(A)FLIP MPM

```
Initialize $m_p$, $V_p$, $\boldsymbol{x}_p$ according to material domain and density
Initialize $\boldsymbol{v}_p = 0$, $\boldsymbol{F}^E = \boldsymbol{I}$
For $n = 0$, ..., max time steps:
    1. Initiate or adapt grid to material domain
    2. Adapt time step: $\Delta t = \min\left(C_{\text{cfl}}\Delta x/\max_p ||\boldsymbol{v}_p^{n+1}||_{L_2},\ C_{\text{elastic}}\Delta x/\sqrt{E/\rho}\right)$
    3. Interpolate particle mass and velocity to grid:
        (a) $m_i^n = \sum_p m_p N_i(\boldsymbol{x}_p^n)$
        EITHER (b) (if PIC/FLIP) $\boldsymbol{v}_i^n = \frac{1}{m_i^n}\sum_p m_p \boldsymbol{v}_p^n N_i(\boldsymbol{x}_p^n)$
        OR (c) (if APIC/AFLIP) $\boldsymbol{v}_i^n = \frac{1}{m_i^n}\sum_p m_p\left(\boldsymbol{v}_p^n + \boldsymbol{B}_p^n \boldsymbol{D}_p^{-1}(\boldsymbol{x}_i^n - \boldsymbol{x}_p^n)\right)N_i(\boldsymbol{x}_p^n)$
        (d) (if MIBF, Sec. 4.3) $\mu_{b,i}^n = \frac{1}{\sum_p N_i(\boldsymbol{x}_p^n)}\sum_p \mu_{m,p} N_i(\boldsymbol{x}_p^n)$
    4. Update grid velocity: $\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \frac{\Delta t}{m_i^n}\boldsymbol{f}_i^n$ where $\boldsymbol{f}_i^n = -\sum_p V_p^0 \boldsymbol{\tau}(\boldsymbol{\varepsilon}_p^{E,n})\nabla N_i(\boldsymbol{x}_p^n)$ is the grid force
    5. Apply boundary conditions on grid: e.g., $\boldsymbol{v}_{i\in D}^{n+1} = 0$ for non-slip
    6. Update particle velocity:
        (a) $\boldsymbol{v}_p^{\text{PIC}} = \sum_{i\in\text{grid}} \boldsymbol{v}_i^{n+1} N_i(\boldsymbol{x}_p^n)$
        (b) $\boldsymbol{v}_p^{\text{FLIP}} = \boldsymbol{v}_p^n + \sum_{i\in\text{grid}} (\boldsymbol{v}_i^{n+1} - \boldsymbol{v}_i^n) N_i(\boldsymbol{x}_p^n)$
        (c) $\boldsymbol{v}_p^{n+1} = C_{\text{transfer}}\boldsymbol{v}_p^{\text{FLIP}} + (1 - C_{\text{transfer}})\boldsymbol{v}_p^{\text{PIC}}$
        (d) (if APIC/AFLIP) $\boldsymbol{B}_p^{n+1} = \sum_{i\in\text{grid}} \boldsymbol{v}_i^{n+1}(\boldsymbol{x}_i^n - \boldsymbol{x}_p^n)^T N_i(\boldsymbol{x}_p^n)$
    7. (If MUSL) revisit step 3 by interpolating $\boldsymbol{v}_p^{n+1}$ from step 4 to the grid, thus overwriting $\boldsymbol{v}_i^{n+1}$
    8. Compute trial elastic strain:
        (a) $\boldsymbol{F}_p^{E,\text{trial}} = \left(\boldsymbol{I} + \Delta t \sum_{i\in\text{grid}} \boldsymbol{v}_i^{n+1}\left(\nabla N_i(\boldsymbol{x}_p^n)\right)^T\right)\boldsymbol{F}_p^{E,n}$
        (b) SVD of $\boldsymbol{F}_p^{E,\text{trial}} = \boldsymbol{U}_p^E \boldsymbol{\Sigma}_p^{E,\text{trial}} (\boldsymbol{V}_p^E)^T$
        (c) $\boldsymbol{\varepsilon}_p^{E,\text{trial}} = \ln \boldsymbol{\Sigma}_p^{E,\text{trial}}$
    9. If $y\left(\boldsymbol{\tau}\left(\boldsymbol{\varepsilon}_p^{E,\text{trial}}\right)\right) > 0$, perform return mapping: $\boldsymbol{\varepsilon}_p^{E,n+1} = RM(\boldsymbol{\varepsilon}_p^{E,\text{trial}}, \ldots)$ where $RM$ is the return mapping and update
       elastic deformation gradient $\boldsymbol{F}_p^{E,n+1} = \boldsymbol{U}_p^E e^{\boldsymbol{\varepsilon}_p^{E,n+1}} (\boldsymbol{V}_p^E)^T$, otherwise accept trial state.
    10. Update positions: $\boldsymbol{x}_p^{n+1} = \boldsymbol{x}_p^n + \Delta t \boldsymbol{v}_p^{\text{PIC}}$
```

4 Terrain and objects

Terrain and objects are described as level sets. This is similar to the treatment in, e.g., Zhao et al. (2023a), although only objects with imposed movements are considered in the context of this work. The terrain and objects can either be analytically constructed or provided as `.vdb` files (OpenVDB, 2012). Analytically creating a terrain or an object requires defining two functions: 1) an "inside"-function which, given a point in space, checks if the point is inside the object, and 2) a "normal"-function which, given a point on the surface of the object, provides the outward unit normal vector at this point. This can be accomplished by deriving from the provided `ObjectGeneral` class. For the common problem of a finite and/or moving plate, a special `ObjectPlate` is available.

As mentioned in Section 3, boundary conditions are enforced on the grid velocities. Thus, every grid node found to be inside the level sets has its velocity altered according to the specified boundary condition. The different options are listed below.

4.1 No-slip

No-slip boundary conditions are realized by overwriting the grid velocities with the velocity of the object/terrain. In particular, in the case of a stationary object D, the grid nodes inside the D have their velocities updated as

$$\boldsymbol{v}_{i\in D}^{n+1} = 0. \tag{32}$$

4.2 Frictional slip

Frictional slip can also be accomplished. As outlined in Blatny et al. (2024), an approximation to Coulomb's friction law on the grid velocities can be considered. Introducing a boundary friction parameter $\mu_b \geq 0$, the velocity $\boldsymbol{v}$ relative to the boundary is obtained as

$$\boldsymbol{v}_{i\in D}^{n+1} = \begin{cases} \boldsymbol{v}_T^* - \mu_b ||\boldsymbol{v}_N^*||_{L_2} \frac{\boldsymbol{v}_T^*}{||\boldsymbol{v}_T^*||_{L_2}} & \text{if} \quad ||\boldsymbol{v}_T^*||_{L_2} > \mu_b ||\boldsymbol{v}_N^*||_{L_2}, \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{33}$$

where $\boldsymbol{v}^*$ denotes the relative velocity (at time $n+1$) before the boundary condition is considered, and $\boldsymbol{v}_T^*$ and $\boldsymbol{v}_N^*$ are its tangential and normal components, respectively. In the special case $\mu_b = 0$, the tangential component of the relative velocity remains unchanged while the normal component vanishes. The boundary condition shown in Equation (33) is typically only applied on grid nodes where $\boldsymbol{v}_N^*$ indicates movement into the boundary such that the material can separate from the boundary.

4.3 Material-induced boundary friction (MIBF)

In the above situation, the friction parameter μ_b is a constant in space and time. However, a spatiotemporally varying friction parameter is also possible in *Matter*. This is particularly relevant in the case of the so-called $\mu(I)$-rheology (see Sections 7.5 and 7.6) where the material is described by a varying internal friction μ depending on the rate of the flow. More generally, consider the situation where a non-constant material property μ_m, which may also depend on flow conditions, can be constructed.

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

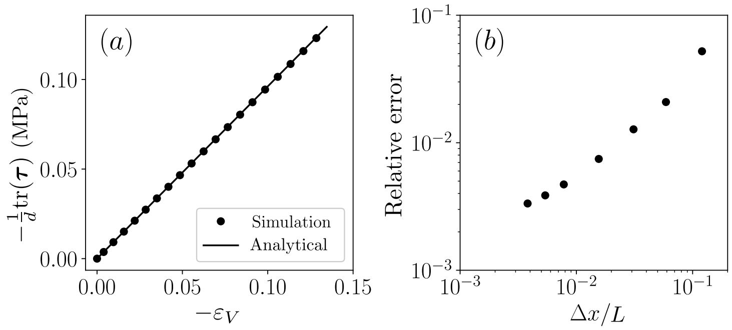

Figure 3. Compression of a cube of side length L with $E = 1$ MPa and $\nu = 0.3$, using PIC-FLIP with $C_{\text{transfer}} = 0.99$. (a) Stress-strain curve from finest discretization. (b) Convergence in the relative error of the measured bulk modulus K under finer discretization. Adapted from Blatny (2023).

