Year : 2022

Elasto-plastic deformations within a material point framework on modern GPU architectures

Wyser Emmanuel

Wyser Emmanuel, 2022, Elasto-plastic deformations within a material point framework on modern GPU architectures

Originally published at : Thesis, University of Lausanne

Posted at the University of Lausanne Open Archive http://serval.unil.ch
Document URN : urn:nbn:ch:serval-BIB_BD8501EFB41E3

Université de Lausanne

Faculté des géosciences et de l'environnement
Institut des sciences de la Terre

Elasto-plastic deformations within a material point framework on modern GPU architectures

---

Thèse de doctorat

présentée à la faculté des géosciences et de l'environnement de l'Université de Lausanne pour l'obtention du grade de Docteur en science de la Terre par

Emmanuel Wyser

Maîtrise ès Sciences en géosciences de l'environnement de l'Université de Lausanne

Jury

Prof. Dr. Klaus Holliger	UNIL, ISTE	Président du jury
Prof. Dr. Michel Jaboyedoff	UNIL, ISTE	Co-directeur de thèse
Prof. Dr. Yury Y. Podladchikov	UNIL, ISTE	Co-directeur de thèse
Prof. Dr. Stefan M. Schmalholz	UNIL, ISTE	Expert interne
Prof. Dr. Johan Gaume	EPFL, SLAB	Expert externe
Prof. Dr. Thibault Duretz	GU, IfG	Expert externe

Lausanne, 2021

UNIL | Université de Lausanne
Faculté des géosciences et de l'environnement
bâtiment Géopolis bureau 4631

IMPRIMATUR

Vu le rapport présenté par le jury d'examen, composé de

Président de la séance publique :	M. le Professeur Klaus Holliger
Président du colloque :	M. le Professeur Klaus Holliger
Co-directeur de thèse :	M. le Professeur Michel Jaboyedoff
Co-directeur de thèse :	M. le Professeur Yury Podladchikov
Expert interne :	M. le Professeur Stefan Schmalholz
Expert externe :	M. le Professeur Thibault Duretz
Expert externe :	M. le Professeur Johan Gaume

Le Doyen de la Faculté des géosciences et de l'environnement autorise l'impression de la thèse de

Monsieur Emmanuel WYSER

Titulaire d'une
Maîtrise universitaire ès Sciences en géosciences de l'environnement
de l'Université de Lausanne

intitulée

ELASTO-PLASTIC DEFORMATIONS WITHIN A MATERIAL POINT FRAMEWORK ON MODERN GPU ARCHITECTURES

Lausanne, le 17 décembre 2021

Pour le Doyen de la Faculté des géosciences et de l'environnement

K. Holliger

Professeur Klaus Holliger

Year : 2021

Elasto-plastic deformations within a material point framework on modern GPU architectures

Emmanuel Wyser

Abstract

Plastic strain localization is an important process on Earth. It strongly influences the mechanical behaviour of natural processes, such as fault mechanics, earthquakes or orogeny. At a smaller scale, a landslide is a fantastic example of elasto-plastic deformations. Such behaviour spans from pre-failure mechanisms to post-failure propagation of the unstable material. To fully resolve the landslide mechanics, the selected numerical methods should be able to efficiently address a wide range of deformation magnitudes.

Accurate and performant numerical modelling requires important computational resources. Mesh-free numerical methods such as the material point method (MPM) or the smoothed-particle hydrodynamics (SPH) are particularly computationally expensive, when compared with mesh-based methods, such as the finite element method (FEM) or the finite difference method (FDM). Still, mesh-free methods are particularly well-suited to numerical problems involving large elasto-plastic deformations. But, the computational efficiency of these methods should be first improved in order to tackle complex three-dimensional problems, *i.e.,* landslides.

As such, this research work attempts to alleviate the computational cost of the material point method by using the most recent graphics processing unit (GPU) architectures available. GPUs are many-core processors originally designed to refresh screen pixels (*e.g.,* for computer games) independently. This allows GPUs to delivers a massive parallelism when compared to central processing units (CPUs).

To do so, this research work first investigates code prototyping in a high-level language, *e.g.,* MATLAB. This allows to implement vectorized algorithms and benchmark numerical results of two-dimensional analysis with analytical solutions and/or experimental results in an affordable amount of time. Afterwards, low-level language such as CUDA C is used to efficiently implement a GPU-based solver, *i.e.,* `ep2-3De v1.0`, can resolve three-dimensional problems in a decent amount of time. This part takes advantages of the massive parallelism of modern GPU architectures. In addition, a first attempt of GPU parallel computing, *i.e.,* multi-GPU codes, is performed to increase even more the performance and to address the on-chip memory limitation. Finally, this GPU-based solver is used to investigate three-dimensional granular collapses and is compared with experimental evidences obtained in the laboratory.

This research work demonstrates that the material point method is well suited to resolve small to large elasto-plastic deformations. Moreover, the computational efficiency of the method can be dramatically increased using modern GPU architectures. These allow fast, performant and accurate three-dimensional modelling of landslides, provided that the on-chip memory limitation is alleviated with an appropriate parallel strategy.

Résumé

La déformation élasto-plastique et la localisation des contraintes sont des processus importants sur Terre. Ils influencent fortement le comportement mécanique des processus naturels, tels que la mécanique des failles, les tremblements de terre ou l'orogenèse. A plus petite échelle, un glissement de terrain est un parfait exemple de déformations élasto-plastiques. Un tel comportement s'étend des mécanismes avant la rupture à la propagation après la rupture du matériel instable. Pour mieux comprendre la mécanique des glissements de terrain, les méthodes numériques utilisées doivent être capables de traiter efficacement une large gamme de niveau de déformation.

Une modélisation numérique précise et performante nécessite des ressources importantes de calcul. Plus précisément, les méthodes numériques sans maillage telles que la méthode des points matériels (MPM) ou l'hydrodynamique des particules lissées (SPH) sont particulièrement coûteuses sur le plan computationnelle, par rapport aux méthodes basées sur des maillages, telles que la méthode des éléments finis (FEM) ou la méthode de différence (FDM). Néanmoins, les méthodes sans maillage sont particulièrement bien adaptées aux problèmes numériques impliquant de grandes déformations élasto-plastiques. Mais, l'efficacité de calcul de ces méthodes doit d'abord être améliorée afin de s'attaquer à des problèmes tridimensionnels complexes, *i.e.,* les glissements de terrain.

Ce travail de recherche tente de diminuer le coût de calcul de la méthode des points matériels en utilisant les architectures d'unité de traitement graphique (carte graphique ou GPU) les plus récentes disponibles. Les GPU sont des processeurs multicœurs conçus à l'origine pour rafraîchir les pixels de l'écran (par exemple, pour les jeux vidéos) de manière indépendante. Cela permet aux GPU de fournir un parallélisme massif par rapport aux unités centrales de traitement (processeur central ou CPU).

Ce travail de recherche étudie d'abord le prototypage de codes dans un langage de haut niveau, *i.e.,* MATLAB. Cela permet de mettre en œuvre des algorithmes vectorisés et de comparer les résultats numériques de l'analyse bidimensionnelle avec des solutions analytiques et/ou des résultats expérimentaux dans un délai acceptable. Ensuite, un langage de bas niveau tel que CUDA C est utilisé pour implémenter efficacement un solveur basé sur GPU, *i.e.,* `ep2-3De v1.0`, pour résoudre rapidement des problèmes tridimensionnels. Cette partie profite du parallélisme massif des architectures GPU modernes. De plus, une première tentative de calcul parallèle GPU, c'est-à-dire de codes multi-GPU, est effectuée pour augmenter encore plus les performances et remédier à la limitation de la mémoire des cartes graphiques. Enfin, ce solveur basé sur GPU est utilisé pour étudier les effondrements granulaires tridimensionnels et est comparé à des résultats expérimentaux obtenus en laboratoire.

Ce travail de recherche a montré que la méthode des points matériels est bien adaptée pour résoudre des déformations élasto-plastiques à différents niveaux. De plus, l'efficacité de calcul de la méthode peut être considérablement aug-

mentée en utilisant des architectures GPU modernes. Ceux-ci permettent une modélisation tridimensionnelle rapide, performante et précise des glissements de terrain, à condition que la limitation de la mémoire soit atténuée par une stratégie parallèle appropriée.

Résumé grand public

Le comportement des solides est souvent déterminé par des processus irréversibles, comme la plasticité. Cette dernière suppose qu'un matériau solide, soumis à des contraintes internes, va se déformer de manière irréversible à partir d'un certain seuil de déformation. Ce type d'interaction prédomine sur la Terre et régit de multiples phénomènes comme la mécanique des failles ou encore la création des chaînes de montagnes. A une plus petite échelle, un bel exemple de cette irréversibilité des processus est un glissement de terrain.

Ce travail de thèse propose d'implémenter des solutions numériques à des problèmes de la mécanique des milieux continus dont les déformations peuvent être faibles à importantes. Ceci dans le but ultime d'acquérir de meilleures connaissances de la mécanique interne des glissements de terrain. Les méthodes numériques traditionnelles sont robustes et validées depuis longtemps, mais peuvent rencontrer certains problèmes lorsque les déformations des matériaux sont très importantes. Ainsi, de nouvelles méthodes numériques sont nécessaires pour prendre en compte les régimes de grande déformations.

Le problème général de tout modèle numérique, en particulier pour les nouvelles méthodes, est son coût en temps de calcul, qui dépend généralement de la résolution numérique utilisée, mais plus spécifiquement, de la méthode numérique choisie. Traditionnellement, l'informatique traite les opérations arithmétiques de manière séquentielle, c'est-à-dire que le processeur central de calcul (CPU) traite les opérations l'une après l'autre. Une parallélisation est cependant possible. Avec le développement technologique des cartes graphiques modernes (GPU), il est maintenant possible de traiter ces opérations de manière massivement parallèle. Ce travail vise donc à utiliser les architectures dite récentes des cartes graphiques afin de permettre des calculs rapides et massivement parallélisés.

Dans un premier temps, ce travail de recherche propose d'implémenter des solutions numériques des déformations élasto-plastiques dans un langage de programmation de haut-niveau, comme MATLAB. Différents tests numériques ont été réalisé afin de valider l'implémentation numérique. Puis, cette structure algorithmique a été implémentée dans un langage de programmation de bas-niveau, orientée vers les cartes graphiques. Ceci a permis d'atteindre un haut niveau de performance et a permis de modéliser des phénomènes complexes tri-dimensionnels comme les glissements de terrain ou les écroulements granulaires secs.

Ce travail a démontré que les cartes graphiques récentes sont d'un très grand intérêt pour accélérer significativement le temps de calcul. De plus, cela donne un éclairage nouveau concernant la mécanique des glissements de terrain en trois dimensions, domaine qui était pour l'instant peu étudié de part les importantes ressources de calcul nécessaire.

Acknowledgements

First and foremost, I would like to thank Michel and Yury, who were two fantastic supervisors during these last past years.

Michel, je voulais sincèrement te remercier pour toutes ces années, qui ont clairement été les meilleures de ce long cursus. Et ça, c'est en grande partie grâce à toi et ton ouverture d'esprit. Merci auss pour ta philosophie générale sur la recherche, qui promeut la curiosité et l'indépendance. C'est précieux.

Юрий Юрьевич Подладчиков, спасибо за безграничный энтузиазм и за то, что научили меня искусству численных методов. Работать с вами было настоящим удовольствием. Наши дискуссии были для меня источником вдохновения на протяжении всего периода работы в университете.

Ich möchte Stefan herzlich danken, vor allem dafür, dass er den Bachelor-Studenten fantastische Subduktionsmodelle gezeigt hat. Dies ist wahrscheinlich das, was meine Aufmerksamkeit zuerst erregt hat.

Merci également à Thibault et Johan pour votre enthousiasme général, c'était vraiment chouette. En particulier, merci à toi, Johan, pour avoir présenté ces simulations d'avalanches de neige à l'EGU. C'est à ce moment que mon intérêt pour les MPM est né.

Особая благодарность Юрию Александровичу Алхименкову. Спасибо за вашу доброту и терпение, а также за то, что познакомили меня с вычислениями на графических ускорителях. Мне было очень интересно в вами работать и обсуждать эластодинамику, упругость, пластичность и локализацию деформаций. Я желаю вам всего наилучшего в вашем будущем.

I would to acknowledge Antoine and François, for all the good memories at the office and abroad, especially in the Yosemite Valley. A huge thump up to Ludovic and Sam and their legacy, i.e., the huge work on Octopus, GPU computing and MPI.

I would also like to acknowledge PhD lads such Richard, Josh, Lorenzo, Dany, Florian and Martin for the good times at EGU. Close friends are important, my thoughts are for Grégoire, Garance, Lila, Niko, Romain, Mak and all these great people from La Chaux-de-Fonds. A special thank to the Park'n'Sun members (and friends) as well. Fun times over there.

I would like to acknowledge my family, especially my parents and my sister, for their extensive support all over these years.

Finally, I would like to thank Liliane and her support during all these years. Thank you for being there all the way.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Rationale

Elasto-plasticity refers to irreversible deformation, as opposed to elasticity, which only treats of reversible deformations. A more formal definition is the following: "*Natural and manufactured materials generally exhibit an irreversible deformation behaviour such that when an applied load is removed only a fraction of deformation is recovered. The extent of reversible deformation is called the elastic range, whereas the extent of irreversible or inelastic deformation, or yield, is the plastic range*" (Borja, 2013, p.1).

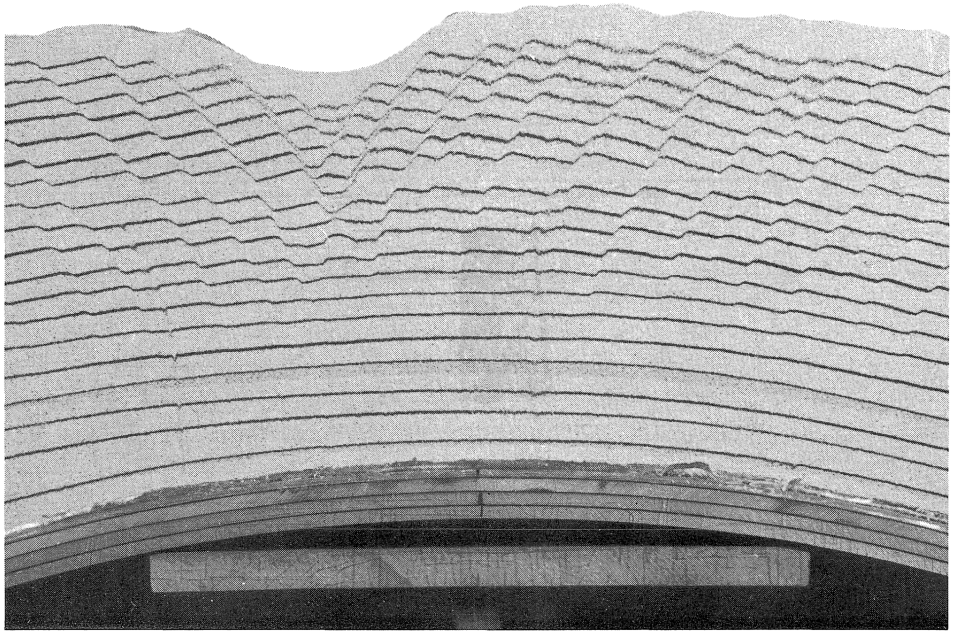


Figure 1.1. | Shear banding in a sandbox (Vermeer et al., 1984).

Shear banding (see Fig. 1.1) is a typical manifestation of localized plastic deformations, i.e., intense zones of shearing deformation (shear strain) within an elasto-plastic material resulting in numerous shear bands. Such feature of elasto-plastic materials plays a significant role over the dynamic of deformation, as non-homogeneous displacements initiate at the plastic onset. A natural phenomenon's example of shear banding is a landslide

On March 22, 2014, a large landslide occurred in north-west Washington state, United States of America (see Fig. 1.2). Fatalities (43 in the community

Figure 1.2. | Aerial photograph of the 2014 mudslide in northwest Washington State, USA. This major event is often referred as the "Oso landslide". Credit: Mark Reid, USGS (https://www.usgs.gov/news/revisiting-oso-landslide, last access: August 4th, 2021).

of Steelhead Haven, Oso, Washington) and significant destruction of properties (about 40 homes) and infrastructures (nearly 1 mi of the State Route 530) were reported. This catastrophic event gives a sense of the destructive power of landslides.

Landslide (or mass-wasting process) refers to "*ownward and outward movement of slope-forming materials composed of natural rock, soils, artificial fills, or combinations of these materials. The moving mass may proceed by any one of three principal types of movement: falling, sliding, or flowing, or by their combination.*" (Varnes, 1958, p.20). Such movements are caused by gravity (De Blasio, 2011). An attempt to classify landslides based on the types of movement and kinds of material (rock or soil) was proposed by Varnes, 1958; Varnes, 1978: this is the Varnes classification of landslide types. More recently, Hungr et al., 2014 proposed an update to this classification, arguing for a more comprehensive and flexible classification. Based on this update, this work focuses on the following landslide types[1]: rotational (i.e., slumps in the original Varnes classification, see Varnes 1958; Varnes 1978) and/or planar slides in clayey/silty soils. An illustration of slumps is depicted in Fig. 1.3.

Such process can be thought of as a complex elasto-plastic system, that develops and evolves in time through several consecutive (or even periodic) stages: the pre-failure deformation, the failure itself and the resulting post-

[1] In the real world, landslides exhibit a variety of types of movement. Therefore, they are usually classified as "*complex*".

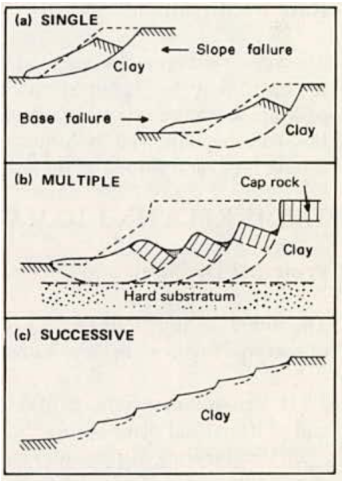


Figure 1.3 | Types of rotational slide (Varnes, 1978, p.23). The slip surface, resulting from plastic strain localization, dictates the displacement of the material. Varnes, 1958 was referring to rotational slides as "*slumps*", which clearly differ from planar slides. This specific term is abandoned in the Varnes classification update of Hungr et al., 2014.

failure deformation, i.e., displacements, scarps or bulging (Terzaghi, 1950; Skempton et al., 1969; Leroueil et al., 1996; Hungr et al., 2014). Shear banding significantly contributes to the landslide mechanics, as it controls the deformation of the whole mass and its motion. As a result of internal shear stresses, strain localization defines a complex pattern of slip surfaces, along which the material can slide.

Varnes, 1978 produced an idealized diagram of a "*complex earth slide - earth flow*" (see Fig. 1.4). It provides a morphological and geometrical description the expected features for a complex landslide. The most important feature is

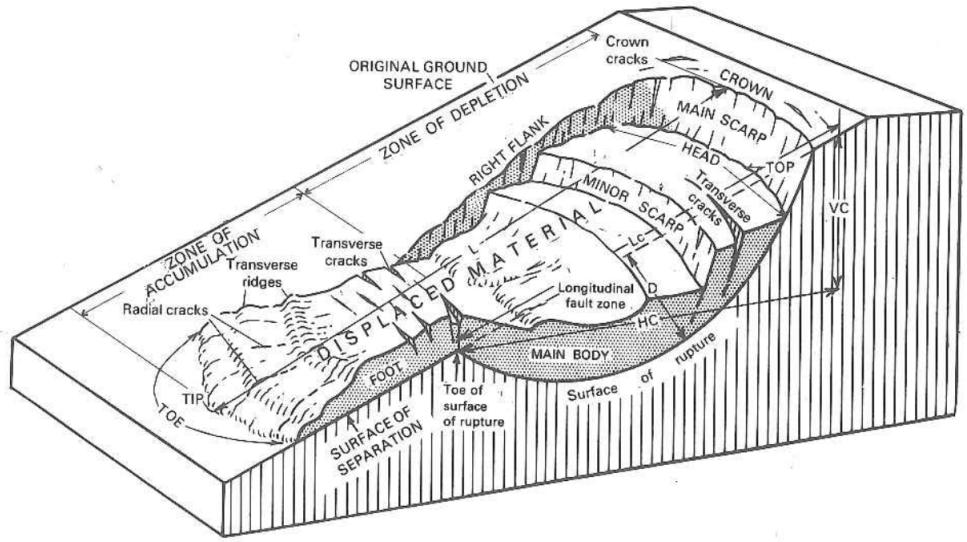


Figure 1.4. | Block diagram for a complex landslide, from Varnes, 1978.

the surface of rupture (or slip surface). This internal surface is the result of intense plastic strain localization, i.e., shear banding. The overall behaviour of the main body entirely depends on the geometry of this internal slip surface.

Another important feature is the main scarp (organised in a crown-like structure), which delimits the upper extent of the landslide body, i.e., the head. On its opposite, there is the toe of the landslide, where the mobilized material accumulates downhill. Generally, the toe results from thrusting mechanisms of the main body over the ground surface. An exhaustive list of these features can be found in Cruden et al., 1996. Two distinct zones exists within a landslide, and these are:

1. a zone of depletion with a complex arrangements of minor scarps and,
2. a zone of accumulation with bulging and thrusting mechanisms.

Elasto-plastic deformations, *i.e.,* shear banding, are closely governed by an interplay between a) pressure, b) shear stresses and, c) the material's shear strength (e.g., cohesion). To trigger strain localization, effective shear stresses should, at least, exceed the maximum shear strength of soils. Hence, the following factors may significantly contribute to irreversible deformations: a) intense rainfall, b) rapid snowmelt, c) water-level change, d) volcanic eruption and, e) earthquake shaking (Varnes, 1958; Wieczorek, 1996).

Moreover, the soil strength and its natural variability (Cho, 2010; De Blasio, 2011; Liu et al., 2015) can affect considerably the mechanical behaviour. In particular, it influences the failure modes and the occurrence probability of landslides (Griffiths et al., 2004; Wang et al., 2020a; Chen et al., 2021).

Landslides pose a significant threat to human life and infrastructure. Prediction of accurate propagation distances and identification of pre- and post-failure mechanisms are of crucial importance (De Blasio, 2011; Bandara et al., 2015; Chen et al., 2021). Within the numerical modelling framework, landslides pose a number of fundamental problems. Perhaps the most significant is that classical mesh-based methods, e.g., finite element method (FEM), suffer from severe mesh distortion errors due to the large deformations involved in the process (Soga et al., 2016; Shan et al., 2021). Hence, an appropriate numerical framework, that can accommodate large deformations, has to be selected. Amongst them, the material point method (MPM, see Sulsky et al. 1994), the large-deformation finite element method (L-DFEM, see Dey et al. 2015; Islam et al. 2019; Shan et al. 2021 or the particle finite element method (PFEM, see Zhang et al. 2018; Zhang et al. 2019; Yuan et al. 2020; Wang et al. 2021). Another two-fold problem is the spatial three-dimensional extent of landslide. From a numerical point of view, this implies a three-dimensional description of the phenomenon at a large scale, *i.e.,* important computational resources are needed (Gerya, 2010; Yerro et al., 2019). Additionally, mesh-free (or hybrid methods) usually require higher computational resources than mesh-based methods (Chen et al., 2017).

1.2. Objectives

The two main objectives of this research work are:

1. Code development of numerical solutions that are able to accommodate large elasto-plastic deformations
2. Validation of the numerical solver with:
   a) numerical benchmarks against analytical solutions
   b) experimental investigations in a laboratory controlled environment

Consequently, several questions arises:

1. How elasto-plastic deformations can be investigated ?
2. Can a high-level programming language (*e.g.,* MATLAB, R or Python) be performant and efficient ?
3. Are three-dimensional elasto-plastic simulations (*i.e.,* landslides) affordable in a decent amount of time ?
4. What is the actual limitation of computer hardware ?

1.3. Approach

A numerical approach, combined with laboratory experiments, is selected in this work to gain insights about elasto-plastic behaviour in geophysical materials, such as soils. This two-fold approach allows to develop efficient and performant numerical solvers, to reproduce experimental results to assess the ability of numerical modelling for more complex and natural cases such as landslides. The recent material point method (MPM) is selected as a numerical framework well suited to accommodate different deformation stages and magnitudes (Dunatunga et al., 2017; Gaume et al., 2019). Toward an application of MPM to landslides, it is essential to ensure that the numerical method itself is well-suited to a wide range of deformation magnitudes, *i.e.,* from pre-failure to propagation stages.

1.3.1. Experimental granular mechanics

The experimental work (within the laboratory controlled environment) consists in two distinct parts (yet somehow closely related) which are presented in the following.

The impacts of free-falling water droplets onto granular beds serve to investigate the experimental evidences of the porosity influence over the elasto-plastic cratering response of non-homogenous granular beds. *A posteriori*, it also serves as a first attempt of experimental prototyping.

The granular collapse experiments focus on significant elasto-plastic deformations during the collapse of a granular column. It is an important phenomenon, because it is an idealization of a general landslide mechanics, which can be greatly controlled in a laboratory environment. It is an appropriate experiment since it can be regarded as a rough approximation of the more complex mechanics of landslides.

1.3.2. Numerical modelling

Within the MPM framework under an explicit formulation, an hybrid numerical approach is chosen. To test and assess different MPM variants, prototyping codes and algorithms are first written in a high-level programming language such as MATLAB. Then, selected benchmarks are performed to assess the efficiency of this MATLAB-based MPM solver. This yields an efficient and effective numerical solver for elasto-plastic problems.

The next step is to transpose (or translate) the algorithmic structure of the MATLAB solver to a more performant language. The low-level C programming language and its syntaxic extension CUDA C is chosen. Standard C allows to implement procedural algorithmic structure, which extensively uses the central processing unit (CPU, or CPU-based) of modern computers. CUDA C is a proprietary syntax extension of Nvidia Corporation to C, which allows algorithm to be executed in parallel on a graphical processing unit (GPU, or GPU-based). Aside of the superior computational features of C over MATLAB, preprocessing activities, *i.e.,* geometry initialization or mesh generation, are still tedious tasks. The hybrid approach mentioned above is the following. MATLAB is used as a pre- and post-processor language. In the meantime, CUDA C is called by MATLAB and, it is used as a processor to explicitly solve for the numerical problems initially defined in MATLAB.

1.4. Thesis outline

The following Chapters 2 and 3 present the theoretical background and the numerical framework, respectively. They introduce and present important notions and numerical methods, *i.e.,* the distinct element method and the material point method. The main contribution of this thesis is further divided into Chapters 4 to 7. These are either published, currently under review or in preparation for an initial submission in peer-reviewed journals. At long last, here comes Chapter 8, which extensively discusses what has been presented so far and concludes this doctoral thesis dissertation. In the remaining of this section §1.4, I briefly present majors outcomes of Chapters 4 to 7.

1.4.1. Chapter 4 Cratering within granular matter

This published work proposes an experimental attempt to better characterize the cratering response induced by liquid droplet impact onto fine granular materials. It also demonstrate the ubiquitous behaviour of non-homogenous granular media and the importance of the packing fraction (or porosity). These laboratory investigations were carried at the beginning of the thesis. At that time, the research topic was to investigate complex impacts (deformable projectile) onto a granular bed. This chapter is important, especially to me, because it encloses my vast interest for the mechanical behaviour of granular matter. Generally speaking, it also widened my perspective from the Micromechanics of granular matter toward Continuum Mechanics and Elasto-plasticity

theories. In this thesis's genesis, it acts as a research tipping point.

1.4.2. Chapter 5 Code prototyping in MATLAB

This published work introduces an efficient and vectorized algorithmic structure of the MATLAB solver `fmpmm-solver v1.1`[2] within the material point framework, limited to two-dimensional configurations. This allows fast prototyping activities to solve for elastodynamic and elasto-plastic problems. The general objective was to provide 1) an efficient MATLAB-based implementation of MPM and, 2) an assessment of different MPM variants (sMPM, GIMP and CPDI). The main outome is that MATLAB is an efficient high-level language to implement MPM, provided that a decent amount of time is spent on vectorization activities. The goal behind this first solver is to propose an algorithmic structure that can be effortlessly transposed to a more performant and lower-level language such as the C (CPU-oriented) and CUDA C (GPU-oriented) programming languages.

1.4.3. Chapter 6 High-performance GPU-based solver

This work, currently under review, is the GPU-based solver `ep2-3De v1.0`[3] written in C and CUDA C languages. It is designed to take advantages of massive parallelism of modern GPU architectures in order to deliver the fastest wall-clock times possible. In addition, it is no longer restricted to plane strain condition but encompasses now three-dimensional geometries as well. Whereas the MATLAB-based solver would take hours of intense computational activities, a GPU equally deliver a time-to-solution of a less-than-a-minute for an equal problem. This chapter is, perhaps, the most important. It convinced me of 1) the terrific computational power of modern GPUs and, 2) that the numerical resolution is no longer nor a limitation neither an argument for low-resolution numerical solutions.

1.4.4. Chapter 7 Granular collapses

This work, currently under preparation for a submission in a peer-reviewed journal, is an attempt to validate the GPU-based solver `ep2-3De v1.0` with experimental evidences (*i.e.,* run-outs) of three-dimensional collapses within a controlled laboratory environment. This combined work demonstrates that

[2] The latest version of `fmpmm-solver v1.1` is available for download from Bitbucket at: https://bitbucket.org/ewyser/fmpmm-solver/src/master/ (last access: October 6, 2020). The fMPMM-solver archive (v1.0 and v1.1) is available from a permanent DOI repository (Zenodo) at: https://doi.org/10.5281/zenodo.4068585 (Wyser et al., 2020c)

[3] The routines of the `ep2-3De v1.0` solver are available for download from GitHub at: https://https://github.com/ewyser/ep2-3De (last access: November 10, 2021). The routines archive (v1.0) (Wyser et al., 2021) is available from a permanent DOI repository (Zenodo) at https://doi.org/10.5281/zenodo.4966590 (June 16, 2021).

both `ep2-3De v1.0` and analytical solutions proposed are in excellent agreement with the experimental run-outs obtained from analog granular collapses.

1.4.5. Appendix A Message passing interface and multi-GPU computing

This work, currently under consideration as additional materials for the revision process of Chapter 6 High-performance GPU-based solver, investigates a message parallel interface standard implementation of the solver `ep2-3De v1.0`, a multi-GPU implementation. Even tough some simplifications were made, this demonstrates the possibilities offered by a parallel implementation of the material point framework to alleviate the on-chip memory limitation of modern GPU architectures. This memory concern was raised in Chapter 6 High-performance GPU-based solver, and it was essential to address it in order to investigate high-resolution three-dimensional geometries on modern GPUs. We report good performances, nearly achieving a prefect weak scaling between the number of GPUs and the wall-clock time.

1.5. General comment

As a last overall comment, the reader may be confused as the general structure of the thesis appears two-folded, *i.e.,* focusing on both granular mechanics and elasto-plastic deformation. I give some further details in the following.

At first, this research work was focused on micromechanics during liquid droplet impact onto granular beds. Experiments were performed in the laboratory to gain insights about the cratering response. The next logical course of action would have been to propose a numerical solution to this complex problem, which could be regarded as a multi-phase elasto-plastic situation.

However, such numerical investigation requires the use of a micromechanical numerical framework, *i.e.,* the distinct element method. The computational resources needed (and some additional concerns which will be further presented) for proper three-dimensional simulations are so much expansive that DEM and such micromechanical framework was progressively disregarded. In particular, an apparatus was designed to reproduce three-dimensional granular collapses in a laboratory environment. At that time, DEM was seriously considered as a valid candidate to reproduce experimental results. But, quick calculation revealed the tremendous amount of discrete particles needed to reproduce accurately the collapse.

What remains from this early stage is the general topic of elastoplastic deformations and the behaviour of complex material, with the computational efficiency concern still in mind.

The progressive shift towards a material point framework focusing on landslide mechanics is the result of the convergence of concerns and interests, *i.e.,* granular collapse experiments, computational efficiency of the chosen numerical method and the topic of elasto-plastic deformations. In addition, one could

see the granular collapse as a perfectly simple case of landslide mechanics. Hence, this experiment is appropriate when studying landslide mechanics.

2. Theoretical background

The theoretical background and general notions related either to Continuum Mechanics, Infinitesimal and Finite Deformation Mechanics or Elasticity and Elasto-plasticity theories presented in this chapter can be found in the following classical text book references: Bower, 2009; Hashiguchi et al., 2012; Spencer, 2012; Borja, 2013; Doghri, 2013. The purpose is to avoid too many references within the text body and to improve the reading.

2.1. Motions and deformations

2.1.1. Description of motion

Let us consider a material point in a continuum body defined by its coordinates $\boldsymbol{X}$ in a reference configuration $\Omega_0 \subset \mathbb{R}^3$ at a reference time $t_0 = 0$. For a given later time $t > 0$ the material point coordinate $\boldsymbol{X}$ has now coordinates $\boldsymbol{x}$ in the current configuration $\Omega_t \subset \mathbb{R}^3$, then the equation

$$\boldsymbol{x} = \boldsymbol{\phi}(\boldsymbol{X}, t), \quad e.g., \quad x_i = \phi_i(X_I, t), \tag{2.1.1}$$

describes a *motion* of the continuum where $\boldsymbol{\phi}$ is a transformation (*e.g.,* a mapping) from the reference configuration to the current configuration.

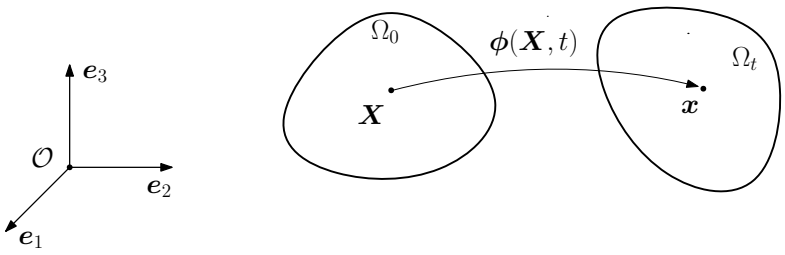


Figure 2.1. | Coordinate $\boldsymbol{X}$ of a material point in the reference configuration Ω_0 and its updated coordinate $\boldsymbol{x} = \boldsymbol{\phi}(\boldsymbol{X}, t)$ in the current configuration, in a fixed Cartesian orthonormal frame $(\mathcal{O}, \boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$.

Any given material point is specified by its position vectors, *e.g.,*

$$\boldsymbol{X} = \{X_I\}, \qquad \text{in the } \textit{reference} \text{ configuration} \tag{2.1.2}$$

$$\boldsymbol{x} = \{x_i\}, \qquad \text{in the } \textit{current} \text{ configuration} \tag{2.1.3}$$

and upper cases (*e.g.,* I, J, K) and lower cases (*e.g.,* i, j, k) indices designate the coordinates in the reference and current configurations, respectively. By

convention, X_I are called *the material coordinates* whereas x_i are called *the spatial coordinates*[1].

During a rigid-body motion made of both translation and rotation, a body moves without changing its shape; the distance and the orientation between two material points do not change during the motion. A *translation* is a rigid-body motion during which every material point undergoes the same displacement. Such motion is described by the following equation

$$\boldsymbol{x} = \boldsymbol{X} + \boldsymbol{c}(t), \tag{2.1.4}$$

where the vector $\boldsymbol{c}(t)$ is frame invariant and only depends on t. A *rotation* is also a rigid-body motion during which every material point undergoes the same rotation about any arbitrary axis of origin $\mathcal{O}$. It is described by the following equation

$$\boldsymbol{x} = \boldsymbol{Q}(t)\boldsymbol{X}, \tag{2.1.5}$$

where $\boldsymbol{Q}(t)$ is an orthogonal tensor.

As such, any rigid-body motion is a combination of both a translational part and a rotation part about an axis and, it can be described by equations of the form

$$\boldsymbol{x} = \boldsymbol{Q}(t)\boldsymbol{X} + \boldsymbol{c}(t), \quad \text{or} \quad \boldsymbol{X} = \boldsymbol{Q}^T(t)\boldsymbol{x} - \boldsymbol{Q}^T(t)\boldsymbol{c}(t). \tag{2.1.6}$$

2.1.2. Deformation

A body will change its shape as well as its position (translation) and its orientation (rotation) during any motion. A motion during which a change in shape takes place is called a deformation, independently of a change of position or orientation.

As such, the main problem in deformation analysis is to separate rigid-body motion from deformation, which has to be *invariant* with respect to rigid-body motion.

The *deformation gradient* is defined by:

$$\boldsymbol{F} \equiv \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}, \quad \textit{e.g.,} \quad F_{iJ} = \frac{\partial x_i}{\partial X_J}, \tag{2.1.7}$$

and describes how much the current coordinate x_i varies w.r.t to the reference coordinate X_J. Taking the inverse of the deformation gradient, *e.g.,* $\boldsymbol{F}^{-1}$ gives the inverse relation between the reference coordinate w.r.t the current configuration. If there is no motion, then $x_i = X_I$ and so F_{iI} reduces to δ_{iI}, *e.g.,* $F_{iI} = \delta_{iI}$.

[1] Similarly, the material coordinates can be referred to as *Lagrangian coordinates* and, the spatial coordinates can be referred to as *Eulerian coordinates*

Alternatively, the current position is defined by $\boldsymbol{x} = \boldsymbol{X} + \boldsymbol{u}$, where $\boldsymbol{u}$ is the displacement. By definition, the deformation gradient is also given by

$$\begin{aligned} \boldsymbol{F} &= \frac{\partial}{\partial \boldsymbol{X}}(\boldsymbol{X} + \boldsymbol{u}), \\ \boldsymbol{F} &= \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}}, \\ \boldsymbol{F} &= \boldsymbol{I} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}}, \end{aligned} \tag{2.1.8}$$

and rearranging terms, the *displacement* gradient tensor is given by

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = \boldsymbol{F} - \boldsymbol{I}, \quad e.g., \quad \frac{\partial u_i}{\partial X_J} = \frac{\partial x_i}{\partial X_J} - \delta_{iJ}. \tag{2.1.9}$$

However, the displacement gradient is expressed w.r.t the reference configuration. Differentiating the displacement w.r.t to the current configuration yields

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \boldsymbol{I} - \boldsymbol{F}^{-1}, \quad e.g., \quad \frac{\partial u_i}{\partial x_j} = \delta_{iJ} - \frac{\partial X_I}{\partial x_j}. \tag{2.1.10}$$

Let consider an infinitesimal line element vector $d\boldsymbol{X}$ with origin $\boldsymbol{X}$ in the reference configuration and an infinitesimal line element vector $d\boldsymbol{x}$ with origin $\boldsymbol{x}$ in the current configuration (see Fig. 2.2). By definition, one has

$$d\boldsymbol{x} = \boldsymbol{F} d\boldsymbol{X} \quad e.g., \quad dx_i = F_{iJ} dX_J, \tag{2.1.11}$$

and demonstrates that the deformation gradient is the fundamental measure of deformation in continuum mechanics.

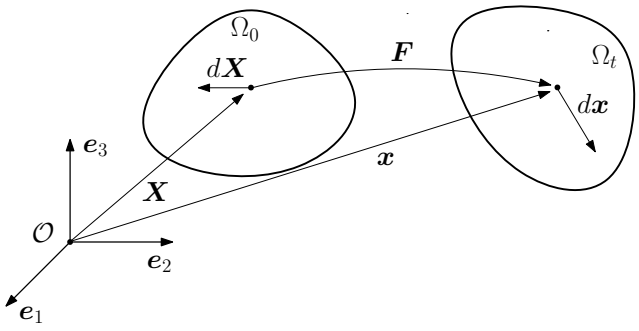


Figure 2.2. | Deformation of an infinitesimal line element $d\boldsymbol{X}$ in a reference configuration.

2.1.3. Finite deformation and strain measure

A *measure* of the deformation (*e.g.*, the *strain*) should be *invariant by rotation*. The deformation gradient tensor $\boldsymbol{F}$ does not have to satisfy this property. In

fact, in the rigid-body motion given by Eq. 2.1.5, the deformation gradient reduces to $\boldsymbol{F} = \boldsymbol{Q}(t)$. As such, $\boldsymbol{F}$ itself is not a suitable measure of the deformation under a rigid-body motion and, it does not allow a proper invariant definition of the strain measure.

What is then a suitable measure of deformation ? The answer lies in the definition of additional deformation tensors which are briefly presented in the following.