The boundary friction μ_b can locally be enforced equal to μ_m by interpolating μ_m to the grid at every time step as shown in Step 3d in Algorithm 1.

This type of frictional boundary condition is named "Material-Induced Boundary Friction" (MIBF), because the friction of the boundary is determined by a material property. If μ_m does not vary in time and space, MIBF reduces to the case of a constant boundary friction.

5 Validation tests

As a simple validation of the finite strain elastic Hencky's model, consider here the compression of an elastic box. The volumetric Hencky strain $\varepsilon_V = \mathrm{tr}(\boldsymbol{\varepsilon})$ is measured, and according to Equation (4), the relation $-\frac{1}{d}\,\mathrm{tr}(\boldsymbol{\tau}) = -K\varepsilon_V$ should hold, with bulk modulus $K = \lambda + 2G/d$. Figure 3 shows that the measured stress response is consistent with the imposed elastic model and demonstrates convergence of the measured bulk modulus under finer discretization. For a more detailed analysis of the convergence properties of MPM, the reader is referred to, e.g., Steffen et al. (2008); Steffen. et al. (2008). In particular, it should be noted that the introduction of boundaries, as was needed in this specific example to compress the material, may reduce the rate of convergence. In fact, this may be more pronounced with higher-order B-splines as these increase the local support of the interpolation, thus increasing the geometric error.

In order to evaluate the various particle-grid transfer schemes (PIC, FLIP, APIC, AFLIP) and to which extent they suffer from numerical dissipation, a simple problem of an elastic box undergoing translation and rotation in the absence of gravity is considered. This is sketched in Figure 4 which also shows the evolution of the kinetic energy with time as the solid undergoes several rotations. Clearly, APIC and AFLIP conserve the kinetic energy well and much better than both PIC and FLIP, with AFLIP slightly more conservative than APIC. Notably, PIC suffers from severe dissipation and is unable to complete a full rotation, instead continuing only with translation corresponding to the drop in kinetic energy by 25% shown in Figure 4.

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

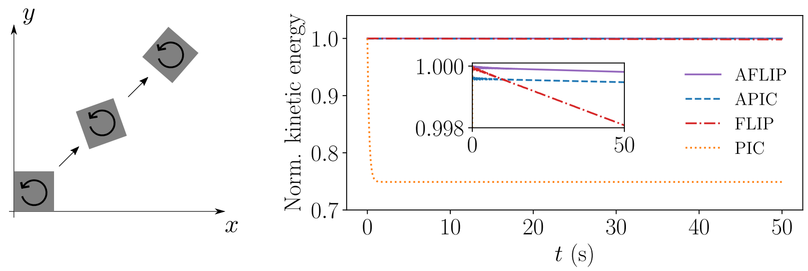

Figure 4. Elastic box with $E = 1$ MPa and $\nu = 0.3$ undergoing rotation and translation. The normalized kinetic energy of the solid is plotted, using various particle-grid transfer schemes. Given an imposed initial angular velocity of 1 rad/s, the material has completed one rotation in $2\pi \approx 6$ s. The inset plot is a zoomed view in order to distinguish FLIP, APIC and AFLIP.

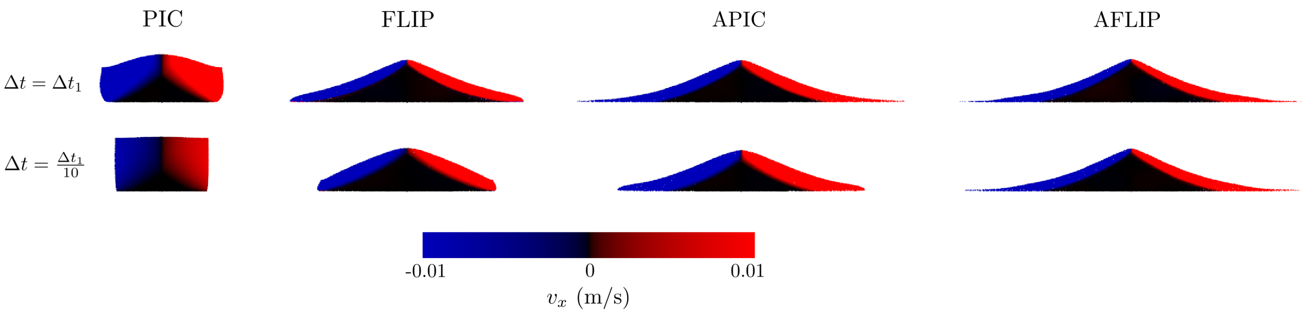

Figure 5. Illustration of the influence of the particle-grid transfer scheme on the granular collapse. Here, all snapshots are taken at time $t = 0.22$ s with $\Delta t_1 = 0.5\Delta x/\sqrt{E/\rho}$. Note that FLIP and AFLIP were used with $C_{\text{transfer}} = 0.99$. Adapted from Blatny (2023).

The above rotating box was a simple toy problem designed specifically to demonstrate the potentially severe effect of particle-grid transfer schemes in systems involving rotations. Choosing a minimally dissipative transfer scheme is especially important when modeling granular mechanics and flow problems as these are typically highly dynamic systems. Even in the relatively simple case of granular collapse under gravity, clear differences can be observed depending on the transfer scheme used, which is demonstrated in Figure 5. This figure shows a collapse of a cohesionless Drucker-Prager granular material (see Section 7.2) at a specific time slightly before the material should come to rest on the horizontal non-slip surface. With a reasonable choice of Δt restricted by the elastic wave speed, it is observed that FLIP, APIC and AFLIP give approximately similar results, while PIC induces too much numerical dissipation resulting in a delayed and viscous response. If Δt is forced 10 times smaller, 10 times as many particle-grid transfers occur, thus inducing more numerical dissipation. In that case, it can be observed from the last row in Figure 5 that only AFLIP is visually unaffected by this large increase in particle-grid transfers.

The implementation of boundary conditions is validated in Figures 6 and 7, respectively. Figure 6 shows that no significant differences between analytically specified level sets and *OpenVDB* level sets can be observed in this specific case of a particle following a frictionless parabola under gravity. In particular, both approaches lead to second-order convergence under mesh

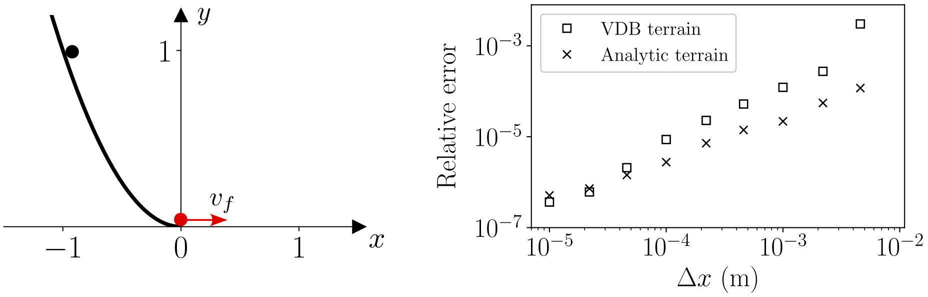

Figure 6. Validation of level set boundaries. A particle initially at rest at (-1,1) is constrained to a frictionless boundary given by $y = x^2$. The relative error between the measured bottom velocity and the expected value $v_f = \sqrt{2g}$ is plotted. Two different simulations are performed, one with an analytically specified level set and another with an *OpenVDB* level set.

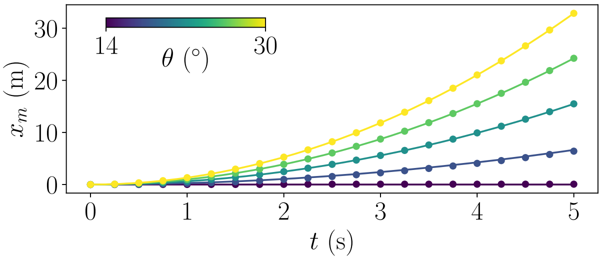

Figure 7. Mean position x_m (dots) of a stiff box sliding on a frictional plane with $\mu_b = \tan(15°)$ on various inclinations θ. Analytical solutions (continuous lines) are given by $x = \max\left(\frac{1}{2}g(\sin\theta - \mu_b\cos\theta)\right)$.

refinement. In the case of frictional slip, Figure 7 demonstrates that the material will slip if and only if $\theta > \tan(\mu_b)$. Furthermore, it confirms that the speed of the material sliding down various inclinations is consistent with the expected value from Coulomb's friction law.

6 Implementation, dependencies and performance

320 *Matter* is implemented in C++ and parallelized on shared memory using *OpenMP* (Dagum and Menon, 1998). The algorithm involves nested loops over particles and grid nodes. In order to reduce the computational complexity, these are generally implemented as outer loops over particles with the inner loop only summing over the grid nodes within the local support of the interpolation functions. Vectors and tensors are handled using the header-only linear algebra library *Eigen* (Gael et al., 2010).

Depending on the problem at hand, the initial sampling of particle positions can affect the results. Regular sampling of 325 the particles is generally discouraged as this may in some cases promote unwanted artifacts during deformation. To this end, Poisson disk distribution sampling (Bridson, 2007) is offered in *Matter* which relies on the single-header zero-dependencies implementation *tph_poisson* (Hinks, 2018). If using the provided particle sampling functions `samplingParticles`, the

Discretization				Compute time	
$N_{\text{particles}}$	Δx	Δt	Time steps	Computer A	Computer B
5k	9.01 mm	142 μs	2807	12 s	7 s
42k	4.57 mm	72 μs	5543	2 min 55 s	1 min 39 s
330k	2.23 mm	36 μs	11015	51 min 57 s	24 min 44 s
2.6M	1.15 mm	18 μs	21952	15h 52 min 8 s	7 h 55 min 31 s

Table 1. Timing of a three-dimensional granular collapse with initial dimension 0.1 m x 0.1 m confined in the in-plane direction with a width 0.05 m, using $E = 1$ MPa, $\nu = 0.3$ $\rho = 1000$ kg/m^3 and the cohesionless Drucker-Prager model (see Section 7.2) with $\mu = 0.58$. The compute time is measured until $t = 0.4$ s after equilibrium has been reached. Initial discretization with 8 particles per grid cell volume Δx^3. Using 8 threads in all cases, Computer A is a 2018 desktop workstation with Intel Xeon Gold 6136 CPUs with 192GB RAM and Computer B is a 2023 Apple MacBook M3 laptop with only 8GB RAM.

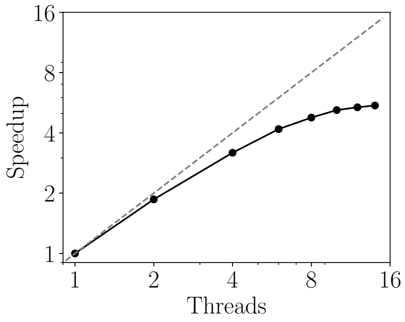

Figure 8. Speedup with number of threads in the simulation with 2.6M particles presented in last row in Table 1 using Computer A.

grid size Δx is determined to ensure a user-provided number of particles per grid cell (typically at least 4 in two dimensions and 8 in three dimensions). Particle data are saved at every chosen time frame as binary `.ply` files using the single-header zero-dependency code *tinyply* (Diakopoulos, 2018), In fact, the only required non-header-only dependency in addition to *OpenMP* is *CMake* (Hoffman, 2000). *OpenVDB* (OpenVDB, 2012) is only required in the case where an object or terrain level set is provided as a `.vdb` file. Such files may also be used to provide the level set which defines the domain of particle sampling. To this end, the `samplingParticlesVdb` function takes a `.vdb` file as input.

Table 1 lists some compute times for a granular collapse under gravity on a horizontal plane using an increasingly fine discretization. For the finest discretization (here 2.6 million particles), Figure 8 shows the speedup with increasing threads, suggesting a max speedup of not more than 6. In comparison, de Vaucorbeil et al. (2021) reports speedups on *Karamelo* with shared-memory parallelization to around 8. Speedup may depend on the system and compiler, and a detailed analysis of this has not been conducted.

7 Elasto-viscoplastic models in *Matter*

This section lists the available elasto-viscoplastic models and rheologies which are currently implemented. Letting as before d denote the number of spatial dimensions, all models are formulated in terms of the stress invariants

$$p = -\frac{1}{d}\,\mathrm{tr}(\boldsymbol{\tau}) \tag{34}$$

and

$$q = \frac{1}{\sqrt{2}}||\,\mathrm{dev}(\boldsymbol{\tau})|| \tag{35}$$

where p is termed the isotropic pressure and q the equivalent shear stress, using the notation $||\boldsymbol{A}|| = \sqrt{\boldsymbol{A}:\boldsymbol{A}} = \mathrm{tr}(\boldsymbol{A}\boldsymbol{A})$ and $\mathrm{dev}(\boldsymbol{A}) = \boldsymbol{A} - \frac{1}{d}\,\mathrm{tr}(\boldsymbol{A})\boldsymbol{I}$ for a symmetric second-order tensor $\boldsymbol{A}$. The invariant $\sqrt{3}q$ is often termed the von Mises equivalent stress.

7.1 Von Mises

Although rarely used in the context of granular media due to its pressure-independency, the von Mises model, typically employed for many metals, has a yield function given by

$$y(p,q) = q - q_y \tag{36}$$

where $q_y > 0$ is the von Mises yield stress. The plastic flow rule is generally chosen associative, i.e., $g = y$.

7.2 Drucker-Prager

The Drucker-Prager model (Drucker and Prager, 1952) extends the von Mises model with pressure-dependency,

$$y(p,q) = q - q_y(p), \quad \text{where} \quad q_y = \mu p + q_c \tag{37}$$

where $\mu > 0$ is the internal friction and $q_c \geq 0$ is a cohesive stress (or shear strength). Since an associative flow rule would result exclusively in volume increase, i.e., $\mathrm{tr}(\boldsymbol{l}^P) \neq 0$, this model is often complemented with a non-associative rule, typically using the von Mises plastic potential, i.e., $g = q$. In *Matter*, such a flow rule is implemented and combined with the standard "cone tip projection" to address the singularity in the Drucker-Prager yield surface, see, e.g., Blatny et al. (2023). As a result of this singularity treatment, some permanent volumetric expansion under tension will occur, which are corrected using the volume correction scheme of Pradhana (2017).

A strain-softening Drucker-Prager model is also introduced, in which q_c is no longer a constant, but decreases with the equivalent plastic shear strain ε_S^P according to a softening parameter $\xi \geq 0$ through

$$q_c = q_c^0 e^{-\xi \varepsilon_S^P} \tag{38}$$

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

where $q_c^0 > 0$ is the initial cohesive stress before any deformation occurs. The equivalent plastic shear strain ε_S^P is a non-negative scalar defined by

$$\varepsilon_S^P = \int ||\operatorname{dev}(\boldsymbol{l}^P)||\, dt \approx \sum_n ||\operatorname{dev}(\Delta \boldsymbol{\varepsilon}^{P,n})||. \tag{39}$$

This formulation of the strain-softening Drucker-Prager model has been used in, e.g., Blatny et al. (2021) and Blatny et al. (2023). If $\xi \to \infty$, this model promotes a brittle behavior as the cohesion would be removed after initial failure. Other formulations of the strain-softening behavior are also possible.

7.3 Modified Cam-Clay

Critical State Soil Mechanics (CSSM) (Schofield and Wroth, 1968) is a well-established theory in the geomechanics/geotechnics community for describing the mechanics of granular media as elastoplastic solids. This theory is conceived from the idea that there exists a critical state, where shear deformations can go on without further changes to volume and stress. In particular, a Critical State Line (CSL) can be defined in (p,q)-space along which all such critical states can exist. Although there exist different models that incorporate the main ideas of CSSM, the Modified Cam-Clay (MCC) model is arguably one of the most commonly used in numerical implementations. First formulated by Roscoe and Burland (1968), the MCC model is characterized by an evolving yield surface of ellipsoidal shape in principal stress space.