Using the polar decomposition theorem, the deformation gradient can be decomposed, in a multiplicative manner, by a product of two second-order tensors. It follows that

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{R} \quad e.g., \quad F_{iJ} = R_{iK}U_{KJ} = V_{ik}R_{kJ}, \tag{2.1.12}$$

where $\boldsymbol{R}$ is a proper orthogonal tensor (*e.g.,* a rotation tensor), and $\boldsymbol{U}$ and $\boldsymbol{V}$ are the right stretch (*e.g.,* defined w.r.t. the reference configuration in the *material* coordinates) and left stretch (*e.g.,* defined w.r.t the current configuration in the *spatial* coordinates) tensors, respectively. The polar decomposition *multiplicatively* decomposes the deformation gradient into orthogonal (*e.g.,* rotation) and stretch tensors.

The right Cauchy-Green deformation tensor $\boldsymbol{C}$ is a first suitable measure of deformation and is given by

$$\boldsymbol{C} = \boldsymbol{F}^T\boldsymbol{F} = \boldsymbol{U}^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{U} = \boldsymbol{U}^2 \quad e.g., \quad C_{IJ} = \frac{\partial x_k}{\partial X_I}\frac{\partial x_k}{\partial X_J}. \tag{2.1.13}$$

and, since $\boldsymbol{R}$ is orthogonal, hence $\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}$, which demonstrates the invariance by rotation of $\boldsymbol{C}$.

Similarly, the left Cauchy-Green deformation tensor $\boldsymbol{b}$ is a second suitable measure of deformation and is given by

$$\boldsymbol{b} = \boldsymbol{F}\boldsymbol{F}^T = \boldsymbol{V}\boldsymbol{R}\boldsymbol{R}^T\boldsymbol{V}^T = \boldsymbol{V}^2 \quad e.g., \quad b_{ij} = \frac{\partial x_i}{\partial X_K}\frac{\partial x_j}{\partial X_K}. \tag{2.1.14}$$

These two tensors allow to define the *Lagrangian strain* tensor $\boldsymbol{\gamma}$ (*e.g.,* the strain is defined w.r.t the reference configuration) and the *Eulerian strain* tensor $\boldsymbol{\eta}$ (*e.g.,* the strain is defined w.r.t the current configuration), respectively, and are defined as

$$\boldsymbol{\gamma} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}), \tag{2.1.15}$$

$$\boldsymbol{\eta} = \frac{1}{2}(\boldsymbol{I} - \boldsymbol{b}^{-1}). \tag{2.1.16}$$

The expressions of both $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ can be expressed in terms of the displacement gradient, which results in the following

$$\gamma_{IJ} = \frac{1}{2}\left(\frac{\partial u_I}{\partial X_J} + \frac{\partial u_J}{\partial X_I} + \frac{\partial u_k}{\partial X_I}\frac{\partial u_k}{\partial X_J}\right), \tag{2.1.17}$$

$$\eta_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{\partial u_K}{\partial x_i}\frac{\partial u_K}{\partial x_j}\right). \tag{2.1.18}$$

2.1.4. Infinitesimal strain

The fundamental assumption of the infinitesimal strain theory is that the current configuration is not significantly different from the reference configuration. This implies that all the components of the displacement gradient are numerically small, *e.g.*, $|\partial u_i/\partial X_J| \ll 1$ and, ii) the squares and products of these quantities are neglected.

To demonstrate the equivalence (Spencer, 2012) , let consider the displacement gradient expressed in the spatial coordinates, *e.g.*,

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \boldsymbol{I} - \boldsymbol{F}^{-1} \quad e.g., \quad \frac{\partial u_i}{\partial x_j} = \delta_{ij} - \frac{\partial X_I}{\partial x_j}, \tag{2.1.19}$$

where, by binomial expansion, $\boldsymbol{I} - \boldsymbol{F}^{-1}$ yields to

$$\boldsymbol{I} - \boldsymbol{F}^{-1} = \boldsymbol{I} - \left(\boldsymbol{I} - (\boldsymbol{F} - \boldsymbol{I}) + (\boldsymbol{F} - \boldsymbol{I})^2 - (\boldsymbol{F} - \boldsymbol{I})^3 + ...\right). \tag{2.1.20}$$

The displacement gradient now reads

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = (\boldsymbol{F} - \boldsymbol{I}) - (\boldsymbol{F} - \boldsymbol{I})^2 + (\boldsymbol{F} - \boldsymbol{I})^3 - ..., \tag{2.1.21}$$

and, considering $\boldsymbol{F} - \boldsymbol{I} = \partial \boldsymbol{u}/\partial \boldsymbol{X}$ and neglecting higher order terms, the displacement gradient formulated in the material coordinates reduces to

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} \quad e.g., \quad \frac{\partial u_i}{\partial x_j} = \frac{\partial u_i}{\partial X_J}. \tag{2.1.22}$$

To first order in the displacement gradient, it follows from Eqs. 2.1.17-2.1.18 that $\gamma_{IJ} \simeq \eta_{ij}$. The *infinitesimal strain* tensor $\boldsymbol{\epsilon}$ is defined as

$$\boldsymbol{\epsilon} = \frac{1}{2}\left(\boldsymbol{F} + \boldsymbol{F}^T\right) - \boldsymbol{I} \quad e.g., \quad \epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_J} + \frac{\partial u_j}{\partial X_I}\right). \tag{2.1.23}$$

Hence, the infinitesimal strain tensor can be regarded as an exact but first order formulation of the displacement gradient tensor.

2.1.5. Finite strain

In the finite strain theory, reference and current configuration are significantly different and, the squares and product of the displacement gradient can no longer be neglected. A comprehensive and detailed introduction to finite strain theory can be found in Hashiguchi et al., 2012. The finite strain theory proposes a formal framework, while considering both infinitesimal and finite deformations. Therefore, it is an extension of infinitesimal strain to finite strain.

When dealing with finite deformation, a useful strain measure is given by the logarithmic strain tensor, which is based on the left Cauchy-Green deformation tensor $\boldsymbol{b}$. Its spectral decomposition reads

$$\boldsymbol{b} = \sum_{i=1}^{3} \lambda_i^2 (\boldsymbol{n}_i \otimes \boldsymbol{n}_i), \tag{2.1.24}$$

where λ_i^2 are the eigenvalues of $\boldsymbol{b}$ and $\boldsymbol{n}_i$ its principal directions, *e.g.,* the eigenvectors of $\boldsymbol{b}$. The eigenvalues define the squares of the principal stretches of the left Cauchy-Green deformation tensor. By the spectral decomposition theorem [2], one can obtain $\boldsymbol{V}$ and the square roots of its eigenvalues as

$$\boldsymbol{V} = \sqrt{\boldsymbol{V}^2} \equiv \sum_{i=1}^{3} \sqrt{\lambda_i^2}(\boldsymbol{n}_i \otimes \boldsymbol{n}_i) = \sum_{i=1}^{3} \lambda_i(\boldsymbol{n}_i \otimes \boldsymbol{n}_i), \tag{2.1.25}$$

where λ_i $(i = 1, 2, 3)$ are the eigenvalues of the left stretch tensor $\boldsymbol{V}$.

The logarithmic strain tensor $\boldsymbol{\varepsilon}$ considers the logarithmic eigenvalues of the left stretch tensor obtained from the left Cauchy-Green deformation tensor, *i.e.,*

$$\boldsymbol{\varepsilon} = \sum_{i=1}^{3} \ln \lambda_i(\boldsymbol{n}_i \otimes \boldsymbol{n}_i) \equiv \frac{1}{2} \ln \boldsymbol{b} \quad e.g., \quad \varepsilon_{ij} = \frac{1}{2} \ln b_{ij}. \tag{2.1.26}$$

Unlike the infinitesimal strain tensor, the logarithmic strain tensor is invariant under rotation, *e.g.,* only the left stretch part of the deformation gradient is considered. Additionally, there exists an exponential map between the logarithmic strain tensor and the left Cauchy-Green deformation tensor, which reads as

$$\boldsymbol{b} = \exp(2\boldsymbol{\varepsilon}) \quad e.g., \quad b_{ij} = \exp(2\varepsilon_{ij}), \tag{2.1.27}$$

which means that the left Cauchy-Green deformation tensor can be recovered from the logarithmic strain tensor.

2.2. Large deformation framework for a linear elasto-plastic continuum

2.2.1. Linear elasticity theory

Elasticity is a reversible process, *e.g.,* a deformable body returns to its original shape when the external load is removed. Hence, a unique relation exists between strains and stresses, *e.g.,* a given strain results in a given stress. The case of a one-dimensional bar in uniaxial tension is instructive (see Fig. 2.3). The Young's modulus is a proportionality constant, which relates the amount of axial strain ϵ to the increase in tensile stress σ in the bar. As such, the stress-strain relation becomes

$$E = \frac{\sigma}{\epsilon} \Longleftrightarrow \sigma = E\epsilon, \tag{2.2.1}$$

[2] The spectral decomposition can be achieved using the singular value decomposition (SVD). It is a generalization of the eigendecomposition of a square orthogonal matrix with an orthonormal eigenbasis to any $m \times n$ matrix. In particular, the singular value decomposition of the deformation gradient $\boldsymbol{F}$ is given by $\boldsymbol{F} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are rotation tensors. They should not be confused with right and left stretch tensors, respectively.

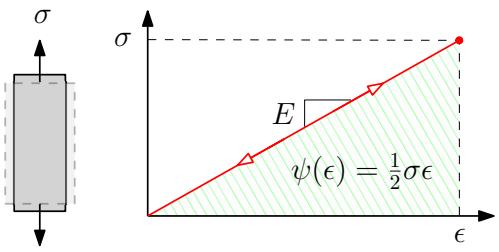


Figure 2.3. | Elastic stress-strain curve for uniaxial tension. The green surface under the curve corresponds to the strain-energy function $\psi(\epsilon)$, *e.g.,* the work done per unit volume to deform a material (Bower, 2009) or the stored elastic energy during elastic loading.

where the Young's modulus characterizes the stiffness of the material, *e.g.,* the higher the modulus, the stiffer the bar. It should not be confused with the material strength, which is the maximum amount of elastic deformation a given material can accommodate before yielding. A geometrical interpretation of the Young's modulus is given by Eq. 2.2.1 in an incremental form, *e.g.,*

$$E = \frac{\Delta\sigma}{\Delta\epsilon} \quad e.g., \quad E = \frac{\partial\sigma}{\partial\epsilon}. \tag{2.2.2}$$

Here, E clearly defines a proportional relation between an increment of strain with an increment of stress.

In this work, the linear elasticity theory is chosen: a linear proportional relation exists between stresses and strains, *e.g.,* the Hooke's Law. It relates the Cauchy stress tensor $\boldsymbol{\sigma}$ to the infinitesimal strain tensor $\boldsymbol{\epsilon}$ by

$$\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\epsilon} \quad e.g., \quad \sigma_{ij} = C_{ijkl}\epsilon_{kl}, \tag{2.2.3}$$

where C_{ijkl} is the fourth rank tangent stiffness tensor (*e.g.,* the Hooke's operator) and $\epsilon_{kl} = (\partial_l u_k + \partial_k u_l)/2$ is the infinitesimal strain tensor, and u_k is the displacement. Eq. 2.2.3 is a *constitutive relation* between stresses and strains. Under the infinitesimal strain theory, Eq. 2.2.3 is derived from a linear strain-energy function $\psi(\epsilon_{ij}) = \frac{1}{2}C_{ijkl}\epsilon_{ij}\epsilon_{kl}$ (Spencer, 2012; Doghri, 2013), that is

$$\boldsymbol{\sigma} = \frac{\partial\psi}{\partial\boldsymbol{\epsilon}} \quad e.g., \quad \sigma_{ij} = \frac{\partial\psi}{\partial\epsilon_{ij}}. \tag{2.2.4}$$

This work is restricted to a linear isotropic material, *e.g.,* its mechanical properties are independent of the axial loading direction and, it reads as

$$C_{ijkl} = \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}) + \lambda\delta_{ij}\delta_{kl}, \tag{2.2.5}$$

where $\mu = E/(2(1+\nu))$ and $\lambda = (E\nu)/((1-2\nu)(1+\nu))$ are the Lamé constants with E is the Young's modulus (the stiffness of the material in the axial loading direction) and ν is the Poisson's ratio (the lateral contraction/expansion of the material in response to the imposed axial loading).

It is often preferred to express the stress-strain constitutive relation using two additional moduli, *e.g.,* the bulk K and shear G moduli. The Hooke's operator for an isotropic elastic material, expressed with these moduli, now reads

$$D_{ijkl} = 2GI^{dev}_{ijkl} + K\delta_{ij}\delta_{kl}, \tag{2.2.6}$$

where $I^{dev}_{ijkl} = (\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})/2 - \delta_{ij}\delta_{kl}/3$ is a fourth-order identity tensor and is the deviatoric part of the fourth-order identity tensor I_{ijkl}. Eq. 2.2.3 can be rewritten using $C_{ijkl} = D_{ijkl}$ and becomes

$$\boldsymbol{\sigma} = 2G\boldsymbol{\epsilon} + \left(K - \frac{2}{3}G\right)\mathrm{tr}(\boldsymbol{\epsilon})\mathbf{1} \quad e.g., \quad \sigma_{ij} = 2G\epsilon_{ij} + \left(K - \frac{2}{3}G\right)\epsilon_{kk}\delta_{ij}, \tag{2.2.7}$$

where the strain tensor is additively decomposed into a deviatoric part, *e.g.,* the shear strains, and a spherical part, *e.g.,* the volumetric strain.

2.2.2. Rate-dependent formulation and objectivity

A rate-dependent formulation of Eq. 2.2.3 may be preferred for larger deformation and reads

$$\dot{\sigma}_{ij} = C_{ijkl}\dot{\epsilon}_{kl}, \tag{2.2.8}$$

where $\dot{\sigma}_{ij} \equiv \partial_t\sigma_{ij}$ is the time derivative of the Cauchy stress and $\dot{\epsilon}_{kl} = (\partial_l v_k + \partial_k v_l)/2$ is the strain rate tensor with v_k the velocity.

A large deformation framework requires an appropriate stress-strain formulation. In this work, a rate-dependent formulation using the Jaumann stress rate is chosen. This formulation provides an objective stress rate measure, *e.g.,* the stress state of the continuum body is *invariant* under rotation. The Jaumann rate of the Cauchy stress tensor is defined as

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = \frac{1}{2}C_{ijkl}\left(\frac{\partial v_k}{\partial x_l} + \frac{\partial v_l}{\partial x_k}\right). \tag{2.2.9}$$

The Jaumann stess derivative can be written as

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = \frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} - \sigma_{ik}\dot{\omega}_{jk} - \sigma_{jk}\dot{\omega}_{ik}, \tag{2.2.10}$$

where $\dot{\omega}_{ij} = (\partial_i v_j - \partial_j v_i)/2$ is the vorticity tensor, and $\mathrm{D}\sigma_{ij}/\mathrm{D}t$ denotes the material derivative of the Cauchy stress tensor, which reads

$$\frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} = \frac{\partial\sigma_{ij}}{\partial t} + v_k\frac{\partial\sigma_{ij}}{\partial v_k}. \tag{2.2.11}$$

Lets note that whereas the strain rate tensor is symmetric, *e.g.,* $\dot{\epsilon}_{ij} = \dot{\epsilon}_{ji}$, the vorticity tensor is antisymmetric, *e.g.,* $\dot{\omega}_{ij} = -\dot{\omega}_{ji}$. Rearranging the Jaumann stress derivative, *e.g.,* Eq. 6.2.12, yields to

$$\frac{\partial\sigma_{ij}}{\partial t} = \frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} + \overbrace{\sigma_{ik}\dot{\omega}_{jk} + \sigma_{jk}\dot{\omega}_{ik}}^{\sigma^{\mathcal{R}}_{ij}}, \tag{2.2.12}$$

where $\sigma_{ij}^{\mathcal{R}}$ represent the rotation of the Cauchy stress tensor which satisfies the stress objectivity for the rate-dependent formulation.

Lets expand the tensorial term $\sigma_{ij}^{\mathcal{R}}$ in Eq. 6.2.14 using the Einstein's convention with $\sigma_{ij} = \sigma_{ji}$, $\dot{\omega}_{ij} = -\dot{\omega}_{ji}$ and $\dot{\omega}_{kk} = 0$. After expanding, collecting and rearranging terms, the rotated stress components $\sigma_{ij}^{\mathcal{R}}$ are

$$\sigma_{xx}^{\mathcal{R}} = 2(\sigma_{xy}\dot{\omega}_{xy} + \sigma_{xz}\dot{\omega}_{xz}), \tag{2.2.13}$$
$$\sigma_{yy}^{\mathcal{R}} = -2(\sigma_{xy}\dot{\omega}_{xy} - \sigma_{yz}\dot{\omega}_{yz}), \tag{2.2.14}$$
$$\sigma_{zz}^{\mathcal{R}} = -2(\sigma_{xz}\dot{\omega}_{xz} + \sigma_{yz}\dot{\omega}_{yz}), \tag{2.2.15}$$
$$\sigma_{xy}^{\mathcal{R}} = \dot{\omega}_{xy}(\sigma_{yy} - \sigma_{xx}) + \sigma_{yz}\dot{\omega}_{xz} + \sigma_{xz}\dot{\omega}_{yz}, \tag{2.2.16}$$
$$\sigma_{yz}^{\mathcal{R}} = \dot{\omega}_{yz}(\sigma_{zz} - \sigma_{yy}) - \sigma_{xy}\dot{\omega}_{xz} - \sigma_{xz}\dot{\omega}_{xy}, \tag{2.2.17}$$
$$\sigma_{xz}^{\mathcal{R}} = \dot{\omega}_{xz}(\sigma_{zz} - \sigma_{xx}) + \sigma_{yz}\dot{\omega}_{xy} - \sigma_{xy}\dot{\omega}_{yz}. \tag{2.2.18}$$

and, for a two-dimensional state of stress assuming plane strain condition (*e.g.,* $\sigma_{xz} = \sigma_{yz} = 0$), Eqs. 6.2.15, 6.2.16 and 6.2.18 reduce to the following

$$\sigma_{xx}^{\mathcal{R}} = 2\sigma_{xy}\dot{\omega}_{xy}, \tag{2.2.19}$$
$$\sigma_{yy}^{\mathcal{R}} = -2\sigma_{xy}\dot{\omega}_{xy}, \tag{2.2.20}$$
$$\sigma_{xy}^{\mathcal{R}} = \dot{\omega}_{xy}(\sigma_{yy} - \sigma_{xx}). \tag{2.2.21}$$

2.2.3. Finite strain formulation

When considering material (or geometrical) non-linearities within the finite deformation framework, a conventional stress-strain relation is the formulation based on the Kirchhoff stress tensor $\boldsymbol{\tau}$ with the logarithmic strain tensor $\boldsymbol{\varepsilon}$ (Coombs et al., 2020b). This is an extension of the linear elastic constitutive relation to a hyper elastic relation. The strain-energy function no longer depends on the quadratic terms of the infinitesimal strain tensor but on the deformation gradient tensor (Hashiguchi et al., 2012), *e.g.,* $\psi(\boldsymbol{F})$. Similarly to Eq. 2.2.3, it now reads

$$\boldsymbol{\tau} = \boldsymbol{C} : \boldsymbol{\varepsilon} \quad e.g., \quad \tau_{ij} = C_{ijkl}\varepsilon_{kl}. \tag{2.2.22}$$

This is a particularly useful stress-strain relation, since one can use the conventional constitutive equations from the infinitesimal strain theory (Coombs et al., 2020b), without the need to reformulate them for the particular use within the finite deformation framework. The Cauchy stress tensor is related to the Kirchhoff stress tensor by $\boldsymbol{\sigma} = \boldsymbol{\tau}/\det(\boldsymbol{F})$.

2.3. Elasto-plasticity

Plasticity (or inelasticity) refers to an irreversible (or permanent) deformation of the material (Poirier, 1985; Lubliner, 2008). It permanently deforms beyond a limit, *e.g.,* plastic yielding, and can never recover its original shape. Unlike elastic deformation, plastic deformation is, by definition, permanent.

2.3.1. Shear banding: orientation and finite thickness

One of the most striking phenomena (in soils or rocks), is the occurrence of shear bands, *e.g.,* a narrow localization of plastic deformation of intensively shearing material. This phenomenon of shear banding was observed by Coulomb as early as 1776. Another feature is that the pressure drops within the shear band, relatively to the pressure outside the shear band (Kaus, 2010; Le Pourhiet, 2013; Duretz et al., 2018).

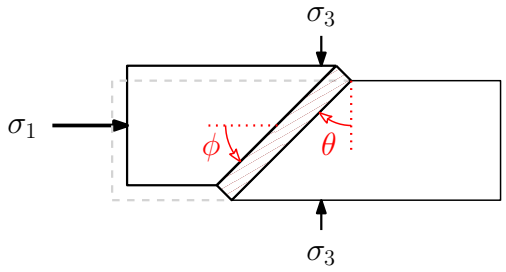


Figure 2.4. | Shear band's orientation θ for a biaxial test (Vermeer, 1990), with the principal stresses $\sigma_1 > \sigma_2 > \sigma_3$.

One of the most important question about shear banding is: which shear band angle (or the shear band's orientation, see Fig. 2.4), *e.g.,* θ, should be expected in a Mohr-Coulomb material ? Coulomb considered the orientation of the shear band against the direction of the minor compressive stress. From a theoretical point of view, this was studied by Vardoulakis, 1980; Vermeer et al., 1984; Vermeer, 1990 for elastoplastic rheologies. In particular, Vermeer, 1990 demonstrated, with a bifurcation and a post-failure analyses, that a range of angle are mechanically stable, *e.g.,*

$$\theta = \frac{\pi}{4} \pm \frac{\phi}{2} \quad \text{(Coulomb's angle)}, \tag{2.3.1}$$

along with the Roscoe's angle, *e.g.,* $\theta = \pi/4 \pm \psi/2$, and the Arthur's (or intermediate) angle, *e.g.,* $\theta = \pi/4 \pm (\phi + \psi)/4$.

Another important feature is the shear band's thickness. In a clay material, it is invisible and the shear band is viewed as a slip line (Vermeer, 1990). The use of strain-softening to induce strain localization is common in the computational mechanics literature: it results in an excessive mesh-dependent numerical solution. This is due to a loss of ellipticity and the lack of an internal length scale that both promote the non-uniqueness of the solution (De Borst, 1988; De Borst et al., 1993; Le Pourhiet, 2013; Sabet et al., 2019). A range of possibilities has been proposed, namely Cosserat plasticity (Mühlhaus et al., 1987), non-local plasticity (Bažant et al., 2002), and gradient plasticity (De Borst et al., 1992). More recently, Duretz et al., 2019; De Borst et al., 2020 investigated the role of elasto-viscoplasticity with a damper in parallel to a plastic slider (Kelvin-type rheology). They successfully demonstrated that models using viscoplastic rheologies converge upon mesh refinement, *e.g.,* a finite shear band's thickness is resolved.

2.3.2. An underlying definition

A key definition is given by Borja, 2013 for a one-dimensional elasto-plastic problem. Let us consider an elasto-plastic bar subjected to uniaxial tension. Let us also assume that the stress-strain curve is given by a bilinear relation, with E the Young's modulus and E^T is the tangential modulus beyond the initial yield stress $\sigma_{y0} > 0$, which is *the material strength*, *e.g.,* the maximal admissible elastic deformation the material can accommodate before yielding. For simplicity, let us also assume that the yield stress is equal in both tension and compression. The *elastic region* $\mathbb{E}$ is defined as

$$\mathbb{E} = \{\sigma \in \mathbb{R} \mid -\sigma_{y0} < \sigma < \sigma_{y0}\}, \tag{2.3.2}$$

and the two yield points as

$$\partial\mathbb{E} = \{\sigma \in \mathbb{R} \mid \sigma = \pm\sigma_{y0}\}. \tag{2.3.3}$$

Hence, the admissible set defined as

$$\bar{\mathbb{E}} = \mathbb{E} \cup \partial\mathbb{E}. \tag{2.3.4}$$

denotes the closure of the elastic region. It is a fundamental notion, because it defines the elastic region $\mathbb{E}$ and its plastic boundary $\partial\mathbb{E}$ as a closed set for the elasto-plastic problem.

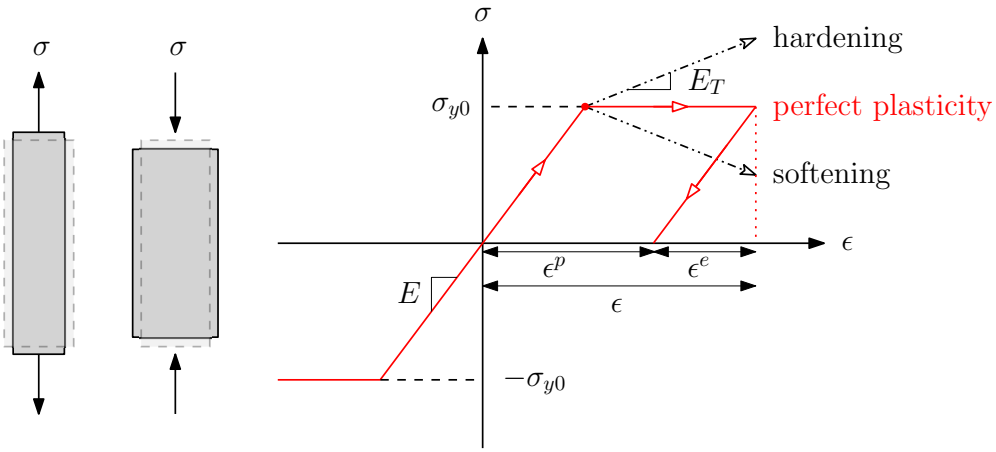


Figure 2.5. | Stress-strain curve for uniaxial loading: elastic loading prior to plastic yielding. Elastic unloading results at the offset of plastic loading.

The stress-strain curve for uniaxial loading is shown in Fig. 2.5. At the plastic yielding onset (*e.g., e.g.,* the yield strength of the material or the yield stress σ_{y0}), the relation between ϵ and σ is no longer linear and, the elasto-plastic problem become non-linear, *e.g.,* plastic loading. Elastic unloading produces at the offset of plastic loading. As showed in Fig. 2.5, the strain ϵ can be decomposed additively into an elastic part ϵ^e and a plastic part ϵ^p, *e.g.,*

$$\epsilon = \epsilon^e + \epsilon^p, \tag{2.3.5}$$

which is one of the fundamental notion in plasticity theory.

2.3.3. The yield function, the consistency condition and the flow rule

The key stone of elasto-plasticity theory is *the yield function* $f(\boldsymbol{\sigma}, \kappa)$, which defines i) a yield surface, *e.g.,* $f(\boldsymbol{\sigma}, \kappa) = 0$ and, ii) an admissible elastic domain, *e.g.,* $f(\boldsymbol{\sigma}, h) \leq 0$. Here, κ is called the hardening parameter, which the evolution is governed by a *hardening law*. The yield condition defines the set of permissible stresses, but also the conditions for which a plastic loading can continue to occur.

The consistency condition governs the condition of plastic loading, that is $\dot{f} = (\partial f/\partial \boldsymbol{\sigma})^T \dot{\boldsymbol{\sigma}}$, with $\dot{f} = 0$ by definition, then

$$0 = \left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)^T \dot{\boldsymbol{\sigma}}, \tag{2.3.6}$$

which states that during plastic loading, the change in stress is tangent to the yield surface.

The plastic flow rule (or law) governs the evolution of plastic strains and, it is defined by the scalar plastic potential function $g(\boldsymbol{\sigma}, h)$, *e.g.,*

$$\dot{\boldsymbol{\epsilon}}^p = \dot{\lambda}\frac{\partial g}{\partial \boldsymbol{\sigma}} \quad e.g., \quad \dot{\epsilon}^p_{ij} = \dot{\lambda}\frac{\partial g}{\partial \sigma_{ij}}, \tag{2.3.7}$$

where the scalar $\dot{\lambda}$ is the plastic multiplier. The loading/unloading conditions can be expressed in the Kuhn–Tucker form (Borja, 2013) as

$$\dot{\lambda} \geq 0\,, f \leq 0\,, \dot{\lambda} f = 0. \tag{2.3.8}$$

The plastic flow law is *associated* whenever the plastic potential function $g = f$ and, *non-associated* when $g \neq f$.

2.3.4. Elasto-plastic constitutive relation

The elasto-plastic constitutive relation (expressed in Voigt's notation) to solve for is

$$\dot{\boldsymbol{\sigma}} = \mathbf{D}\dot{\boldsymbol{\epsilon}} - \mathbf{D}\dot{\lambda}\frac{\partial g}{\partial \boldsymbol{\sigma}}, \tag{2.3.9}$$

where $\mathbf{D}$ is the elastic tangent operator. Eq. 2.3.9 can be substituted in Eq. 2.3.6, which yields

$$0 = \left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)^T \mathbf{D}\dot{\boldsymbol{\epsilon}} - \left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)^T \mathbf{D}\dot{\lambda}\frac{\partial g}{\partial \boldsymbol{\sigma}}, \tag{2.3.10}$$

solving for $\dot{\lambda}$, substituting it back into the Hooke's law and collecting terms yield to the elasto-plastic constitutive relation

$$\dot{\boldsymbol{\sigma}} = \underbrace{\left(\mathbf{D} - \frac{\mathbf{D}\frac{\partial g}{\partial \boldsymbol{\sigma}}\left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)^T \mathbf{D}}{\left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)^T \mathbf{D}\frac{\partial g}{\partial \boldsymbol{\sigma}}}\right)}_{\mathbf{D}^{\mathrm{ep}}} \dot{\boldsymbol{\epsilon}}, \tag{2.3.11}$$

where $\mathbf{D}^{\text{ep}}$ is the elasto-plastic tangent operator, for which its numerical solution (or approximation) is the main topic of Computational Elasto-plasticity (Simo et al., 1998; Souza Neto et al., 2011). Solving an elasto-plastic problem is all about solving (or approximating) Eq. 2.3.11.

2.3.5. Integration of the constitutive elasto-plastic relation

The numerical techniques for integration of the constitutive equations are commonly referred to as *return mapping*. These fall into three types: i) explicit forward Euler (Zienkiewicz et al., 1969; Nayak et al., 1972), ii) implicit backward Euler (Simo et al., 1985a; Simo et al., 1985b; Ortiz et al., 1986; Krabbenhoft et al., 2012), and iii) closed-form exact stress integration. The latter is only available for a limited number of simple plasticity model, *e.g.,* the widely used Drucker-Prager (D-P) formulation (Drucker et al., 1952; Szabó et al., 2012; Huang et al., 2015).

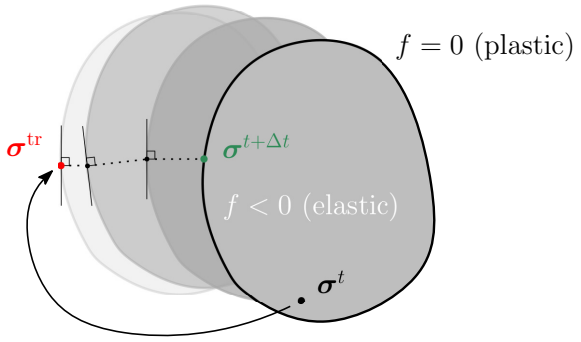


Figure 2.6. | Typical return mapping procedure (Simo et al., 1985b) to return the trial stress $\boldsymbol{\sigma}^{tr}$ onto the yield surface to satisfy $f(\boldsymbol{\sigma}^{t+\Delta t}) = 0$. Algorithm 1 shows the procedure for CPA.

A return mapping (see Fig. 2.6) involves i) an elastic predictor step (*e.g.,* trial stresses), followed by ii) a relaxation of predicated stresses onto a suitably updated yield surface by an iterative correction of the plastic strains. In particular, Simo et al., 1985a; Simo et al., 1985b; Ortiz et al., 1986; Simo et al., 1998 proposed the implicit closest point projection method (CPPM) and the implicit cutting plane algorithm (CPA), which is described in Algorithm 1 and is used in this work. A clear discussion of CPPM and CPA and their application to elasto-plastic problems in geomechanics can be found in Huang et al., 2008; Huang et al., 2009; De Borst et al., 2012; Starman et al., 2014. These two return mapping techniques are further presented. Note that a very comprehensive discussion of the limitation of an explicit treatment of return mapping is found in De Borst et al., 2012, p.239, *e.g.,* the corrected stress drifts away from the yield surface proportionally to its local curvature. This implies that the yield condition $f \leq 0$ is not satisfied locally.

Cutting-plane algorithm (CPA)

It was initially proposed by Ortiz et al., 1986 as a successive application of a number of Euler forward steps. This iterative algorithm progressively constructs tangent planes of inadmissible elastic regions, on which the yield function is evaluated. Hence, some authors (De Borst et al., 2012) refer to this algorithm as the "tangent" cutting-plane algorithm. The tangent cutting-plane algorithm is summarized in Alg. 1.

A trial stress $\boldsymbol{\sigma}^{\mathrm{tr}}$ is first computed, assuming a pure elastic behaviour, *e.g.,* an initial guess. Then, an initial yield function is estimated by $f(\boldsymbol{\sigma}_0)$, *e.g.,* $j = 0$. When $f(\boldsymbol{\sigma}_0) \leq 0$, the elastic state is admissible and, no correction is needed. Whenever $f(\boldsymbol{\sigma}_0) > 0$, the elastic state is inadmissible, by definition (see Eq. 2.3.4), and, the stress must be corrected by an iterative process to satisfy the yield condition within a given tolerance ϵ_{tol}. It is repeated as long as the corrected stress is not compliant with the yield condition evaluated on the tangent plane, *e.g.,* $f(\boldsymbol{\sigma}_{j+1}) \leq \pm\epsilon_{\mathrm{tol}}$ provided that $\epsilon_{\mathrm{tol}} \to 0$.

Algorithm 1: Cutting-plane algorithm (CPA).

$\boldsymbol{\sigma}_j = \boldsymbol{\sigma}^{tr}$ for iteration $j = 0$;
while $f(\boldsymbol{\sigma}_{j+1}) \geq \epsilon_{\mathrm{tol}}$ **do**

$$\begin{aligned} \Delta\lambda &= \frac{f(\boldsymbol{\sigma})_j}{((\partial f/\partial \boldsymbol{\sigma})_j)^T \mathbf{D} (\partial g/\partial \boldsymbol{\sigma})_j} \\ \Delta\boldsymbol{\sigma} &= \Delta\lambda \mathbf{D} (\partial g/\partial \boldsymbol{\sigma})_j \\ \boldsymbol{\sigma}_{j+1} &= \boldsymbol{\sigma}_j - \Delta\boldsymbol{\sigma} \end{aligned}$$

Implicit backward Euler method

To overcome limitations of the explicit forward Euler approach, an implicit solution (Simo et al., 1985b) is preferred to evaluate the unknown quantities $\boldsymbol{\sigma}_{j+1}$, $\Delta\lambda$ and κ_{j+1} at the end of plastic step. In addition, the yield condition is evaluated at the end of the time step and should satisfy the consistency condition, *e.g.,* $f(\boldsymbol{\sigma}_{j+1}, \kappa_{j+1}) = 0$.

To solve for these unknowns, local residuals can be formed by a set of non-linear equations. For isotropic hardening, a linear relation exists between the rate of the hardening parameter $\dot{\kappa}$ and the plastic multiplier $\dot{\lambda}$. Exploiting this proportionality (De Borst et al., 2012), the set of non-linear equations to solve for becomes

$$\begin{cases} \boldsymbol{r}_{\boldsymbol{\sigma}} &= \boldsymbol{\sigma}_{j+1} - \boldsymbol{\sigma}_0 + \Delta\lambda \mathbf{D} \partial_{\boldsymbol{\sigma}} g(\boldsymbol{\sigma}_{j+1}, \lambda_{j+1}), \\ r_\lambda &= f(\boldsymbol{\sigma}_{j+1}, \lambda_{j+1}). \end{cases} \tag{2.3.12}$$

where κ_{j+1} is computed from

$$\kappa_{j+1} = \kappa_0 + \Delta\lambda p(\boldsymbol{\sigma}_{j+1}, \lambda_{j+1}). \tag{2.3.13}$$

The system of non-linear equations can be iteratively solved, *e.g.,* the Newton-Raphson (N-R) method, such as

$$\begin{pmatrix} \boldsymbol{\sigma}_{j+1}^{k+1} \\ \lambda_{j+1}^{k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\sigma}_{j+1}^{k} \\ \lambda_{j+1}^{k} \end{pmatrix} - \begin{bmatrix} \partial_{\boldsymbol{\sigma}} \boldsymbol{r}_{\boldsymbol{\sigma}} & \partial_{\lambda} \boldsymbol{r}_{\boldsymbol{\sigma}} \\ \partial_{\boldsymbol{\sigma}} r_f & \partial_{\lambda} r_f \end{bmatrix}^{-1} \begin{pmatrix} \boldsymbol{r}_{\boldsymbol{\sigma}}^{k} \\ r_{\lambda}^{k} \end{pmatrix}, \tag{2.3.14}$$

where k denotes the iteration counter of the local N-R iteration at the material point level. The differentials of the local residuals are elaborated as

$$\partial_{\boldsymbol{\sigma}} \boldsymbol{r}_{\boldsymbol{\sigma}} = \boldsymbol{I} + \Delta\lambda \mathbf{D} \partial^2_{\boldsymbol{\sigma}\boldsymbol{\sigma}} g(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k}), \tag{2.3.15}$$

$$\partial_{\lambda} \boldsymbol{r}_{\boldsymbol{\sigma}} = \mathbf{D}[\partial_{\boldsymbol{\sigma}} g(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k}) + \Delta\lambda \partial^2_{\lambda\boldsymbol{\sigma}} g(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k})], \tag{2.3.16}$$

$$\partial_{\boldsymbol{\sigma}} r_f = \partial_{\boldsymbol{\sigma}} f(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k}), \tag{2.3.17}$$

$$\partial_{\lambda} r_f = \partial_{\kappa} f(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k}) \partial_{\lambda} \kappa(\boldsymbol{\sigma}_{j+1}^{k}, \lambda_{j+1}^{k}), \tag{2.3.18}$$

where $\partial_{\kappa} f \partial_{\lambda} \kappa = -h$ the hardening modulus. The N-R iteration is stopped once the residuals fall within a given tolerance ϵ_{tol}.

2.3.6. Pressure-sensitive Mohr-Coulomb model

An important class of elasto-plastic problems, in rocks or soils, are those derived from the Mohr-Coulomb (M-C) yield criterion. It is a pressure-dependent yield function and is expressed, in plane strain condition, as

$$f(\boldsymbol{\sigma}, \kappa) = \tau + \sigma \sin\phi - c(\kappa)\cos\phi, \tag{2.3.19}$$

where $c(\kappa)$ is the cohesion, which depends on a hardening parameter usually defined by the history of the accumulated plastic strain ϵ^p_{acc}, ϕ is the angle of internal friction, $\sigma = (\sigma_{xx} + \sigma_{yy})/2$ is the normal stress and $\tau = [(\sigma_{xx} - \sigma_{yy})^2/4 + \sigma_{xy}^2]^{1/2}$ is the shear stress.