An MCC yield function that incorporates cohesion can be defined as

$$y(p,q) = q - q_y(p), \quad \text{where} \quad q_y = \mu\sqrt{(p+\beta p_c)(p_c - p)}, \tag{40}$$

where $\mu > 0$ defines the slope of the CSL, $p_c \geq 0$ is an isotropic compressive strength and $\beta \geq 0$ is a dimensionless measure of cohesion. All critical states (on the CSL) are found on a Drucker-Prager-like surface $q = \mu(p + \beta p_c)$, and for exactly this reason the symbol μ is here also used for the slope of the CSL. In Equation (40), $-\beta p_c$ represents the isotropic tensile strength, and β can therefore also be described as the ratio between tensile and compressive strength. The case $\beta = 0$ represents the cohesionless case. In this case, the material can not sustain any tensile stress states. If $\beta > 0$, there will always be a non-zero tensile strength (unless $p_c = 0$ which represents the stress-free state). As such, cohesion is always present in this particular formulation of the model.

An associative flow rule is used, that is, $g = y$. As such, both dilation and compaction can occur. An imposed hardening law ensures that the yield surface will shrink or expand, respectively, until critical state is reached. Two hardening laws are considered, both dependent on the equivalent plastic volumetric Hencky strain ε_V^P defined as

$$\varepsilon_V^P = \int \operatorname{tr}(\boldsymbol{l}^P)\, dt \approx \sum_n \operatorname{tr}(\Delta \boldsymbol{\varepsilon}^{P,n}). \tag{41}$$

The first hardening law is given by

$$p_c(\varepsilon_V^P) = p_c^0 e^{-\xi \varepsilon_V^P} \tag{42}$$

reminiscent of the hardening laws typically used in Cam-Clay models. The second, as presented in Gaume et al. (2018) inspired by Ortiz and Pandolfi (2004), is given by

$$p_c(\varepsilon_V^P) = K\sinh\left(-\xi\varepsilon_V^P + \sinh^{-1}\left(\frac{p_c^0}{K}\right)\right) \tag{43}$$

where in both cases $\xi \geq 0$ is a dimensionless parameter and $p_c^0 \geq 0$ is an initial compressive strength before any plastic compaction/dilation has occurred.

7.4 Overstress models

Rate-dependent plasticity is here considered through the general overstress model of Perzyna (1963, 1966) where a yield surface $y = 0$ defines the onset of viscoplastic deformation, and a separate expression for the plastic strain rate is imposed. Given a yield stress q_y, potentially dependent on p, the plastic shear strain rate $\dot{\gamma}_S = \sqrt{2}||\mathrm{dev}(\boldsymbol{l}^P)||$ is imposed using the formulation of Perić (1993), in particular,

$$\dot{\gamma}_S = \begin{cases} 0 & \text{if } y < 0 \\ \frac{1}{t_{\text{visc}}}\left(\left(\frac{q}{q_y(p)}\right)^{1/s} - 1\right) & \text{if } y \geq 0 \end{cases} \tag{44}$$

where t_{visc} is a so-called viscous time determining the viscosity of the material and s is a non-negative exponent. The formulation of Perić (1993) has the advantage of recovering the rate-independent limit as $s \to 0$ (Neto et al., 2008). The yield stress $q_y(p)$ in Equation (44) can be chosen as that from von Mises, i.e., Equation (36), Drucker-Prager, i.e., Equation (37) or Modified Cam-Clay, i.e., Equation (40), respectively. Not considering the elastic behavior below yield, the overstress model of Equation (44) with the choice of a von Mises yield describes Herschel–Bulkley viscoplasticity (Herschel and Bulkley, 1926). In the special case $s = 1$, this further reduces to a Bingham fluid (Bingham, 1922).

7.5 The $\mu(I)$-rheology

The $\mu(I)$-rheology (GDR MiDi, 2004; Jop et al., 2006) is a well-established theory to describe dense granular flow where the apparent friction depends on, and only on, a non-dimensional number termed the inertial number I defined by the pressure, shear rate as well as the intrinsic grain diameter d_g and density ρ_g. Following the general idea of Dunatunga and Kamrin (2015), this rheology can be incorporated in an elastoplastic framework as a Drucker-Prager model with varying friction $\mu = \mu(I)$. The typical functional form of $\mu(I)$ from Jop et al. (2005) is assumed where μ is bounded by a lower value μ_1 and an upper value μ_2, in particular,

$$\mu(I) = \mu_1 + \frac{\mu_2 - \mu_1}{I_0/I + 1} = \mu_1 + \frac{\mu_2 - \mu_1}{\omega\frac{\sqrt{p}}{\dot{\gamma}_S} + 1} \tag{45}$$

where I_0 is a constant parameter and for convenience defining $\omega = \frac{I_0}{d_g\sqrt{\rho_g}}$. As such, this model has three rheological parameters, μ_1, μ_2 and ω, in addition to the cohesion q_c and the two elastic moduli. In order to treat cohesive problems, i.e., when the pressure p can become negative, the inertial number is here defined following Berger et al. (2015); Vo et al. (2020) as

$$I = \frac{\dot{\gamma}_S d_g}{\sqrt{\bar{p}/\rho_g}}. \tag{46}$$

where an effective pressure

$$\bar{p} = p + q_c/\mu_1 \tag{47}$$

is introduced by adding to p the isotropic tensile strength (i.e., the minimal p).

7.6 Critical state $\mu(I)$-rheology

The critical state $\mu(I)$-rheology introduced by Blatny et al. (2024) is available in *Matter* for modeling compressible and cohesive granular flow, and the reader is referred to this article for further details and for the validation of its implementation. This model is conceived as a combination of the MCC model from Section 7.3 and the $\mu(I)$-rheology from Section 7.5. In essence, this model constructs the CSL to be dependent on the inertial number I such that in the flowing limit it retrieves the properties of the $\mu(I)$-rheology. The inertial number is defined similarly as in Equation (46), using

$$\bar{p} = p + \beta p_c \tag{48}$$

instead of Equation (47) following an analogous reasoning, and employing the same functional form of $\mu(I)$ as in Equation (45).

In addition to being able to combine the solid and liquid granular phase, the elastoplastic MPM approach has some clear advantages over fluid solvers that implement the $\mu(I)$-rheology. First, due to the introduction of elasticity, there is no need for regularization for low inertial numbers as is otherwise required (Barker and Gray, 2017). Second, no special treatment of the free-surface is necessary as MPM by construction handles this naturally. Third, connectivity to the boundary is not required and lift-offs are trivially handled.

8 Examples

Figures 9, 10 and 11 demonstrate some three-dimensional granular collapse and flow simulations using different constitutive models on small/simple and large/complex terrains. Figure 9 shows a cohesive granular collapse where the material is described by the Drucker-Prager model of Section 7.2. In addition to the tilted bottom plate, the collapse is restricted by a back wall and two side walls, all using the available plate objects described in Section 4. The material is visualized according to the plastic shear strain rate, highlighting the shear bands along which significant deformation occurs during the collapse. A similar visualization of the plastic shear strain rate is also used in Figure 10 which shows granular flow through a silo where the material is described by the elastoplastic $\mu(I)$-rheology of Section 7.5. Here, the silo is created analytically, specifically as a hyperbolic tangent curve $x = \tanh(y) + 1$ rotated along the vertical y-axis with a cut $y = y_{\text{cut}}$ to define the silo opening through which the material flows. Using a mountain provided as a `.vdb`-file, Figure 11 shows the flow of a material described by an

https://doi.org/10.5194/egusphere-2025-1157
Preprint. Discussion started: 1 April 2025

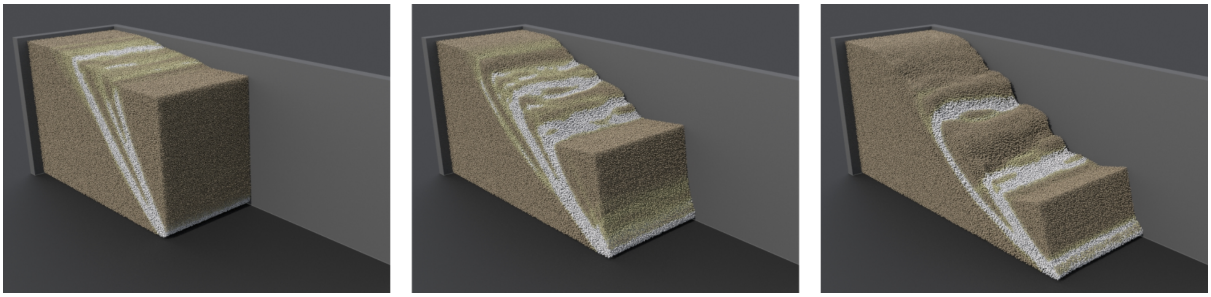

Figure 9. Three-dimensional cohesive granular collapse on a 10° tilted non-slip terrain using the cohesive Drucker-Prager model ($\mu = 0.58$ and $q_c = 100$ Pa). The initial dimensions of the granular column was 20 x 14 x 10 cm, discretized with 1.6 million particles with initially 8 particles per cell. The displayed snapshots are taken at 0.1, 0.2 and 0.3 s, respectively, after release. In order to visualize the material inside the left side wall is not shown. The material is colored according to its plastic shear strain rate $\dot{\gamma}_S$, with brighter (whiter) colors representing higher strain rates, thus highlighting the shear bands.

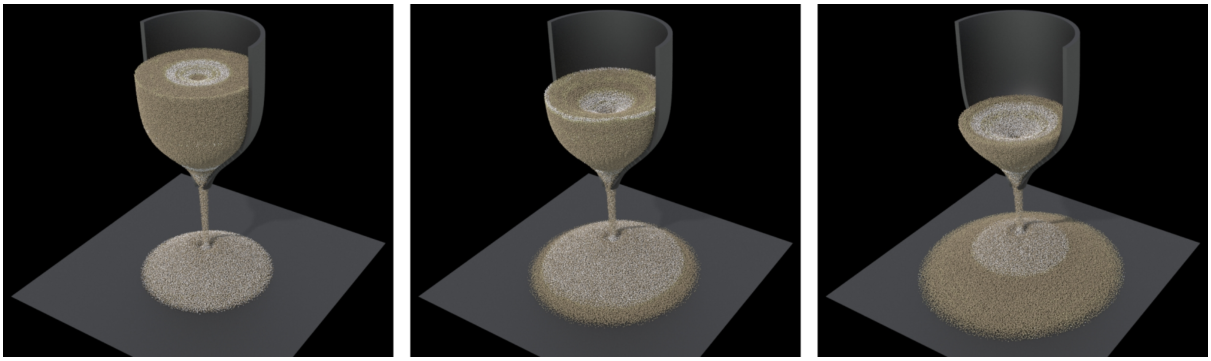

Figure 10. Three-dimensional flow from a silo using the elastoplastic $\mu(I)$-rheology ($\mu_1 = 0.58$, $\mu_2 = 0.84$, $\omega = 2.2$ kg$^{-\frac{1}{2}}$m$^{\frac{1}{2}}$). With a maximum radius of 2 m and height of 4 m, the silo is elevated 4 m above the ground where the deposit comes to rest. Only half of the silo boundary is shown in order to visualize the material moving inside. The displayed snapshots are taken at 50, 100 and 150 s, respectively, after release. The material is colored according to its plastic shear strain rate $\dot{\gamma}_S$, with brighter (whiter) colors representing higher strain rates, thus highlighting the shear bands.

overstress model (Section 7.4) with a Drucker-Prager yield. Enforcing MIBF, the non-constant internal friction is used as the basal friction. Notably, Figure 11 also demonstrates that particle lift-off from the terrain is possible.

9 Concluding remarks

This article has presented *Matter*, an open-source MPM solver with a wide selection of elasto-viscoplastic models to describe the mechanics and rheology of granular media. The few cases presented in this article provide a glimpse into its capabilities in simulating granular media, from small-scale laboratory settings to flow on large-scale complex terrains. This includes terrains

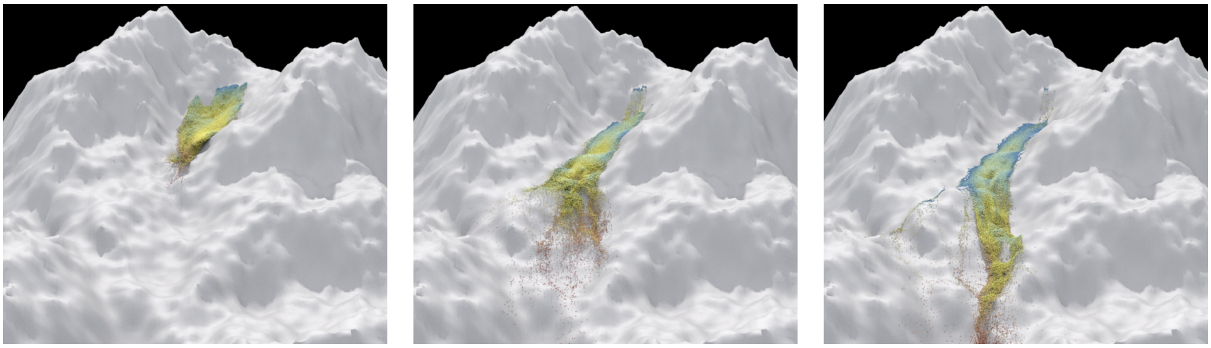

Figure 11. Three-dimensional flow on complex terrain with MIBF, using an overstress model with ($t_{\text{visc}} = 0.12$ s, $s = 1$) combined with a non-cohesive Drucker-Prager yield ($\mu = 0.36$). The displayed snapshots are taken at 6, 10 and 14 seconds, respectively, after release. Using 425k particles and $\Delta x = 0.43$ m (with initially 8 particles per grid cell) the compute time from the leftmost to rightmost snapshot during 8 s was 47 minutes on 8 Intel Xeon Gold 6136 CPUs.

with various boundary conditions, including the proposed method of material-induced boundary friction. Built on a validated finite strain elastoplastic framework, *Matter* leverages recent advances in MPM particle-grid transfer schemes that reduce numerical dissipation.

The main limitations of the software should be highlighted. For example, modeling granular media immersed in or interacting with water or other fluids is beyond the scope of this software. Along these lines, the MPM community has proposed various strategies for handling multiple phases on one or more grids (e.g., Yerro et al., 2015; Higo et al., 2025), or proposed CFD coupling strategies (e.g., Higo et al., 2010; Tran et al., 2024). Moreover, including more than one type of material in the same simulation is currently not supported. Nevertheless, different constitutive parameters can be assigned to different particles with minimal modifications to the code. In addition, schemes to circumvent volumetric locking could also be considered in the future, in particular with the simple and elegant method of Zhao et al. (2023b). Although *Matter* represents a lightweight and efficient option for most problems that can be run on laptops and desktops, its current implementation is not optimized for sparse problems for which excessive simulation times can be witnessed. This could be addressed primarily by transitioning away from dense *Eigen* grids, however, the current strategy ensures a readable and adaptable code.

With the growing interest in understanding and modeling cohesive granular flow, particularly cohesive flows, this software aims to be a useful tool for both fundamental research and practical applications. This includes real-scale natural hazard applications, e.g., snow avalanches. While such mass movements are typically assessed with depth- or thickness-integrated models (Sampl and Zwinger, 2004; Christen et al., 2010; Tonnel et al., 2023), *Matter* represents an alternative three-dimensional approach to simulate these processes. Future developments aim to incorporate more advanced constitutive models for granular rheology, including non-local models, which are not yet considered. The community is also encouraged to contribute by integrating their own models into the framework.

Code availability. The source code and user manual is available at https://github.com/larsblatny/matter/ (Blatny and Gaume, 2025) (DOI:10.5281/zenodo.15052337) under the GPL-3.0 license.

Author contributions. L. B. initiated and developed the software, and wrote the paper. J. G. supervised the work and reviewed the manuscript.

Competing interests. The authors declare no competing interests.

Disclaimer. The software is provided "as is" with no guarantees of any kind. The creators are not responsible for any issues, damages, or losses that may result from using the software.

Acknowledgements. The authors wish to thank Tobias Verheijen for improving return mapping schemes, as well as everyone who has inspired the development of *Matter*, in particular its users who have shared valuable feedback.

References

Abramian, A., Staron, L., and Lagrée, P.-Y.: The Slumping of a Cohesive Granular Column: Continuum and Discrete Modeling, Journal of Rheology, 64, 1227–1235, https://doi.org/10.1122/8.0000049, 2020.

Anura3D: Anura3D, https://github.com/Anura3D, 2017.

Badetti, M., Fall, A., Hautemayou, D., Chevoir, F., Aimedieu, P., Rodts, S., and Roux, J.-N.: Rheology and Microstructure of Unsaturated Wet Granular Materials: Experiments and Simulations, Journal of Rheology, 62, 1175–1186, https://doi.org/10.1122/1.5026979, 2018.

Bardenhagen, S. G. and Kober, E. M.: The Generalized Interpolation Material Point Method, Computer Modeling in Engineering and Sciences, 5, 477–496, https://doi.org/10.3970/cmes.2004.005.477, 2004.