The plastic flow rule is given by $\dot{\boldsymbol{\epsilon}}^p = \dot{\lambda}(\partial g/\partial \boldsymbol{\sigma})$, where $\dot{\lambda}$ is the plastic multiplier (*e.g.,* $\dot{\lambda} \geq 0$ and $\dot{f} = 0$). The plastic strain rate $\dot{\boldsymbol{\epsilon}}^p$ is derived from a plastic potential function, such as

$$g = \tau + \sigma \sin\psi + \alpha, \tag{2.3.20}$$

where ψ is the dilatancy angle of the material and α a constant. The chain rule (*e.g.,* $\partial_{\boldsymbol{\sigma}} g \equiv \partial_{\tau} g \partial_{\boldsymbol{\sigma}} \tau + \partial_{\sigma} g \partial_{\boldsymbol{\sigma}} \sigma$), after expanding and collecting terms, yields to

$$\frac{\partial g}{\partial \boldsymbol{\sigma}} \begin{cases} \frac{\partial g}{\partial \sigma_{xx}} = \frac{\sin\psi}{2} + \frac{(\sigma_{xx} - \sigma_{yy})}{4\tau}, \\ \frac{\partial g}{\partial \sigma_{yy}} = \frac{\sin\psi}{2} - \frac{(\sigma_{xx} - \sigma_{yy})}{4\tau}, \\ \frac{\partial g}{\partial \sigma_{xy}} = \frac{\sigma_{xy}}{\tau}. \end{cases} \tag{2.3.21}$$

In view of the implicit backward Euler stress integration, the second derivatives of the plastic potential are also required and read

$$\frac{\partial^2 g}{\partial \boldsymbol{\sigma}^2} \begin{cases} \frac{\partial^2 g}{\partial \sigma_{xx}^2} = \frac{1}{4\tau}, \\ \frac{\partial^2 g}{\partial \sigma_{yy}^2} = -\frac{1}{4\tau}, \\ \frac{\partial^2 g}{\partial \sigma_{xy}^2} = \frac{1}{\tau}. \end{cases} \tag{2.3.22}$$

The derivatives of the yield function are derived similarly. A proper return mapping (*e.g.,* CPA or the implicit backward Euler method) can be used to solve for the elasto-plastic problem. However, Simpson, 2017 proposed the following algorithm to return stresses to the yield surface,

$$\sigma_{xx}^{j+1} = \sigma + (\sigma_{xx}^{j} - \sigma_{yy}^{j})\beta/2, \tag{2.3.23}$$
$$\sigma_{yy}^{j+1} = \sigma - (\sigma_{xx}^{j} - \sigma_{yy}^{j})\beta/2, \tag{2.3.24}$$
$$\sigma_{xy}^{j+1} = \sigma_{xy}^{j}\beta, \tag{2.3.25}$$

where $\beta = (|c\cos\phi - \sigma\sin\phi|)/\tau$, and σ_{xx}^{j+1}, σ_{yy}^{j+1} and σ_{xy}^{j+1} are the corrected stresses, *e.g.,* $f = 0$. It could be considered as a closed-form solution for the stress integration of the Mohr-Coulomb model, which bypasses the iterative procedure to integrate the elasto-plastic problem. From a computational point of view, it is interesting to avoid a more expansive Newton-Raphson procedure to solve for the residuals.

2.3.7. Pressure-sensitive Drucker-Prager model

The Drucker-Prager model has been established as an approximation of the M-C model (Alejano et al., 2012; Krabbenhoft et al., 2012), *e.g.,* a conical yield surface which approximates the Mohr-Coulomb yield surface in the principal stress space. The former can be adjusted by parameters, so it passes either through the outer or inner edges of the Mohr-Coulomb yield surface (Jiang et al., 2011; De Borst et al., 2012).

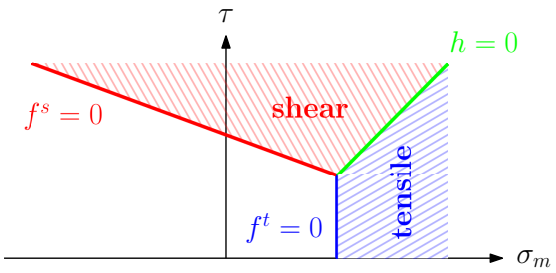


Figure 2.7. | Drucker-Prager yield surface in the (σ_m-τ) space. The yield surface is made of a shear line segment (in red) and a tensile line segment (in blue).

The D-P yield function f (see Fig. 6.1) is typically defined in terms of stress tensor invariants. The first invariant I_1 of the Cauchy stress tensor $\sigma_{ij} = \sigma_{kk}$ and the second invariant $J_2 = \frac{1}{2}\tau_{ij}\tau_{ji}$ of its deviatoric part τ_{ij}, where the deviatoric part of the Cauchy stress $\tau_{ij} = \sigma_{ij} + \delta_{ij}p$ with the pressure $p = -\frac{1}{3}\sigma_{kk}$. The D-P yield surface (in the (σ_m-τ) space, see Fig. 6.1) is made of two surfaces (*e.g.,* representing **s**hear and **t**ensile yield criteria) respectively delimited by

$$f^s(\sigma_m, \tau) = \tau + q_\phi\sigma_m - k_\phi, \tag{2.3.26}$$
$$f^t(\sigma_m) = \sigma_m - \sigma^t, \tag{2.3.27}$$

where $\tau = \sqrt{J_2}$ is the effective shear stress, $\sigma_m = -p$ is the mean stress, q_ϕ and k_ϕ are material parameters estimated with the friction angle ϕ, σ^t is the tensile strength and, c is the cohesion, which varies with the accumulated plastic strain $\bar{\epsilon}_p$ when considering a strain softening material, *e.g.*, $c = f(\bar{\epsilon}_p)$. These two surfaces define two plastic regions (domains 1 & 2, see Fig. 6.1), corresponding either to shear or tensile failure mode.

When considering shear and tensile failures for a non-associated plastic flow law, the plastic potential function g reads

$$g^s(\sigma_m, \tau) = \tau + q_\psi \sigma_m, \tag{2.3.28}$$

$$g^t(\sigma_m) = \sigma_m, \tag{2.3.29}$$

where q_ψ is a material parameter estimated with the dilation angle ψ. Note that if $q_\psi = q_\phi$, then the plastic flow rule is associated.

The line segment $h(\sigma_m, \tau) = 0$ represents the diagonal line between $f^s(\sigma_m, \tau) = 0$ and $f^t(\sigma_m, \tau) = 0$ in the (σ_m, τ) plane, *e.g.*, h is the boundary between shear and tensile failure modes. The function $h(\sigma_m, \tau)$ is given by

$$h(\sigma_m, \tau) = \tau - \tau^P - \alpha^P(\sigma_m - \sigma^t), \tag{2.3.30}$$

with the constants $\tau^P = k_\phi - q_\phi \sigma^t$ and $\alpha^P = (1 - q_\phi^2)^{1/2} - q_\phi^2$.

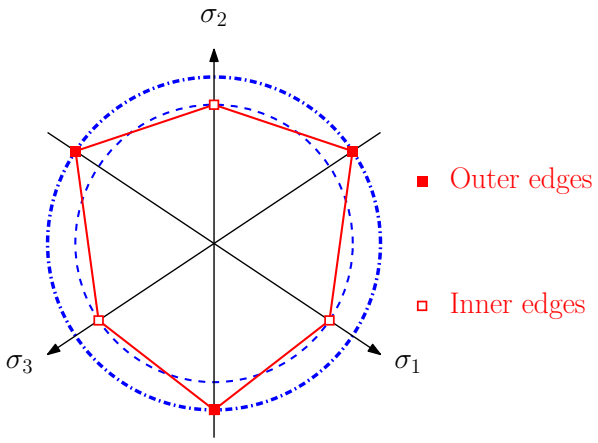


Figure 2.8. | Representation of M-C and D-P yield surfaces in the $\pi-$ plane (*e.g.*, orthogonal to the space diagonal $\sigma_1 = \sigma_2 = \sigma_3$). The D-P yield surfaces represent the inner and outer approximation of M-C yield surface.

If considering an inner adjustment (see Fig. 2.8) of the D-P yield surface w.r.t the M-C yield surface (Souza Neto et al., 2011), the model parameter

used in Eqs. 6.2.24 & 6.2.26 are given by

$$q_\phi = \frac{6\sin\phi}{\sqrt{3}(3+\sin\phi)}, \tag{2.3.31}$$

$$q_\psi = \frac{6\sin\psi}{\sqrt{3}(3+\sin\psi)}, \tag{2.3.32}$$

$$k_\phi = \frac{6c\cos\phi}{\sqrt{3}(3+\sin\phi)}. \tag{2.3.33}$$

The plastic correction algorithm is given in details in Algorithm 2 (Huang et al., 2015). This closed-form exact solution of the D-P model is used in this work and others (Liu et al., 2018a; Chalk et al., 2020; Liu et al., 2020; Nguyen et al., 2020), because of its ease of implementation within explicit numerical solvers. Compared to the M-C closed-form solution of Simpson, 2017, the D-P closed-form solution allows i) to account for dilatant material, *e.g.,* $\psi > 0$ and $\epsilon_v^p \neq 0$, ii) to consider both associated and non-associated flow rules and, iii) is not restricted only to two-dimensional configurations.

Algorithm 2: Closed-form solution for D-P.

$\boldsymbol{\sigma}^{tr}, \sigma_m^{tr}, \tau^{tr}$
if $f^s(\sigma_m^{tr}, \tau^{tr}) > 0 \wedge \sigma_m^{tr} < \sigma^t$ **then**
  $\Delta\lambda^s = f^s(\boldsymbol{\sigma}^{tr})/(G + Kq_\phi q_\psi)$
  $\Delta\epsilon^p_{\text{eqv}} = \Delta\lambda^s(1/3 + (2/9)q_\psi^2)^{1/2}$
  $\sigma_m^{t+\Delta t} = \sigma_m^{tr} - Kq_\psi\Delta\lambda$
  $\tau^{t+\Delta t} = k_\phi - q_\phi\sigma_m^{t+\Delta t}$
  $\boldsymbol{\sigma}^{t+\Delta t} = \boldsymbol{\tau}^{tr}(\tau^{t+\Delta t}/\tau^{tr}) + \sigma_m^{t+\Delta t}\boldsymbol{\delta}$
else if $h(\sigma_m^{tr}, \tau^{tr}) \leq 0 \wedge \sigma_m^{tr} \geq \sigma^t$ **then**
  $\Delta\lambda^t = (\sigma_m^{tr} - \sigma^t)/K$
  $\Delta\epsilon^p_{\text{eqv}} = \sqrt{2}/3\Delta\lambda^t$
  $\boldsymbol{\sigma}^{t+\Delta t} = \boldsymbol{\sigma}^{tr} + (\sigma^t - \sigma_m^{tr})\boldsymbol{\delta}$
else if $h(\sigma_m^{tr}, \tau^{tr}) > 0 \wedge \sigma_m^{tr} \geq \sigma^t$ **then**
  $\Delta\lambda^s = f^s(\boldsymbol{\sigma}^{tr})/(G + Kq_\phi q_\psi)$
  $\Delta\epsilon^p_{\text{eqv}} = \Delta\lambda^s(1/3 + 2q_\psi^2/9)^{1/2}$
  $\sigma_m^{t+\Delta t} = \sigma_m^{tr} - Kq_\psi\Delta\lambda^s$
  $\tau^{t+\Delta t} = k_\phi - q_\phi\sigma_m^{t+\Delta t}$
  $\boldsymbol{\sigma}^{t+\Delta t} = \boldsymbol{\tau}^{tr}(\tau^{t+\Delta t}/\tau^{tr}) + \sigma_m^{t+\Delta t}\boldsymbol{\delta}$
else
  $\boldsymbol{\sigma}^{t+\Delta t} = \boldsymbol{\sigma}^{tr}$

3. Numerical framework

3.1. The distinct element method

3.1.1. Essential overview

The distinct element method (DEM) was originally proposed by Cundall, 1971; Cundall et al., 1979 to approximate numerically the bulk behaviour of granular materials. Particles are approximated by circles (2D) & spheres (3D) to investigate the ubiquitous behaviour of granular material. The motion of the discrete assembly of particles is governed by the equations of motion of Newtonian Mechanics to resolve translation and rotation. The particles eventually overlap during contacts, which implies a smooth deformation process as opposed to the non-smooth mechanics of the more recent Contact Dynamics (CD, see e.g., Jean 1999; Moreau 2000; Dubois et al. 2018).

3.1.2. Mathematical formulations

Two contact models are commonly used : linear spring-dashpot (LSD) model and nonlinear spring-dashpot (HSD) model (often referred as the Hertz-Mindlin model, see e.g., Malone et al., 2008; Morris et al., 2016). The impact force f is strongly governed by the normal overlap ξ_n between a pair of particles: linear (LSD) and nonlinear (HSD) response, i.e. $f \propto \xi_n$ and $f \propto \xi_n^{\lambda}$, respectively. The deformation is decomposed into i) an elastic contribution as an elastic spring κ (i.e., the material stiffness) and, ii) a viscous contribution as a viscous dashpot ν (i.e., the viscous dissipation). This provides a linear or nonlinear visco-elastic force law to resolve the impact forces within the system.

Due to the widely accepted use of spherical discs, the Hertz-Mindlin contact force model is often preferred. Indeed, other geometries also requires greater computational ressources.

Contact detection and relative motion

A contact between the i-th and j-th particles occurs when

$$\xi_n = \max(0, (r_i + r_j) - ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2), \tag{3.1.1}$$

which implies $\xi_n \geq 0$ for any contact and $\xi_n < 0$ when particle i and particle j do not overlap. The normal vector $\boldsymbol{n}_{i \mapsto j}$ ($\boldsymbol{n}_{i,j}$ in the following) at contact is given by

$$\boldsymbol{n}_{ij} = \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{||\boldsymbol{x}_i - \boldsymbol{x}_j||_2}. \tag{3.1.2}$$

An illustration of typical contact detections between multiple particles in showed in Fig. 5.3.20. Considering spherical (3D) or circular (2D) geometries for the shape of particle greatly simplifies the contact detection.

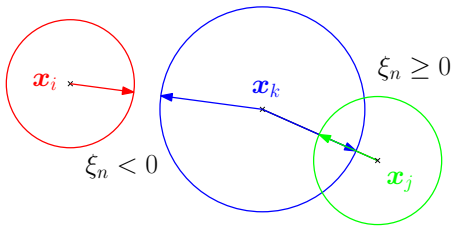


Figure 3.1. | Scheme of possible overlaps between particles i, j and k of different radii.

Newton's equation of motions and interaction forces

As stated in Poschel et al., 2005, the translational motion of a particle i interacting with a particle j is expressed as

$$m_i \partial_t \boldsymbol{v}_i = \sum_j (\boldsymbol{f}^n_{i,j} + \boldsymbol{f}^s_{i,j}) + m_i \boldsymbol{g}, \tag{3.1.3}$$

where m_i is the mass of the i-th particle, $\boldsymbol{v}_i$ its velocity vector, $\boldsymbol{g}$ the gravity and $\boldsymbol{f}^n_{i,j}$ and $\boldsymbol{f}^s_{i,j}$ are the normal and tangential forces at the point of contact between the two interacting particles.

The rotational motion of a particle i interacting with a particle j is expressed as

$$J_i \partial_t \boldsymbol{\omega}_i = \sum_j (r_i \boldsymbol{n}_{i,j} \times \boldsymbol{f}^s_{i,j}), \tag{3.1.4}$$

where J_i is the moment of inertie of the i-th particle and $\boldsymbol{\omega}_i$ is the angular velocity.

Normal and tangential forces during an arbitrary particle contact i, j are (Brilliantov et al., 1996; Poschel et al., 2005; Boac et al., 2014)

$$\boldsymbol{f}^n_{i,j} = \max(0, \kappa_n \xi_n^{3/2} - \nu_n(\xi_n)\partial_t \xi_n)\boldsymbol{n}_{i,j}, \tag{3.1.5}$$

$$\boldsymbol{f}^s_{i,j} = -\kappa_s \boldsymbol{\zeta}_s - \nu_s(\zeta_s)\partial_t \boldsymbol{\zeta}_s, \tag{3.1.6}$$

where κ_n and κ_s are normal and tangential stiffness coefficients (elastic springs), ν_n and ν_s are normal and tangential damping coefficients (viscous dash-pot) and finally, ξ_n is the normal overlap (normal strain) and $\boldsymbol{\zeta}_s$ is accumulated tangential overlap (shear strain). The latter is given by

$$\boldsymbol{\zeta}_s = \int_{t \in t_c} \boldsymbol{v}_s \mathrm{d}t, \tag{3.1.7}$$

with $\boldsymbol{v}_s$ the projection of the relative velocity $\boldsymbol{v}^r_{ij} = \boldsymbol{v}_i - \boldsymbol{v}_j$ in the tangential contact plane of particle i and t_c the collision duration of the particle i with

the particle j. The Mohr-Coulomb failure criterion restricts the tangential force to a maximum shear force, such as

$$\boldsymbol{f}^s_{i,j} = \min(||\ \boldsymbol{f}^s_{i,j}||_2, \mu||\boldsymbol{f}^n_{i,j}\ ||_2)\boldsymbol{s}_{i,j}, \tag{3.1.8}$$

where μ is the friction coefficient and $\boldsymbol{s}_{i,j} = \boldsymbol{f}^s_{i,j}/||\boldsymbol{f}^s_{i,j}||_2$ is the unit tangential vector.

Normal and tangential stiffness coefficients are defined as following (Boac et al., 2014)

$$\kappa_n = \frac{4}{3}E^\star\sqrt{R^\star} \quad \text{and} \quad \kappa_s = 8G^\star\sqrt{R^\star\xi_n} \tag{3.1.9}$$

where $R^\star$,$E^\star$ and $G^\star$ are respectively the equivalent radius, equivalent Young's shear moduli. Similarly, normal & shear damping coefficients are defined as following (Tsuji et al., 1992; Remy et al., 2009; Langlois et al., 2015)

$$\nu_n(\xi_n) = \gamma\sqrt{m^\star\kappa_n}\xi_n^{1/4} \quad \text{and} \quad \nu_s(\zeta_s) = \gamma\sqrt{m^\star\kappa_s}\zeta_s^{1/4}, \tag{3.1.10}$$

where $\gamma = -\ln(e)/(\ln^2(e) + \pi^2)^{1/2}$, e is the restitution coefficient and $m^\star$ is the equivalent mass.

Critical time step

Accordingly to Li et al., 2005, the critical time step is expressed as:

$$\tau_{crit} = \frac{\pi\langle R\rangle}{\beta}\sqrt{\frac{\rho}{G}} \tag{3.1.11}$$

where $\langle R\rangle$ is the average particle radius, G is the shear modulus and β is given by:

$$\beta = 0.8766 + 0.163\nu \tag{3.1.12}$$

where ν is the Poisson's ratio.

3.1.3. Exploratory numerical simulations

Selected numerical results are presented: static friction, impact cratering and the force chain network. These preliminary investigations are showed only to demonstrate the potential of DEM.

A simple numerical example

A rapid numerical demonstration of a contact between two spherical particles, i.e., a binary system in a two-dimensional configuration with coefficient of restitution is $e = 0.1$, is shown in Fig. 3.2.

The evolution of the binary system is showed in Fig. 3.2 for two contact angles α_{ij}. When $\alpha_{ij} \leq \tan^{-1}\mu$, the tangential interaction is sticky, i.e., the tangential force is not sufficient to exceed the Mohr-Coulomb criterion to generate a slip. As a consequence, any tangential relative motion is prevented,

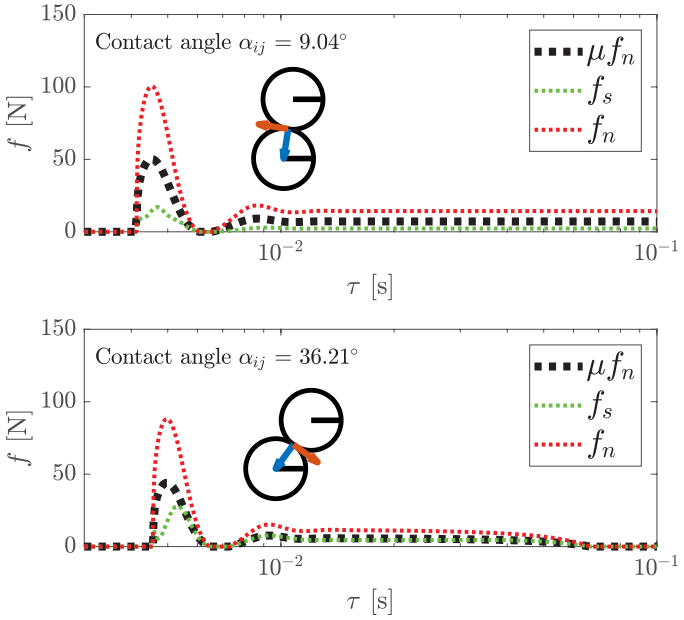


Figure 3.2. | Static and dynamic friction (stick-slip): two different contact angle α_{ij} for a constant friction coefficient $\mu = 0.5 \Rightarrow \tan^{-1}\mu \approx 26°$. The blue and orange arrows denote the normal and tangential vectors, respectively.

i.e., the binary system is in a static equilibrium. This stick-slip behaviour is particularly obvious if one looks at the time evolution of the tangential force (the dotted green line in Fig. 3.2). The black dotted line in Fig. 3.2 expresses the Mohr-Coulomb failure envelope. As long as the tangential force is inferior, no relative tangential motion is observed.

Impact cratering

An other potential investigation is the case of a solid impact at moderate velocity (i.e., 5 - 10 m·s^{-1}) of a circular intruder onto a cohesion-less and poly-disperse granular layer, which is horizontally confined. In case of a cohesive material, the cohesion force is simply added in the formulation of the normal contact force in Eq. 3.1.3 (Luding, 2008; Rojek et al., 2012; Parteli et al., 2014).

A macroscopic description of a given property is not directly available. For instance, the porosity of the granular layer is not directly given and, it must be reconstructed (Rycroft et al., 2009).

An example of a reconstructed porosity field is given in Fig. 3.3. The polydispersity of the granular material results in a heterogeneous porosity field, as demonstrated by the spatial variations in Fig. 3.3. Each cell has to be sufficiently large, compared to the particles, to be a representative element of the porosity field, i.e., its length scale should be higher than the maximum size of a particle within the system.

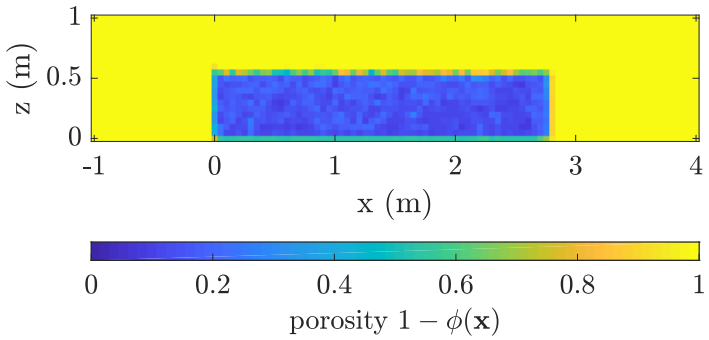


Figure 3.3. | Reconstruction of the porosity field for a granular layer in a static equilibrium before the impact.

The numerical method allows to simulate the motions and the velocities of every particles in the system at the particle level and, such microscopic description can be resolved in DEM (see Fig. 3.4).

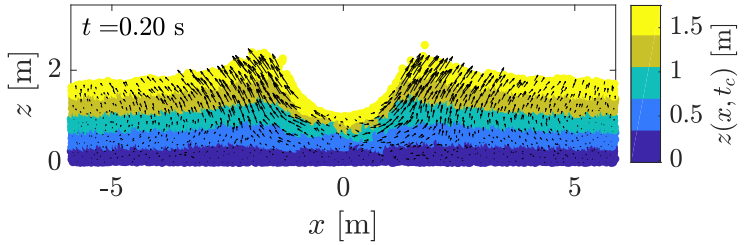


Figure 3.4. | Impact of a circular intruder onto a cohesion-less and poly-disperse granular material.

Force chain network in static assemblies

One of the most important characteristic of granular matter (in a static state) is the force chain network (Ostojic et al., 2006; Abed Zadeh et al., 2019). It describes the self-organizing behaviour of the contact forces within the granular assembly. It is responsible for arching and jamming phenomena (Vivanco et al., 2012; Behringer, 2015), both influencing the mechanical stability of the granular assembly.

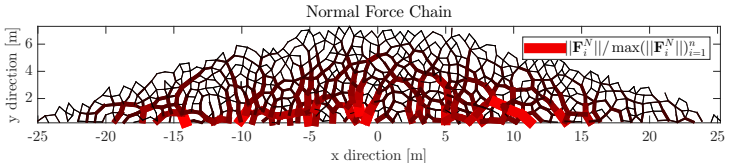


Figure 3.5. | Normalized force chain network within a static polydisperse granular pile.

The force chain network (see Fig. 3.5) gives a unique microscopic information about the normal forces at the particle level. It results also in a highly non-linear bulk behaviour, i.e., asymmetric patterns of deformation during solid impacts. The force chain network is typically a microscopic characteristic of granular matter that is far beyond the scope of continuum models governed by constitutive relations.

3.1.4. Different type of problems

Calibration

DEM introduce numerical parameters that need experimental calibrations to produce accurate numerical results (Yan et al., 2015). Among those, the restitution coefficient is a parameter significantly discussed an estimated for a variety of problems (Roessler et al., 2019; Jiang et al., 2020; Wang et al., 2020b). As a consequence, a consistent numerical analysis prior to an experimental calibration in laboratory is precluded.

Scale

Whereas the traditional continuum mechanics relies on a macroscopic description, i.e., the macroscopic stress-strain response, the discontinuum framework of DEM implies a microscopic description of matter, i.e., the contact forces within the system. The scale, at which processes are described, is significantly different between these two formulations. Averaging techniques are required (Lätzel et al., 2000; Nicot et al., 2013; Weinhart et al., 2016; Cui et al., 2017) to define macroscopic tensorial quantities, i.e., to obtain comparable results for the stress-strain response. As an example, Christoffersen et al., 1981; Lätzel et al., 2000; Bagi, 2006 gave an average description of the Cauchy stress and the strain tensors. This procedure is necessary and complementary to any DEM-based numerical analysis. However, the advantage is the constitutive behaviour of the system is based on the micromechanics of contacts and not on a constitutive model.

3.1.5. Particle geometry

Even tough the spherical assumption is convenient, this far from the reality. DEM suffers from a poor representation of actual shapes within a granular assembly (Kawamoto et al., 2018; Li et al., 2019), i.e., granular matter is rarely made of spherical particles. The shape has a significant influence. As such, DEM does not adequately reproduce the bulk behaviour (Kawamoto et al., 2018). Staron et al., 2005 demonstrated that the circular shape leaded to greater run-out distance of granular collapses when using DEM. This was also thoroughly demonstrated by Lim et al., 2014b; Lim et al., 2014c; Lim et al., 2014a, who proposed the Granular Element Method combined with CD, using Non-uniform radial basis spline (NURBS) to simulate a variety of non-convex shapes. More recently, Kawamoto et al., 2018 proposed to use the level set

method in DEM (LS-DEM, see also Li et al. 2019) to accurately resolve the complex shapes of particles within a granular assembly, obtained by X-ray tomography imaging. Specifically, their model was able to predict the onset and evolution of shear banding in sands (during a triaxial test) with a high degree of agreement with experimental results, as shown in Fig. 3.6. It shows the incremental particle rotations between the experiment and the numerical model.

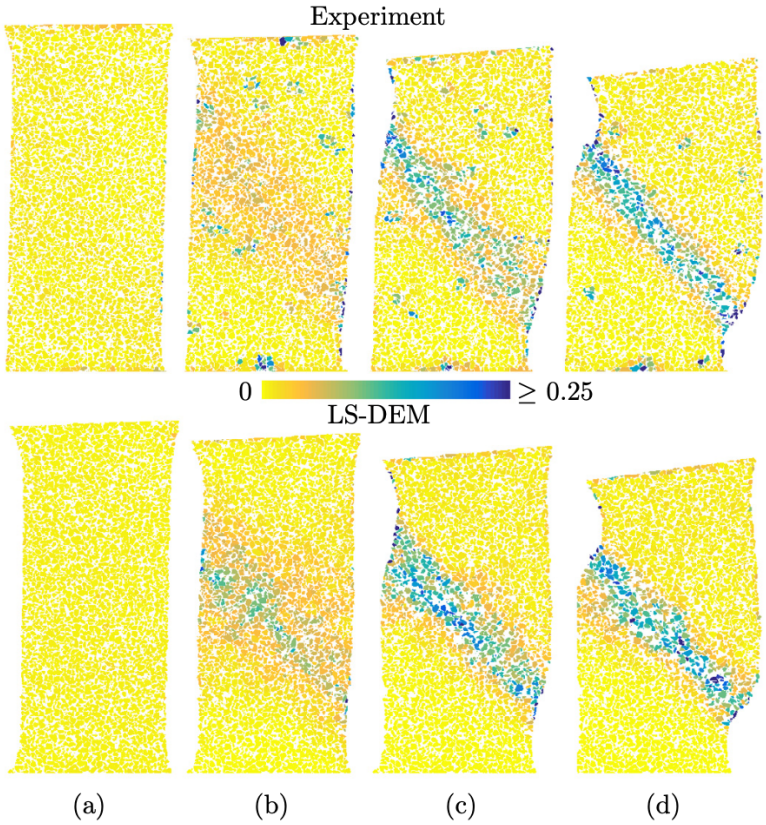


Figure 3.6. | Experimental and LS-DEM incremental particle rotations for different ranges of axial strain, a) 0.0-0.6 %, b) 3.9-5.1 %, c) 8.6-10.0 % and d) 13.3-14.8 %, taken from Kawamoto et al., 2018. Both timing and location of strain localization is captured by the numerical model.

3.2. The material point method (MPM)

3.2.1. Essential overview

The standard material point method (sMPM) originates from the Particle-In-Cell method, as an extension to solve for large deformation problems in Solid Mechanics. It was first referred by Sulsky et al., 1994; Sulsky et al., 1995; Sulsky et al., 1996, who provided the first formal definition of the numerical

method. Hence, it is a relatively new numerical method compared to the well-established Finite Difference and/or Finite Element Methods.

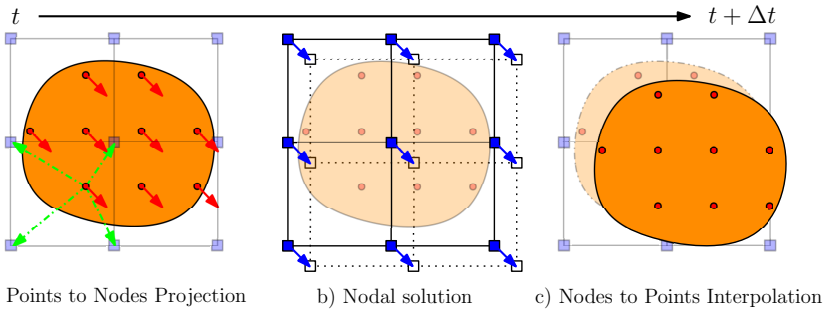


Figure 3.7. | Typical calculation cycle of a MPM solver for a homogeneous velocity field, inspired by Dunatunga et al., 2017. a) The continuum (orange) is discretized into a set of Lagrangian material points (red dots), at which state variables or properties (e.g., mass, stress, and velocity) are defined. The latter are mapped to an Eulerian finite element mesh made of nodes (blue square). b) Momentum equations are solved at the nodes and, the solution is explicitly advanced forward in time. c) The nodal solutions are interpolated back to the material points and, their properties are updated.

MPM can be categorized as an advanced formulation of FEM, where the continuum body is represented by a set of Lagrangian points, called *material points*. They serve the dual purpose of carriers (i.e., they carry state variables such as stresses or velocities) and integration points. They move through an Eulerian background FE mesh, on which equations are solved. Whereas the Gauss-Legendre quadrature is used in FEM, it is replaced by a so-called point or point-wise quadrature in MPM. It can be regarded as a Finite Element procedure during which integration points are allowed to move (Guilkey et al., 2003). This allows MPM to handle large deformations, unlike FEM, but at the expense of a lower numerical accuracy since material points are not always located at the optimal Gauss-Legendre locations (Steffen et al., 2008b; Steffen et al., 2008a). A typical computational step in MPM is showed in Fig. 3.7.

3.2.2. The cell-crossing instability: the generalized interpolation material point method (GIMPM)

One of the most important problem of any sMPM formulation is the cell-crossing instability (or error). As material points move through the mesh, they cross element boundaries. The discontinuous gradient due to the C_0 continuity of the basis functions results in spurious oscillations of the stress field and internal forces (Bardenhagen et al., 2004; González Acosta et al., 2019; González Acosta et al., 2020) when material points cross element boundaries.

To solve for this instability, Bardenhagen et al., 2004 introduced the Generalized Interpolation material point method (GIMPM). Whereas the material point is treated as a point in sMPM, Bardenhagen et al., 2004 assigned a *spa-*

tial extent or a *domain* to the material point. Alternative basis functions are constructed, i.e., to consider the material point's domain, as followed

$$\phi_{np} \equiv \phi_n(\boldsymbol{x}_p) = \frac{1}{v_p} \int_{\Omega_p \subset \Omega} \chi_p(\boldsymbol{x}) N_n(\boldsymbol{x}) \mathrm{d}\Omega, \tag{3.2.1}$$

where v_p is the material point's volume, Ω_p denotes the material point's domain, $\chi_p(\boldsymbol{x})$ is the *particle characteristic function*, $N_n(\boldsymbol{x})$ is the basis function (or shape function) for the mapping between the material point p and its associated node n and $\boldsymbol{x} = \boldsymbol{x}_p - \boldsymbol{x}_n$ are the local coordinates between the node n and the material point p.

The particle characteristic function must satsify the partition of unity property, i.e., $\sum_p \chi_p(\boldsymbol{x}) = 1$ (Bardenhagen et al., 2004). The simplest particle characteristic function is given by the hat function, i.e.,

$$\chi_p(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} \subset \Omega_p, \\ 0 & \text{otherwise.} \end{cases} \tag{3.2.2}$$

The GIMPM basis functions and derivatives are constructed analytically (Charlton et al., 2017; Coombs et al., 2020b) in one dimension from a convolution of the standard finite element basis functions and the material point characteristic function (Steffen et al., 2008b), i.e.,

$$\phi_n(x_p) = \begin{cases} 1 - (4x^2 + l_p^2)/(4hl_p) & \text{if } |x| < l_p/2 \\ 1 - |x|/h & \text{if } l_p/2 \leq |x| < h - l_p/2 \\ (h + l_p/2 - |x|)^2 / (2hl_p) & \text{if } h - l_p/2 \leq |x| < h + l_p/2 \\ 0 & \text{otherwise ,} \end{cases} \tag{3.2.3}$$

with l_p the length of the material point domain, h the mesh spacing, $x = x_p - x_n$ where x_p is the coordinate of a material point and x_n the coordinate of its associated node n. The two-dimensional basis function of a node n with its material point p is constructed as

$$\phi_{np} \equiv \phi_n(\boldsymbol{x}_p) = \phi_n(x_p)\phi_n(y_p), \tag{3.2.4}$$

for which the gradient is defined as

$$\nabla\phi_{np} \equiv \nabla\phi_n(\boldsymbol{x}_p) = (\partial_x\phi_n(x_p)\phi_n(y_p), \phi_n(x_p)\partial_y\phi_n(y_p)). \tag{3.2.5}$$

Sadeghirad et al., 2011 and Sadeghirad et al., 2013 proposed the Convected Particle Domain Interpolation (CPDI) and its second-order variant (CPDI2q). Essentially, the material point's domain is considered here as a deforming parallelogram (CPDI) or as a deforming quadrilateral (CPDI2q). However, Wang et al., 2019 showed that i) CPDI2q suffers from highly deformed material points under given deformation modes and, ii) CPDI is only superior to GIMPM for problems involving massive stretching deformation modes. Similarly, Charlton et al., 2017; Coombs et al., 2020b argued for the simplicity and

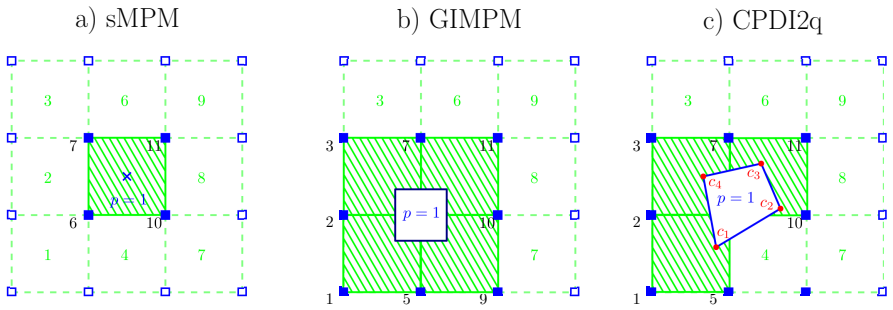


Figure 3.8. | Nodal connectivities of a) sMPM, b) GIMPM and c) CPDI2q variants. The material point's location is marked by the blue cross. Note that for sMPM, the particle domain does not exist, unlike GIMPM or CPDI2q (the blue square enclosing the material point). Nodes associated with the material point are denoted by filled blue squares, and the element number appears in green in the centre of the element.

robustness of analytical solutions to the basis functions (e.g., GIMPM) rather than linear numerical approximations involved in the most recent CPDI and CPDI2q techniques. The major improvements from the original sMPM are shown in Fig. 3.8. Aside from domain-based material point methods, further variants exists, i.e., the B-spline material point method (BSMPM, see Gan et al. 2018; Wobbes et al. 2019; Koster et al. 2021) or the dual domain material point method (DDMPM, see Tran et al. 2019b)

3.2.3. Domain updating method in GIMPM

Two main variants of domain-based method exists: The material point's domain is a square for which the deformation is always aligned with the mesh axis, i.e., a non-deforming domain uGIMPM (Bardenhagen et al., 2004) or, a deforming domain cpGIMPM (Wallstedt et al., 2008), the latter being usually related to a measure of the deformation, e.g., the determinant of the deformation gradient.

Four domain updating methods exists: i) the domain is not updated, ii) the deformation of the domain is proportional to the determinant of the deformation gradient $\det(F_{ij})$ (Bardenhagen et al., 2004), iii) the domain lengths l_p are updated accordingly to the principal component of the deformation gradient F_{ii} (Sadeghirad et al., 2011) or, iv) are updated with the principal component of the stretch part of the deformation gradient U_{ii} (Charlton et al., 2017), which is rotation-free by definition of the left stretch part. Coombs et al., 2020b highlighted the suitability of generalised interpolation domain updating methods accordingly to distinct deformation modes. Four different deformation modes were considered by Coombs et al., 2020b: simple stretch, hydrostatic compression/extension, simple shear and, pure rotation. Coombs et al., 2020b concluded the following:

- Not updating the domain is not suitable for simple stretch and hydro-

static compression/extension.

- A domain update based on $\det(F_{ij})$ will results in an artificial contraction/expansion of the domain for simple stretch.
- The domain will vanish with increasing rotation when using F_{ii}.
- The domain volume will change under isochoric deformation when using U_{ii}.

Consequently, Coombs et al., 2020b proposed a hybrid domain update inspired by CPDI2q approaches: the corners of the material point domain are updated accordingly to the nodal deformation but, the midpoints of the domain limits are used to update domain lengths l_p to maintain a rectangular domain. Even tough Coombs et al., 2020b reported an excellent numerical stability, the drawback is to compute specific basis functions between nodes and material point's corners, which has an additional computational cost.

Regarding the recent CPDI/CPDI2q, Wang et al., 2019 investigated the numerical stability under stretch, shear and torsional deformation modes. CPDI2q was found to be erroneous in some case, especially when torsion mode is involved, due to distortion of the domain. In contrast, CPDI and even sMPM performed better in modelling torsional deformations. Even tough CPDI2q can exactly represent the deformed domain (Sadeghirad et al., 2013), care must be taken when dealing with very large distortion, especially when the material has yielded, which is common in geotechnical engineering (Wang et al., 2019).

3.2.4. Governing equations and weak formulation

The governing equations for mass (mass balance equation) and momentum (momentum balance equation) conservation, for a continuum material enclosed within the domain Ω, are

$$\partial_t \rho + \rho \partial_i v_i = 0, \tag{3.2.6}$$

$$\partial_j \sigma_{ij} + \rho b_i = \rho \partial_t v_i, \tag{3.2.7}$$

where ρ is the density, v_i is the velocity, σ_{ij} is the Cauchy stress tensor and b_i is the body force. Dirichlet and Neumann boundary conditions are:

$$u_i = \hat{u}_i \quad \text{on} \quad \partial\Omega_u, \tag{3.2.8}$$

$$\sigma_{ij} n_j = \hat{\tau}_i \quad \text{on} \quad \partial\Omega_\tau, \tag{3.2.9}$$

where u_i is the displacement, $\hat{u}_i$ and $\hat{\tau}_i$ are prescribed displacement and traction on a boundary surface $\partial\Omega$ with an outward unit normal vector n_i, respectively. Conservation of angular momentum simply results in the symmetry of the Cauchy stress tensor, i.e., $\sigma_{ji} = \sigma_{ij}$.