Barker, T. and Gray, J. M. N. T.: Partial Regularisation of the Incompressible μ(I)-Rheology for Granular Flow, Journal of Fluid Mechanics, 828, 5–32, https://doi.org/10.1017/jfm.2017.428, 2017.

Belytschko, T., Lu, Y. Y., and Gu, L.: Element-Free Galerkin Methods, International Journal for Numerical Methods in Engineering, 37, 229–256, https://doi.org/10.1002/nme.1620370205, 1994.

Berger, N., Azéma, E., Douce, J.-F., and Radjai, F.: Scaling Behaviour of Cohesive Granular Flows, Europhysics Letters, 112, 64 004, https://doi.org/10.1209/0295-5075/112/64004, 2015.

Bingham, E.: Fluidity and Plasticity, McGraw-Hill, 1922.

Blatny, L. and Gaume, J.: Matter, https://github.com/larsblatny/matter/, https://doi.org/10.5281/zenodo.15052337, 2025.

Blatny, L., Löwe, H., Wang, S., and Gaume, J.: Computational Micromechanics of Porous Brittle Solids, Computers and Geotechnics, 140, 104 284, https://doi.org/10.1016/j.compgeo.2021.104284, 2021.

Blatny, L., Löwe, H., and Gaume, J.: Microstructural Controls on the Plastic Consolidation of Porous Brittle Solids, Acta Materialia, 250, 118 861, https://doi.org/10.1016/j.actamat.2023.118861, 2023.

Blatny, L., Gray, J., and Gaume, J.: A Critical State μ(I)-Rheology Model for Cohesive Granular Flows, Journal of Fluid Mechanics, 997, A67, https://doi.org/10.1017/jfm.2024.643, 2024.

Blatny, L. K. U.: Modeling the Mechanics and Rheology of Porous and Granular Media: An Elastoplastic Continuum Approach, Ph.D. thesis, Lausanne, EPFL, https://doi.org/10.5075/EPFL-THESIS-10267, 2023.

Brackbill, J., Kothe, D., and Ruppel, H.: Flip: A Low-Dissipation, Particle-in-Cell Method for Fluid Flow, Computer Physics Communications, 48, 25–38, https://doi.org/10.1016/0010-4655(88)90020-3, 1988.

Bridson, R.: Fast Poisson Disk Sampling in Arbitrary Dimensions, ACM SIGGRAPH 2007 Sketches Program, 2007.

Bruhns, O., Xiao, H., and Meyers, A.: Self-Consistent Eulerian Rate Type Elasto-Plasticity Models Based upon the Logarithmic Stress Rate, International Journal of Plasticity, 15, 479–520, https://doi.org/10.1016/S0749-6419(99)00003-0, 1999.

Chen, H., Zhao, S., and Zhao, J.: A Sparse-Memory-Encoding GPU-MPM Framework for Large-Scale Simulations of Granular Flows, Computers and Geotechnics, 180, 107 113, https://doi.org/10.1016/j.compgeo.2025.107113, 2025.

Christen, M., Kowalski, J., and Bartelt, P.: RAMMS: Numerical Simulation of Dense Snow Avalanches in Three-Dimensional Terrain, Cold Regions Science and Technology, 63, 1–14, https://doi.org/10.1016/j.coldregions.2010.04.005, 2010.

Coombs, W. M. and Augarde, C. E.: AMPLE: A Material Point Learning Environment, Advances in Engineering Software, 139, 102 748, https://doi.org/10.1016/j.advengsoft.2019.102748, 2020.

Cremonesi, M., Franci, A., Idelsohn, S., and Oñate, E.: A State of the Art Review of the Particle Finite Element Method (PFEM), Archives of Computational Methods in Engineering, 27, 1709–1735, https://doi.org/10.1007/s11831-020-09468-4, 2020.

Dadvand, P., Rossi, R., and Oñate, E.: An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications, Archives of Computational Methods in Engineering, 17, 253–297, https://doi.org/10.1007/s11831-010-9045-2, 2010.

Dagum, L. and Menon, R.: OpenMP: an industry standard API for shared-memory programming, IEEE Computational Science and Engineering, 5, 46–55, 1998.

Davison De St. Germain, J., McCorquodale, J., Parker, S., and Johnson, C.: Uintah: A Massively Parallel Problem Solving Environment, in: Proceedings the Ninth International Symposium on High-Performance Distributed Computing, pp. 33–41, IEEE Comput. Soc, Pittsburgh, PA, USA, ISBN 978-0-7695-0783-5, https://doi.org/10.1109/HPDC.2000.868632, 2000.

de Goes, F., Wallez, C., Huang, J., Pavlov, D., and Desbrun, M.: Power Particles: An Incompressible Fluid Solver Based on Power Diagrams, ACM Transactions on Graphics, 34, 1–11, https://doi.org/10.1145/2766901, 2015.

de Vaucorbeil, A., Nguyen, V. P., and Hutchinson, C. R.: A Total-Lagrangian Material Point Method for Solid Mechanics Problems Involving Large Deformations, Computer Methods in Applied Mechanics and Engineering, 360, 112 783, https://doi.org/10.1016/j.cma.2019.112783, 2020a.

de Vaucorbeil, A., Nguyen, V. P., Sinaie, S., and Wu, J. Y.: Material Point Method after 25 Years: Theory, Implementation, and Applications, in: Advances in Applied Mechanics, vol. 53, pp. 185–398, Elsevier, ISBN 978-0-12-820989-9, https://doi.org/10.1016/bs.aams.2019.11.001, 2020b.

de Vaucorbeil, A., Nguyen, V. P., and Nguyen-Thanh, C.: Karamelo: An Open Source Parallel C++ Package for the Material Point Method, Computational Particle Mechanics, 8, 767–789, https://doi.org/10.1007/s40571-020-00369-8, 2021.

Diakopoulos, D.: Tinyply, https://github.com/ddiakopoulos/tinyply, 2018.

Donea, J., Giuliani, S., and Halleux, J.: An Arbitrary Lagrangian-Eulerian Finite Element Method for Transient Dynamic Fluid-Structure Interactions, Computer Methods in Applied Mechanics and Engineering, 33, 689–723, https://doi.org/10.1016/0045-7825(82)90128-1, 1982.

Drucker, D. and Prager, W.: Soil Mechanics and Plastic Analysis or Limit Design, Quarterly of applied mathematics, 10, 157–165, 1952.

Dunatunga, S. and Kamrin, K.: Continuum Modelling and Simulation of Granular Flows through Their Many Phases, Journal of Fluid Mechanics, 779, 483–513, https://doi.org/10.1017/jfm.2015.383, 2015.

Duverger, S., Duriez, J., Philippe, P., and Bonelli, S.: Critical Comparison of Motion Integration Strategies and Discretization Choices in the Material Point Method, Archives of Computational Methods in Engineering, https://doi.org/10.1007/s11831-024-10170-y, 2024.

Fei, Y., Guo, Q., Wu, R., Huang, L., and Gao, M.: Revisiting Integration in the Material Point Method: A Scheme for Easier Separation and Less Dissipation, ACM Transactions on Graphics, 40, 1–16, https://doi.org/10.1145/3450626.3459678, 2021.

Fu, C., Guo, Q., Gast, T., Jiang, C., and Teran, J.: A Polynomial Particle-in-Cell Method, ACM Transactions on Graphics, 36, 1–12, https://doi.org/10.1145/3130800.3130878, 2017.

Gael, G., Benoit, J., and others: Eigen, http://eigen.tuxfamily.org, 2010.

Gao, M., Wang, X., Wu, K., Pradhana, A., Sifakis, E., Yuksel, C., and Jiang, C.: GPU Optimization of Material Point Methods, ACM Transactions on Graphics, 37, 1–12, https://doi.org/10.1145/3272127.3275044, 2018.

Gaume, J., Gast, T., Teran, J., van Herwijnen, A., and Jiang, C.: Dynamic Anticrack Propagation in Snow, Nature Communications, 9, 3047, https://doi.org/10.1038/s41467-018-05181-w, 2018.

GDR MiDi: On Dense Granular Flows, The European Physical Journal E, 14, 341–365, https://doi.org/10.1140/epje/i2003-10153-0, 2004.

Gingold, R. A. and Monaghan, J. J.: Smoothed Particle Hydrodynamics: Theory and Application to Non-Spherical Stars, Monthly Notices of the Royal Astronomical Society, 181, 375–389, https://doi.org/10.1093/mnras/181.3.375, 1977.