Using the principle of virtual work, the weak form of Eq. 3.2.7 is found by taking the product of Eq. 3.2.7 with a test function (or virtual displacement)

and integrating over the current configuration, i.e., an updated-Lagrangian configuration. The weak form of Eq. 3.2.7 is given by, e.g., see De Borst et al., 2012; Belytschko et al., 2013,

$$\int_\Omega \rho\delta u_i b_i \mathrm{d}\Omega - \int_\Omega \rho\partial_j \delta u_i \sigma^s_{ij} \mathrm{d}\Omega + \int_\Gamma \rho\delta u_i \tau^s_i \mathrm{d}\Gamma = \int_\Omega \rho\delta u_i \partial_t v_i \mathrm{d}\Omega, \tag{3.2.10}$$

where δu_i is the test function, $\sigma^s_{ij} = \sigma_{ij}/\rho$ is the specific Cauchy stress tensor, $\tau^s_i = \tau_i/\rho$ is the specific traction at the boundary surface $\Gamma = \partial\Omega$ of the continuum body in current configuration Ω.

By convention in the following, the subscripts n refer to nodes, p refer to material points and i refer to the spatial components n or p.

The continuum Ω is discretized into a set of n_p material point sub-domains Ω_p. Here, the mass of Ω_p is concentrated at the material point location, and the density field at the coordinate x_i is

$$\rho(x_i) = \sum_{p=1}^{n_p} m_p \delta(x_i - x_{i,p}), \tag{3.2.11}$$

where δ is the Dirac delta function and m_p is the mass of the material point p.

Using the identity $\int f(x_i)\delta(x_i - x_{i,p}) = f(x_{i,p}) \equiv f_{i,p}$ (i.e., f evaluated at i^{th} component of the p^{th} material point), neglecting traction on boundary (for simplification) and substituting Eq. 3.2.11 into Eq. 3.2.10 yields to

$$\sum_{p=1}^{n_p} m_p \delta u_{i,p} b_{i,p} - \sum_{p=1}^{n_p} v_p \partial_j \delta u_{i,p} \sigma_{ij,p} = \sum_{p=1}^{n_p} m_p \delta u_{i,p} \partial_t v_{i,p}, \tag{3.2.12}$$

where the volume of the material point is $v_p = m_p/\rho$.

The domain Ω is further decomposed into a set of finite subdomains Ω_e, i.e., the finite elements of the background mesh. Each element is connected with its surrounding nodes. The discretized form is obtained by approximating the acceleration $\boldsymbol{a} = \partial_t \boldsymbol{v}$ using basis functions ϕ_n, i.e.,

$$a_i(\boldsymbol{x}_p) = \sum_{n=1}^{n_{no}} \phi_n(\boldsymbol{x}_p) a_{i,n}, \tag{3.2.13}$$

and, the virtual displacement field and its spatial derivatives are approximated as

$$\delta u_i(\boldsymbol{x}_p) = \sum_{n=1}^{n_{no}} \phi_n(\boldsymbol{x}_p) \delta u_{i,n}, \tag{3.2.14}$$

$$\partial_j u_i(\boldsymbol{x}_p) = \sum_{n=1}^{n_{no}} \partial_j \phi_n(\boldsymbol{x}_p) \delta u_{i,n}. \tag{3.2.15}$$

Substituting the Finite Element approximation of Eqs. 3.2.13 - 3.2.15 in Eq. 3.2.12 leads to

$$\sum_{p=1}^{n_p} m_p \phi_n(\boldsymbol{x}_p) b_{i,p} - \sum_{p=1}^{n_p} v_p \partial_j \phi_n(\boldsymbol{x}_p) \sigma_{ij} = \sum_{p=1}^{n_p} m_p \phi_n(\boldsymbol{x}_p) \left(\sum_{m=1}^{n_{no}} \phi_n(\boldsymbol{x}_p) a_{i,m} \right), \tag{3.2.16}$$

which can be rewritten in the following compact form,

$$[M_{ij} a_j]_k = [f_i^{\text{ext}} - f_i^{\text{int}}]_k, \tag{3.2.17}$$

where $k = \overline{1, ..., 3}$, $M_{ij} = \sum_{p=1}^{n_p} m_p \phi_i(\boldsymbol{x}_p) \phi_j(\boldsymbol{x}_p)$ is the consistent mass matrix with $\phi_i(\boldsymbol{x}_p)$ the basis function between the node i with the material point p. This work adopts a lumped mass matrix, i.e., $m_i \equiv M_{ii} = \sum_{p=1}^{n_p} m_p \phi_i(\boldsymbol{x}_p)$, to avoids an expensive matrix inversion (Sulsky et al., 1994; Bardenhagen et al., 2004; González Acosta et al., 2020).

3.2.5. Procedure for an explicit formulation

A description of the computational procedure of MPM is presented. A typical computational cycle is made of three phases (Wang et al., 2016b; Wang et al., 2016c; Zhang et al., 2016):

1. Mapping phase between material points and their associated nodes, i.e., namely the material point's forces, momentum and mass.

2. updated-Lagrangian (FEM) solution to the momentum balance equation by an explicit foward-Euler integration in time to Eq. 6.2.5.

3. convection phase, in which i) updated nodal solution is mapped back to the material points, and ii) material point state variables are updated.

Note that implicit MPM procedures exists and are clearly presented in Beuth et al., 2008; Iaconeta et al., 2017; Coombs et al., 2020a; Zhao et al., 2020.

The standard formulation, i.e., Update Stress Last (USL), is traditionally employed. The material point's stresses are updated at the end of a computational cycle. A Modified Update Stress Last (MUSL) update is chosen here (Sulsky et al., 1994; Nairn, 2003): a remapping of the updated nodal momentum is performed after the updated nodal velocity is obtained. Nairn, 2003 showed it improved the accuracy and the stability of the numerical solution.

To solve for the acceleration in Eq. 6.2.5, a mapping of material point's contribution is performed. The external $f_{i,n}^{\text{ext}}$ and internal $f_{i,n}^{\text{int}}$ forces at the node n are then defined by

$$f_{i,n}^{\text{ext}} = \sum_{p=1}^{n_p} m_p \phi_n(\boldsymbol{x}_p) g_{i,p}, \tag{3.2.18}$$

$$f_{i,n}^{\text{int}} = \sum_{p=1}^{n_p} v_p \partial_j \phi_n(\boldsymbol{x}_p) \sigma_{ij,p}, \tag{3.2.19}$$

where m_p is the material point's mass, v_p is the material point's volume and $\sigma_{ij,p}$ is the material point's Cauchy stress tensor.

Solving Eq. 6.2.5 for the acceleration $a_{i,n}$, the updated velocity is obtained using a forward-Euler scheme, that is

$$v_{i,n}^{t+\Delta t} = v_{i,n}^{t} + \Delta t a_{i,n}, \tag{3.2.20}$$

where the velocity is given by $v_{i,n}^{t} = m_i^{-1} \sum_{p=1}^{n_p} \phi_i(\boldsymbol{x}_p) m_p v_{i,p}^{t}$ with $v_{i,p}$ the material point's velocity. Finally, boundary conditions are applied on the boundary nodes.

Since nodal information is deleted at the end of a computational step, nodal solutions have to be mapped back to the material points. The material point velocity $v_{i,p}$ is defined as an interpolation of the solution of the updated nodal accelerations, given by

$$v_{i,p}^{t+\Delta t} = v_{i,p}^{t} + \Delta t \sum_{n=1}^{n_n} \phi_n(\boldsymbol{x}_p) a_{i,n}, \tag{3.2.21}$$

which correspond to the classical FLIP update. More advanced schemes can be used, *i.e.,* the FLIP/PIC update.

The double mapping procedure (DM or MUSL) of the nodal velocity $v_{in}^{t+\Delta t}$ consists of the remapping of the updated material point momentum on the background mesh, divided by the nodal mass, given as

$$v_{i,n}^{t+\Delta t} = m_n^{-1} \sum_{p \in n} \phi_n(\boldsymbol{x}_p) m_p v_{i,p}^{t+\Delta t}, \tag{3.2.22}$$

and, for which boundary conditions are enforced. Finally, the material point coordinates are updated based on the following:

$$x_{i,p}^{t+\Delta t} = x_{i,p}^{t} + \Delta t \sum_{n=1}^{n_n} \phi_n(\boldsymbol{x}_p) v_{i,n}^{t+\Delta t}. \tag{3.2.23}$$

The last step is to update the Cauchy stress tensor $\sigma_{ij,p}$ for the material points using the incremental nodal displacement $\Delta u_{i,n} = \Delta t v_{i,n}^{t+\Delta t}$. The strain increment $\Delta \epsilon_{ij,p}$ is obtained with the incremental deformation gradient tensor $\Delta F_{ij,p}$, defined as

$$\Delta F_{ij,p} = \delta_{ij} + \sum_{n=1}^{n_n} \partial_j \phi_n(\boldsymbol{x}_p) \Delta u_{i,n}, \tag{3.2.24}$$

and, the incremental infinitesimal strain tensor (Eq. 2.1.23) becomes

$$\Delta \epsilon_{ij,p} = \frac{1}{2}(\Delta F_{ij,p} + \Delta F_{ji,p}) - \delta_{ij}. \tag{3.2.25}$$

The objective rate-formulation (Eq. 6.2.14) of the constitutive model is used to update the Cauchy stress tensor $\sigma_{ij,p}$ and, the material point's volume v_p and domain dimension $l_{i,p}$ are updated accordingly using the incremental deformation gradient $\Delta F_{ij,p}$.

3.2.6. Finite strain implementation

Within the finite deformation framework (i.e., the finite strain theory), the procedure to calculate strains is different (Souza Neto et al., 2011; Iaconeta et al., 2019). The spectral decomposition of the previous logarithmic strain tensor $\boldsymbol{\varepsilon}^0$ is used to obtain the previous left Cauchy-Green deformation tensor, such as

$$\boldsymbol{b}^0 = \sum_{i=1}^{3} \exp(2\lambda_i^0)(\boldsymbol{n}_i^0 \otimes \boldsymbol{n}_i^0), \tag{3.2.26}$$

where λ_i^0 and $\boldsymbol{n}_i^0$ are the eigenvalues and eigenvectors of $\boldsymbol{\varepsilon}^0$. Next, the left Cauchy-Green deformation tensor $\boldsymbol{b}^0$ is updated using the incremental deformation gradient, i.e.,

$$\boldsymbol{b} = \Delta\boldsymbol{F} \cdot \boldsymbol{b}^0 \cdot \Delta\boldsymbol{F}^T. \tag{3.2.27}$$

Calculating λ_i^2 and $\boldsymbol{n}_i$ of $\boldsymbol{b}$, one can use again the spectral decomposition theorem to obtain the updated logarithmic strain tensor, that is

$$\boldsymbol{\varepsilon} = \frac{1}{2} \sum_{i=1}^{3} \ln \lambda_i (\boldsymbol{n}_i \otimes \boldsymbol{n}_i). \tag{3.2.28}$$

The updated logarithmic strain tensor $\boldsymbol{\varepsilon}$ can be used to calculate the Kirchhoff stress tensor $\boldsymbol{\tau}$ using Eq. 2.2.22.

3.2.7. Rate-dependent or finite strain formulation ?

A majority of MPM implementation relies on the rate-dependent formulation (Abe et al., 2014; Bandara et al., 2015; Soga et al., 2016; Baumgarten et al., 2019a; Liu et al., 2020; Li et al., 2021) with an objective stress-rate measure, whereas a less significant number of researchers preferred the finite strain theory (Gaume et al., 2018; Iaconeta et al., 2019; Müller et al., 2019). A comparison of both rate-dependent and finite strain formulations (implemented in MATLAB under a plane strain configuration) for an elastodynamic problem in a one dimensional configuration is presented in the following.

One dimensional compression under self weight

The benchmark case of the the vertical compression of an elastic column (Charlton et al., 2017) of an initial height $l = 50$ m, subjected to an external load (e.g., the gravity), is investigated. The material has a Young's modulus $E = 1 \cdot 10^3$ kPa and a Poisson's ratio $\nu = 0$ with a density $\rho = 80$ kg·m^{-3}. The gravity g is increased from 0 to 9.81 m·s^{-2}. The domain update based on the diagonal component of the left stretch tensor U_{ii} is selected, as in Charlton et al., 2017; Coombs et al., 2020b. At the end of the analyis, the final height of the column should be ≈ 24 m. The difference between the two formulations

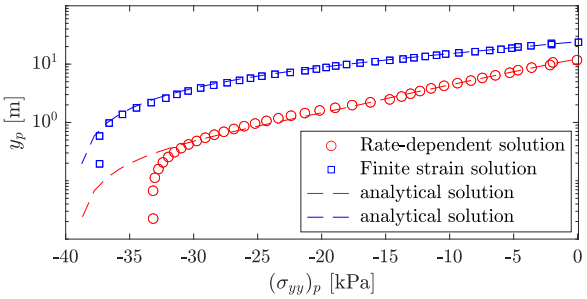


Figure 3.9. | When considering a soft material (i.e., $E = 1 \cdot 10^3$ Pa), the difference between rate-dependent and finite strain formulations is significant: the latter better resolves the elastic compression of the column as well as the final height of the column.

is significant (see Fig. 3.9). For an elastic compression of the column $\approx 50\%$, the rate-dependent formulation is not able to resolve the elastic deformation of the column. Not only the rate-dependent formulation struggles to resolve vertical stresses at the base of the column but, it also predicts a higher elastic compression of the column. In this case, the finite strain formulation is a better candidate for the vertical elastic compression of a soft material. This

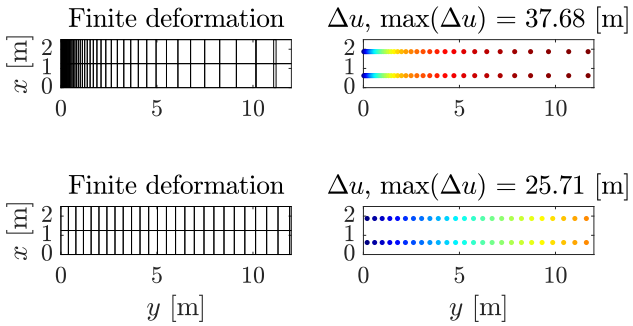


Figure 3.10. | Spatial extent of the material point's domain and associated displacement. One can see that both are significantly different.

is also confirmed when comparing i) the material point's spatial extent (i.e., its finite deformation) and, ii) the accumulated displacement Δu of material points during the elastic loading (see Fig. 3.10). For the finite strain formulation, both deformaiton and displacement are homogeneous whereas this is not true for the rate-dependent formulation. Overlaps of material point's spatial extents are observed, i.e., at the base of the column.

Two dimensional elasto-plastic slump

When considering an elasto-plastic Couloumb-type material (i.e., the pressure-sensitive Mohr-Coulomb model described in §2.3.6) with a linear strain-softening behaviour, the difference is less significant. The case of an ideal elasto-plastic

slump with $\rho = 2700$ kg$\cdot$ m^{-3}, $E = 1 \cdot 10^6$ Pa, $\nu = 0.3$, a friction angle $\phi = 20°$ and an initial peak cohesion $c = 20 \cdot 10^3$ Pa is now presented. An elastic

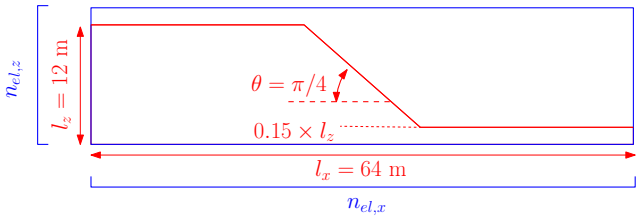


Figure 3.11. | Geometry for the elasto-plastic slump for a two dimensional configuration. The number of material points per initially filled element is $n_{pe} = 4$ and the number of element along the $x-$direction is $n_{el,x} = 80$ whereas the number of element in the $z-$direction in $n_{el,z} = 20$. This results in a total number of material point of $n_p = 2598$

loading relaxation phase is first performed and, the elasto-plastic behaviour is resolved once this relaxation stage has completed. For the properties previously listed, this corresponds to a time of $t \approx 8$ s. The domain update is based on the determinant of the deformation gradient. The geometry for the elasto-plastic slump is shown in Fig. 3.11 and, roller boundary conditions are applied on the boundaries of the computational domain.

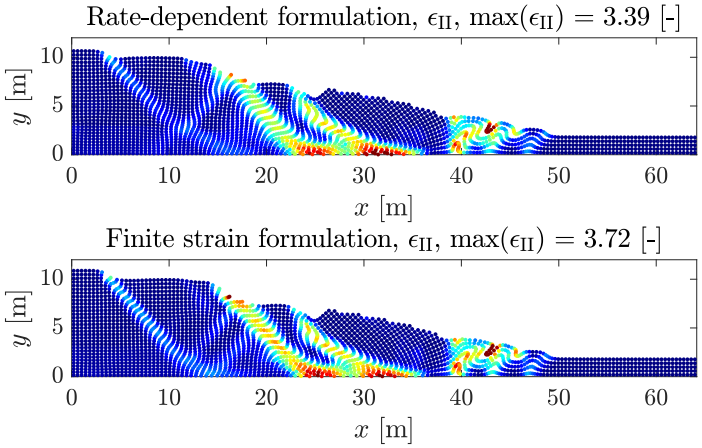


Figure 3.12. | Finite deformation and second invariant of the plastic strain ϵ_{II} for the rate-dependent and finite strain formulations. The color range is $[0; 3]$ for both cases, for the sake of comparison.

One can observe in Fig. 3.12, that i) the elasto-plastic behaviour is similar for both formulations, ii) strain localization (e.g., progressive development of shear bands, see the second invariant of the accumulated plastic strain ϵ_{II} in Fig. 3.12) initiates in very similar locations (e.g., bulging, folding and thrusting at the toe of the slope or the internal major and secondary shear bands) and, iii) a retrogressive propagation of localization is observed for both formulations. The magnitude of ϵ_{II} is also very similar for both formulations.

The overall elasto-plastic behaviour of the slump is homogeneous, regardless of the deformation framework chosen, and, no major differences are identified.

Synthesis

These two examples demonstrates that, accordingly to material parameters and expected magnitude of elastic deformation, the rate-dependent formulation is as valid as the finite strain formulation. Since this work is focused on elasto-plastic deformation in rocks and soils (i.e., small elastic deformation prior to the onset of plastic yielding), the rate-dependent formulation is a valid candidate.

Because of the explicit updated-Lagrangian formulation, Δt is required to be small between the reference configuration Ω_t and the current configuration $\Omega_{t+\Delta t}$, i.e., small displacement gradient. This satisfies $|\partial u_i/\partial X_J| \ll 1$ in the framework of the infinitesimal strain theory. It would have been different in a implicit formulation, which allows larger time steps.

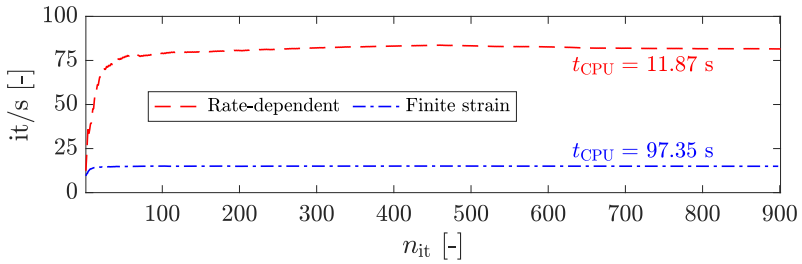


Figure 3.13. | Number of averaged iteration per second for both formulations implemented under MATLAB running within a macOS Catalina environment on an Quad-Core Intel i5 CPU @ 2 GHz. The finite strain formulation is roughly ten times slower compared to the rate-dependent formulation on this computing system.

In addition, the successive calculation of the left Cauchy-Green deformation tensor and the logarithmic strain tensor necessitate additional computational expenses, because of the required spectral decomposition (see Fig. 3.13). From a computational point of view, it is far more trivial to calculate the incremental strain tensor with the incremental deformation tensor than calculating the necessary eigenvalues and eigenvectors, which are required for the spectral decomposition in the finite strain framework.

3.2.8. Closed-form or implicit stress integration ?

Following the comparison between the rate-dependent and the finite strain formulation, one may also question the potential differences between an implicit or a closed-form solution of the stress integration during plastic loading. One can see that the differences (Fig. 3.14) are not significant between these two approaches, even tough some minor discrepancies are existing between the

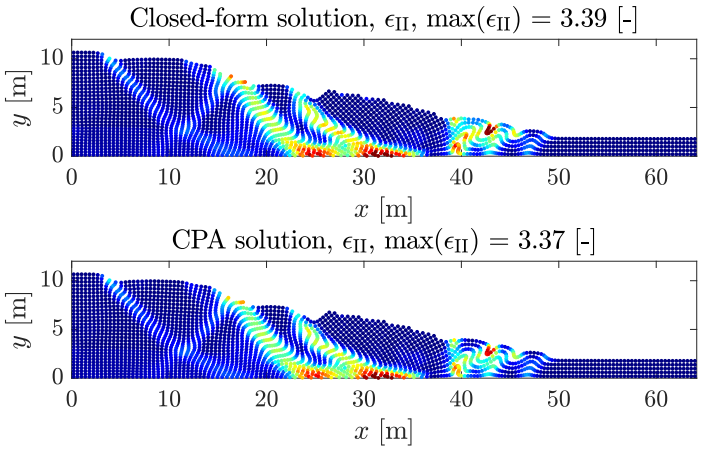


Figure 3.14. | Second invariant of the plastic strain ϵ_{II} considering either a closed-form or an implicit stress integration. The color range is $[0; 3]$ for both cases, for the sake of comparison.

two implementations. However, the computational runtime needed to solve for an implicit stress integration is more important than the closed-form solution (see Fig. 3.15), which i) does not necessitate to iterate upon convergence and, ii) it can be more easily vectorized than CPA or the implicit backward Euler method.

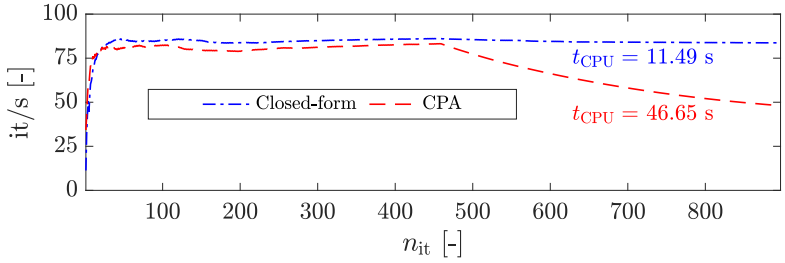


Figure 3.15. | Number of averaged iteration per second for both formulations implemented under MATLAB running within a macOS Catalina environment on an Quad-Core Intel i5 CPU @ 2 GHz. The CPA is roughly 5 times slower compared to the closed-form solution of Mohr-Coulomb model on this computing system.

3.2.9. Volumetric locking

Volumetric locking occurs when modelling near-incompressible elastic materials (or isochoric elasto-plastic deformation). It is caused by excessive constraints on the element's deformation by the integration points and was first reported by Nagtegaal et al., 1974. The element will lock, resulting in an over-stiff behaviour. It is primarily linked to the basis of the elements, i.e., bi-linear quadrilaterals or tri-linear hexahedrals. Such low-order elements are

widely used in MPM and, volumetric locking is likely to occur (Coombs et al., 2018).

When low-order elements are used in the GIMP formulation, spurious oscillations of the stress field arise (Coombs et al., 2018; González Acosta et al., 2019; González Acosta et al., 2021). The typical check-board pattern of the pressure field is showed in Fig. 3.16 for a near-incompressible Coulomb-type elasto-plastic material, for which the properties and the geometry are defined in §3.2.7 except the Poisson's coefficient is $\nu = 0.49$. Volumetric locking for ischoric plastic deformation is obvious and, the pressure field becomes highly oscillatory. This is of crucial importance, especially when pressure-sensitive plastic flow laws are chosen. A smooth pressure field is needed to properly resolve shear-banding.

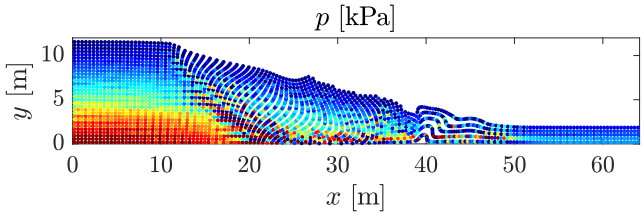


Figure 3.16. | Check-board pattern of the pressure field p after plastic loading at $t = 15$ s. Shear banding is pressure-sensitive and, it necessitates a smooth evaluation of the field.

Mast et al., 2012; Zhang et al., 2017 proposed different techniques to mitigate volumetric locking in sMPM. In particular, Bandara et al., 2015 implemented the linear $\bar{B}$ approach for coupled hydro-mechanical problems, Coombs et al., 2018 implemented an $\bar{F}$ approach (see Souza Neto et al. 2011) in both sMPM and GIMPM, whereas Bisht et al., 2021 recently proposed a non-linear extension of $\bar{B}$ approach for GIMPM. Cuomo et al., 2019; Lei et al., 2020 proposed an element-based averaging approach, following Mast et al., 2012. Selected material point properties are reconstructed based on an average value calculated over the element at the end of a time step. We average only the volumetric part of the stress tensor, i.e., the pressure $p = -\frac{1}{3}\sigma_{kk}$ while its deviatoric part $\tau_{ij} = \sigma_{ij} - p\delta_{ij}$ remains unchanged. This results in the following

$$p_e = \frac{\sum_{p\in e} v_p p_p}{\sum_{p\in e} v_p}, \tag{3.2.29}$$

where v_p is the material point's volume. This gives a constant distribution of the pressure field over an element because of its zero-order reconstruction (Lei et al., 2020). The Cauchy stress tensor $\sigma_{ij,p}$ of a material point p in an element e is corrected as

$$\sigma_{ij,p} = \tau_{ij,p} + \delta_{ij}(p_e)_p, \tag{3.2.30}$$

where δ_{ij} is the Kronecker Delta tensor and $(p_e)_p$ is the averaged pressure within an element e assigned to a material point p.

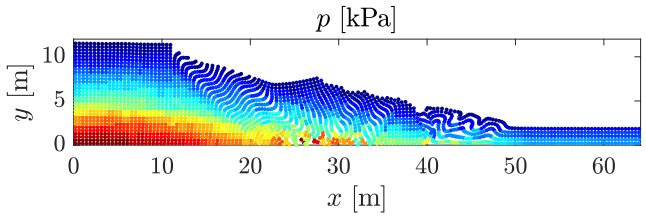


Figure 3.17. | Smooth pressure field is obtained with the proposed technique when compared to Fig. 3.16.

A smooth pressure field is resolved when considering Eqs. 3.2.29 & 3.2.30. The typical check-board pattern is greatly reduced and, the resulting pressure field is smoother when compared to Fig. 3.16.

3.2.10. High-resolution shear banding on a recent GPU architecture

As a synthesis of this chapter, a last numerical experiment is presented. Shear banding is an important phenomenon in both soils and rocks and, its influence over the material's behaviour is of crucial importance. An application of an explicit implementation of GIMPM under a GPU architecture provides useful insights on strain localization.

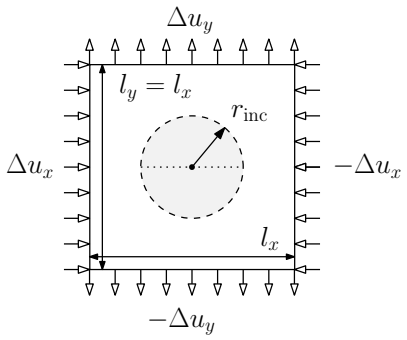


Figure 3.18. | Geometrical setting for the weak circular inclusion elasto-plastic problem during pure shear deformation.

A typical benchmark for shear banding is the case of an elasto-plastic domain of dimension $l_x \times l_y$ (i.e., 2000×2000 m), subjected to a pure shear deformation, see Fig. 3.18, within a zero-gravity field. Similar studies can be found in Kaus, 2010; Duretz et al., 2018. A circular weak inclusion of radius $r = l_x/10$ is introduced within a stronger surrounding matrix, to trigger strain localization with a discontinuous shear modulus G. Here, the modulus of the surrounding matrix is $G_{\text{mat}} = 10^{10}$ Pa, whereas within the weaker inclusion, it is $G_{\text{inc}} = 2.5 \cdot 10^9$ Pa. The other mechanical properties are spatially homogeneous: the bulk modulus is $K = 2G_{\text{mat}}$ Pa, the cohesion is $c = 3 \cdot 10^7$ Pa and the friction

angle is $\phi = 30°$. The elasto-plastic behaviour is modelled using the pressure-sensitive Drucker-Prager model presented in §2.3.7.

The pressure is progressively building up, as velocities $\dot{u}_{x,y}$ are applied on the boundaries, where $|\dot{u}_x| = |\dot{u}_y| = 10^{-3}$ m·s^{-1}. Shear localization starts close to the weak inclusion and propagates outward from the centre of the inclusion. Shear banding progressively develops and the pressure within drops

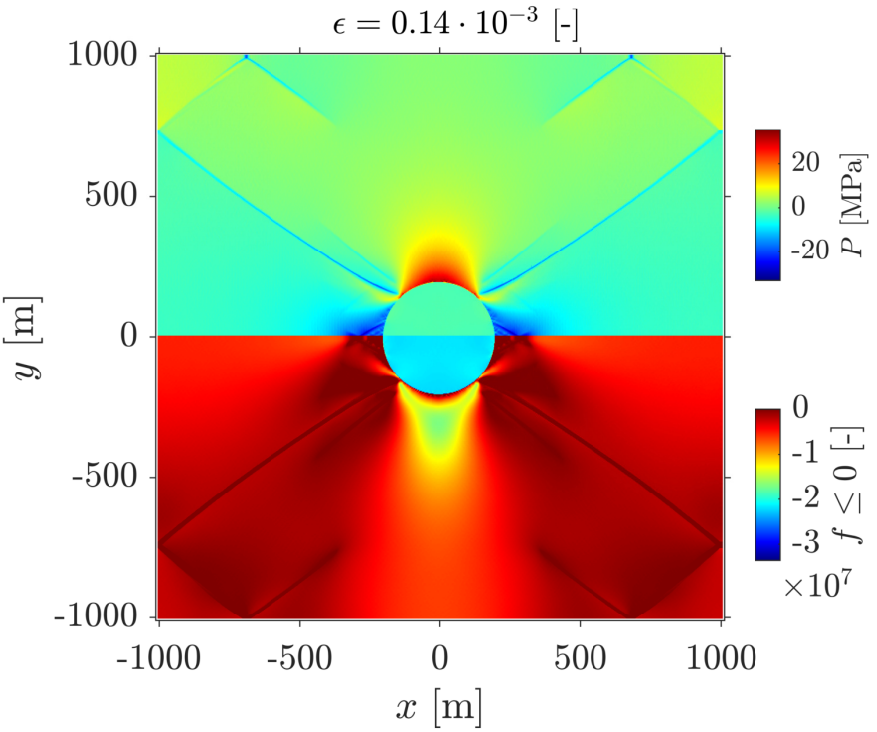


Figure 3.19. | Plastic strain localization, after a strain $\epsilon = 1.4 \cdot 10^{-4}$, for the pressure field P and the yield function f, for $n_p \approx 1.6 \cdot 10^6$ and $n_{el} = 414736$.

in comparison to the local pressure field, as expected. As soon as the shear bands reach the boundaries of the computational domain, they immediately reflect onto it because of the free-slip condition imposed at boundaries.

4. Cratering within granular matter

Cratering response during droplet impacts on granular beds

Emmanuel Wyser, Dario Carrea, Michel Jaboyedoff & Shiva P. Pudasaini

Published in *The European Physical Journal E*, **42** (2019), p. 111-122

Abstract This experimental work focuses on the cratering response of granular layers induced by liquid droplet impacts. A droplet impact results in severe granular layer deformation, crater formation and deposits in the vicinity of the impact center. High-precision three-dimensional imaging of the granular layer surface revealed important characteristics of liquid impacts on granular matter, such as singular asymmetric deformations of the layer. Our analysis also demonstrated that the impact energy and the granular packing, and its inherent compressibility, are not the unique parameters controlling the bed response, for which granular fraction heterogeneities may induce strong variations. Such heterogeneous conditions primarily influence the magnitude but not the dynamics of liquid impacts on granular layers. Finally, a general equation can be used to relate the enery released during cratering to both the impact energy and the compressibility of the granular matter. However, our results do not support any transition triggered by the compaction-dilation regime. Hence, higly detailed numerical simulations could provide considerable insights regarding the remaining questions related to heterogeneous packing conditions and its influence over the bulk compressibility and the compaction-dilation phase transition.

4.1. Introduction

Granular matter, which exhibits a hybrid behavior between solid and fluid (Goldman et al., 2008; Pudasaini et al., 2007), has been extensively studied over the past decades. Specifically, solid-to-granular impacts have been investigated to understand several aspects, such as the role of impact energy or the chain force in the penetration depth (Nordstrom et al., 2014; Clark et al., 2014; Deboeuf et al., 2009), the dynamics of intruder deceleration (Marston et al., 2015; Clark et al., 2013; Clark et al., 2012; Goldman et al., 2008) and the confinement influence (Seguin et al., 2008). The laser scanning technique is used to digitize the crater surface (Umbanhowar et al., 2010; Vet et al., 2007). (Jong et al., 2017; Walsh et al., 2003; Ambroso et al., 2005) experimentally studied impact craters and acknowledged scaling laws for various relevant crater dimensions (*e.g.,* the crater diameter D_c or the crater depth d_c) with the impact energy E_μ, such as $D_c \propto E_\mu^{0.25}$. For solid-to-granular impacts, Deboeuf et al., 2009 proposed a power-law scaling of the crater volume, $v(E_\mu) = \alpha E_\mu^\beta$ with $\alpha = 3.60 \cdot 10^4$ and $\beta = 0.67$, thereby confirming a previously observed relation between the impact energy and the volume of the crater (Vet et al., 2007).

However, little is known about liquid-to-granular impacts, and this field of study remains to be investigated. Most of the recent studies are experimental studies (Zhao et al., 2015a; Nefzaoui et al., 2012a; Nefzaoui et al., 2012b; Katsuragi, 2010; Katsuragi, 2011) and focus on specific topics, such as the spatial distribution of the splashed particles (Long et al., 2014; Ahn et al., 2013; Brodowski, 2013; Furbish et al., 2009; Furbish et al., 2007); the influence of impact conditions, such as the granular bed packing ϕ_0, which largely determines the mechanical response of sheared material (Nedderman, 1992) or the impacting droplet Bond number (Marston et al., 2010); the wetting regimes (Hapgood et al., 2002); and the crater morphology (Ghadiri, 2004; Ghadiri et al., 1979) and scaling laws for its diameter (Zhao et al., 2015a).

Using recent PIV techniques and high-speed cameras, Long et al., 2014 reported two types of radial distributions of the splashed particles for a constant initial packing fraction ϕ_0: the water-encapsulated particles ejected from the periphery of the crater (high ejection velocities and low ejection angles), and the dry particles ejected from the periphery of the crater (slower ejection velocities and higher ejection angles). Since the results provided by Long et al., 2014 and by the pioneering work of Furbish et al., 2007 were promising, one major investigation is still needed, *i.e.,* considering the volumetric distribution of mass rather than particle distribution under various initial packings.

The packing fraction ϕ_0 governs the mechanical response and flow structure in slow granular shearing: the material dilates locally due to localized shear bands at large ϕ_0, whereas it compacts and flows globally at low ϕ_0 (Nedderman, 1992). Recent works focused on a transition signature between the compaction and dilation regime within slightly polydisperse granular matter (Royer et al., 2011; Royer et al., 2007) and discussed its importance on the im-

pact response over a solid intruder for fine-grained layers. Umbanhowar et al., 2010 demonstrated that a critical packing state $\phi_{cs} = 0.591 \pm 0.005$ governs this transition phase and is a principle determinant of both the intruder and the crater dynamics. This critical threshold was also observed by (Zhao et al., 2015b), where $\phi_{cs} = 0.58$ was found.

In addition, Marston et al., 2012 demonstrated different impact dynamics between spherical glass beads and natural sands. The use of natural grains such as sand is the most interesting case. Furthermore, the empirical estimation of the excavated volume provided by Deboeuf et al., 2009 for the solid impact has to be compared with the excavated volume for the liquid droplet impact on granular beds. It is of interest to set in perspective both liquid and solid impacts and compare the results. Hence, we suspect some common features between solid and liquid impacts on a fine granular media.

Our work focuses on the crater response of granular layers induced by liquid droplet impacts, with a potential application for raindrop-induced erosion (Cooper et al., 2012; Dunne et al., 2010; Foot et al., 2005). Following the studies of Marston et al., 2015; Marston et al., 2010 for steel sphere impacts, we investigate the influence of the bed packing fraction on the spatial distribution of induced volumes. We focus on the cratering response through quantifying both the excavated and deposited volumes for a constant impact energy and various bed packings.

4.2. Methods

4.2.1. Overview of the experiment

The experiments performed in this study are based on previous works (Long et al., 2014; Carrea et al., 2011; Furbish et al., 2009; Furbish et al., 2007). The sample preparation and experimental setup are shown in Fig. 4.1. First, an accurate 3D imaging of the granular bed sample is conducted using a non-contact 3D digitizer VIVID 9i equipped with a TELE 25 mm lens with a focal distance of $f = 25$ mm and a vertical precision (Z, σ) of $Z = 0.008 \pm 0.024$ mm within a typical range distance of 0.6 to 1 m (KONICA, 2004). A device above the granular bed at a predefined height ($h_1 = 6.25$ cm, $h_2 = 12.5$ cm, $h_3 = 25$ cm, $h_4 = 50$ cm , $h_5 = 100$ cm, $h_6 = 200$ and $h_7 = 290$ cm) releases a droplet of water with a 2.34 mm radius, which is a typical value for higher natural drops in high-intensity rainfall (Zhao et al., 2015a; Villermaux et al., 2009; Dufour, 1962; Marshall et al., 1948), and the droplet impacts fine natural silty samples with a diameter of $d \leq 200$ μm within a range of various granular bed packings ϕ_0. The upper limit of 200 μm was chosen to ensure a combined effect of both frictional and microscopic forces (friction, electrostatic and van der Waals forces), which should not be neglected for powders (Andreotti et al., 2013). The impact dynamics and the splashing of particles are recorded using a high-speed camera, SpeedCam MiniVIS ECO-2 (500 fps), equipped with an EX SIGMA 24-70 mm F 2.8 DG MACRO lens.

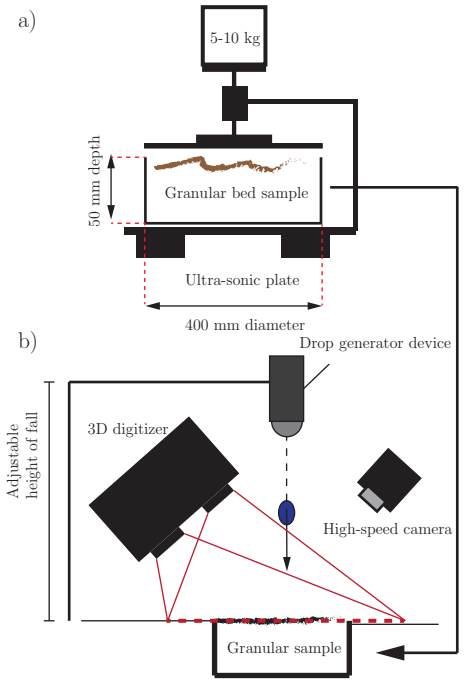


Figure 4.1. | (a) Sample preparation setup with a pressing plate, a shaking rotational motor and the cylindrical granular sample. (b) Experimental setup with the drop generator device with adjustable vertical distance from the sample, an impact surface with an adjustable slope, a 3D scanner and a high-speed camera, inspired by the works of Long et al., 2014; Furbish et al., 2007.

4.2.2. Preparing granular bed samples

Natural sand, which is from the Black Marls of Draix (France), was sieved with mesh sizes of 100 μm and 63 μm. Particle size analyses were performed using a Malvern Mastersizer 3000 to estimate the features of the grain size distribution, *e.g.,* the median diameter d_{50}, the volume density (the proportion of a grain size class over the total granular volume processed) and the cumulative volume (Fig. 4.2(a) and (b)).

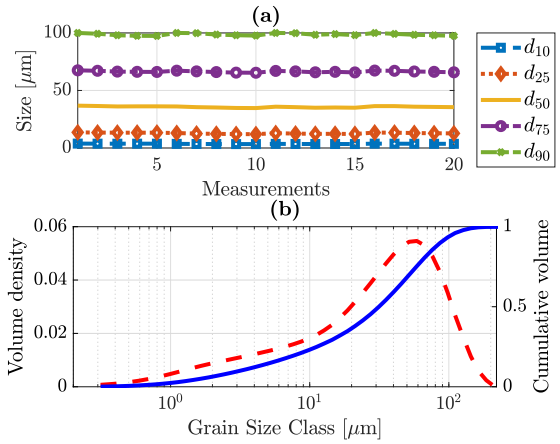


Figure 4.2. | a) Twenty measurements were performed to ensure a consistent quantification of the grain size distribution. Variations in the characteristic diameter are small in magnitude. b) Volume density (red dashed line) and cumulative volume (continuous blue line) of the granular samples used.