Harlow, F. H.: The Particle-in-Cell Computing Method for Fluid Dynamics, Methods in Computational Physics, 3, 319–343, 1964.

Herschel, W. H. and Bulkley, R.: Konsistenzmessungen von Gummi-Benzollösungen, Kolloid-Zeitschrift, 39, 291–300, https://doi.org/10.1007/BF01432034, 1926.

Higo, Y., Oka, F., Kimoto, S., Morinaka, Y., Goto, Y., and Chen, Z.: A Coupled MPM-FDM Analysis Method for Multi-Phase Elasto-Plastic Soils, Soils and Foundations, 50, 515–532, https://doi.org/10.3208/sandf.50.515, 2010.

Higo, Y., Takegawa, Y., Zhu, F., and Uchiyama, D.: A Three-Phase Two-Point MPM for Large Deformation Analysis of Unsaturated Soils, Computers and Geotechnics, 177, 106 860, https://doi.org/10.1016/j.compgeo.2024.106860, 2025.

Hinks, T.: Tph_poisson, https://github.com/thinks/poisson-disk-sampling, 2018.

Hoffman, W.: CMake, https://cmake.org/, 2000.

Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., and Jiang, C.: A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling, ACM Transactions on Graphics, 37, 1–14, https://doi.org/10.1145/3197517.3201293, 2018a.

Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., and Jiang, C.: A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling, ACM Transactions on Graphics, 37, 1–14, https://doi.org/10.1145/3197517.3201293, 2018b.

Hu, Y., Li, T.-M., Anderson, L., Ragan-Kelley, J., and Durand, F.: Taichi: A Language for High-Performance Computation on Spatially Sparse Data Structures, ACM Transactions on Graphics, 38, 1–16, https://doi.org/10.1145/3355089.3356506, 2019.

Iaconeta, I., Larese, A., Rossi, R., and Oñate, E.: A Stabilized Mixed Implicit Material Point Method for Non-Linear Incompressible Solid Mechanics, Computational Mechanics, 63, 1243–1260, https://doi.org/10.1007/s00466-018-1647-9, 2019.

Idelsohn, S., Oñate, E., and Pin, F. D.: The Particle Finite Element Method: A Powerful Tool to Solve Incompressible Flows with Free-Surfaces and Breaking Waves, International Journal for Numerical Methods in Engineering, 61, 964–989, https://doi.org/10.1002/nme.1096, 2004.

Jiang, C., Schroeder, C., Selle, A., Teran, J., and Stomakhin, A.: The Affine Particle-in-Cell Method, ACM Transactions on Graphics, 34, 1–10, https://doi.org/10.1145/2766996, 2015.

Jiang, C., Schroeder, C., Teran, J., Stomakhin, A., and Selle, A.: The Material Point Method for Simulating Continuum Materials, in: ACM SIGGRAPH 2016 Courses, SIGGRAPH '16, Association for Computing Machinery, New York, NY, USA, ISBN 978-1-4503-4289-6, https://doi.org/10.1145/2897826.2927348, 2016.

Jiang, C., Schroeder, C., and Teran, J.: An Angular Momentum Conserving Affine-Particle-in-Cell Method, Journal of Computational Physics, 338, 137–164, https://doi.org/10.1016/j.jcp.2017.02.050, 2017.

Jop, P., Forterre, Y., and Pouliquen, O.: Crucial Role of Sidewalls in Granular Surface Flows: Consequences for the Rheology, Journal of Fluid Mechanics, 541, 167, https://doi.org/10.1017/S0022112005005987, 2005.

Jop, P., Forterre, Y., and Pouliquen, O.: A Constitutive Law for Dense Granular Flows, Nature, 441, 727–730, https://doi.org/10.1038/nature04801, 2006.

Khamseh, S., Roux, J.-N., and Chevoir, F.: Flow of Wet Granular Materials: A Numerical Study, Physical Review E, 92, 022 201, https://doi.org/10.1103/PhysRevE.92.022201, 2015.

Klár, G.: Simulation of Granular Media with the Material Point Method, Ph.D. thesis, UCLA, 2016.

Ku, Q., Zhao, J., Mollon, G., and Zhao, S.: Compaction of Highly Deformable Cohesive Granular Powders, Powder Technology, 421, 118 455, https://doi.org/10.1016/j.powtec.2023.118455, 2023.

Kumar, K., Salmond, J., Kularathna, S., Wilkes, C., Tjung, E., Biscontin, G., and Soga, K.: Scalable and Modular Material Point Method for Large-Scale Simulations, in: 2nd International Conference on the Material Point Method for Modeling Soil-Water-Structure Interaction, p. mpm2019.43, arXiv, Cambridge, UK, https://doi.org/10.48550/ARXIV.1909.13380, 2019.

Libersky, L. D., Petschek, A. G., Carney, T. C., Hipp, J. R., and Allahdadi, F. A.: High Strain Lagrangian Hydrodynamics, Journal of Computational Physics, 109, 67–75, https://doi.org/10.1006/jcph.1993.1199, 1993.

Liu, W. K., Jun, S., and Zhang, Y. F.: Reproducing Kernel Particle Methods, International Journal for Numerical Methods in Fluids, 20, 1081–1106, https://doi.org/10.1002/fld.1650200824, 1995.

Love, E. and Sulsky, D.: An Unconditionally Stable, Energy–Momentum Consistent Implementation of the Material-Point Method, Computer Methods in Applied Mechanics and Engineering, 195, 3903–3925, https://doi.org/10.1016/j.cma.2005.06.027, 2006.

Lucy, L. B.: A Numerical Approach to the Testing of the Fission Hypothesis, The Astronomical Journal, 82, 1013, https://doi.org/10.1086/112164, 1977.

Macaulay, M. and Rognon, P.: Viscosity of Cohesive Granular Flows, Soft Matter, 17, 165–173, https://doi.org/10.1039/D0SM01456G, 2021.

Mandal, S., Nicolas, M., and Pouliquen, O.: Insights into the Rheology of Cohesive Granular Media, Proceedings of the National Academy of Sciences, 117, 8366–8373, https://doi.org/10.1073/pnas.1921778117, 2020.

Mandal, S., Nicolas, M., and Pouliquen, O.: Rheology of Cohesive Granular Media: Shear Banding, Hysteresis, and Nonlocal Effects, Physical Review X, 11, 021 017, https://doi.org/10.1103/PhysRevX.11.021017, 2021.

Monaghan, J.: Simulating Free Surface Flows with SPH, Journal of Computational Physics, 110, 399–406, https://doi.org/10.1006/jcph.1994.1034, 1994.

Moresi, L., Dufour, F., and Mühlhaus, H.-B.: A Lagrangian Integration Point Finite Element Method for Large Deformation Modeling of Viscoelastic Geomaterials, Journal of Computational Physics, 184, 476–497, https://doi.org/10.1016/S0021-9991(02)00031-1, 2003.

Nairn, J. A.: NairnMPM, https://github.com/nairnj/nairn-mpm-fea, 2015.

Neto, E. A. d. S., Perić, D., and Owen, D. R. J.: Computational Methods for Plasticity: Theory and Applications, Wiley, Chichester, West Sussex, UK, ISBN 978-0-470-69452-7 978-0-470-69462-6, 2008.

Nguyen, V. P., de Vaucorbeil, A., and Bordas, S.: The Material Point Method: Theory, Implementations and Applications, Scientific Computation, Springer International Publishing, Cham, ISBN 978-3-031-24069-0 978-3-031-24070-6, https://doi.org/10.1007/978-3-031-24070-6, 2023.

Oliver, J., Cante, J. C., Weyler, R., González, C., and Hernandez, J.: Particle Finite Element Methods in Solid Mechanics Problems, in: Computational Plasticity, edited by Oñate, E. and Owen, R., vol. 7, pp. 87–103, Springer Netherlands, Dordrecht, ISBN 978-1-4020-6576-7 978-1-4020-6577-4, https://doi.org/10.1007/978-1-4020-6577-4_6, 2007.

OpenVDB: OpenVDB, http://www.openvdb.org., 2012.

Ortiz, M. and Pandolfi, A.: A Variational Cam-clay Theory of Plasticity, Computer Methods in Applied Mechanics and Engineering, 193, 2645–2666, https://doi.org/10.1016/j.cma.2003.08.008, 2004.

Perić, D.: On a Class of Constitutive Equations in Viscoplasticity: Formulation and Computational Issues, International Journal for Numerical Methods in Engineering, 36, 1365–1393, https://doi.org/10.1002/nme.1620360807, 1993.