A granular sample preparation procedure was used to ensure a reproducible packing fraction using a specific device, as shown in Fig. 4.1 (a). A cylindrical container with dimensions of 50 $\times$ 400 mm and volume of v_c was filled with the sieved natural sand. The excess sand was scraped to ensure a flat surface, and the granular sample was pressed and gently vibrated using the compacting device described in Fig. 4.1 (a). Various packing fractions ranging from $0.49 < \phi_0 < 0.61$ were achieved.

A simple method was used to resolve the average grain density ρ_g. Considering a small initial volume v_i of water in a partially filled container, a granular mass m_g is poured into the water, which increases the water volume, resulting in a final volume v_f. The granular volume v_g is given by $v_g = v_f - v_i$. The average density is given by $\rho_g = m_g/v_g$, thus leading to $\rho_g = 2.54 \pm 0.26$ gr·cm^{-3}.

4.2.3. Modeling impact energy

Quantifying the kinetic energy of the drop at the impact is achieved using a one-dimensional numerical model (Middleton et al., 1994), assuming that horizontal forces and air turbulences can be neglected in a controlled environment. Although air resistance is generally neglected for solid impacts, it is

no longer negligible for a liquid droplet impact. Most studies (Birch et al., 2014; Deboeuf et al., 2009; Ambroso et al., 2005) use the gravitational potential energy $E = mgh$ for the impact energy, where m is the mass in kg of the impacting object, g is the gravitational constant in m·s^{-2}, and h is the height above the granular sample in m. We introduce i) the use of an efficient quantification of the impact kinetic energy by means of a numerical model of the average velocity with the corresponding standard deviation $u_{\mu\pm\sigma}$ of the droplet, and ii) our experimental validation of this numerical estimation of the kinetic energy by means of empirical measurements of the impact velocity $\hat{u}_h$ of the droplet.

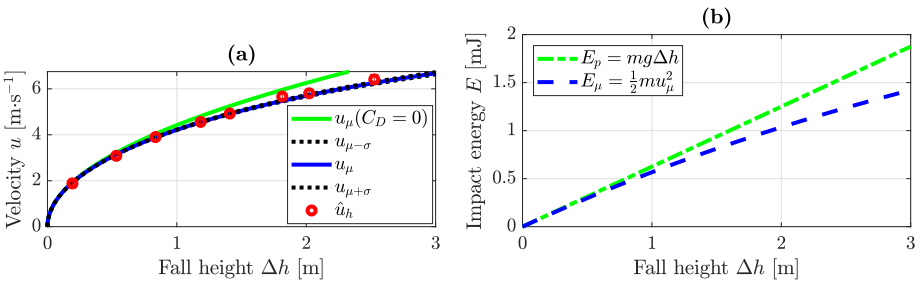


Figure 4.3. | a) The numerical velocity $u_{\mu\pm\sigma}$ of a 2.34 ± 0.14 mm radius droplet and the average values of empirical velocities $\hat{u}_h$ measured using the high-speed camera. $\hat{u}_h$ is in good agreement with $u_{\mu\pm\sigma}$. b) Increasing difference between the kinetic energy E_μ and the potential energy E_p with respect to Δh, assuming that $E_p \approx E_\mu$ remains valid for the height of the fall $\Delta h \leq 0.5$ m. The difference between the dotted-dashed green line and the dashed blue line indicates the need to consider drag force when estimating the impact energies of liquid droplets.

The numerical solution of the one-dimensional equation of translational motion is approximated using a forward Euler integration scheme considering gravity and the drag force (Middleton et al., 1994; Andreotti et al., 2013). The general formulation of the equation of motion is $F - D = ma$, where $F = mg$ is the body force with $g = 9.81$, $D = \lambda u^2 = -0.12\pi C_D \rho_f d^2 u^2$ is the drag force with $C_D = 0.5$, the fluid density $\rho_f = 1.20$ in kg·m^{-3}, the droplet dimension d in m and the velocity u in m·s^{-1} (Andreotti et al., 2013). A constant drag coefficient of 0.5 (Anderson et al., 2010; Furbish et al., 2007) is used, which is commonly admitted for a sphere, and a sufficiently small time step $dt = 10^{-5}$ is used to ensure numerical stability.

The instrumental error of the droplet radius ($r_{\mu\pm\sigma} = 2.34 \pm 0.14$ mm) is also included within the numerical model. Several laboratory measurements were performed on the mass of a droplet generated by the device. Assuming a spherical shape and the droplet volume $v_d = 1.33\pi r_\mu^3$ with water density $\rho_w = 998.20$ kg·m^{-3} at temperature $T_w = 293.15$ K, the quantification of the average droplet radius and its standard deviation using the relation $m = \rho_w v_d$ is straightforward.

To ensure the efficiency of the numerical model, empirical measurements

of the velocity of the droplet were achieved in the laboratory using the high-speed camera and a small time step between images. Differences in the vertical positions of droplets were evaluated to estimate an average velocity $\hat{u}$ for a 2 ms time interval. However, we assume no horizontal motion of the droplet (no lateral effect due to air drag). Experiments were performed for different fall heights. The empirical impact velocity $\hat{u}$ was estimated and compared to the numerical model. As shown in Fig. 4.3 (a), the experimental data are in good agreement with our 1D numerical model (solid blue and green curves).

We also compared the mean numerical terminal velocity $u_\mu^T = \text{cst.}$ when $\partial_t u_\mu := 0$, which can be considered a steady-state velocity, to the analytical solution $W^T = ((4\rho_w g C_D)/(3\rho_f 2r_\mu))^{0.5}$ provided in Furbish et al., 2007. Both W^T and u_μ^T are identical since they are constrained by the accuracy limit (three significant digits) of material properties, *i.e.,* $W^T \rightarrow u_\mu^T = 10.070$ in $\text{m}\cdot\text{s}^{-1}$. This provides a consistent approximation of the terminal velocity of a water droplet in the stagnant air of a laboratory. However, this terminal velocity differs from the actual terminal velocity of natural raindrops. Under different atmospheric pressure conditions, Foote et al., 01 Apr. 1969 computed terminal velocities for a 2.34 mm radius droplet ranging from 9 up to 11.8 $\text{m}\cdot\text{s}^{-1}$. The difference can be explained since the drag coefficient that we use is set to be constant, whereas it is actually inversely proportional to the Reynolds number, $C_D \propto \text{Re}^{-1}$ (Edwards et al., 2001; Foote et al., 01 Apr. 1969).Indeed, C_D varies with the droplet size, *i.e.,* it increases when $r_\mu \geq 3$ mm, and a more consistent value for C_D would be $C_D \approx 0.56$ for $r_\mu \approx 2$ mm according to Mason, 1971.

The quantification of the kinetic energy of the impact $E_\mu = 0.5mu_\mu^2$ is then fully based on the numerical model that we used. We also calculated the gravitational potential energy $E_p = mgh$ at the impact to determine the major differences between impact energy quantification. Fig. 4.3 (b) clearly shows a significant difference between E_μ and E_p that increases steadily as Δh increases, mainly for $\Delta h \geq 0.5$ m.

4.2.4. Assessing excavated and deposited volumes

We perform a preimpact imaging of the granular sample. Then, a droplet impacts the granular bed, resulting in severe granular bed deformation, *i.e.,* crater formation and deposits in its vicinity. Postimpact imaging is performed at different locations around the granular bed to ensure that there are no shadows during the processing of the resulting 3D point clouds, as shown in Fig. 4.4. Two opposite locations ensure a sufficient overlap between the acquisitions. An accurate alignment of the 3D scans is also performed during the data processing using the commercial software PolyWorks.

To enhance the point cloud density and the imaging accuracy, an alignment of several postimpact 3D scans is achieved using the ImAlign module of the software PolyWorks (Carrea et al., 2012; Carrea et al., 2011). Subsequently, the distance between the preimpact point cloud (reference) and the postimpact

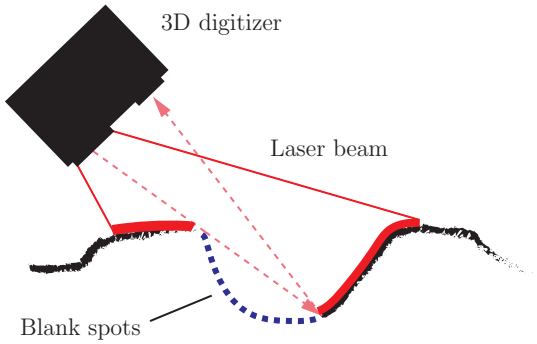


Figure 4.4. | To avoid shadows, several 3D acquisitions have been performed around the geometry of interest.

aligned point cloud is computed using several steps:

1. The preimpact point cloud is transformed into a meshed surface denoted as $\mathbf{S} = (x, y, z)$. A best-fitting plane Π is fitted to this reference surface to calculate the corresponding averaged normal unit vector $\mathbf{n}_\Pi$, which will be further used to calculate the orthogonal distance.

2. The normal Euclidean distance $d_2(\bullet, \bullet)$ from the meshed surface to a single point $P(x, y, z)$ denoted as $d_2(\mathbf{D}, \mathbf{S}) = \|\mathbf{D} - \mathbf{S}\|_2$ is calculated using $\mathbf{n}_\Pi$, and the aligned point cloud after impact with a vector position denoted as $\mathbf{D} = (x, y, z)^T$ to project the distance in the averaged normal direction.

3. Sets of negative (excavation) and positive (deposition) normal distances are obtained for further postprocessing tasks and transformed into a relative position, such as $(x, y, z)^T \mapsto (x, y, d_2(\mathbf{D}, \mathbf{S}))^T$.

Using a MATLAB routine, a method of point selection based on a minimum positive or negative threshold of normal change $\xi^\pm$ allowed a semiautomatic recognition of (x, y, z) points corresponding to either an excavation or deposition area. These thresholds were defined by the operator during the processing of the data using the ImInspect module of the PolyWorks software. The average positive $\langle \xi^+ \rangle = 0.06 \pm 0.02$ mm and negative $\langle \xi^- \rangle = -0.08 \pm 0.03$ mm thresholds define the minimum detection metric in the normal direction. However, we argue for symmetry, *i.e.,* $\xi^\pm = \pm 0.10$ mm.

Let the subset $\mathbf{D}_+ \subset \mathbf{D}$ be the set of all positive normal distances, *i.e.,* $\mathbf{D}_+ = \{d_2(\mathbf{D}, \mathbf{S}) \mid d_2(\mathbf{D}, \mathbf{S}) \in \mathbf{D}_+ : d_2(\mathbf{D}, \mathbf{S}) \geq \xi\}$, where ξ is the defined positive threshold. Additionally, assume a regular grid spacing in the xy plane with a sufficiently small spacing distance Δ. We can define the vertical position $z(x, y) \equiv d_2(\mathbf{D}, \mathbf{S})$. The approximated volume v_D of the subset $\mathbf{D}_+$ is then

$$v_\mathrm{D} = \lim_{\Delta \to 0} \sum_{z \in \mathbf{D}_+} z(x, y) \Delta x \Delta y. \tag{4.2.1}$$

The approximated volume is closely related to the grid spacing of the point cloud. Hence, a linear interpolation was used to obtain a high-density grid with a typical spacing $\Delta x = \Delta y = 0.02$ mm to satisfy the limit condition in (4.2.1).

The average deposited and excavated volumes were calculated for each impact energy E_μ considering loose packing, *i.e.*, $\phi_0 \leq 0.5$, respectively defined as

$$\langle v_{\mathrm{D,C}} \rangle = \frac{1}{n} \sum_{\phi_0 \leq 0.5} v_{\mathrm{D,C}}(\phi_0, E_\mu) \tag{4.2.2}$$

with n the number of impact for a given energy with a packing lower than 0.5.

4.3. Results

4.3.1. Local granular bed uplifts

We conducted a set of experiments using the four highest impact energies to investigate the general features (Fig. 4.5) of liquid-to-granular impacts considering various initial packing conditions. We did not investigate in detail the experiments for which the packing was kept constant because they mostly result in identical morphologies with no clear differences. Fig. 4.5 intuitively

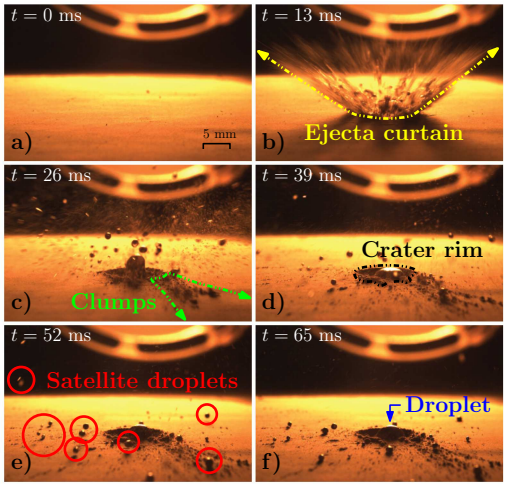


Figure 4.5. | Representative sequence for an impact on an intermediate initial packing, *i.e.*, $\phi_0 \approx 0.55$ for an impact energy $E_\mu = 0.57$ mJ corresponding to a fall height of $h_5 = 100$ cm.

represents major sub-processes occurring during a liquid-to-granular impact: immediately after impact, an ejecta curtain grows and satellite droplets may further develop if the drop deformation overcomes the surface tension, resulting in a final crater rim and granular clumps in the vicinity of the crater. Usually, the remaining drop progressively vanishes while infiltrating within the granular matter.

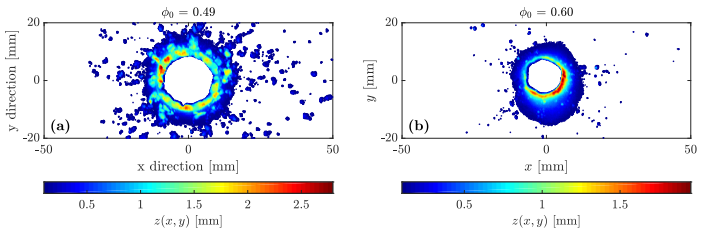


Figure 4.6. | Mapping of positive elevations for loose ($\phi_0 = 0.49$) and tight ($\phi_0 = 0.59$) packings of processed data of Fig. 4.7, which shows clear variations in positive differences regarding the granular bed packing. The positive threshold was set to $\xi^+ = 0.10$ mm. Local granular uplifts are noticeable. However, this feature is more evident for a tight packing, as in panel b).

Three types of positive elevations are observed, which result in three postimpact morphologies: a) the crater rim, b) the ejecta deposits consisting of granular clumps of various size, unitary particles and satellite droplets, and c) a partially symmetric uplift of the granular layer in the vicinity of the impact location (Fig. 4.6 and 4.7).

The latter implies that the granular packing and the structural assemblies of grains are locally modified close to the impact crater by the deforming drop during the cratering process.

For tight packing, shearing processes result in an uplift of the granular bed, whereas for loose packings, the granular bed is merely compacted, which correspond to dilation and compaction, respectively (Zhao et al., 2015b; Métayer et al., 2011; Gravish et al., 2010; Schröter et al., 2007; Thompson et al., 1991). Regarding the mapping of positive elevations, we observe a smooth radial decrease in the positive difference, suggesting shearing-induced uplifts, particularly for tight packing (Fig. 4.6(b)). Such observations are hardly visible for loose packing because of overlying granular clumps deposited above granular uplift areas (Fig. 4.6).

It is not easy to determine deformation features, *e.g.,* uplifts, from materials deposited beyond the crater rim for loose packing. Hence, we investigated the proportion of volume induced by the granular layer deformation with the overall deposited volume of six experiments for the two highest impact energies in which the packing was considered to be tight. Certainly, large deformations and associated strains of the surface layer are substantially decreased within tightly packed layers. Areas in which deformations are identified (Fig. 4.6(b), where granular uplifts are clearly observed) are further selected to acknowledge the ratio between pure deposition and deformation. Hence, the volumes of deposited materials were calculated and normalized by the overall deposited volume. Positive deformations (*e.g.,* uplifts) act for a proportion p of 94% to 97% of the overall deposited volume, which demonstrates a high energy dissipation due to numerous elasto-plastic collisional processes during deformation of the layer, *i.e.,* compaction and dilation mechanisms.

Following the methodology used by Jong et al., 2017, we infer an average

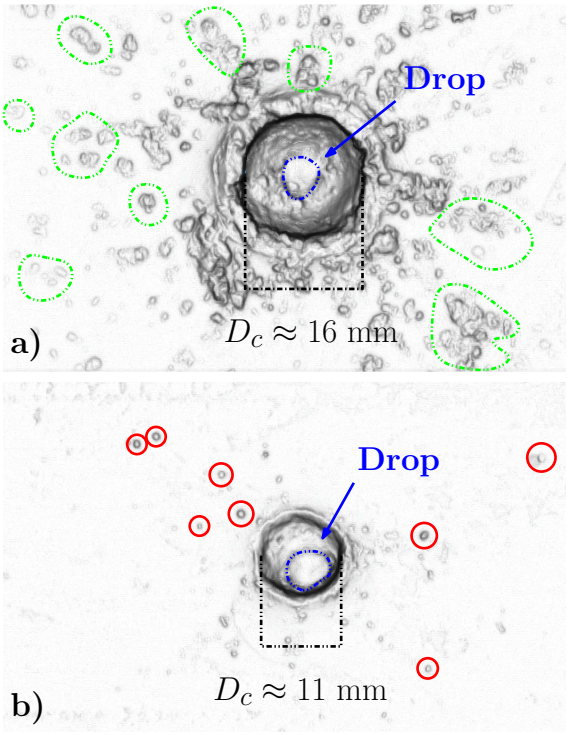


Figure 4.7. | Shaded-relief mesh corresponding to Fig. 4.6 a) and b), respectively for loose and tight initial packings. The average impact energy is $E_\mu = 1.39$ mJ, corresponding to the fall height of $h_7 = 290$ cm. We identify clearly the final drop in the crater (blue dash dotted lines), various granular clumps (green dash dotted lines) and satellite droplets (red continuous lines). Large clumps appear close to the crater rim, whereas smaller clumps are located away from the crater.

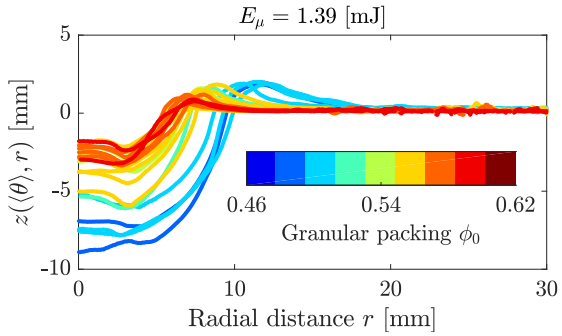


Figure 4.8. | Relaxed cratering stage: Various averaged radial profiles are shown for a constant impact energy E_μ and for various initial packings. We observe that i) the crater slope becomes gentler as ϕ_0 increases, and ii) the maximum height of the crater rim appears to be related to the initial packing, mainly the loose packing.

polar profile $z(\langle\theta\rangle, r)$. We observe an unexpected common feature of these liquid impacts: the crater rim is higher for loose packings than for tight ones, as shown in Fig. 4.8, whereas Umbanhowar et al., 2010 observed the strict opposite for granular impacts. Moreover, the cratering response regarding the packing is well defined for the packing extrema (loose or tight packings), but it remains unclear for the intermediate packing state.

4.3.2. Radial distribution of deposited volumes

Considering previous works (Long et al., 2014; Furbish et al., 2007), we further investigate the radial distance of deposited material. First, we use deposited volumes as $v_{\mathrm{D}}(\theta, r)$, where θ and r are angular and radial coordinates, respectively and then we average it with respect to θ, *i.e.,* $v_{\mathrm{D}}(\langle\theta\rangle, r)$ for the four highest impact energies. Finally, we investigate the corresponding normalized cumulative polar distribution $\Theta(r)$. Fig. 4.9 shows that i) a positively skewed distribution appears to well define the radial distribution of deposited mass, and ii) the greater the impact energy is, the wider is the distribution.

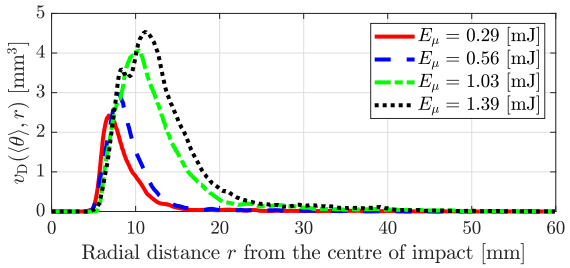


Figure 4.9. | Radial distributions of average deposited volume corresponding to the four highest impact energies. Influence of the energy during the impact over the spatial distribution of the mass is evident.

We observe a zero deposit range for low radial distance, *i.e.,* $r < 6$ mm. It is an artefact from the averaging procedure since naturally, no deposited volume can be found within the crater. Such range reflects purely the radius of the impact crater and provides some information about its increase with the impact energy. Similarly, the maximal value of deposited volume corresponds to the radial distance for which the crater rim is the highest. An interesting observation is that such maximal value increases more rapidly than the radius of the crater does.

When examining the bell shape of the skewed distribution, it is evident that for increasing impact energy, the overall spread increases. This result suggests dependences of the spatial distribution over the impact energy, *i.e.,* the higher the energy is, the farther is the average radial distance.

Interestingly, this result also reveals an increasing contribution of the crater rim over the overall excavated volume. Indeed, the increasing skew of the bell part of the distribution can be explained by a higher contribution of excavated volumes in the vicinity of the impact, clearly induced by an interplay between

granular clump deposits and local granular bed uplifts.

As demonstrated in Long et al., 2014, the particle distribution of a liquid impact is defined by a Burr XII distribution. However, we visually observe that such a distribution also performs well when considering a radial volume distribution. We conduct a two-side Kolmogorov-Smirnov goodness-of-fit test. Overall, low p-values are found, except for a few experiments in which the p-value is close to 0.5. These values are not satisfactory when considering the average radial distributions for the four impact energies (Fig. 4.9).

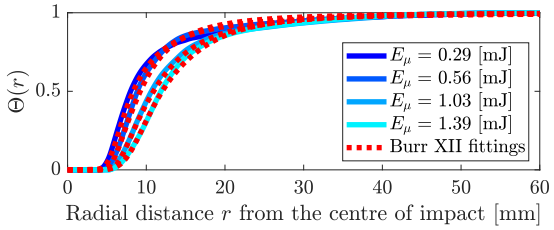


Figure 4.10. | Burr XII fitting of the four average radial distributions. It is in agreement with close radial distance from the impact center and starts to diverge for farther distances.

Hence, Burr XII fitting performs poorly in modeling the heavy-tailed size of the radial volume distribution, as clearly suggested in Fig. 4.9, whereas it appears to accurately fit the smaller radial distances. These statistical results indicate that a Burr XII distribution is not a suitable candidate for assessing the radial mass distribution for a liquid droplet impact on a granular bed.

4.3.3. Asymmetric and random patterns of liquid impacts

We observed few asymmetric patterns of the polar distribution of the deposited volume (Fig. 4.11 a)). It could be due to an interplay between heterogeneous packings and nonlinear rearrangement of the force chain network during the layer deformation and also immediately after the impact.

We further performed a quantitative analysis of these asymmetric patterns. Let the normalized polar distribution be $f(\theta, r) = (\int_{\delta\theta}\int_r v_\mathrm{D}(\theta, r)\mathrm{d}r\mathrm{d}\theta)/v_\mathrm{D}$ (continuous red line in Fig. 4.11(b)) with v_D being the overall deposited volume. Let us also consider a theoretically symmetric average distribution $\mu(f(\theta, r)) = 1/n$ (dashed blue line in Fig. 4.11(b)) with a selected number of increment $n = 25$ and $\delta\theta = 2\pi/n$. For notational convenience, now let $f \equiv f(\theta, r)$ and $\mu \equiv \mu(f(\theta, r))$. We can further define an average, strictly positive distance quantifying the relative distance between both distributions, *i.e.*, $\langle d(f, \mu)\rangle = (n^{-1}\sum_{i=1}^{n}(f - \mu)^2)^{0.5}$. μ would correspond to a perfectly symmetric distribution of deposited volume, *i.e.*, $\mu = 4$ % with $d\theta = 0.25$ rad. Thus, $\langle d(f, \mu)\rangle$ provides a divergent metric of the experimentally obtained polar distribution from a purely symmetric distribution, *i.e.*, $\langle d(f, \mu)\rangle = 0$. As soon as $\langle d(f, \mu)\rangle > 0$, the metric is positively diverging from 0, and then the polar distribution increasingly tends to asymmetric conditions.

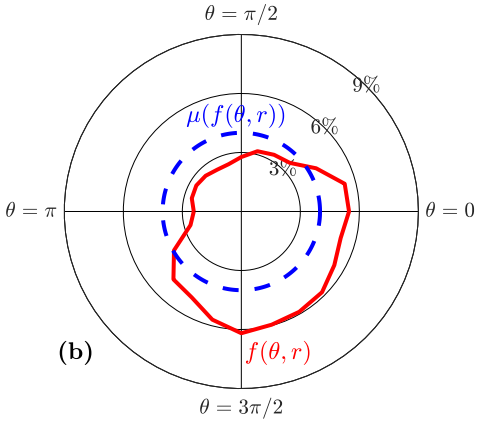


Figure 4.11. | Asymmetric patterns resulting from an impact on a tight packing $\phi_0 = 0.60$ for an impact energy $E_\mu = 1.39$ mJ, *i.e.,* it is exactly the same realization as in Fig. 4.6(b). The dashed blue line indicates a purely symmetric distribution $\mu(f(\theta,r))$.

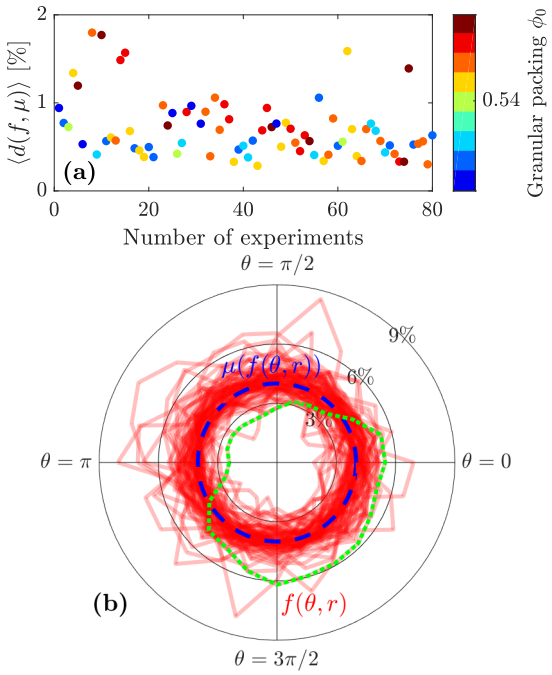


Figure 4.12. | a) Average distances $\langle d(f,\mu)\rangle$ for 80 experiments. Red and green squares are the selected distributions in b). As a result of the randomness, no significant trend can be found. b) Overlapping polar distribution of every experiment, regarding angular coordinates θ. The dotted green line denotes the asymmetric patterns of the experiment shown in Fig. 4.11b).

Figure 4.12 a) indicates random fluctuations (*e.g.,* scattering) of the average distance between distributions $\langle d(f,\mu)\rangle$ for four different impact energies. Moreover, the packing has no influence over $\langle d(f,\mu)\rangle$ since no significant trends can be found in Fig. 4.12 a). All experimental data are shown in Fig. 4.12 b). Overall, the results suggest that i) most distributions are symmetric and show inherent random fluctuations, but ii) few asymmetric distributions emerge with a maximum magnitude of 5 %. We explain this as the main result of the heterogeneities of the initial packing, i.e., the granular are highly polydisperse (Fig. 4.2b)) and it seems reasonable to assume strongly localized packing heterogeneities during the sample preparation. Moreover, we can suspect an influence of the drop shape at the contact time due to oscillations of the drop due to the pinch-off mechanism.

4.3.4. Average volumes as a function of impact energies

We further investigate the relation between the impact energy and volumes, considering three additional impact energies for which the packing was kept constant and loose, *i.e.,* $\phi_0 < 0.50$, to acknowledge the relation between the impact energy with overall volumes. Since we work with average volumes, we consider only volumes resulting from impacts on loose granular beds to ensure consistency between selected experiments. The coefficient of determination R^2 provides an estimate of the goodness-of-fit and is expressed as $R^2 = 1 - \sum_i (y_i - \hat{y}_i)^2 / \sum_i (y_i - \langle y\rangle)^2$, with an experimental value y_i estimated by its least-squares estimate $\hat{y}_i$ and the average value $\langle y\rangle$ of experimental values.

As mentioned earlier, we suspect, as in Zhang et al., 2015; Zhao et al., 2015a; Zhao et al., 2015b; Deboeuf et al., 2009, a relation between volumes and impact energies. Moreover, we consider the strength or plastic regime as the significant scaling, which governs the crater volume, based on the well established proportional relation $\langle v_C\rangle \simeq E_\mu / Y$, where Y is the yield stress during the deformation (Zhang et al., 2015). Assuming a affine form $\langle v_{C,D}\rangle = f_{C,D} E_\mu / Y$, we infer from a robust fitting that $Y = 990$ [Pa] and $f_C = 0.93$ or $f_D = 0.64$ with significantly high R^2, *i.e.,* $R^2 \geq 0.98$ in Fig. 4.13 a). The

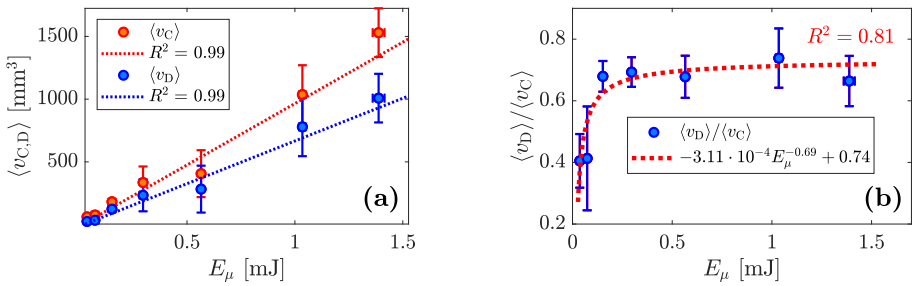


Figure 4.13. | a) Affine strength regime for different impact energy E_μ considering loose packings, *i.e.,* $\phi_0 \leq 0.5$. b) Power-law scaling between E_μ and $\langle v_D\rangle / \langle v_C\rangle$. The constant 0.74 certainly defines the compressibility limit.

coefficient f_C can be regarded as the proportion of energy transferred to the

cratering, *i.e.,* it account for 94 % of the initial impact energy. The rest of impact energy turns into the droplet deformation during impact, which was also demonstrated in Jong et al., 2017. However, the proportion of transferred energy is much lower when considering $\langle v_{\mathrm{D}} \rangle$, *i.e.,* $f_D = 0.64$. The value of Y, inferred from the robust fitting. Zhang et al., 2015 demonstrated a similar range when considering dry (liquid saturation $S = 0.0$ %) granular samples of similar size, thus confirming consistency between our experiments with others. Finally, we infer a constant volume ratio $\langle v_{\mathrm{D}} \rangle / \langle v_{\mathrm{C}} \rangle = 0.69$ from the scaling. This corresponds, to a certain extent, to the asymptotic value 0.74 in Fig. 4.13 b) where a power-law scaling was used. We also tested other fitting functions, *i.e.,* step functions and exponentials, but no more significant results were obtained. Therefore, we quantified the statistical significance to be consistent, *i.e.,* we performed a one-way ANOVA resulting in a p-value $p << 0.01$. The volume ratio reaches a plateau regardless of the increase of the impact energy.

4.3.5. The cratering energy release as a result of the energy dissipation

Since mass and volume of the crater are related to each other by the granular packing, we further address the crater mass and its corresponding energy, excluding uplifted materials, as being governed by both the impact energy and the initial packing of the layer.

As mentioned in Jong et al., 2017, the potential energy budget E_{C} of the crater (*i.e.,* neglecting deposited materials) is likely to depend on ϕ_0. In addition, we assume that the energy ratio E_{C}/E_μ is influenced by the packing ratio $\phi_r = \phi_0/\phi_{cp}$ because of the state of compression of the material. We calculate the minimal gravitational potential energy of the crater, *i.e.,* $E_{\mathrm{C}} = \sum_{z<0} \rho_g \phi_0 g z(x,y)^2 \Delta x \Delta y$, and find it represents at most 5 % of the impact energy released during cratering (Fig. 4.14). Our results are of similar

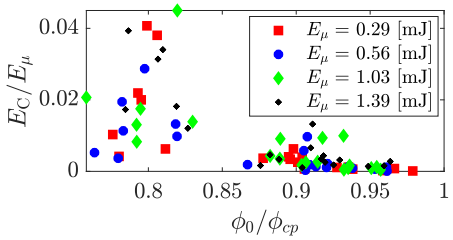


Figure 4.14. | Plot of the energy ratio E_{C}/E_μ as a function of the state of compression given by the packing ratio ϕ_0/ϕ_{cp}. We observe the energy ratio and its corresponding deviation both decrease as the compression state decreases, ultimately to 0 when $\phi_0 \to \phi_{cp}$.

magnitude with those obtained by Jong et al., 2017. The deviation of such energy ratio, as in Fig. 4.14 demonstrates that i) E_{C}/E_μ depends on ϕ_0/ϕ_{cp} and, ii) its scattering is proportional, i.e. the lower the compressibility, the

lower the scattering of the energy balance. In addition, whereas we previously found $\phi_{cp} = 0.74$, the ratio E_C/E_μ shows a greater convergence to 0 when a lower ϕ_{cp} value is used, *i.e.,* $\phi_{cp} \approx 0.63$ in Fig. 4.14.

We further question the relation between E_C considering both E_μ and $\Delta\phi = \phi_{cp} - \phi_0$ as a dynamic compressibility (Heyman et al., 2017), *i.e.,* the interplay between E_μ and $\Delta\phi$ should alter the energy dissipation and thus, the energy released during cratering. The best-fit exponent $b = 1.79$ is relatively closed in magnitude to the value 3/2 proposed in Heyman et al., 2017, see Fig. 4.15. The result suggest that i) the energy released during cratering

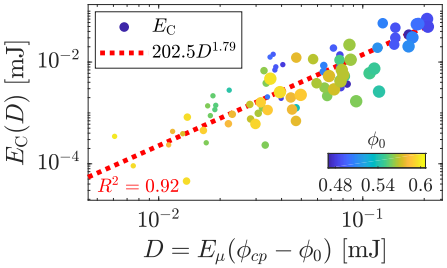


Figure 4.15. | Data collapse when the impact energy E_μ and the compressibility $\Delta\phi$ are combined. Color indicates the granular fraction whereas the circle's size indicates the magnitude of the impact energy.

obviously depends on two main variables, namely the impact energy and the compressibility of the granular matter, ii) such relation is highly nonlinear and, iii) an evident scattering prevails. It suggests the state of compression of the material to be non-homogeneous, which stands to reason if one considers a highly polydisperse media.

4.3.6. Discussion

Our work highlighted the overall symmetric distribution of deposited volumes with random fluctuations. We interpret such patterns as being governed by random processes of ejectas, *i.e.,* deposition of granular clumps, whereas asymmetric distributions are singular and possibly due to anisotropic deformations during the cratering process. In addition, we previously calculated the contribution of ejectas and clumps to the overall deposited volume to be at most 5 %. The magnitude of these fluctuations confirms our previous estimations. Moreover, and as stated by Zhao et al., 2015a, the rim of the spreading lamella develops a fingering instability during the drop deformation. These instabilities could certainly explain the random patterns of the deposited volumes, *i.e.,* granular clumps are likely to be transported by fingerings during the growth of the spreading lamella. In addition, Sabuwala et al., 2018 recently demonstrated that the ray system of the ejecta blanket due to a ball impact is governed by the undulations of the granular surface. The origin of random fluctuations in our experiment certainly lies in a interplay between fingering instabilities and the ray system due to surface undulations. The very few

asymmetric distributions that we observed are due to nonhomogeneous internal deformations during cratering. Packing heterogeneities could localize collapses within the force chain network, resulting in asymmetric deformations of the granular layer.

Whereas a Burr XII distribution was established by Long et al., 2014 as an accurate distribution of ejected particles, we found that this distribution was insufficient to fit the radial distribution of the excavated mass. The radial distance (Fig. 4.9 and Fig. 4.10) indicates another type of distribution that slightly deviates from a Burr XII distribution, particularly for the heavy-tailed part of the distributions. However, we also need to note the methodology that we used: the main concern was to describe how deposited volume was distributed, including single particle deposit to general uplift material. The Burr XII distribution significantly fits single particle travel distance, but it is still insufficient when dealing with transported mass, *e.g.,* granular clumps or uplift-induced volume.

We demonstrated an influence of the impact energy over the cratering response, considering both deposited and excavated materials. Let us recall the scaling between the crater diameter D_c and energy (assuming that material ejection is the predominant dissipation mechanism, as suggested by Amato et al., 1998) given by $D_c \sim E_\mu^\beta$ with $\beta = 0.25$ as reported in Katsuragi, 2011; Nefzaoui et al., 2012b. Approximating the crater as a half sphere and neglecting the constant 1.33π yields the crater volume $v_\mathrm{C} \propto (D_c/2)^3$. The following proportional relationship is derived for the crater volume $v_\mathrm{C} \propto E_\mu^{0.75}$. However, we obtained a higher exponent for the crater volume than initially predicted. Deboeuf et al., 2009 suggested an exponent $\beta = 0.67$ for solid spheres impacting a granular layer and as such are dominated by material ejection. Let us now consider that plastic deformation is the most dissipating process and that the crater dimension is determined by the crater depth $d_c \propto E_\mu^{0.33}$, as claimed in Katsuragi et al., 2007; Uehara et al., 2003. Then, we have $v_\mathrm{C} \propto (d_c/2)^3$; thus, an exponent $\beta = 1.00$ is obtained, confirmed by our experimental observations. These considerations confirm again that i) plastic deformations are dominant for liquid droplet impact experiments, and ii) when this type of deformation prevails, average cratering volume considering loose initial packings scales as $v_\mathrm{C} \propto E_\mu/Y$.

To better understand $\langle v_\mathrm{D}\rangle/\langle v_\mathrm{C}\rangle$ when considering average volumes, we assume simply i) the ejecta contribution to be negligible (*e.g.,* the proportion of uplift-induced volume was $p > 0.94$) and, ii) a conservation of mass, defined as $\phi_0\rho_g\langle v_\mathrm{C}\rangle = \phi_{cp}\rho_g\langle v_\mathrm{D}\rangle$, results from uplifts due to dilation, assuming that the close packing ϕ_{cp} is reached during internal deformations, along localized shear bands. Then, the conservation of mass combined to a plastic regime yields to $\phi_0/\phi_{cp} = \langle v_\mathrm{D}\rangle/\langle v_\mathrm{C}\rangle$. The latter states that the granular compression is constant, accordingly to initial and final packings, which strictly defines the volume ratio, *i.e.,* the greater the compression state, the higher the volume ratio. In addition, this also states that the granular capability to collapse vanishes when $\phi_0 \to \phi_{cp}$, which stands to reason since the material is not able

to compress more beyond its close packing state. The limit $\phi_0 \to \phi_{cp}$ thus defines the bulk incompressibility limit of the granular matter. Considering $\langle v_D \rangle / \langle v_C \rangle = 0.69$, we infer $\phi_{cp} = 0.72$ for $\phi_0 = 0.5$, which is similar for the close random packing of equally sized spheres but higher than the widely accepted $\phi_{cp} = 0.64$ for polydisperse system (Farr et al., 2009; Torquato et al., 2000). We may suspect the geometry of grains to cause higher values, *i.e.,* ellipsoid shape of grains typically results in $\phi_{cp} \in [0.68; 0.74]$ (Jin et al., 2017; Donev et al., 2004). However, such inferred value of ϕ_{cp} is not equivalent when considering the packing ratio and the energy ratio, *i.e.,* $\phi_{cp} \approx 0.63$ typically, which is closer to an polydisperse assembly of spheres. Again, the energy released during cratering E_C is null when the bulk incompressiblity limit is reached, *i.e.,* $\Delta\phi = 0$.

As stated by Royer et al., 2011; Royer et al., 2007 for solid-to-granular impacts, the compaction-dilation transition phase is strongly influenced by the packing. The latter still prevails for liquid-to-granular impacts because local granular uplifts are clearly observed. However, it was not possible to investigate in detail such a transition due to a CPS, as mentioned in Umbanhowar et al., 2010 for glass beads (dry noncohesive granular media of slightly polydisperse particles). Identifying an accurate compaction-dilation transition value was not possible in our experiments. Furthermore, Umbanhowar et al., 2010 suggests a critical packing state of $\phi_{cs} = 0.591 \pm 0.005$, but this value did not explicitly appear. Again, when comparing our results with those of Umbanhowar et al., 2010, we also observe a major difference: the crater rim magnitude is proportional to the initial packing, whereas Umbanhowar et al., 2010 clearly indicated an inversely proportional relationship.

4.4. Conclusion

We demonstrated that volumes resulting from cratering are mainly correlated to the impact energy but are also proportional to the initial granular packing. In particular, the amount of the impact energy transferred to the cratering phase seems to depend on the compressibility of the granular layer. Considering the deposited volume also revealed overall symmetric patterns, even though singular impacts led to asymmetric cratering responses during internal layer deformations. Deeper insights could benefit from highly accurate numerical models, *e.g.,* distinct element simulations, and could lead to potential novel understandings of anisotropic but singular deformations during cratering, ultimately due to granular packing heterogeneities within the granular layer.

Acknowledgements

Authors would like to thank Pierre-Etienne Cherix (Mécanix Sàrl, 1880 Bex, Switzerland), who created the whole experimental device and thus made this

research possible. Shiva P. Pudasaini gratefully thanks the Herbette Foundation for providing financial support for Sabbatical visit to the University of Lausanne, Switzerland for the year 2018, April - June.

Authors contributions

EM prepared the manuscript and performed all the experiments, together with DC. SP considerably improved the manuscript by providing insightful suggestions, perspectives and comments. MJ supervised the experimental project, designed the experimental setup and provided strong assistance and guidance throughout all the experimental work.

5. Code prototyping in MATLAB

A fast and efficient MATLAB-based MPM solver: fMPMM-solver v1.1

Emmanuel Wyser, Yury Alkhimenkov, Michel Jaboyedoff & Yury Y. Podladchikov

Published in *Geoscientific Model Development*, **13** (2020), p. 6265-6284

Abstract We present an efficient MATLAB-based implementation of the material point method (MPM) and its most recent variants. MPM has gained popularity over the last decade, especially for problems in solid mechanics in which large deformations are involved, i.e., cantilever beam problems, granular collapses, and even large-scale snow avalanches. Although its numerical accuracy is lower than that of the widely accepted finite element method (FEM), MPM has been proven useful in overcoming some of the limitations of FEM, such as excessive mesh distortions. We demonstrate that MATLAB is an efficient high-level language for MPM implementations that solve elasto-dynamic and elasto-plastic problems. We accelerate the MATLAB-based implementation of MPM method by using the numerical techniques recently developed for FEM optimization in MATLAB. These techniques include vectorisation, the usage of native MATLAB functions, the maintenance of optimal RAM-to-cache communication, and others. We validate our in-house code with classical MPM benchmarks including i) the elastic collapse under self-weight of a column, ii) the elastic cantilever beam problem, and iii) existing experimental and numerical results, i.e., granular collapses and slumping mechanics respectively. We report a performance gain by a factor of 28 for a vectorised code compared to a classical iterative version. The computational performance of the solver is at least 2.8 times greater than those of previously reported MPM implementations in Julia under a similar computational architecture.

5.1. Introduction

The material point method (MPM), developed in the 1990s (Sulsky et al., 1994), is an extension of a particle-in-cell (PIC) method to solve solid mechanics problems involving massive deformations. It is an alternative to Lagrangian approaches (updated Lagrangian finite element method) that is well suited to problems with large deformations involved in geomechanics, granular mechanics or even snow avalanche mechanics. Vardon et al., 2017; Wang et al., 2016c investigated elasto-plastic problems of strain localization of slumping processes relying on an explicit or implicit MPM formulation. Similarly, Bandara et al., 2016; Bandara et al., 2015; Abe et al., 2014 proposed a poro-elasto-plastic MPM formulation to study levee failures induced by pore pressure increases. Additionally, Baumgarten et al., 2019a; Dunatunga et al., 2017; Dunatunga et al., 2015; Więckowski, 2004 proposed a general numerical framework of granular mechanics, i.e., silo discharge or granular collapses. More recently, Gaume et al., 2019; Gaume et al., 2018 proposed a unified numerical model in the finite deformation framework to study the whole process, i.e., from failure to propagation, of slab avalanche releases.

The core idea of MPM is to discretize a continuum with material points carrying state variables (e.g., mass, stress, and velocity). The latter are mapped (accumulated) to the nodes of a regular or irregular background FE mesh, on which an Eulerian solution to the momentum balance equation is explicitly advanced forward in time. Nodal solutions are then mapped back to the material points, and the mesh can be discarded. The mapping from material points to nodes is ensured using the standard FE hat function that spans over an entire element (Bardenhagen et al., 2004). This avoids a common flaw of FEM, which is an excessive mesh distortion. We will refer to this first variant as the standard material point method (sMPM).

MATLAB© allows a rapid code prototyping but, at the expense of significantly lower computational performances than compiled language. An efficient MATLAB implementation of FEM called MILAMIN (Million a Minute) was proposed by Dabrowski et al., 2008 that was capable of solving two-dimensional linear problems with one million unknowns in one minute on a modern computer with a reasonable architecture. The efficiency of the algorithm lies on a combined use of vectorised calculations with a technique called blocking. MATLAB uses the Linear Algebra PACKages (LAPACK), written in Fortran, to perform mathematical operations by calling Basic Linear Algebra Subroutines (BLAS, Moler 2000). The latter results in an overhead each time a BLAS call is made. Hence, mathematical operations over a large number of small matrices should be avoided and, operations on fewer and larger matrices preferred. This is a typical bottleneck in FEM when local stiffness matrices are assembled during the integration point loop within the global stiffness matrix. Dabrowski et al., 2008 proposed an algorithm, in which a loop reordering is combined with operations on blocks of elements to address this bottleneck. However, data required for a calculation within a block should

entirely resides in the CPUs cache. Otherwise, an additional time is spent on the RAM-to-cache communication and the performance decreases. Therefore, an optimal block size exists and, is solely defined by the CPU architecture. This technique of vectorisation combined with blocking significantly increases the performance.

More recently, Bird et al., 2017 extended the vectorised and blocked algorithm presented by Dabrowski et al., 2008 to the calculation of the global stiffness matrix for Discontinuous Galerkin FEM considering linear elastic problems using only native MATLAB functions. Indeed, the optimisation strategy chosen by Dabrowski et al., 2008 also relied on non-native MATLAB functions, e.g., `sparse2` of the SuiteSparse package (Davis, 2013). In particular, Bird et al., 2017 showed the importance of storing vectors in a column-major form during calculation. Mathematical operations are performed in MATLAB by calling LAPACK, written in FORTRAN, in which arrays are stored in column-major order form. Hence, element-wise multiplication of arrays in column-major form is significantly faster and thus, vectors in column-major form are recommended, whenever possible. Bird et al., 2017 concluded that vectorisation alone results in a performance increase between 13.7 and 23 times, while blocking only improved vectorisation by an additional 1.8 times. O'Sullivan et al., 2019 recently extended the works of Bird et al., 2017; Dabrowski et al., 2008 to optimised elasto-plastic codes for Continuous Galerkin (CG) or Discontinuous Galerkin (DG) methods. In particular, they proposed an efficient native MATLAB function, i.e., `accumarray()`, to efficiently assemble the internal force vector. Such function constructs an array by accumulation. More generally, O'Sullivan et al., 2019 reported a performance gain of x25.7 when using an optimised CG code instead of an equivalent non-optimised code.

Since MPM and FEM share common grounds, we aim at increasing the performances of MATLAB up to what was reported by Sinaie et al., 2017 using Julia language environment. In principal, Julia is significantly faster than MATLAB for a MPM implementation. We combine the most recent and accurate versions of MPM: the explicit generalized interpolation material point method (GIMPM, Bardenhagen et al. 2004) and the explicit convected particle domain interpolation with second-order quadrilateral domains (CPDI2q and CPDI, Sadeghirad et al. 2013; Sadeghirad et al. 2011) variants with some of the numerical techniques developed during the last decade of FEM optimisation in MATLAB. These techniques include the use of `accumarray()`, optimal RAM-to-cache communication, minimum BLAS calls and the use of native MATLAB functions. We did not consider the blocking technique initially proposed by Dabrowski et al., 2008 since an explicit formulation in MPM excludes the global stiffness matrix assembly procedure. The performance gain mainly comes from the vectorisation of the algorithm, whereas blocking has a less significant impact over the performance gain, as stated by Bird et al., 2017. The vectorisation of MATLAB functions is also crucial for a straight transpose of the solver to a more efficient language, such as the C-CUDA language, which allows the parallel execution of computational kernels of graphics processing

units (GPUs).

In this contribution, we present an implementation of an efficiently vectorised explicit MPM solver (fMPMM-solver, which v1.1 is available for download from Bitbucket at: https://bitbucket.org/ewyser/fmpmm-solver/src/master/), taking advantage of vectorisation capabilities of MATLAB©. We extensively use native functions of MATLAB© such as `repmat( )`, `reshape( )`, `sum( )` or `accumarray( )`. We validate our in-house code with classical MPM benchmarks including i) the elastic collapse under self-weight of a column, ii) the elastic cantilever beam problem, and iii) existing experimental results, i.e., granular collapses and slumping mechanics. We demonstrate the computational efficiency of a vectorised implementation over an iterative one for the case of an elasto-plastic collapse of a column. We compare the performances of Julia and MATLAB language environments for the collision of two elastic discs problem.

5.2. Overview of the Material Point Method (MPM)

5.2.1. A Material Point Method implementation

The material point method (MPM), originally proposed by Sulsky et al., 1995; Sulsky et al., 1994 in an explicit formulation, is an extension of the particle-in-cell (PIC) method. The key idea is to solve the weak form of the momentum balance equation on a FE mesh while state variables (e.g., stress, velocity or mass) are stored at Lagrangian points discretizing the continuum, i.e., the material points, which can move according to the deformation of the grid (Dunatunga et al., 2017). MPM could be regarded as a finite element solver in which integration points (material points) are allowed to move (Guilkey et al., 2003) and are thus not always located at the Gauss-Legendre location within an element, resulting in higher quadrature errors and poorer integration estimates, especially when using low-order basis functions (Steffen et al., 2008b; Steffen et al., 2008a).

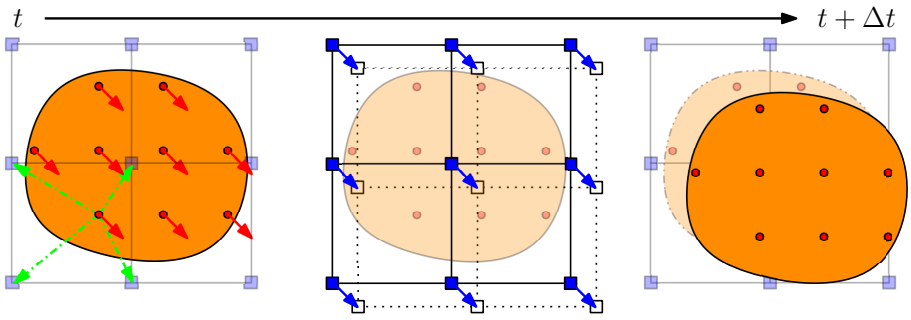
A typical calculation cycle (see Fig. 5.1) consists of the three following steps (Wang et al., 2016a):

1. A Mapping phase, during which properties of the material point (mass, momentum or stress) are mapped to the nodes.

2. An updated-Lagrangian FEM (UL-FEM) phase, during which the momentum equations are solved on the nodes of the background mesh and, the solution is explicitly advanced forward in time.

3. A Convection phase, during which i) the nodal solutions are interpolated back to the material points, and ii) the properties of the material point are updated.

Since the 1990's, several variants were introduced to resolve a number of numerical issues. The generalized interpolation material point method (GIMPM)



a) Points to Nodes Projection b) Nodal solution c) Nodes to Points Interpolation

Figure 5.1. | Typical calculation cycle of a MPM solver for a homogeneous velocity field, inspired by Dunatunga et al., 2017. a) The continuum (orange) is discretized into a set of Lagrangian material points (red dots), at which state variables or properties (e.g., mass, stress, and velocity) are defined. The latter are mapped to an Eulerian finite element mesh made of nodes (blue square). b) Momentum equations are solved at the nodes and, the solution is explicitly advanced forward in time. c) The nodal solutions are interpolated back to the material points and, their properties are updated.

was first presented by Bardenhagen et al., 2004. They proposed a generalization of the basis and gradient functions that were convoluted with a characteristic domain function of the material point. A major flaw in sMPM is the lack of continuity of the gradient basis function, resulting in spurious oscillations of internal forces as soon as a material point crosses an element boundary while entering into its neighbour. This is referred to as cell-crossing instabilities due to the C_0 continuity of the gradient basis functions used in sMPM. Such issue is minimized by the GIMPM variant (González Acosta et al., 2020).

GIMPM is categorized as a domain-based material point method, unlike the later development of the B-spline material point method (BSMPM, e.g. Koster et al. 2021; Gan et al. 2018; Gaume et al. 2018; Stomakhin et al. 2013) which cures cell-crossing instabilities using B-spline functions as basis functions. Whereas in sMPM only nodes belonging to an element contribute to a given material point, GIMPM requires an extended nodal connectivity, i.e., the nodes of the element enclosing the material point and the nodes belonging to the adjacent elements (see Fig. 5.2). More recently, the convected particle domain interpolation (CPDI and its most recent development CPDI2q) has been proposed by Sadeghirad et al., 2013; Sadeghirad et al., 2011.

We choose the explicit GIMPM variant with the modified update stress last scheme (MUSL, see Nairn, 2003; Bardenhagen et al., 2000 for a detailed discussion), i.e., the stress of material point is updated after the nodal solutions are obtained. The updated momentum of the material point is then mapped back a second time to the nodes in order to obtain an updated nodal velocity, further used to calculate derivative terms such as strains or the deformation gradient of the material point. The explicit formulation also implies the well-

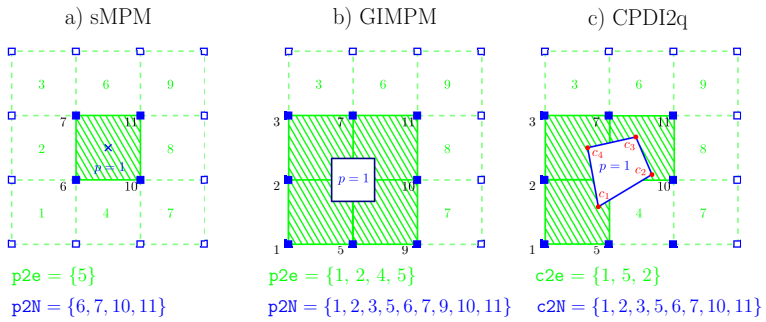


Figure 5.2. | Nodal connectivities of a) standard MPM, b) GIMPM and c) CPDI2q variants. The material point's location is marked by the blue cross. Note that for sMPM (and similarly BSMPM) the particle domain does not exist, unlike GIMPM or CPDI2q (the blue square enclosing the material point). Nodes associated with the material point are denoted by filled blue squares, and the element number appears in green in the centre of the element. For sMPM and GIMPM, the connectivity array between the material point and the element is p2e and, the array between the material point and its associated nodes is p2N. For CPDI2q, the connectivity array between the corners (filled red circles) of the quadrilateral domain of the material point and the element is c2e and, the array between the corners and their associated nodes is c2N.

known restriction on the time step, which is limited by the Courant-Friedrich-Lewy (CFL) condition to ensure numerical stability.

Additionally, we implemented a CPDI/CPDI2q version (in an explicit and quasi-static implicit formulation) of the solver. However, in this paper, we do not present the theoretical background of the CPDI variant nor the implicit implementation of a MPM-based solver. Therefore, interested readers are referred to the original contributions of Sadeghirad et al., 2013; Sadeghirad et al., 2011 and González Acosta et al., 2020; Charlton et al., 2017; Iaconeta et al., 2017; Beuth et al., 2008; Guilkey et al., 2003, respectively. Regarding the quasi-static implicit implementation, we strongly adapted our vectorisation strategy to some aspects of the numerical implementation proposed by Coombs et al., 2020a in the MATLAB code AMPLE v1.0. However, we did not consider blocking, because our main concern for performance is on the explicit implementation.

5.2.2. Domain-based material point method variants

Domain-based material point method variants could be treated as two distinct groups:

- The material point's domain is a square for which the deformation is always aligned with the mesh axis, i.e., a non-deforming domain uGIMPM (Bardenhagen et al., 2004) or, a deforming domain cpGIMPM (Wallstedt et al., 2008), the latter being usually related to a measure of the

deformation, e.g., the determinant of the deformation gradient.

- The material point's domain is either a deforming parallelogram for which its dimensions are specified by two vectors, i.e., CPDI (Sadeghirad et al., 2011), or a deforming quadrilateral solely defined by its corners, i.e., CPDI2q (Sadeghirad et al., 2013). However, the deformation is not necessarily aligned with the mesh anymore.

We first focus on the different domain updating methods for GIMPM. Four domain updating methods exists: i) the domain is not updated, ii) the deformation of the domain is proportional to the determinant of the deformation gradient $\det(F_{ij})$ (Bardenhagen et al., 2004), iii) the domain lengths l_p are updated accordingly to the principal component of the deformation gradient F_{ii} (Sadeghirad et al., 2011) or, iv) are updated with the principal component of the stretch part of the deformation gradient U_{ii} (Charlton et al., 2017). Coombs et al., 2020b highlighted the suitability of generalised interpolation domain updating methods accordingly to distinct deformation modes. Four different deformation modes were considered by Coombs et al., 2020b: simple stretch, hydrostatic compression/extension, simple shear and, pure rotation. Coombs et al., 2020b concluded the following:

- Not updating the domain is not suitable for simple stretch and hydrostatic compression/extension.
- A domain update based on $\det(F_{ij})$ will results in an artificial contraction/expansion of the domain for simple stretch.
- The domain will vanish with increasing rotation when using F_{ii}.
- The domain volume will change under isochoric deformation when using U_{ii}.

Consequently, Coombs et al., 2020b proposed a hybrid domain update inspired by CPDI2q approaches: the corners of the material point domain are updated accordingly to the nodal deformation but, the midpoints of the domain limits are used to update domain lengths l_p to maintain a rectangular domain. Even tough Coombs et al., 2020b reported an excellent numerical stability, the drawback is to compute specific basis functions between nodes and material point's corners, which has an additional computational cost. Hence, we did not selected this approach in this contribution.

Regarding the recent CPDI/CPDI2q, Wang et al., 2019 investigated the numerical stability under stretch, shear and torsional deformation modes. CPDI2q was found to be erroneous in some case, especially when torsion mode is involved, due to distortion of the domain. In contrast, CPDI and even sMPM performed better in modelling torsional deformations. Even tough CPDI2q can exactly represent the deformed domain (Sadeghirad et al., 2013), care must be taken when dealing with very large distortion, especially when

the material has yielded, which is common in geotechnical engineering (Wang et al., 2019).

Consequently, the domain-based method as well as the domain updating method should be carefully chosen accordingly to the deformation mode expected for a given case. The latter will be always specify in the following and, the domain update method will be clearly stated.

5.3. MATLAB-based MPM implementation

5.3.1. Rate formulation and elasto-plasticity

The large deformation framework in a linear elastic continuum requires an appropriate stress-strain formulation. One approach is based on the finite deformation framework, which relies on a linear relationship between elastic logarithmic strains and Kirchoff stresses (Coombs et al., 2020b; Gaume et al., 2018; Charlton et al., 2017). In this study, we adopt another approach, namely, a rate dependent formulation using the Jaumann stress rate (e.g. Huang et al. 2015; Bandara et al. 2016; Wang et al. 2016c; Wang et al. 2016b). This formulation provides an objective (invariant by rotation or frame-indifferent) stress rate measure (Souza Neto et al., 2011) and is simple to implement. The Jaumann rate of the Cauchy stress is defined as

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = \frac{1}{2}C_{ijkl}\left(\frac{\partial v_l}{\partial x_k} + \frac{\partial v_k}{\partial x_l}\right), \tag{5.3.1}$$

where C_{ijkl} is the fourth rank tangent stiffness tensor and v_k is the velocity. Thus, the Jaumann stress derivative can be written as

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = \frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} - \sigma_{ik}\omega_{jk} - \sigma_{jk}\omega_{ik}, \tag{5.3.2}$$

where $\omega_{ij} = (\partial_i v_j - \partial_j v_i)/2$ is the vorticity tensor and $\mathrm{D}\sigma_{ij}/\mathrm{D}t$ denotes the material derivative

$$\frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} = \frac{\partial \sigma_{ij}}{\partial t} + v_k\frac{\partial \sigma_{ij}}{\partial x_k}. \tag{5.3.3}$$

Plastic deformation is modelled with a pressure dependent Mohr-Coulomb law with non-associated plastic flow, i.e., both the dilatancy angle ψ and the volumetric plastic strain ϵ_v^p are null (Vermeer et al., 1984). We have adopted the approach of Simpson, 2017 for a two dimensional linear elastic, perfectly plastic (elasto-plasticity) continuum because of its simplicity and its ease of implementation. The yield function is defined as

$$f = \tau + \sigma\sin\phi - c\cos\phi, \tag{5.3.4}$$

where c is the cohesion and ϕ the angle of internal friction,

$$\sigma = (\sigma_{xx} + \sigma_{yy})/2, \tag{5.3.5}$$

and

$$\tau = \sqrt{(\sigma_{xx} - \sigma_{yy})^2/4 + \sigma_{xy}^2}. \tag{5.3.6}$$

The elastic state is defined when $f < 0$. However when $f > 0$, plastic state is declared and stresses must be corrected (or scaled) to satisfy the condition $f = 0$, since $f > 0$ is an inadmissible state. Simpson, 2017 proposed the following simple algorithm to return stresses to the yield surface,

$$\sigma_{xx}^{\star} = \sigma + (\sigma_{xx} - \sigma_{yy})\beta/2, \tag{5.3.7}$$

$$\sigma_{yy}^{\star} = \sigma - (\sigma_{xx} - \sigma_{yy})\beta/2, \tag{5.3.8}$$

$$\sigma_{xy}^{\star} = \sigma_{xy}\beta, \tag{5.3.9}$$

where $\beta = (\mid c\cos\phi - \sigma\sin\phi \mid)/\tau$, and $\sigma_{xx}^{\star}$, $\sigma_{yy}^{\star}$ and $\sigma_{xy}^{\star}$ are the corrected stresses, i.e., $f = 0$.

A similar approach is used to return stresses when considering a non-associated Drucker-Prager plasticity (see Huang et al., 2015 for a detailed description of the procedure). In addition, their approach allows also to model associated plastic flows, i.e., $\psi > 0$ and $\epsilon_v^p \neq 0$.

5.3.2. Structure of the MPM solver

The solver procedure is shown in Fig. 5.3. In the `main.m` script, both functions `meSetup.m` and `mpSetup.m`, respectively, define the geometry and related quantities such as the nodal connectivity (or element topology) array, e.g., the `e2N` array. The latter stores the nodes associated with a given element. As such, a material point p located in an element e can immediately identify which nodes n it is associated with.

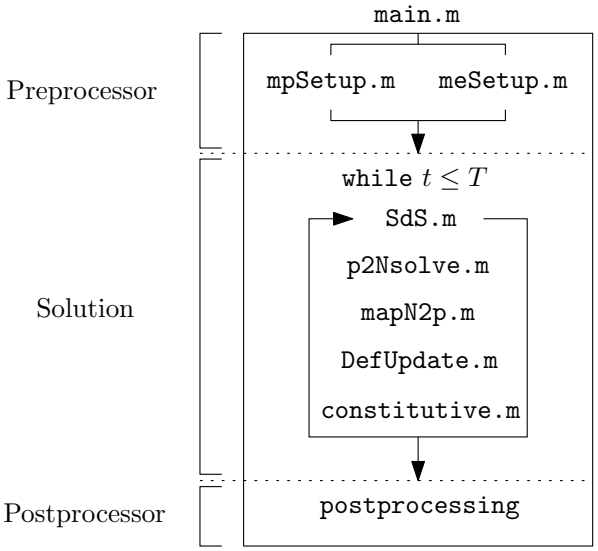


Figure 5.3. | Workflow of the explicit GIMPM solver and the calls to functions within a calculation cycle. The role of each function is described in the text.

After initialization, a while loop solves the elasto-dynamic (or elasto-plastic) problem until a time criterion T is reached. This time criterion could be restricted to the time needed for the system to reach an equilibrium, or if the global kinetic energy of the system has reached a threshold.

At the beginning of each cycle, a connectivity array `p2e` between the material points and their respective element (a material point can only reside in a single element) is constructed. Since i) the nodes associated with the elements and ii) the elements enclosing the material points are known, it is possible to obtain the connectivity array `p2N` between the material points and their associated nodes, e.g., `p2N=e2N(p2e,:)` in a MATLAB syntax (see Fig. 5.2 for an example of these connectivity arrays). This array is of dimension (n_p, n_n), with n_p the total number of material points, n_n the total number of nodes associated with an element (16 in two-dimensional problems) and $n_{i,j}$ the node number where i corresponds to the material point and j corresponds to its j-th associated nodes, which results in the following:

$$\texttt{p2N} = \begin{pmatrix} n_{1,1} & \cdots & n_{1,n_n} \\ \vdots & \ddots & \vdots \\ n_{n_p,1} & \cdots & n_{n_p,n_n} \end{pmatrix}. \tag{5.3.10}$$

The following functions are called successively during one calculation cycle:

1. `SdS.m` calculates the basis functions, derivatives and, assembles the strain-displacement matrix for each material points.

2. `p2Nsolve.m` projects the quantities of the material point (e.g., mass and momentum) to the associated nodes, solves the equations of motion and sets boundary conditions.

3. `mapN2p.m` interpolates nodal solutions (acceleration and velocity) to the material points with a double mapping procedure (see Zhang et al., 2017 or Nairn, 2003 for a clear discussion of USF, USL and MUSL algorithms).

4. `DefUpdate.m` updates incremental strains and deformation-related quantities (e.g., the volume of the material point or the domain half-length) at the level of the material point based on the remapping of the updated material point momentum.

5. `constitutive.m` calls two functions to solve for the constitutive elasto-plastic relation, i.e.,

   a) `elastic.m`, which predicts an incremental objective stress assuming a purely elastic step, further corrected by

   b) `plastic.m`, which corrects the trial stress by a plastic correction if the material has yielded.

When a time criterion is met, the calculation cycle stops and further post-processing tasks (visualization, data exportation) can be performed.

The numerical simulations are conducted using MATLAB© R2018a within a Windows 7 64-bit environment on an Intel Core i7-4790 (4th generation CPU with 4 physical cores of base frequency at 3.60 GHz up to a maximum turbo frequency of 4.00 GHz) with 4 × 256 kB L2 cache and 16 GB DDR3 RAM (clock speed 800 MHz).

5.3.3. Vectorisation

Basis functions and derivatives

The GIMPM basis function (Coombs et al., 2018; Steffen et al., 2008b; Bardenhagen et al., 2004) results from the convolution of a characteristic particle function χ_p (i.e., the material point spatial extent or domain) with the standard basis function $N_n(x)$ of the mesh, which results in:

$$S_n(x_p) = \begin{cases} 1 - (4x^2 + l_p^2)/(4hl_p) & \text{if } \mid x \mid < l_p/2 \\ 1 - \mid x \mid /h & \text{if } l_p/2 \leq \mid x \mid < h - l_p/2 \\ (h + l_p/2 - \mid x \mid)^2 /(2hl_p) & \text{if } h - l_p/2 \leq \mid x \mid < h + l_p/2 \\ 0 & \text{otherwise ,} \end{cases} \tag{5.3.11}$$

with l_p the length of the material point domain, h the mesh spacing, $x = x_p - x_n$ where x_p is the coordinate of a material point and x_n the coordinate of its associated node n. The basis function of a node n with its material point p is constructed for a two-dimensional model, as follows:

$$S_n(\boldsymbol{x}_p) = S_n(x_p) S_n(y_p), \tag{5.3.12}$$

for which the derivative is defined as:

$$\nabla S_n(\boldsymbol{x}_p) = (\partial_x S_n(x_p) S_n(y_p), S_n(x_p) \partial_y S_n(y_p)). \tag{5.3.13}$$

Similar to the FEM, the strain-displacement matrix $\boldsymbol{B}$ consists of the derivatives of the basis function and is assigned to each material point, which results in the following:

$$\boldsymbol{B}(\boldsymbol{x}_p) = \begin{pmatrix} \partial_x S_1 & 0 & \cdots & \partial_x S_{n_n} & 0 \\ 0 & \partial_y S_1 & \cdots & 0 & \partial_y S_{n_n} \\ \partial_y S_1 & \partial_x S_1 & \cdots & \partial_y S_{n_n} & \partial_x S_{n_n} \end{pmatrix}, \tag{5.3.14}$$

where n_n is the total number of associated nodes to an element e, in which a material point p resides.

The algorithm outlined in Fig. 5.4 (the function `[mpD] = SdS(meD,mpD,p2N)` called at the beginning of each cycle, see Fig. 5.4) represents the vectorised solution of the computation of basis functions and their derivatives.

Coordinates of the material points `mpD.x(:,1:2)` are first replicated and then subtracted by their associated nodes coordinates, e.g., `meD.x(p2N)` and `meD.y(p2N)` respectively (Lines 3 or 5 in Fig. 5.4). This yields the array `D`

with the same dimension of `p2N`. This array of distance between the points and their associated nodes is sent as an input to the nested function `[N,dN] = NdN(D,h,lp)`, which computes 1D basis function and its derivative through matrix element-wise operations (operator `.*`) (either in Line 4 for x coordinates or Line 6 for y coordinates in Fig. 5.4).

Given the piece-wise Eq. 6.6.3, three logical arrays (`c1`, `c2` and `c3`) are defined (Lines 21-24 in Fig. 5.4), whose elements are either 1 (the condition is true) or 0 (the condition is false). Three arrays of basis functions are calculated (`N1`, `N2` and `N3`, Lines 26-28) according to Eq. 6.6.4. The array of basis functions `N` is obtained through a summation of the element-wise multiplications of these temporary arrays with their corresponding logical arrays (Line 29 in Fig. 5.4). The same holds true for the calculation of the gradient basis function (Lines 31-34 in Fig. 5.4). It is faster to use logical arrays as multipliers of precomputed basis function arrays rather than using these in a conditional indexing statement, e.g., `N(c2==1) = 1-abs(dX(c2==1))./h`. The performance gain is significant between the two approaches, i.e., an intrinsic 30 % gain over the wall-clock time of the basis functions and derivatives calculation. We observe an invariance of such gain with respect to the initial number of material point per element or to the mesh resolution.

```
1  function [mpD] = SdS(meD,mpD,p2N)
2  %% COMPUTE (X,Y)-BASIS FUNCTION
3  D         = (repmat(mpD.x(:,1),1,meD.nNe) - meD.x(p2N)) ;%
4  [Sx,dSx] = NdN(D,meD.h(1),repmat(mpD.l(:,1),1,meD.nNe));%
5  D         = (repmat(mpD.x(:,2),1,meD.nNe) - meD.y(p2N)) ;%
6  [Sy,dSy] = NdN(D,meD.h(2),repmat(mpD.l(:,2),1,meD.nNe));%
7  %% CONVOLUTION OF BASIS FUNCTIONS
8  mpD.S     =  Sx.* Sy                                    ;%
9  mpD.dSx   = dSx.* Sy                                    ;%
10 mpD.dSy   =  Sx.*dSy                                    ;%
11 %% B MATRIX ASSEMBLY
12 iDx             = 1:meD.DoF:meD.nDoF(1)-1               ;%
13 iDy             = iDx+1                                 ;%
14 mpD.B(1,iDx,:) = mpD.dSx'                               ;%
15 mpD.B(2,iDy,:) = mpD.dSy'                               ;%
16 mpD.B(3,iDx,:) = mpD.dSy'                               ;%
17 mpD.B(3,iDy,:) = mpD.dSx'                               ;%
18 end
19 function [N,dN]=NdN(dX,h,lp)
20 %% COMPUTE BASIS FUNCTIONS
21 lp = 2*lp                                                ;%
22 c1 = ( abs(dX)< (  0.5*lp)                        )     ;%
23 c2 = ((abs(dX)>=(  0.5*lp)) & (abs(dX)<(h-0.5*lp)))     ;%
24 c3 = ((abs(dX)>=(h-0.5*lp)) & (abs(dX)<(h+0.5*lp)))     ;%
25 % BASIS FUNCTION
26 N1 = 1-((4*dX.^2+lp.^2)./(4*h.*lp))                     ;%
27 N2 = 1-(abs(dX)./h)                                     ;%
28 N3 = ((h+0.5*lp-abs(dX)).^2)./(2*h.*lp)                 ;%
29 N  = c1.*N1+c2.*N2+c3.*N3                               ;%
30 % BASIS FUNCTION GRADIENT
31 dN1= -((8*dX)./(4*h.*lp))                               ;%
32 dN2= sign(dX).*(-1/h)                                   ;%
33 dN3=-sign(dX).*(h+0.5*lp-abs(dX))./(h*lp)               ;%
34 dN = c1.*dN1+c2.*dN2+c3.*dN3                            ;%
35 end
```

Figure 5.4. | Code Fragment 1 shows the vectorised solution to the calculation of the basis functions and their derivatives within `SdS.m`. Table 5.3 lists the variables used.

Integration of internal forces

Another computationally expensive operation for MATLAB© is the mapping (or accumulation) of the material point contributions to their associated nodes. It is performed by the function `p2Nsolve.m` in the workflow of the solver.

The standard calculations for the material point contributions to the lumped mass m_n, the momentum $\boldsymbol{p}_n$, the external $\boldsymbol{f}_n^e$ and internal $\boldsymbol{f}_n^i$ forces are given by:

$$m_n = \sum_{p \in n} S_n(\boldsymbol{x}_p) m_p, \tag{5.3.15}$$

$$\boldsymbol{p}_n = \sum_{p \in n} S_n(\boldsymbol{x}_p) m_p \boldsymbol{v}_p, \tag{5.3.16}$$

$$\boldsymbol{f}_n^e = \sum_{p \in n} S_n(\boldsymbol{x}_p) m_p \boldsymbol{b}_p, \tag{5.3.17}$$

$$\boldsymbol{f}_n^i = \sum_{p \in n} v_p \boldsymbol{B}^T(\boldsymbol{x}_p) \boldsymbol{\sigma}_p, \tag{5.3.18}$$

with m_p the material point mass, $\boldsymbol{v}_p$ the material point velocity, $\boldsymbol{b}_p$ the body force applied to the material point and $\boldsymbol{\sigma}_p$ the material point Cauchy stress tensor in the Voigt notation.

Once the mapping phase is achieved, the equations of motions are explicitly solved forward in time on the mesh. Nodal accelerations $\boldsymbol{a}_n$ and velocities $\boldsymbol{v}_n$ are given by:

$$\boldsymbol{a}_n^{t+\Delta t} = m_n^{-1}(\boldsymbol{f}_n^e - \boldsymbol{f}_n^i), \tag{5.3.19}$$

$$\boldsymbol{v}_n^{t+\Delta t} = m_n^{-1}\boldsymbol{p}_n + \Delta t \boldsymbol{a}_n^{t+\Delta t}. \tag{5.3.20}$$

Finally, boundary conditions are applied to the nodes that belong to the boundaries.

The vectorised solution comes from the use of the built-in function `accumarray( )` of MATLAB© combined with `reshape( )` and `repmat( )`. The core of the vectorization is to use `p2N` as a vector (i.e., flattening the array `p2N(:)` results in a row vector) of subscripts with `accumarray`, which accumulates material point contributions (e.g., mass or momentum) that share the same node.

In the function `p2Nsolve` (Code Fragment 2 shown in Fig. 5.5), the first step is to initialize nodal vectors (mass, momentum, forces, etc.) to zero (Lines 4-5 in Fig. 5.5). Then, temporary vectors (`m`, `p`, `f` and `fi`) of material point contributions (namely, mass, momentum, and external and internal forces) are generated (Lines 10-17 in Fig. 5.5). The accumulation (nodal summation) is performed (Lines 19-26 in Fig. 5.5) using either the flattened `p2n(:)` or `l2g(:)` (e.g., the global indices of nodes) as the vector of subscripts. Note that for the accumulation of material point contributions of internal forces, a short for-loop iterates over the associated node (e.g., from 1 to `meD.nNe`) of every material point to accumulate their respective contributions.

```
function [meD] = p2Nsolve(meD,mpD,g,dt,l2g,p2N,bc)
%% INITIALIZATION
% NODAL VECTOR INITIALIZATION
meD.m(:) = 0.0 ; meD.mr(:) = 0.0 ; meD.f(:) = 0.0 ; meD.d(:) = 0.0        ;%
meD.a(:) = 0.0 ;  meD.p(:) = 0.0 ; meD.v(:) = 0.0 ; meD.u(:) = 0.0        ;%
%% CONTRIBUTION TO NODES
% PREPROCESSING
m = reshape( mpD.S.*repmat(mpD.m,1,meD.nNe)      ,mpD.n*meD.nNe    ,1)     ;%
p = reshape([mpD.S.*repmat(mpD.p(:,1),1,meD.nNe);...
             mpD.S.*repmat(mpD.p(:,2),1,meD.nNe)],mpD.n*meD.nDoF(1),1)     ;%
f = reshape([mpD.S.*0.0                          ;...
             mpD.S.*repmat(mpD.m,1,meD.nNe).*-g ],mpD.n*meD.nDoF(1),1)     ;%
fi= squeeze(sum(mpD.B.*repmat(reshape(mpD.s,size(mpD.s,1),1,mpD.n)...
                    ,1,meD.nDoF(1)),1)).*repmat(mpD.V',meD.nDoF(1),1)     ;%
% CONTRIBUTION FROM p TO N
meD.m = accumarray(p2N(:),m,[meD.nN     1])                               ;%
meD.p = accumarray(l2g(:),p,[meD.nDoF(2) 1])                              ;%
meD.f = accumarray(l2g(:),f,[meD.nDoF(2) 1])                              ;%
for n = 1:meD.nNe
    l = [(meD.DoF*p2N(:,n)-1);(meD.DoF*p2N(:,n))]                         ;%
    meD.f = meD.f - accumarray(l,[fi(n*meD.DoF-1,:)';...
                                  fi(n*meD.DoF  ,:)'],[meD.nDoF(2) 1])    ;%
end                                                                        %
%% SOLVE EXPLICIT MOMENTUM BALANCE EQUATION
% UPDATE GLOBAL NODAL INFORMATIONS
iDx        = 1:meD.DoF:meD.nDoF(2)-1                                      ;%
iDy        = iDx+1                                                        ;%
% COMPUTE GLOBAL NODAL FORCE
meD.d(iDx) = sqrt(meD.f(iDx).^2+meD.f(iDy).^2)                            ;%
meD.d(iDy) = meD.d(iDx)                                                   ;%
meD.f      = meD.f - meD.vd*meD.d.*sign(meD.p)                            ;%
% UPDATE GLOBAL NODAL MOMENTUM
meD.p      = meD.p + dt*meD.f                                             ;%
% COMPUTE GLOBAL NODAL ACCELERATION AND VELOCITY
meD.mr     = reshape(repmat(meD.m',meD.DoF,1),meD.nDoF(2),1)              ;%
iD         = meD.mr==0                                                    ;%
meD.a      = meD.f./meD.mr                                                ;%
meD.v      = meD.p./meD.mr                                                ;%
meD.a(iD)  = 0.0                                                          ;%
meD.v(iD)  = 0.0                                                          ;%
% BOUNDARY CONDITIONS: FIX DIRICHLET BOUNDARY CONDITIONS
meD.a(bc.x(:,1))=bc.x(:,2)                                                ;%
meD.a(bc.y(:,1))=bc.y(:,2)                                                ;%
meD.v(bc.x(:,1))=bc.x(:,2)                                                ;%
meD.v(bc.y(:,1))=bc.y(:,2)                                                ;%
end
```

Figure 5.5. | Code Fragment 2 shows the vectorised solution to the nodal projection of material point quantities (e.g., mass and momentum) within the local function `p2Nsolve.m`. The core of the vectorization process is the extensive use of the built-in function of MATLAB© `accumarray( )`, for which we detail the main features in the text. Table 5.3 lists the variables used.