Perzyna, P.: The Constitutive Equations for Rate Sensitive Plastic Materials, Quarterly of applied mathematics, 20, 321–332, https://doi.org/10.1090/qam/144536, 1963.

Perzyna, P.: Fundamental Problems in Viscoplasticity, in: Advances in Applied Mechanics, vol. 9, pp. 243–377, Elsevier, ISBN 978-0-12-002009-6, https://doi.org/10.1016/S0065-2156(08)70009-7, 1966.

Pradhana, A.: Multiphase Simulation Using Material Point Method, Ph.D. thesis, UCLA, 2017.

Qu, Z., Li, M., De Goes, F., and Jiang, C.: The Power Particle-in-Cell Method, ACM Transactions on Graphics, 41, 1–13, https://doi.org/10.1145/3528223.3530066, 2022.

Radjai, F. and Richefeu, V.: Bond Anisotropy and Cohesion of Wet Granular Materials, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 367, 5123–5138, https://doi.org/10.1098/rsta.2009.0185, 2009.

Radjai, F., Roux, J.-N., and Daouadji, A.: Modeling Granular Materials: Century-Long Research across Scales, Journal of Engineering Mechanics, 143, 04017 002, https://doi.org/10.1061/(ASCE)EM.1943-7889.0001196, 2017.

Raymond, S. J., Jones, B., and Williams, J. R.: A Strategy to Couple the Material Point Method (MPM) and Smoothed Particle Hydrodynamics (SPH) Computational Techniques, Computational Particle Mechanics, 5, 49–58, https://doi.org/10.1007/s40571-016-0149-9, 2018.

Rognon, P. G., Roux, J.-N., Wolf, D., Naaïm, M., and Chevoir, F.: Rheophysics of Cohesive Granular Materials, Europhysics Letters (EPL), 74, 644–650, https://doi.org/10.1209/epl/i2005-10578-y, 2006.

Rognon, P. G., Roux, J.-N., Naaïm, M., and Chevoir, F.: Dense Flows of Cohesive Granular Materials, Journal of Fluid Mechanics, 596, 21–47, https://doi.org/10.1017/S0022112007009329, 2008.

Roscoe, K. H. and Burland, J. B.: On the Generalized Stress-Strain Behaviour of Wet Clay, in: Engineering Plasticity, pp. 535–609, Cambridge University Press, Cambridge, UK, 1968.

Sampl, P. and Zwinger, T.: Avalanche Simulation with SAMOS, Annals of Glaciology, 38, 393–398, https://doi.org/10.3189/172756404781814780, 2004.

Schofield, A. N. and Wroth, P.: Critical State Soil Mechanics, McGraw-Hill London, 1968.

Simo, J. C.: Algorithms for Static and Dynamic Multiplicative Plasticity That Preserve the Classical Return Mapping Schemes of the Infinitesimal Theory, Computer Methods in Applied Mechanics and Engineering, 99, 61–112, https://doi.org/10.1016/0045-7825(92)90123-2, 1992.

Sołowski, W. T., Berzins, M., Coombs, W. M., Guilkey, J. E., Möller, M., Tran, Q. A., Adibaskoro, T., Seyedan, S., Tielen, R., and Soga, K.: Material Point Method: Overview and Challenges Ahead, in: Advances in Applied Mechanics, vol. 54, pp. 113–204, Elsevier, ISBN 978-0-323-88519-5, https://doi.org/10.1016/bs.aams.2020.12.002, 2021.

Steffen, M., Kirby, R. M., and Berzins, M.: Analysis and Reduction of Quadrature Errors in the Material Point Method (MPM), International Journal for Numerical Methods in Engineering, 76, 922–948, https://doi.org/10.1002/nme.2360, 2008.

Steffen., M., Wallstedt, P. C., Guilkey, J. E., Kirby, R. M., and Berzins, M.: Examination and Analysis of Implementation Choices within the Material Point Method (MPM), Computer Modeling in Engineering & Sciences, 31, 107–128, https://doi.org/10.3970/cmes.2008.031.107, 2008.

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., and Selle, A.: A Material Point Method for Snow Simulation, ACM Transactions on Graphics, 32, 1–10, https://doi.org/10.1145/2461912.2461948, 2013.

Sulsky, D., Chen, Z., and Schreyer, H. L.: A Particle Method for History-Dependent Materials, Computer Methods in Applied Mechanics and Engineering, 118, 179–196, https://doi.org/10.1016/0045-7825(94)90112-0, 1994.

Sulsky, d., Zhou, S.-J., and Schreyer, H. L.: Application of a Particle-in-Cell Method to Solid Mechanics, Computer Physics Communications, 87, 236–252, https://doi.org/10.1016/0010-4655(94)00170-7, 1995.

Tonnel, M., Wirbel, A., Oesterle, F., and Fischer, J.-T.: AvaFrame com1DFA (v1.3): A Thickness-Integrated Computational Avalanche Module – Theory, Numerics, and Testing, Geoscientific Model Development, 16, 7013–7035, https://doi.org/10.5194/gmd-16-7013-2023, 2023.

Tran, Q. A., Grimstad, G., and Ghoreishian Amiri, S. A.: MPMICE: A Hybrid MPM-CFD Model for Simulating Coupled Problems in Porous Media. Application to Earthquake-induced Submarine Landslides, International Journal for Numerical Methods in Engineering, 125, e7383, https://doi.org/10.1002/nme.7383, 2024.

Vacondio, R., Altomare, C., De Leffe, M., Hu, X., Le Touzé, D., Lind, S., Marongiu, J.-C., Marrone, S., Rogers, B. D., and Souto-Iglesias, A.: Grand Challenges for Smoothed Particle Hydrodynamics Numerical Schemes, Computational Particle Mechanics, 8, 575–588, https://doi.org/10.1007/s40571-020-00354-1, 2021.

Vo, T. T., Nezamabadi, S., Mutabaruka, P., Delenne, J.-Y., and Radjai, F.: Additive Rheology of Complex Granular Flows, Nature Communications, 11, 1476, https://doi.org/10.1038/s41467-020-15263-3, 2020.

Wallstedt, P. C.: MPM-GIMP, https://sourceforge.net/projects/mpmgimp/, 2011.

Wallstedt, P. C. and Guilkey, J. E.: An Evaluation of Explicit Time Integration Schemes for Use with the Generalized Interpolation Material Point Method, Journal of Computational Physics, 227, 9628–9642, https://doi.org/10.1016/j.jcp.2008.07.019, 2008.

Wang, X., Qiu, Y., Slattery, S. R., Fang, Y., Li, M., Zhu, S.-C., Zhu, Y., Tang, M., Manocha, D., and Jiang, C.: A Massively Parallel and Scalable Multi-GPU Material Point Method, ACM Transactions on Graphics, 39, https://doi.org/10.1145/3386569.3392442, 2020.

Wyser, E., Alkhimenkov, Y., Jaboyedoff, M., and Podladchikov, Y. Y.: A Fast and Efficient MATLAB-based MPM Solver: fMPMM-solver v1.1, Geoscientific Model Development, 13, 6265–6284, https://doi.org/10.5194/gmd-13-6265-2020, 2020.

Xiao, H. and Chen, L. S.: Hencky's Elasticity Model and Linear Stress-Strain Relations in Isotropic Finite Hyperelasticity, Acta Mechanica, 157, 51–60, https://doi.org/10.1007/BF01182154, 2002.

Yerro, A., Alonso, E., and Pinyol, N.: The Material Point Method for Unsaturated Soils, Géotechnique, 65, 201–217, https://doi.org/10.1680/geot.14.P.163, 2015.

Zhang, X., Chen, Z., and Liu, Y.: The Material Point Method: A Continuum-Based Particle Method for Extreme Loading Cases, Elsevier and Tsinghua University Press Computational Mechanics Series, Academic Press, Amsterdam, [Netherlands], ISBN 978-0-12-407855-0, 2017.

Zhao, Y., Choo, J., Jiang, Y., and Li, L.: Coupled Material Point and Level Set Methods for Simulating Soils Interacting with Rigid Objects with Complex Geometry, Computers and Geotechnics, 163, 105 708, https://doi.org/10.1016/j.compgeo.2023.105708, 2023a.

Zhao, Y., Jiang, C., and Choo, J.: Circumventing Volumetric Locking in Explicit Material Point Methods: A Simple, Efficient, and General Approach, International Journal for Numerical Methods in Engineering, 124, 5334–5355, https://doi.org/10.1002/nme.7347, 2023b.