To calculate the temporary vector of internal forces (`fi` at Lines 15-17 in Fig. 5.5), the first step consists of the matrix multiplication of the strain-displacement matrix `mpD.B` with the material point stress vector `mpD.s`. The vectorised solution is given by i) element-wise multiplications of `mpD.B` with a replication of the transposed stress vector `repmat(reshape(mpD.s,size(mpD.s,1),1,mpD.n),1,meD.` whose result is then ii) summed by means of the built-in function `sum( )` along the columns and, finally multiplied by a replicated transpose of the material point volume vector, e.g., `repmat(mpD.V',meD.nDoF(1),1)`.

To illustrate the numerical efficiency of the vectorised multiplication between a matrix and a vector, we have developed an iterative and vectorised solution of $\boldsymbol{B}(\boldsymbol{x}_p)^T\boldsymbol{\sigma}_p$ with an increasing n_p and considering single (4 bytes) and double (8 bytes) arithmetic precision. The wall-clock time increases with n_p with a sharp transition for the vectorised solution around $n_p \approx 1000$, as showed in Fig 5.6a. The mathematical operation requires more memory than available in the L2 cache (1024 kB under the CPU architecture used), which inhibits cache reuse (Dabrowski et al., 2008). A peak performance of at least 1000 Mflops, showed in Fig. 5.6b, is achieved when $n_p = 1327$ or $n_p = 2654$ for simple or double arithmetic precision respectively, i.e., it corresponds exactly to 1024 kB for both precisions. Beyond, the performance dramatically drops to approximately the half of the peak value. This drop is even more severe for a double arithmetic precision.

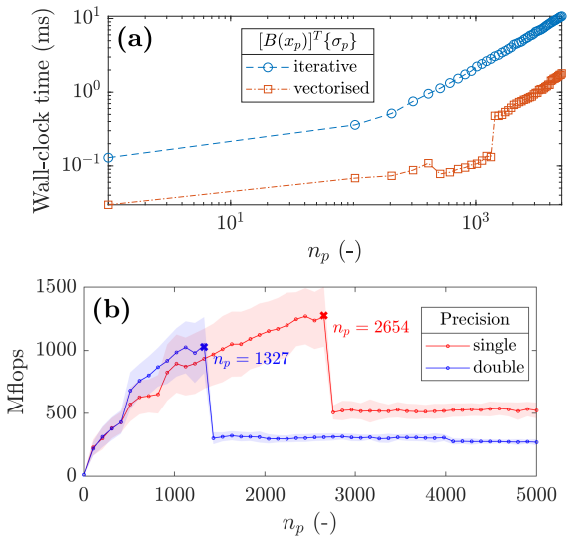


Figure 5.6. | a) Wall-clock time to solve for a matrix multiplication between a multidimensional array and a vector with an increasing number of the third dimension with a double arithmetic precision and, b) number of floating point operations per second (flops) for single and double arithmetic precisions. The continuous line represents the averages value whereas the shaded area denotes the standard deviation.

Update of material point properties

Finally, we propose a vectorisation of the function `mapN2p.m` that i) interpolates updated nodal solutions to the material points (velocities and coordinates) and ii) the double mapping (DM or MUSL) procedure (see Fern et al. 2019). The material point velocity $\boldsymbol{v}_p$ is defined as an interpolation of the solution of the updated nodal accelerations, given by:

$$\boldsymbol{v}_p^{t+\Delta t} = \boldsymbol{v}_p^t + \Delta t \sum_{n=1}^{n_n} S_n(\boldsymbol{x}_p) \boldsymbol{a}_n^{t+\Delta t}. \tag{5.3.21}$$

The material point updated momentum is found by $\boldsymbol{p}_p^{t+\Delta t} = m_p \boldsymbol{v}_p^{t+\Delta t}$. The double mapping procedure of the nodal velocity $\boldsymbol{v}_n$ consists of the remapping of the updated material point momentum on the mesh, divided by the nodal mass, given as:

$$\boldsymbol{v}_n^{t+\Delta t} = m_n^{-1} \sum_{p \in n} S_n(\boldsymbol{x}_p) \boldsymbol{p}_p^{t+\Delta t}, \tag{5.3.22}$$

and for which boundary conditions are enforced. Finally, the material point coordinates are updated based on the following:

$$\boldsymbol{x}_p^{t+\Delta t} = \boldsymbol{x}_p^t + \Delta t \sum_{n=1}^{n_n} S_n(\boldsymbol{x}_p) \boldsymbol{v}_n^{t+\Delta t}. \tag{5.3.23}$$

To solve for the interpolation of updated nodal solutions to the material points, we rely on a combination of element-wise matrix multiplication between the array of basis functions `mpD.S` with the global vectors through a transform of the `p2N` array, i.e., `iDx=meD.DoF*p2N-1` and `iDy=iDx+1` (Lines 3-4 in Code Fragment 3 in Fig. 5.7), which are used to access to x and y components of global vectors.

When accessing global nodal vectors by means of `iDx` and `iDy`, the resulting arrays are naturally of the same size as `p2N` and are therefore dimension-compatible with `mpD.S`. For instance, a summation along the columns (e.g., the associated nodes of material points) of an element-wise multiplication of `mpD.S` with `meD.a(iDx)` results in an interpolation of the x-component of the global acceleration vector to the material points.

This procedure is used for the velocity update (Line 6 in Fig. 5.7) and for the material point coordinate update (Line 10 in Fig. 5.7). A remapping of the nodal momentum is carried out (Lines 12 to 16 in Fig. 5.7), which allows calculating the updated nodal incremental displacements (Line 17 in Fig. 5.7). Finally, boundary conditions of nodal incremental displacements are enforced (Lines 21-22 in Fig. 5.7).

5.3.4. Initial settings and adaptive time step

Regarding the initial setting of the background mesh of the demonstration cases further presented, we select a uniform mesh and a regular distribution

```
function [meD,mpD] = mapN2p(meD,mpD,dt,l2g,p2N,bc)
%% INTERPOLATE SOLUTIONS N to p
iDx         = meD.DoF*p2N-1                                                 ;%
iDy         = iDx+1                                                         ;%
% VELOCITY UPDATE
mpD.v       = mpD.v+dt*[sum(mpD.S.*meD.a(iDx),2) sum(mpD.S.*meD.a(iDy),2)] ;%
% MOMENTUM UPDATE
mpD.p       = mpD.v.*repmat(mpD.m,1,meD.DoF)                                ;%
%% UPDATE NODAL MOMENTUM WITH UPDATED MP MOMENTUM (MUSL OR DOUBLE MAPPING)
meD.p(:) = 0.0                                                              ;%
p           = reshape([mpD.S.*repmat(mpD.p(:,1),1,meD.nNe) ;...
                       mpD.S.*repmat(mpD.p(:,2),1,meD.nNe)],...
                       mpD.n*meD.nDoF(1),1                  )                ;%
meD.p       = accumarray(l2g(:),p,[meD.nDoF(2) 1])                          ;%
meD.u       = dt*(meD.p./meD.mr)                                            ;%
iD          = meD.mr==0                                                     ;%
meD.u(iD)= 0.0                                                              ;%
%% BOUNDARY CONDITIONS: FIX DIRICHLET BOUNDARY CONDITIONS
meD.u(bc.x(:,1))=bc.x(:,2)                                                  ;%
meD.u(bc.y(:,1))=bc.y(:,2)                                                  ;%
%% UPDATE COORDINATE AND DISPLACEMENT
mpD.x       = mpD.x+[sum(mpD.S.*meD.u(iDx),2) sum(mpD.S.*meD.u(iDy),2)]     ;%
mpD.u       = mpD.u+[sum(mpD.S.*meD.u(iDx),2) sum(mpD.S.*meD.u(iDy),2)]     ;%
end
```

Figure 5.7. | Code Fragment 3 shows the vectorised solution for the interpolation of nodal solutions to material points with a double mapping procedure (or MUSL) within the function `mapN2p.m`.

of material points within the initially populated elements of the mesh. Each element is evenly filled with 4 material points, e.g., $n_{pe} = 2^2$, unless otherwise stated.

In this contribution, Dirichlet boundary conditions are resolved directly on the background mesh, as in the standard finite element method. This implies that boundary conditions are resolved only in contiguous regions between the mesh and the material points. Deviating from this contiguity or having the mesh not aligned with the coordinate system requires specific treatments for boundary conditions (Cortis et al., 2018). Furthermore, we ignore the external tractions as their implementation is complex.

As explicit time integration is only conditionally stable, any explicit formulation requires a small time step Δt to ensure numerical stability (Ni et al., 2020), e.g., smaller than a critical value defined by the Courant-Friedrich-Lewy (CFL) condition. Hence, we employ an adaptive time step (de Vaucorbeil et al., 2020), which considers the velocity of the material points. The first step is to compute the maximum wave speed of the material using (Zhang et al., 2017; Anderson, 1987)

$$(c_x, c_y) = \left(\max_p (V + \mid (v_x)_p \mid), \max_p (V + \mid (v_y)_p \mid) \right), \tag{5.3.24}$$

where the wave speed is $V = ((K+4G/3)/\rho)^{\frac{1}{2}}$, K and G are the bulk and shear moduli respectively, ρ is the material density, $(v_x)_p$ and $(v_y)_p$ are the material point velocity components. Δt is then restricted by the CFL condition as followed:

$$\Delta t = \alpha \min \left(\frac{h_x}{c_x}, \frac{h_y}{c_y} \right), \tag{5.3.25}$$

where $\alpha \in [0; 1]$ is the time step multiplier, and h_x and h_y are the mesh spacings.

5.4. Results

In this section, we first demonstrate our MATLAB-based MPM solver to be efficient in reproducing results from other studies, i.e., the compaction of an elastic column (Coombs et al., 2020b) (e.g., quasi-static analysis), the cantilever beam problem (Sadeghirad et al., 2011) (e.g., large elastic deformation) and an application to landslide dynamics (Huang et al., 2015) (e.g., elasto-plastic behaviour). Then, we present both the efficiency and the numerical performances for a selected case, e.g., the elasto-plastic collapse. We conclude and compare the performances of the solver with respect to the specific case of an impact of two elastic disks previously implemented in a Julia language environment by (Sinaie et al., 2017).

Regarding the performance analysis, we investigate the performance gain of the vectorised solver considering a double arithmetic precision with respect to the total number of material point because of the following reasons: i) the mesh resolution, i.e., the total number of elements n_{el}, influences the wall-clock time of the solver by reducing the time step due to the CFL condition hence increasing the total number of iterations. In addition, ii) the total number of material points n_p increases the number of operations per cycle due to an increase of the size of matrices, i.e., the size of the strain-displacement matrix depends on n_p and not on n_{el}. Hence, n_p consistently influences the performance of the solver whereas n_{el} determines the wall-clock time of the solver. The performance of the solver is addressed through both the number of floating point operations per second (flops), and by the average number of iteration per second (it/s). The number of floating point operations per second was manually estimated for each function of the solver.

5.4.1. Validation of the solver and numerical efficiency

Convergence: elastic compaction under self-weight of a column

Following the convergence analysis proposed by Coombs et al., 2020a; Wang et al., 2019; Charlton et al., 2017, we analyse an elastic column of an initial height $l_0 = 10$ m subjected to an external load (e.g. the gravity). We selected the cpGIMPM variant with a domain update based on the diagonal components of the deformation gradient. Coombs et al., 2020b showed that such domain update is well suited for hydrostatic compression problems. We also selected the CPDI2q variant as a reference, because of its superior convergence accuracy for such problem compared to GIMPM (Coombs et al., 2020b).

The initial geometry is shown in Fig. 5.8. The background mesh is made of a bi-linear four-noded quadrilaterals, and roller boundary conditions are applied on the base and the sides of the column, initially populated by 4 material points per element. The column is 1 element wide and n elements

tall and, the number of element in the vertical direction is increased from 1 to a maximum of 1280 elements. The time step is adaptive and we selected a time step multiplier of $\alpha = 0.5$, e.g., minimal and maximal time step values of $\Delta t_{\min} = 3.1 \cdot 10^{-4}$ s and $\Delta t_{\max} = 3.8 \cdot 10^{-4}$ s respectively for the finest mesh of 1280 elements.

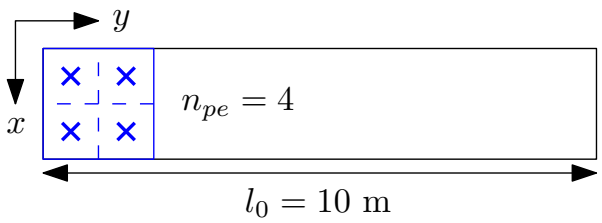


Figure 5.8. | Initial geometry of the column.

To consistently apply the external load for the explicit solver, we follow the recommendation of Bardenhagen et al., 2004, i.e., a quasi-static solution (given an explicit integration scheme is chosen) is obtained if the total simulation time is equal to 40 elastic wave transit times. The material has a Young's modulus $E = 1 \cdot 10^4$ Pa and a Poisson's ratio $\nu = 0$ with a density $\rho = 80$ kg m^{-3}. The gravity g is increased from 0 to its final value, i.e., $g = 9.81$ m s^{-2}. We performed additional implicit quasi-static simulations (named iCPDI2q) in order to consistently discuss the results with respect to what was reported in Coombs et al., 2020a. The external force is consistently applied over 50 equal load steps. The vertical normal stress is given by the analytical solution (Coombs et al., 2020a) $\sigma_{yy}(y_0) = \rho g(l_0 - y_0)$, where l_0 is the initial height of the column and y_0 is the initial position of a point within the column.

The error between the analytical and numerical solutions is as follows:

$$\text{error} = \sum_{p=1}^{n_p} \frac{||(\sigma_{yy})_p - \sigma_{yy}(y_p)||(V_0)_p}{(\rho g l_0) V_0}, \tag{5.4.1}$$

where $(\sigma_{yy})_p$ is the stress along the y-axis of a material point p (Fig. 5.8) of an initial volume $(V_0)_p$ and V_0 is the initial volume of the column, i.e., $V_0 = \sum_{p=1}^{n_p} (V_0)_p$.

The convergence toward a quasi-static solution is shown in Fig. 5.9 (a). Is is quadratic for both cpGIMPM and CPDI2q, but contrary to Coombs et al., 2020b; Coombs et al., 2020a who reported a full convergence, it stops at error $\approx 2 \cdot 10^{-6}$ for the explicit implementation. This was already outlined by Bardenhagen et al., 2004 as a saturation of the error caused by resolving the dynamic stress wave propagation, which is inherent to any explicit scheme. Hence, a static solution could never be achieved because, unlike quasi-static implicit methods, the elastic waves propagate indefinitely and the static equilibrium is never resolved. This is consistent when compared to the iCPDI2q solution we implemented, whose behaviour is still converging below the limit error $\approx 2 \cdot 10^{-6}$ reached by the explicit solver. However, the convergence rate

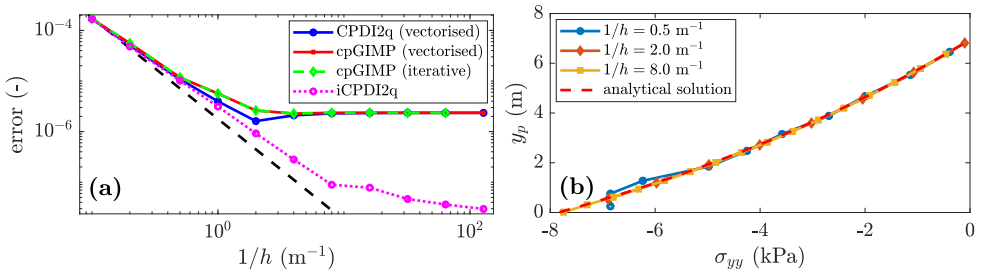


Figure 5.9. | a) Convergence of the error: a limit is reached at error $\approx 2 \cdot 10^{-6}$ for the explicit solver, whereas the quasi-static solution still converges. This was already demonstrated in Bardenhagen et al., 2004 as an error saturation due to the explicit scheme, i.e., the equilibrium is never resolved. b) The stress σ_{yy} along the y-axis predicted at the deformed position y_p by the CPDI2q variant is in good agreements with the analytical solution for a refined mesh.

of the implicit algorithm decreases as the mesh resolution increases. We did not investigate this since our focus is on the explicit implementation. The vertical stresses of material points are in good agreements with the analytical solution (see Fig. 5.9 b)). Some oscillations are observed for a coarse mesh resolution but these rapidly decrease as the mesh resolution increases.

We finally report the wall-clock time for the cpGIMPM (iterative), cpGIMPM (vectorised) and the CPDI2q (vectorised) variants. As claimed by Sadeghirad et al., 2013; Sadeghirad et al., 2011, the CPDI2q variant induces no significant computational cost compared to the cpGIMPM variant. However, the absolute value between vectorised and iterative implementations is significant. For $n_p = 2560$, the vectorised solution completed in 1161 s whereas the iterative solution completed in 52'856 s. The vectorised implementation is roughly 50 times faster than the iterative implementation.

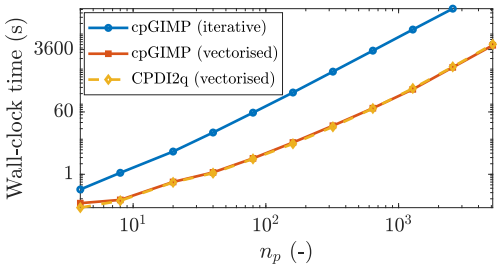


Figure 5.10. | Wall-clock time for cpGIMPM (vectorised and iterative solutions) and the CPDI2q solution with respect to the total number of material points n_p. There is no significant differences between CPDI2q and cpGIMPM variants regarding the wall-clock time. The iterative implementation is also much slower than the vectorised implementation.

Large deformation: the elastic cantilever beam problem

The cantilever beam problem (Sinaie et al., 2017; Sadeghirad et al., 2011) is the second benchmark which demonstrates the robustness of the MPM solver. Two MPM variants are implemented, namely, i) the contiguous GIMPM (cpGIMPM) which relies on the stretch part of the deformation gradient (see Charlton et al. 2017) to update the particle domain since large rotations are expected during the deformation of the beam, and ii) the convected particle domain interpolation (CPDI, Leavy et al. 2019; Sadeghirad et al. 2011). We selected the CPDI variant since it is more suitable to large torsional deformation modes (Coombs et al., 2020b) than the CPDI2q variant. Two constitutive elastic models are selected, i.e., neo-Hookean Guilkey et al., 2003 or linear elastic York II et al., 1999 solids. For consistency, we use the same physical quantities as in Sadeghirad et al., 2011, i.e., an elastic modulus $E = 10^6$ Pa, a Poisson's ratio $\nu = 0.3$, a density $\rho = 1050$ kg/m^3, the gravity $g = 10.0$ m/s and a real-time simulation $t = 3$ s with no damping forces introduced.

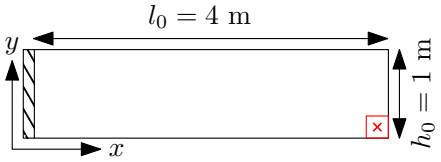


Figure 5.11. | Initial geometry for the cantilever beam problem; the free end material point appears in red where a red cross marks its centre.

The beam geometry is depicted in Fig. 5.11 and is discretized by 64 four-noded quadrilaterals, each of them initially populated by 9 material points (e.g., $n_p = 576$) with a adaptive time step determined by the CFL condition, i.e., the time step multiplier is $alpha = 0.1$, which yields minimal and maximal time step values of $\Delta t_{\min} = 5.7 \cdot 10^{-4}$ s and $\Delta t_{\max} = 6.9 \cdot 10^{-4}$ s respectively. The large deformation is initiated by suddenly applying the gravity at the beginning of the simulation, i.e., $t = 0$ s.

As indicated in Sadeghirad et al., 2011, the cpGIMPM simulation failed when using the diagonal components of the deformation gradient to update the material point domain, i.e., the domain vanishes under large rotations as stated in (Coombs et al., 2020b). However and as as expected, the cpGIMPM simulation succeeded when using the diagonal terms of the stretch part of the deformation gradient, as proposed by Coombs et al., 2020b; Charlton et al., 2017. The numerical solutions, obtained by the latter cpGIMPM and CPDI, to the vertical deflection Δu of the material point at the bottom free end of the beam (e.g., the red cross in Fig. 5.11) are shown in Fig. 5.12. Some comparative results reported by Sadeghirad et al., 2011 are depicted by black markers (squares for the FEM solution and circles for the CPDI solution), whereas the results of the solver are depicted by lines.

The local minimal and the minimal and maximal values (in timing and magnitude) are in agreement with the FEM solution of Sadeghirad et al., 2011. The

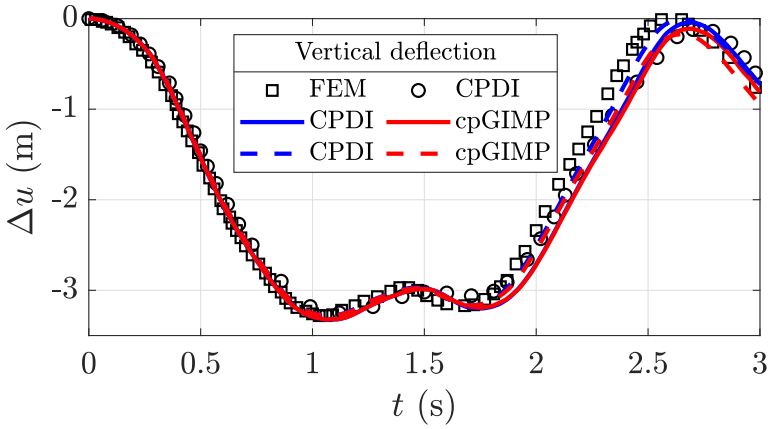


Figure 5.12. | Vertical deflection Δu for the cantilever beam problem. The black markers denote the solutions of Sadeghirad et al., 2011 (circles for CPDI and squares for FEM). The line colour indicates the MPM variant (blue for CPDI and red for cpGIMP), solid lines refer to a linear elastic solid, whereas dashed lines refer to a neo-Hookean solid. Δu corresponds to the vertical displacement of the bottom material point at the free end of the beam (the red cross in Fig. 5.11).

elastic response is in agreement with the CPDI results reported by Sadeghirad et al., 2011 but, it differs in timing with respect to the FEM solution. This confirms our numerical implementation of CPDI when compared to the one proposed by Sadeghirad et al., 2011. In addition, the elastic response does not substantially differ from a linear elastic solid to a neo-Hookean one. It demonstrates the incremental implementation of the MPM solver to be relevant in capturing large elastic deformations for the cantilever beam problem.

Figure 5.13 shows the finite deformation of the material point domain, i.e., a) or c), and, the vertical Cauchy stress field, i.e., b) or d), for CPDI and cpGIMPM. The stress oscillations due to the cell-crossing error are partially cured when using a domain-based variant compared to the standard MPM. However, spurious vertical stresses are more developed in Fig. 5.13 (d) compared to Fig. 5.13 (b) where the vertical stress field appears even smoother. Both CPDI and cpGIMPM give a decent representation of the actual geometry of the deformed beam.

We also report a quite significant difference in execution time between the CPDI variant compared to the CPDI2q and cpGIMPM variants, i.e., CPDI executes in an average 280.54 it/s whereas both CPDI2q and CPGIMPM execute in an average 301.42 it/s and an average 299.33 it/s, respectively.

Application: the elasto-plastic slumping dynamics

We present an application of the MPM solver (vectorised and iterative version) to the case of landslide mechanics. We selected the domain-based CDPI variant since it performs better than the CPDI2q variant in modelling torsional and

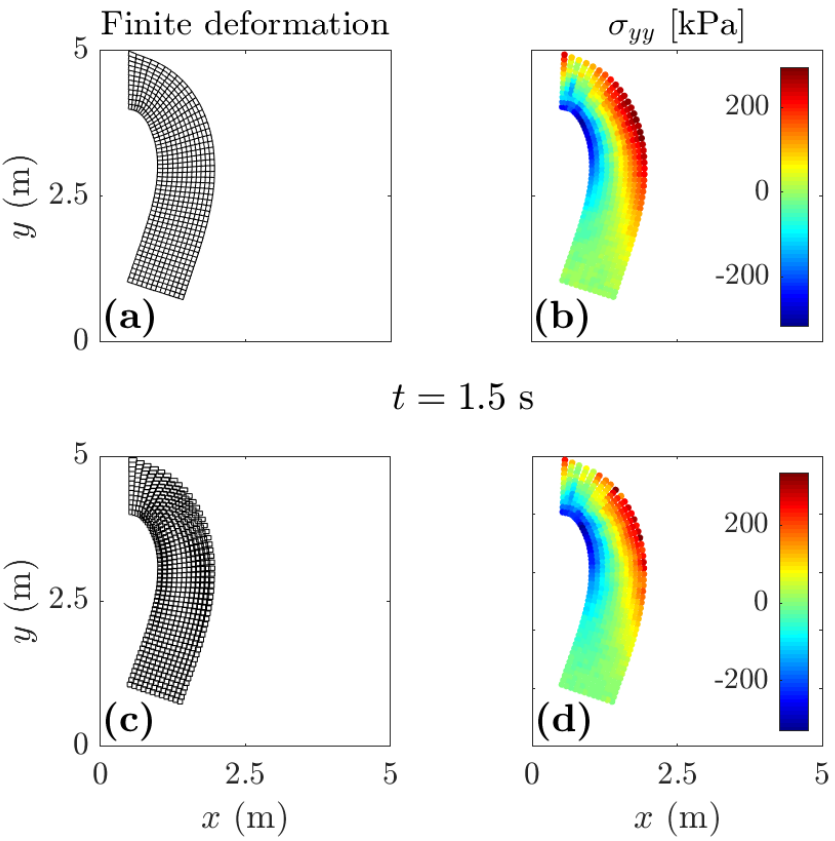


Figure 5.13. | Finite deformation of the material point domain and vertical Cauchy stress σ_{yy} for CPDI, i.e., a) & b), and for cpGIMPM, i.e., c) & d). The CPDI variant gives a better and contiguous description of the material point's domain and a slightly smoother stress field, compared to the cpGIMPM variant, which is based on the stretch part of the deformation gradient.

stretch deformation modes (Wang et al., 2019) coupled to an elasto-plastic constitutive model based on a non-associated Mohr-Coulomb (M-C) plasticity (Simpson, 2017). We i) analyse the geometrical features of the slump and, ii) compare the results (the geometry and the failure surface) to the numerical simulation of Huang et al., 2015, which is based on a Drucker-Prager model with tension cut-off (D-P).

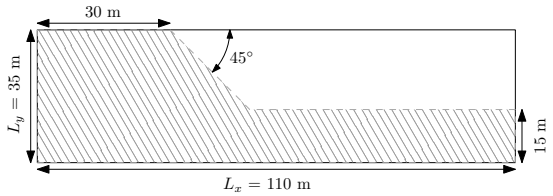


Figure 5.14. | Initial geometry for the slump problem from Huang et al., 2015. Roller boundary conditions are imposed on the left and right of the domain while a no-slip condition is enforced at the base of the material.

The geometry of the problem is shown in Fig. 5.14, the soil material is discretized by 110×35 elements with $n_{pe} = 9$, resulting in $n_p = 21'840$ material points. A uniform mesh spacing $h_{x,y} = 1$ m is used and, rollers are imposed at the left and right domain limits while a no-slip condition is enforced at the base of the material. We closely follow the numerical procedure proposed in Huang et al., 2015, i.e., no local damping is introduced in the equation of motion and the gravity is suddenly applied at the beginning of the simulation. As in Huang et al., 2015, we consider an elasto-plastic cohesive material of density $\rho = 2100$ kg·m^3, with an elastic modulus $E = 70$ MPa and a Poisson's ratio $\nu = 0.3$. The cohesion is $c = 10$ Pa, the internal friction angle is $\phi = 20°$ with no dilatancy, i.e., the dilatancy angle is $\psi = 0$. The total simulation time is 7.22 s and, we select a time step multiplier $\alpha = 0.5$. The adaptive time steps (considering the elastic properties and the mesh spacings $h_{x,y} = 1$ m) yield minimal and maximal values $\Delta t_{\min} = 2.3 \cdot 10^{-3}$ s and $\Delta t_{\max} = 2.4 \cdot 10^{-3}$ s respectively.

The numerical solution to the elasto-plastic problem is shown in Fig. 5.15. An intense shear zone, highlighted by the second invariant of the accumulated plastic strain ϵ_{II}, develops at the toe of the slope as soon as the material yields and propagates backwards to the top of the material. It results in a rotational slump. The failure surface is in good agreement with the solution reported by Huang et al., 2015 (continuous and discontinuous red lines in Fig. 5.15) but, we also observe differences, i.e., the crest of the slope is lower compared to the original work of Huang et al., 2015. This may be explained by the problem of spurious material separation when using sMPM or GIMPM (Sadeghirad et al., 2011), the latter being overcome with the CPDI variant, i.e., the crest of the slope experiences considerable stretch deformation modes. Despite some differences, our numerical results appear coherent with those reported by Huang et al., 2015.

The vectorised and iterative solutions are resolved within approximately

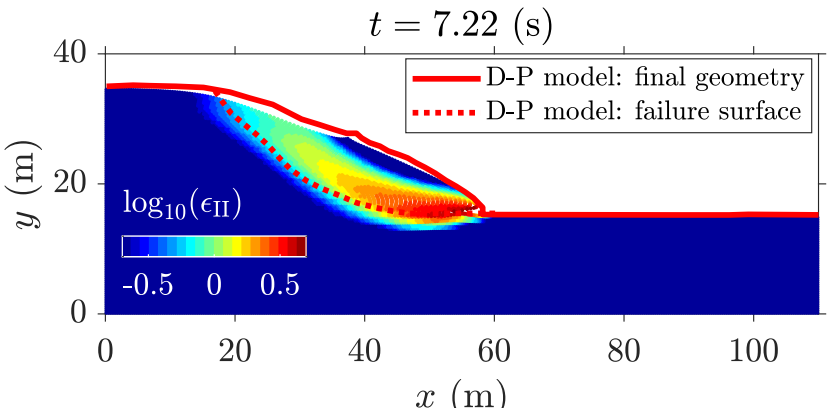


Figure 5.15. | MPM solution to the elasto-plastic slump. The red lines indicate the numerical solution of Huang et al., 2015 and, the coloured points indicate the second invariant of the accumulated plastic strain ϵ_{II} obtained by the CPDI solver. An intense shear zone progressively develops backwards from the toe of the slope, resulting in a circular failure mode.

630 s (a wall-clock time of $\approx$ 10 min. and an average 4.20 it/s) and 14'868 s (a wall-clock time of $\approx$ 4.1 hrs. and an average 0.21 it/s) respectively. This corresponds to a performance gain of 23.6. The performance gain is significant between an iterative and a vectorised solver for this problem.

5.4.2. Computational performance

Iterative and vectorised elasto-plastic collapses

We evaluate the computational performance of the solver, using the MATLAB version R2018a on an Intel Core i7-4790, with a benchmark based on the elasto-plastic collapse of the aluminium-bar assemblage, for which numerical and experimental results were initially reported by Bui et al., 2008 and Huang et al., 2015 respectively.

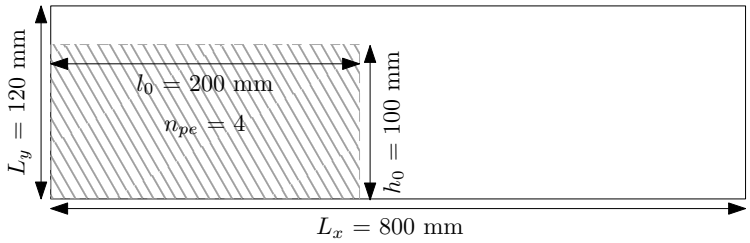


Figure 5.16. | Initial geometry for the elasto-plastic collapse (Huang et al., 2015). Roller boundaries are imposed on the left and right boundaries of the computational domain while a no-slip condition is enforced at the bottom of the domain. The aluminium-bar assemblage has dimensions of $l_0 \times h_0$ and is discretized by $n_{pe} = 4$ material points per initially populated element.

We vary the number of elements of the background mesh, which results in a variety of different regular mesh spacings $h_{x,y}$. The number of elements along the x- and y- directions are $n_{el,x} = [10, 20, 40, 80, 160, 320, 640]$ and $n_{el,y} = [1, 2, 5, 11, 23, 47, 95]$ respectively. The number of material points per element is kept constant, i.e., $n_{pe} = 4$, and this yields a total number of material points $n_p = [10, 50, 200, 800, 3'200, 12'800, 51'200]$. The initial geometry and boundary conditions used for this problem are depicted in Fig. 5.16. The total simulation time is 1.0 s and, the time step multiplier is $\alpha = 0.5$. Accordingly to Huang et al., 2015, the gravity $g = 9.81$ m·s^{-2} is applied to the assemblage and, no damping is introduced. We consider a non-cohesive granular material (Huang et al., 2015) of density $\rho = 2650$ kg·m^3, with a bulk modulus $K = 0.7$ MPa and a Poisson's ratio $\nu = 0.3$. The cohesion is $c = 0$ Pa, the internal friction angle is $\phi = 19.8°$ and there is no dilatancy, i.e., $\psi = 0$.

We conducted preliminary investigations using either uGIMPM or cpGIMPM variants, the latter with a domain update based either on the determinant of the deformation gradient or on the diagonal components of the stretch part of the deformation gradient. We concluded the uGIMPM was the most reliable, even tough its suitability is restricted to both simple shear and pure rotation deformation modes (Coombs et al., 2020b).

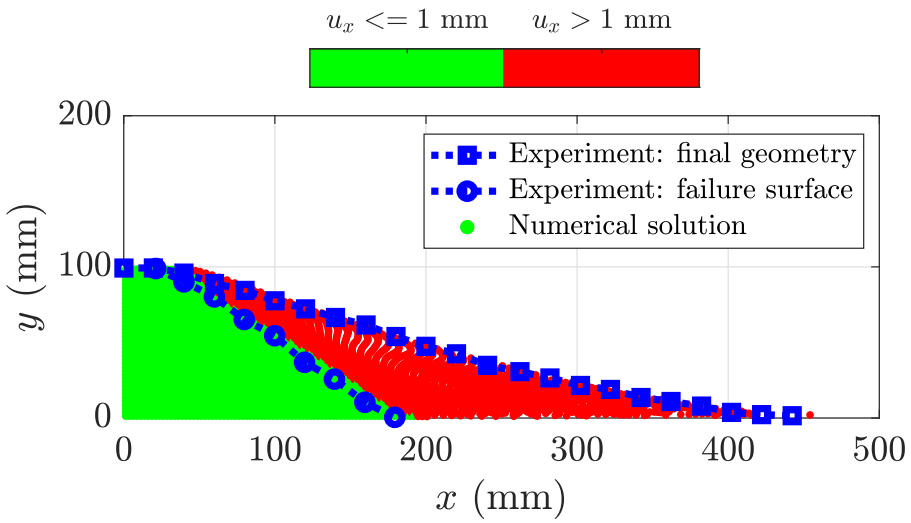


Figure 5.17. | Final geometry of the collapse: in the intact region (horizontal displacement $u_x < 1$ mm), the material points are coloured in green, whereas in the deformed region (horizontal displacement $u_x > 1$ mm), they are coloured in red and indicate plastic deformations of the initial mass. The transition between the deformed and undeformed region marks the failure surface of the material. Experimental results of (Bui et al., 2008) are depicted by the blue dotted lines. The computational domain is discretized by a background mesh made of 320 × 48 quadrilateral elements with $n_p = 4$ per initially populated element, i.e., a total $n_p = 12'800$ material points discretize the aluminium assemblage.

We observe a good agreement between the numerical simulation and the

experiments (see Fig. 5.17), considering either the final surface (blue square dotted line) or the failure surface (blue circle dotted line). The repose angle in the numerical simulation is approximately 13°, which is in agreements with the experimental data reported by Bui et al., 2008, e.g., they reported a final angle of 14°.

The vectorised and iterative solutions (for a total of $n_p = 12'800$ material points) are resolved within approximately 1595 s (a wall-clock time of ≈ 0.5 hrs. and an average 10.98 it/s) and 43'861 s (a wall-clock time of ≈ 12 hrs. and an average 0.38 it/s) respectively. This corresponds to a performance gain of 28.24 for a vectorised code over an iterative code to solve this elasto-plastic problem.

The performance of the solver is demonstrated in Fig. 5.18. A peak performance of ≈ 900 Mflops is reached, as soon as n_p exceeds 1000 material points and, a residual performance of ≈ 600 Mflops is further resolved (for $n_p \approx 50'000$ material points). Every functions provide an even and fair contribution on the overall performance, except the function `constitutive.m` for which the performance appears delayed or shifted. First of all, this function treats the elasto-plastic constitutive relation, in which the dimensions of the matrices are smaller when compared to the other functions. Hence, the amount of floating point operations per second is lower compared to other functions, e.g., `p2Nsolve.m`. This results in less performance for an equivalent number of material points. It also requires a greater number of material points to increase the dimensions of the matrices in order to exceed the L2 cache maximum capacity.

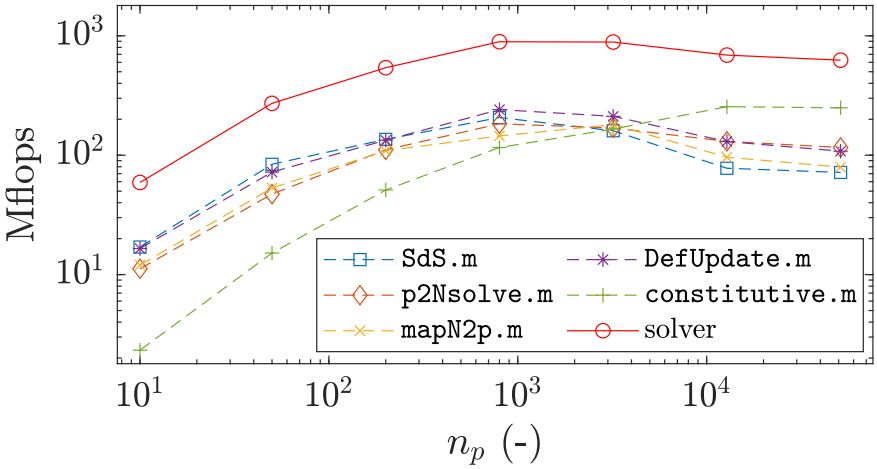


Figure 5.18. | Number of floating point operation per seconds (flops) with respect to the total number of material point n_p for the vectorised implementation. The discontinuous lines refer to the functions of the solver, whereas the continuous line refer to the solver. A peak performance of 900 Mflops is reached by the solver for $n_p > 1000$ and, a residual performance of 600 Mflops is further resolved for an increasing n_p.

This considerations provide a better understanding of the performance gain of the vectorised solver showed in Fig. 5.19: the gain increases and then, reaches a plateau and ultimately, decreases to a residual gain. This is directly related to the peak and the residual performances of the solver showed in Fig. 5.18.

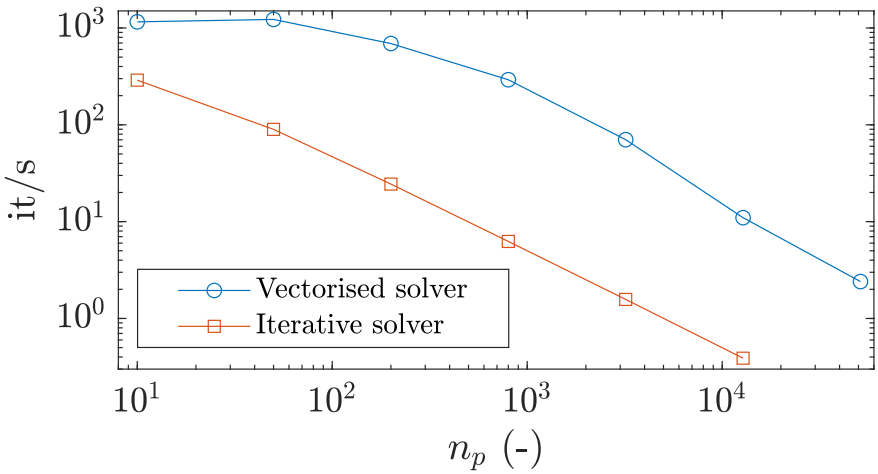


Figure 5.19. | Number of iterations per second with respect to the total number of material point n_p. The greatest performance gain is reached around $n_p = 1000$, which is related to the peak performance of the solver (see Fig. 5.18). The gains corresponding to the peak and residual performances are 46 and 28 respectively.

Comparison between Julia and MATLAB

We compare the computational efficiency of the vectorised CPDI2q MATLAB implementation and the computational efficiency reported by Sinaie et al., 2017 of a Julia-based implementation of the collision of two elastic disks problem. However, we note a difference between the actual implementation and the one used by Sinaie et al., 2017; the latter is based on a USL variant with a cut-off algorithm, whereas the present implementation relies on the MUSL (or double mapping) procedure, which necessitates a double mapping procedure. The initial geometry and parameters are the same as those used in Sinaie et al., 2017. However, the time step is adaptive and, we select a time step multiplier $\alpha = 0.5$. Given the variety of mesh resolution, we do not present minimal and maximal time step values.

Our CPDI2q implementation, in MATLAB R2018a, is, at least, 2.8 times faster than the Julia implementation proposed by Sinaie et al., 2017 for similar hardware (see Table 5.1). Sinaie et al., 2017 completed the analysis with an Intel Core i7-6700 (4 cores with a base frequency of 3.40 GHz up to a turbo frequency of 4.00 GHz) with 16 GB RAM, whereas we used an Intel Core i7-4790 with similar specifications (see Section 5.2). However, the performance ratio between MATLAB and Julia seems to decrease as the mesh resolution

increases.

Table 5.1. | Efficiency comparison of the Julia implementation of Sinaie et al., 2017, and the MATLAB-based implementation for the two elastic disk impact problems.

mesh	n_{pe}	n_p	Its/s		
			Julia	MATLAB	Gain
20×20	2^2	416	132.80	450.27	3.40
20×20	4^2	1'624	33.37	118.45	3.54
40×40	2^2	1'624	26.45	115.59	4.37
80×80	4^2	25'784	1.82	5.21	2.86

5.5. Discussion

In this contribution, a fast and efficient explicit MPM solver is proposed that considers two variants (e.g., the uGIMPM/cpGIMPM and the CPDI/CPDI2q variants).

Regarding the compression of the elastic column, we report a good agreement of the numerical solver with previous explicit MPM implementations, such as Bardenhagen et al., 2004. The same flaw of an explicit scheme is also experienced by the solver, i.e., a saturation of the error due to the specific usage of an explicit scheme that resolves the wave propagation, thus preventing any static equilibrium to be reached. This confirms that our implementation is consistent with previous MPM implementations. However, the implicit implementation suffers from a decrease of the convergence rate for a fine mesh resolution. Further work would be needed to investigate this decrease of convergence rate. This case also demonstrated that cpGIMPM and CPDI variants have a similar computational cost and, this confirms the suitability of cpGIMPM with respect to CPDI, as previously mentioned by Coombs et al., 2020b; Charlton et al., 2017.

For the cantilever beam, we report a good agreement of the solver with the results of Sadeghirad et al., 2011, i.e., we report the vertical deflection of the beam to be very close in both magnitude and timing (for the CPDI variant) to the FEM solution. However, we also report a slower execution time for the CPDI variant when compared to both cpGIMPM and CPDI2q variants.

The elasto-plastic slump also demonstrates the solver to be efficient in capturing complex dynamics in the field of geomechanics. The CDPI solution showed that the algorithm proposed by Simpson, 2017 to return stresses when the material yields is well suited to the slumping dynamics. However and as mentioned by Simpson, 2017, such return mapping is only valid under the assumption of a non-associated plasticity with no volumetric plastic strain. This particular case of isochoric plastic deformations rises the issue of volumetric locking. In the actual implementation, no regularization techniques are con-

sidered. As a result, the pressure field experience severe locking for isochoric plastic deformations. One way to overcome locking phenomenons would be to implement the regularization technique initially proposed by Coombs et al., 2018 for quasi-static sMPM and GIMPM implementations.

Regarding the elasto-plastic collapse, the numerical results demonstrate the solver to be in agreement with both previous experimental and numerical results (Huang et al., 2015; Bui et al., 2008). This confirms the ability of the solver to address elasto-plastic problems. However, the choice of whether to update or not the material point domain remains critical. Such question remains open and would require a more thorough investigation of the suitability of each of these domain updating variants. Nevertheless, the uGIMPM variant is a good candidate since, i) it is able to reproduce the experimental results of Bui et al., 2008 and, ii) it ensures numerical stability. However, one has to keep in mind its limited range of suitability regarding the deformation modes involved. If a cpGIMPM is selected, the splitting algorithm proposed in Gracia et al., 2019; Homel et al., 2016 could be implemented to mitigate the amount of distortion experienced by the material point domains during deformation. We did not selected the domain updating method based on the corners of the domain as suggested in Coombs et al., 2020b. This is because such domain updating method necessitates to calculate additional shape functions between the corners of the domain of the material point with their associated nodes. This results in an additional computational cost. Nevertheless, such variant is of interest and should be addressed as well when the computational performances are not the main concern.

The computational performance comes from the combined use of the connectivity array `p2N` with the built-in function `accumarray( )` to i) accumulate material point contributions to their associated nodes or, ii) to interpolate the updated nodal solutions to the associated material points. When a residual performance is resolved, an overall performance gain (e.g., the amount of it/s) of 28 is reported. As an example, the functions `p2nsolve.m` and `mapN2p.m` are 24 and 22 times faster than an iterative algorithm when the residual performance is achieved. The overall performance gain is in agreement to other vectorised FEM codes, i.e., O'Sullivan et al., 2019 reported an overall gain of 25.7 for a optimised continuous Galerkin finite element code.

An iterative implementation would require multiple nested for-loops and a larger number of operations on smaller matrices, which increase the number of BLAS calls, thus inducing significant BLAS overheads and decreasing the overall performance of the solver. This is limited by a vectorised code structure. However and as showed by the matrix multiplication problem, the L2 cache reuse is the limiting factor and, it ultimately affects the peak performance of the solver due to these numerous RAM-to-cache communications for larger matrices. Such problem is serious and, its influence is demonstrated by the delayed response in terms of performance for the function `constitutive.m`. However, we also have to mention that the overall residual performance was resolved only for a limited total number of material points. The performance

drop of the function `constitutive.m` has never been achieved. Consequently, we suspect an additional decrease of overall performances of the solver for larger problems.

The overall performance achieved by the solver is higher than expected and, is even higher with respect to what was reported by Sinaie et al., 2017. We demonstrate that MATLAB is even more efficient than Julia, i.e., a minimum 2.86 performance gain achieved compared to a similar Julia CPDI2q implementation. This confirms the efficiency of MATLAB for solid mechanics problems, provided a reasonable amount of time is spent on the vectorisation of the algorithm.

5.6. Conclusion

We have demonstrated the capability of MATLAB as an efficient language in regard to a material point method (MPM) implementation in an explicit formulation when bottleneck operations (e.g., calculations of the shape function or material point contributions) are properly vectorised. The computational performances of MATLAB are even higher than those previously reported for a similar CPDI2q implementation in Julia, provided that built-in functions such as `accumarray( )` are used. However, the numerical efficiency naturally decreases with the level of complexity of the chosen MPM variant (sMPM, GIMPM or CPDI/CPDI2q).

The vectorisation activities we performed provide a fast and efficient MATLAB-based MPM solver. Such vectorised code could be transposed to a more efficient language, such as the C-CUDA language, that is known to efficiently take advantage of vectorised operations.

As a final word, a future implementation of a poro-elasto-plastic mechanical solver could be applied to complex geomechanical problems such as landslide dynamics while benefiting from a faster numerical implementation in C-CUDA, thus resolving high three-dimensional resolutions in a decent and affordable amount a time.

Code availability The fMPMM-solver developed in this study is licensed under the GPLv3 free software licence. The latest version of the code is available for download from Bitbucket at: `https://bitbucket.org/ewyser/fmpmm-solver/src/master/` (last access: October 6, 2020). The fMPMM-solver archive (v1.0 and v1.1) is available from a permanent DOI repository (Zenodo) at: `https://doi.org/10.5281/zenodo.4068585` (Wyser et al., 2020c). The fMPMM-solver software includes the reproducible codes used for this study.

Appendix A: Acronyms

Table 5.2. | Acronyms used throughout the manuscript

PIC	**P**article-**i**n-**C**ell
FLIP	**FL**uid **I**mplicit **P**article
FEM	**F**inite **E**lement **M**ethod
sMPM	**s**tandard **M**aterial **P**oint **M**ethod
GIMPM	**G**eneralized **M**aterial **P**oint **M**ethod
uGIMPM	**u**ndeformed **G**eneralized **M**aterial **P**oint **M**ethod
cpGIMPM	**c**ontiguous **p**article **G**eneralized **M**aterial **P**oint **M**ethod
CPDI	**C**onvected **P**article **D**omain **I**nterpolation
CPDI2q	**C**onvected **P**article **D**omain **I**nterpolation **2**nd order **q**uadrilateral

Appendix B: fMPMM-solver Variables

Table 5.3. | Variables of the structure arrays for the mesh `meD` and the material point `mpD` used in Code Fragment 1 & 2 shown in Figs. 5.4 & 5.5. `nDF` stores the local and global number of degrees of freedom, i.e., `nDF=[nNe,nN*DoF]`. The constant `nstr` is the number of stress components, according to the standard definition of the Cauchy stress tensor using the Voigt notation, e.g., $\boldsymbol{\sigma}_p = (\sigma_{xx}, \sigma_{yy}, \sigma_{xy})$.

Variable		Description	Dimension
meD.	nNe	nodes per element	(1)
	nN	number of nodes	(1)
	DoF	degree of freedom	(1)
	nDF	number of DoF	(1,2)
	h	mesh spacing	(1,DoF)
	x	node coordinates	(nN,1)
	y	node coordinates	(nN,1)
	m	nodal mass	(nN,1)
	p	nodal momentum	(nDF(2),1)
	f	nodal force	(nDF(2),1)
mpD.	n	number of points	(1)
	l	domain half-length	(np,DoF)
	V	volume	(np,1)
	m	mass	(np,1)
	x	point coordinates	(np,DoF)
	p	momentum	(np,DoF)
	s	stress	(np,nstr)
	S	basis function	(np,nNe)
	dSx	derivative in x	(np,nNe)
	dSy	derivative in y	(np,nNe)
	B	B matrix	(nstr,nDF(1),np)

Author contributions EW wrote the original manuscript and developed, together with YP, the first version the solver (fMPMM-solver, v1.0). YA provided technical supports, assisted EW in the revision of the latest version of the solver (v1.1) and corrected specific parts of the solver. EW and YA wrote together the revised version of the manuscript. MJ and YP supervised the early stages of the study and provided guidance. All authors have reviewed and approved the final version of the paper.

Competing interests The authors declare that they have no conflicts of interest.

Acknowledgements Yury Alkhimenkov gratefully acknowledges the support from the Swiss National Science Foundation (grant no. 172691). Yury Alkhimenkov and Yury Y. Podladchikov gratefully acknowledge support from the Russian Ministry of Science and Higher Education (project No. 075-15-2019-1890). The authors gratefully thank Johan Gaume for his comments that contributed to improve the overall quality of the manuscript.

6. High-performance GPU-based solver

ep2-3De v1.0: an explicit GPU-based GIMPM solver for selected elasto-plastic problems

Emmanuel Wyser, Yury Alkhimenkov, Michel Jaboyedoff & Yury Y. Podladchikov

Under review in *Geoscientific Model Development Discussions*

Abstract We propose an explicit GPU-based solver within the material point method (MPM) framework on a single graphics processing unit (GPU) to resolve elastoplastic problems under two- and three-dimensional configurations (*i.e.,* granular collapses and slumping mechanics). Modern GPU architectures, including Ampere, Turing and Volta, provide a computational framework that is well suited to the locality of the material point method in view of high-performance computing. For intense and nonlocal computational aspects (*i.e.,* the back-and-forth mapping between the nodes of the background mesh and the material points), we use straightforward atomic operations (the scattering paradigm). We select the generalized interpolation material point method (GIMPM) to resolve the cell-crossing error, which typically arises in the original MPM, because of the C_0 continuity of the linear basis function. We validate our GPU-based in-house solver by comparing numerical results for granular collapses with the available experimental data sets. Good agreement is found between the numerical results and experimental results for the free surface and failure surface. We further evaluate the performance of our GPU-based implementation for the three-dimensional elastoplastic slumping mechanics problem. We report i) a maximum performance gain of x200 between a CPU- and GPU-based implementation, provided that ii) the hardware limit (*i.e.,* the peak memory bandwidth) of the device is reached. We finally showcase an application to slumping mechanics and demonstrate the importance of a three-dimensional configuration coupled with heterogeneous properties to resolve complex material behaviour.

6.1. Introduction

The material point method (MPM) was first proposed by Sulsky et al., 1994 and was further advanced by the generalized interpolation material point method (GIMPM) by Bardenhagen et al., 2004. It can be think of as a finite element method (FEM) in which a) integration points (*i.e.,* material points) move and b) convey state variables, *e.g.,* stress and strain components. The continuum is discretized by material points. The nodal momentum equations are solved on a background mesh and nodal basis functions provide a mapping framework between the mesh and the material points to transfer either the updated nodal solution or material point properties. The background mesh can be reset and actually never deforms. It has been widely used for large deformation geomechanical problems such as retrogressive failure, coupled hydromechanical landslides or granular collapses (Tran et al., 2019a; Bandara et al., 2015; Dunatunga et al., 2015).

From a computational point a view, it is critical that the MPM be able to simulate large-scale problems in both two- and three-dimensional configurations. From this perspective, a few researchers have exploited parallel computing using a single or multiple GPU strategy (Dong et al., 2015; Dong et al., 2018) to efficiently implement an explicit GIMPM for two-dimensional configurations. More recently, some researchers in the graphics community presented a similar implementation (Gao et al., 2018; Hu et al., 2019; Wang et al., 2020c) for three-dimensional configurations. One of the most computationally expensive operations in MPM is mapping between material points and their associated nodes, which is supported by basis functions. When implementing a GPU, the two most common approaches are *gathering* and *scattering*. The former gathers the material point's state variables (*i.e.,* mass, velocity component or stresses) to the nodes, whereas the latter scatters (*i.e.,* distributes) the material point's state variables to their associated nodes. This leads to write conflicts, as several threads are writing into the same memory location at the same time. Gao et al., 2018 demonstrated the superiority of *scattering* over *gathering*, provided that the write conflicts are handled without atomic operations. Gao et al., 2018 proposed parallel scattering that results in a performance of an order of magnitude higher than that of a naive atomic implementation. Recently, Wang et al., 2020c proposed an Array of Structures of Arrays (AoSoA) as an efficient layout. It is largely responsible for CPU or GPU performances, as it dictates the memory access pattern (Wang et al., 2020c) by ensuring coalesced memory accesses.

We propose an explicit GIMPM implementation in a three-dimensional configuration on a single GPU (`ep2-3De v1.0`), taking advantage of the efficient vectorized algorithmic structure of the MPM solver proposed by Wyser et al., 2020a. Our GPU-based solver relies on built-in functions of atomic operations for the mapping between material points and their associated nodes (*i.e.,* scattering). For large-scale simulations, the main hardware limit is the GPU on-chip memory, which was well documented by Dong et al., 2018. The GPU

solver `ep2-3De v1.0`[1] combines MATLAB for pre- and postprocessing activities with the massive power of the most recent GPU architectures available (Ampere, Turing and Tesla architectures). This approach allows the user to easily set the problem's geometry and initialize the material points as well as their state variables. Everything needed is then passed to the GPU, which further performs the computations. We propose a formal framework to evaluate the performance of our GPU-based implementation based on the metric for memory-bounded codes, *i.e.,* the effective memory throughput (Omlin, 2017). Since the memory wall has been reached, the memory bandwidth becomes the limiting factor for performance. In addition, it is an easily comparable metric. Similarly, we also report the average number of iterations per second for the same reason: it indicates a relative performance, and it does not depend on material properties (*e.g.,* bulk or shear moduli). We also implement the solver `ep2-3De v1.0` under a single-CPU architecture to provide a reference baseline for the performance evaluation of the GPU-based implementation. For the validation of our solver, we simulate the granular collapse problem in a three-dimensional configuration and compare the result against the well-known experimental results of Bui et al., 2008.

6.2. Numerical implementation

In this section, we briefly describe the governing equations implemented in the MPM solver. We use a linear elastoplastic rheology. Large deformations are carried out via a rate-dependent formulation with the Jaumann stress rate.

6.2.1. Governing equations

The conservation of linear momentum is given by (using the Einstein summation convention)

$$\rho \frac{\partial v_k}{\partial t} = \frac{\partial \sigma_{kl}}{\partial x_l} + \rho g_k, \tag{6.2.1}$$

where σ_{kl} is the Cauchy stress tensor, $v_k = \partial u_k / \partial t$ is the velocity, u_k is the displacement, g_k is the body force, and $k, l = \overline{1..3}$. The conservation of angular momentum is given by $\sigma_{kl} = \sigma_{lk}$. Dirichlet and Neumann boundary conditions are

$$u_k = \bar{u}_k \quad \text{on} \quad \partial\Omega_u, \tag{6.2.2}$$

$$\sigma_{kl} n_l = \bar{\tau}_k \quad \text{on} \quad \partial\Omega_\tau, \tag{6.2.3}$$

where $\hat{u}_k$ and $\hat{\tau}_k$ are prescribed displacements, and n_k is a unit normal vector pointing outward from the boundary $\partial\Omega$ of the domain Ω. Following the

[1] The routines of the `ep2-3De v1.0` solver are available for download from Bitbucket at: https://bitbucket.org/ewyser/ep2-3de/src/master/ (last access: June 16, 2021). The routines archive (v1.0) (Wyser et al., 2021) is available from a permanent DOI repository (Zenodo) at https://doi.org/10.5281/zenodo.4966590 (June 16, 2021).

standard FEM procedure, we use the updated Lagrangian framework; thus, the weak form of Eq. 6.2.1 is written in the current spatial configuration. The weak form of Eq. 6.2.1 can be obtained by multiplying it with a test function ϕ and then applying integration by parts and divergence theorem, leading to

$$\int_{\Omega} \phi \rho a_k \mathrm{d}\Omega = \int_{\Omega} \phi \rho g_k \mathrm{d}\Omega - \int_{\Omega} \frac{\partial \phi}{\partial x_l} \sigma_{kl} \mathrm{d}\Omega + \int_{\partial\Omega_\tau} \phi \bar{\tau}_k \mathrm{d}S, \tag{6.2.4}$$

where $\partial v_k / \partial t = a_k$ is the acceleration, ϕ is any test function that vanishes on $\partial\Omega_u$, and $\bar{\tau}_k$ is the external traction applied on the boundary $\partial\Omega$, $k = \overline{1..3}$. However, in our MPM implementation, tractions on the boundary are not used. Eq. 6.2.4 can be solved using a finite element approach leading to the following compact form:

$$\left[M_{ij} a_j\right]_k = \left[f_i^{\text{ext}} - f_i^{\text{int}}\right]_k, \tag{6.2.5}$$

where $M_{ij} = \sum_{p=1}^{n_p} m_p \phi_i(\boldsymbol{x}_p) \phi_j(\boldsymbol{x}_p)$ is the consistent mass matrix with $\phi_i(\boldsymbol{x}_p)$ being the basis function between node i and material point p. This work adopts a lumped mass matrix, *i.e.*, $m_i \equiv M_{ii} = \sum_{p=1}^{n_p} m_p \phi_i(\boldsymbol{x}_p)$, to avoid an expensive matrix inversion (Sulsky et al., 1994; Bardenhagen et al., 2004; González Acosta et al., 2020). The external $f_{k,n}^{\text{ext}}$ and internal $f_{k,n}^{\text{int}}$ forces at node n are then defined by

$$f_{k,n}^{\text{ext}} = \sum_{p=1}^{n_p} m_p \phi_n(\boldsymbol{x}_p) g_k, \tag{6.2.6}$$

$$f_{k,n}^{\text{int}} = \sum_{p=1}^{n_p} v_p \frac{\partial \phi_n}{\partial x_l}(\boldsymbol{x}_p) \sigma_{kl,p}, \tag{6.2.7}$$

where m_p is the material point's mass, v_p is the material point's volume and $\sigma_{kl,p}$ is the material point's Cauchy stress tensor. Solving Eq. 6.2.5 for the acceleration $a_{k,n}$, the updated velocity is obtained via a forward-Euler scheme,

$$v_{k,n}^{t+\Delta t} = v_{k,n}^{t} + \Delta t a_{k,n}, \tag{6.2.8}$$

where the velocity is given by $v_{k,n}^t = m_n^{-1} \sum_{p=1}^{n_p} \phi_n(\boldsymbol{x}_p) m_p v_{k,p}$ and $v_{k,p}$ is the material point's velocity. Boundary conditions are enforced on the boundary nodes. The material point velocity $v_{k,p}$ and coordinates $x_{k,p}$ are defined by mapping (*i.e.*, an interpolation) between the updated solution on the mesh and the material points, *i.e.*,

$$v_{k,p}^{t+\Delta t} = v_{k,p}^{t} + \Delta t \sum_{n=1}^{n_n} \phi_n(\boldsymbol{x}_p) a_{k,n}, \tag{6.2.9}$$

$$x_{k,p}^{t+\Delta t} = x_{k,p}^{t} + \Delta t \sum_{n=1}^{n_n} \phi_n(\boldsymbol{x}_p) v_{k,n}^{t+\Delta t}, \tag{6.2.10}$$

where n_n is the number of associated nodes n to a material point p. The remaining tasks are i) to update the material point volume and ii) to solve for the constitutive stress-strain relationship.

6.2.2. Rate formulation

The large deformation framework necessitates a suitable stress-strain formulation. Some studies prefer the finite deformation framework and employ a linear relationship between Kirchhoff stresses and logarithmic strains (Charlton et al., 2017; Gaume et al., 2018; Coombs et al., 2020b). In the present work, we adopt a rate-dependent framework by applying the Jaumann rate (*e.g.,* Huang et al. 2015; Wang et al. 2016c; Wang et al. 2016b; Bandara et al. 2016), which yields an objective stress rate measure.

The Jaumann rate of the Cauchy stress is given by

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = C_{ijkl}\frac{1}{2}\left(\frac{\partial v_l}{\partial x_k} + \frac{\partial v_k}{\partial x_l}\right), \tag{6.2.11}$$

where C_{ijkl} is the 4th rank tangent stiffness tensor. Thus, the Jaumann stress derivative may be written as

$$\frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} = \frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} - \sigma_{ik}\dot{\omega}_{jk} - \sigma_{jk}\dot{\omega}_{ik}, \tag{6.2.12}$$

where $\omega_{ij} = (\partial_i v_j - \partial_j v_i)/2$ is the vorticity tensor, $\mathrm{D}\sigma_{ij}/\mathrm{D}t$ corresponds to the material derivative

$$\frac{\mathrm{D}\sigma_{ij}}{\mathrm{D}t} = \frac{\partial\sigma_{ij}}{\partial t} + v_k\frac{\partial\sigma_{ij}}{\partial x_k}. \tag{6.2.13}$$

By rearranging the Jaumann stress derivative in Eq. 6.2.12, we obtain

$$\frac{\partial\sigma_{ij}}{\partial t} = \frac{\mathcal{D}\sigma_{ij}}{\mathcal{D}t} + \overbrace{\sigma_{ik}\dot{\omega}_{jk} + \sigma_{jk}\dot{\omega}_{ik}}^{\sigma^{\mathcal{R}}_{ij}}, \tag{6.2.14}$$

where $\sigma^{\mathcal{R}}_{ij}$ represents the rotation of the Cauchy stress tensor, which satisfies the stress objectivity for the rate-dependent formulation.

Let us expand $\sigma^{\mathcal{R}}_{ij}$ in Eq. 6.2.14 using identities $\sigma_{ij} = \sigma_{ji}$, $\dot{\omega}_{ij} = -\dot{\omega}_{ji}$ and $\dot{\omega}_{kk} = 0$. The Cauchy stress tensor is written using the so-called Voigt notation (as a vector $\boldsymbol{\sigma} = \{\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{xz}\}$). After expanding, collecting and rearranging terms, the objective stress terms $\sigma^{\mathcal{R}}_{ij}$ for a three-dimensional configuration are

$$\sigma^{\mathcal{R}}_{xx} = 2(\sigma_{xy}\dot{\omega}_{xy} + \sigma_{xz}\dot{\omega}_{xz}), \tag{6.2.15}$$
$$\sigma^{\mathcal{R}}_{yy} = -2(\sigma_{xy}\dot{\omega}_{xy} - \sigma_{yz}\dot{\omega}_{yz}), \tag{6.2.16}$$
$$\sigma^{\mathcal{R}}_{zz} = -2(\sigma_{xz}\dot{\omega}_{xz} + \sigma_{yz}\dot{\omega}_{yz}), \tag{6.2.17}$$
$$\sigma^{\mathcal{R}}_{xy} = \dot{\omega}_{xy}(\sigma_{yy} - \sigma_{xx}) + \sigma_{yz}\dot{\omega}_{xz} + \sigma_{xz}\dot{\omega}_{yz}, \tag{6.2.18}$$
$$\sigma^{\mathcal{R}}_{yz} = \dot{\omega}_{yz}(\sigma_{zz} - \sigma_{yy}) - \sigma_{xy}\dot{\omega}_{xz} - \sigma_{xz}\dot{\omega}_{xy}, \tag{6.2.19}$$
$$\sigma^{\mathcal{R}}_{xz} = \dot{\omega}_{xz}(\sigma_{zz} - \sigma_{xx}) + \sigma_{yz}\dot{\omega}_{xy} - \sigma_{xy}\dot{\omega}_{yz}, \tag{6.2.20}$$

and, for a two-dimensional configuration assuming plane strain conditions, Eqs. 6.2.15, 6.2.16 and 6.2.18 reduce to

$$\sigma_{xx}^{\mathcal{R}} = 2\sigma_{xy}\dot{\omega}_{xy}, \tag{6.2.21}$$
$$\sigma_{yy}^{\mathcal{R}} = -2\sigma_{xy}\dot{\omega}_{xy}, \tag{6.2.22}$$
$$\sigma_{xy}^{\mathcal{R}} = \dot{\omega}_{xy}(\sigma_{yy} - \sigma_{xx}). \tag{6.2.23}$$

6.2.3. Elastoplastic deformation

A nonassociated Drucker-Prager model (D-P model) with a tension cutoff is used in this study, similar to Huang et al., 2015; Liu et al., 2020; Nguyen et al., 2020; Zuo et al., 2020, because of its straightforward implementation within explicit numerical solvers. The D-P model has been established as an approximation of the Mohr-Couloumb (M-C) model (Krabbenhoft et al., 2012; Alejano et al., 2012), *i.e.,* a conical yield surface that approximates the M-C yield surface in the principal stress space. The former can be adjusted by parameters, so it passes either through the outer or inner edges of the M-C yield surface (Jiang et al., 2011; De Borst et al., 2012).

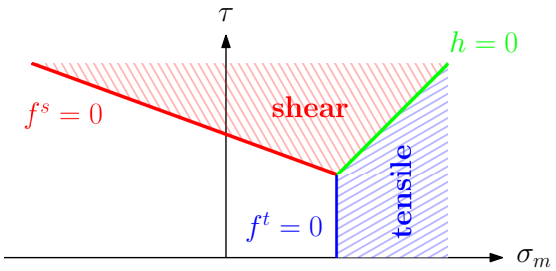


Figure 6.1. | Drucker-Prager yield surface in the (σ_m-τ) space. The yield surface is made of a shear line segment (in red) and a tensile line segment (in blue).

The D-P yield function f (see Fig. 6.1) is typically defined in terms of invariants; The first invariant of the Cauchy stress tensor $I_1 = \sigma_{kk}$, and the second invariant $J_2 = \frac{1}{2}\tau_{ij}\tau_{ji}$ of its deviatoric part τ_{ij}, where the deviatoric part of the Cauchy stress is $\tau_{ij} = \sigma_{ij} + \delta_{ij}p$ with the pressure $p = -\frac{1}{3}\sigma_{kk}$. The D-P yield surface is made of two surfaces (*i.e.,* representing **s**hear and **t**ensile yield criteria), delimited by

$$f^s(\sigma_m, \tau) = \tau + q_\phi \sigma_m - k_\phi, \tag{6.2.24}$$
$$f^t(\sigma_m) = \sigma_m - \sigma^t, \tag{6.2.25}$$

where $\tau = \sqrt{J_2}$ is the effective shear stress, $\sigma_m = -p$ is the mean stress, q_ϕ and k_ϕ are the material parameters defined by ϕ as the internal friction angle, σ^t is the tensile strength and c is the cohesion. Cohesion varies with the accumulated plastic strain $\bar{\epsilon}_p$ when considering a strain softening material, *i.e.,* $c = f(\bar{\epsilon}_p)$. These two surfaces define two plastic regions (see Fig. 6.1)

corresponding to either the shear or tensile failure mode. We use a nonassociated plastic flow law for shear and tensile failures; thus, the plastic potential function g is written as

$$g^s(\sigma_m, \tau) = \tau + q_\psi \sigma_m, \tag{6.2.26}$$

$$g^t(\sigma_m) = \sigma_m, \tag{6.2.27}$$

where q_ψ is a material parameter estimated with the dilation angle ψ.

The line segment $h(\sigma_m, \tau) = 0$ represents the diagonal line between $f^s(\sigma_m, \tau) = 0$ and $f^t(\sigma_m, \tau) = 0$ in the (σ_m, τ) plane, *i.e.,* h is the boundary between shear and tensile failure modes. The function $h(\sigma_m, \tau)$ is given by

$$h(\sigma_m, \tau) = \tau - \tau^P - \alpha^P(\sigma_m - \sigma^t), \tag{6.2.28}$$

with the constants $\tau^P = k_\phi - q_\phi \sigma^t$ and $\alpha^P = (1 - q_\phi^2)^{1/2} - q_\phi^2$. We consider an inner adjustment of the D-P yield surface with respect to the M-C yield surface (Souza Neto et al., 2011), and the model parameter used in Eqs. 6.2.24 & 6.2.26 are given by

$$q_\phi = \frac{6 \sin \phi}{\sqrt{3}(3 + \sin \phi)}, \tag{6.2.29}$$

$$q_\psi = \frac{6 \sin \psi}{\sqrt{3}(3 + \sin \psi)}, \tag{6.2.30}$$

$$k_\phi = \frac{6c \cos \phi}{\sqrt{3}(3 + \sin \phi)}. \tag{6.2.31}$$

In the following, we briefly detail the return mapping strategy used to return the trial Cauchy stress σ_{ij}^{tr} (*i.e.,* assuming pure elastic deformation only) onto the yield surfaces considering $\psi = 0$. A complete description of such return mapping can be found in Huang et al., 2015. Shear failure is declared when i) $f^s(\sigma_m^{tr}, \tau^{tr}) > 0$ and $\sigma_m^{tr} < \sigma^t$ or if ii) $h(\sigma^{tr}, \tau^{tr}) > 0$ and $\sigma_m^{tr} \geq \sigma^t$. The corrected Cauchy stress tensor now reads

$$\sigma_{ij}^{t+\Delta t} = \tau_{ij}^{tr} \left(\frac{k_\phi - q_\phi \sigma^{tr}}{\tau^{tr}} \right) + \sigma^{tr} \delta_{ij}, \tag{6.2.32}$$

with $\boldsymbol{\delta}$ the Kronecker tensor. Tensile failure is declared when $h(\sigma^{tr}, \tau^{tr}) \leq 0$ and $\sigma_m^{tr} \geq \sigma^t$. The corrected Cauchy stress tensor reads as

$$\sigma_{ij}^{t+\Delta t} = \sigma_{ij}^{tr} + (\sigma^t - \sigma_m^{tr}) \delta_{ij}. \tag{6.2.33}$$

6.3. GIMPM implementation under a GPU architecture

We propose an explicit generalized interpolation material point method (GIMPM) implementation (Dong et al., 2018; Wang et al., 2020c) in a three-dimensional configuration on a GPU, taking advantage of the efficient vectorized algorithmic structure (Wyser et al., 2020a; Wyser et al., 2020b). We select explicit

GIMPM implementation, which is valid for a variety of problems compared to other latest variants (Wang et al., 2019; Coombs et al., 2020b), *i.e.,* CPDI or CPDI2q. Additionally, we use a double-mapping approach (MUSL, see Nairn 2003; Buzzi et al. 2008), which consists of updating the stress at the end of the time step. We implement the following domain-update methods: a) no update of the material point domain, further labelled uGIMPM, and b) a domain update controlled by the determinant of the deformation gradient, *i.e.,* $\det(F_{ij})$, further labelled cpGIMPM. These domain-update methods are commonly used in the literature (Baumgarten et al., 2019b; Tran et al., 2019a). The limitation of the two methods is that they are not ideally suited for specific tests: simple stretching and compression modes (Coombs et al., 2020b).

6.3.1. Implementation on a graphical processing unit (GPU)

Graphical processing units (GPUs) are many-core processors originally designed to refresh screen pixels (*e.g.,* for computer games) independently. A schematic representation of the main architecture differences between a CPU and a GPU is depicted in Fig. 6.2. On the GPU chip, most of the physical space is dedicated to arithmetic logical units, whereas on a CPU, most of the physical space is dedicated to chip host scheduling and control microsystems. GPUs feature many more cores, a lower thread-scheduling cost and a higher memory bandwidth than CPUs. The programming model is based on a parallel principle called single instruction - multiple data (or SIMD), *i.e.,* every single instruction is executed on different data. GPUs feature a hierarchical structure. The lowest computational unit is the thread. Threads are organized into blocks of threads, the whole constituting a hierarchical grid of blocks of threads. A GPU typically launches thousands of threads, which execute the same instruction in parallel, thus achieving massive parallelism. Additionally, the most recent GPUs offer a high throughput (close to a TB per second peak memory throughput).

Currently, most of the algorithms are memory-bounded, meaning that memory transfers limit the performance, in contrast to computer-bounded algorithms, where floating point (arithmetic) operations limit the performance. Thus, for an efficient implementation of an algorithm, one must consider a) limiting the memory transfers to the bare minimum and b) avoiding complex data structures (Räss et al., 2020) to benefit from the high throughput capabilities of GPUs. The ability of a GPU is particularly well suited to efficiently execute a large number of local operations in parallel, *i.e.,* single instruction, multiple data (SIMD) programming. In the case of a GIMPM implementation, this includes the calculation of shape functions and the update of various quantities at the material point level (*i.e.,* stresses, domain lengths, material point volumes, etc.). Below, we present key aspects of our GPU-based implementation using the Computed Unified Device Architecture (CUDA C) language of the Nvidia Corporation, which is a syntax extension of the C programming

language.

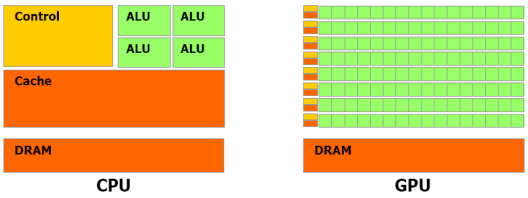


Figure 6.2. | Schematic chip representation for both the central processing unit (CPU) and the graphical processing unit (GPU) architecture (Nvidia, 2021). The latter is made of thousands of arithmetic logical units (ALUs). The CPU architecture is primarily dedicated to controlling units and cache memory, and the physical space allowed for ALUs is considerably reduced compared to a GPU architecture.

Algorithm workflow

In our implementation, MATLAB acts as an architect (see Fig. 6.3). It 1) defines the problem geometry (*i.e.,* the background mesh, material point locations and related quantities, etc.), which can be tedious to initialize in a CUDA C environment. It also calls an external MATLAB script, which compiles the necessary source codes, *i.e.,* `gpu.cu` or `cpu.cu`. It further 2) calls either a CUDA C or plain C executable, *i.e.,* `gpu.exe` or `cpu.exe`, within a Windows OS to solve for the numerical problem and finally 3) imports the results of calculations for further postprocessing tasks.

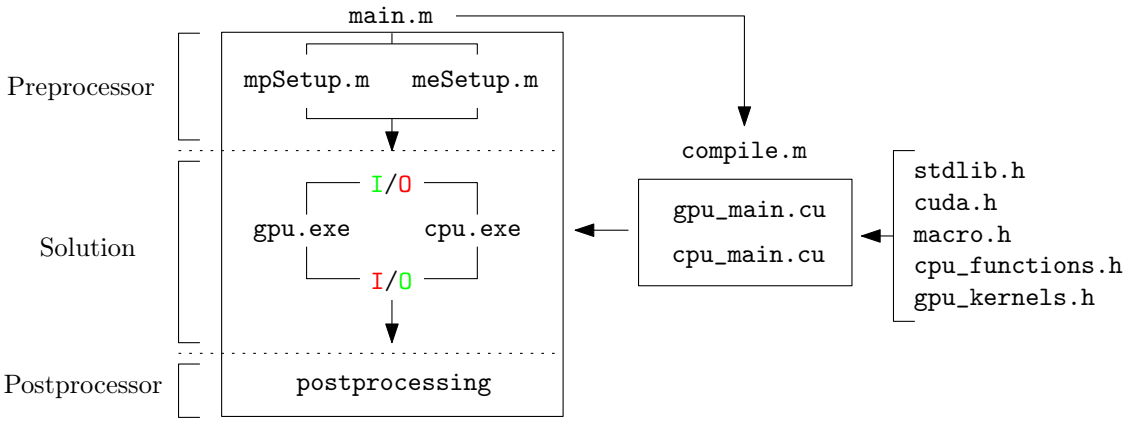


Figure 6.3. | Multifunctional workflow: 1) usage of MATLAB for data initialization, compilation and postprocessing activities and 2) system calls to a performant compiled language such as C (CPU-based) and CUDA C (GPU-based) for heavy calculations. Here, `I/O` stands for input/output, and the colouring (red or green) specifies which one is active, *i.e.,* `I/O` means data only are transferred to the GPU (or CPU) for further calculation activities.

This is a powerful combination between a high-level language such as MATLAB and a performant low-level language such as CUDA C or plain C. It is also easy to invoke system commands directly via MATLAB, *i.e.,* to compile source codes and/or run executables using the built-in command `system('...')`. We focus on OS-free scripting in MATLAB using a built-in command (*i.e.,* `isunix` or `ispc`) to ensure that it performs well under all operating system (OS) architectures. In addition, such a workflow can be easily extended to other high-level languages such as Python.

Kernels and launch configuration

We briefly describe our GPU-based implementation (`gpu_main.cu`) while focusing mainly on the computational aspects of the implementation. Implementation of an explicit GIMPM solver into the CUDA C language requires dispatching computational activities into several kernels, *i.e.,* similar to classic functions for a serial implementation in the C language. Each kernel is operated by the GPU only, and kernel launch configuration parameters must be defined for its proper execution. Among them, one must define the number of active threads per block (*i.e.,* the block size) and the number of blocks (*i.e.,* the grid size). A typical kernel is executed N times in parallel by N distinct threads organized into blocks of threads, *i.e.,* a grid of blocks of threads. The principal hardware limitation is the total number of threads within a block: it cannot exceed 1024 threads per block. One must ensure that the maximal size of a block is lower than or equal to this limit.

The computational activities are handled by multiple GPU kernels; 11 kernels are successively launched over a computational cycle. An overall description is given in Fig. 6.4. A `while` loop is used to perform the computational cycles, and an MPM step is solved at every cycle. n_{IO} (*i.e.,* the number of accesses to the GPU global memory) is reported in Fig. 6.4 for each kernel and is estimated by a careful examination of relevant operations within the kernels. Note that all calculations are performed on the GPU, except the calculation of the adaptive time step, which is serially executed by the CPU.

In our GPU-based implementation, we define two distinct types of kernel launch parameters: 1) those used for mapping between material points and background nodes (*i.e.,* accumulations and projections between material points with their associated nodes and back and forth) and 2) those used for local calculation at the material point or node level (*i.e.,* update of material point stresses or the solution to the momentum balance equations on the Eulerian background mesh).

Adaptative time step

An adaptive time step is implemented. For three-dimensional configurations, the maximum elastic wave speed of the material (Anderson, 1987; Zhang et

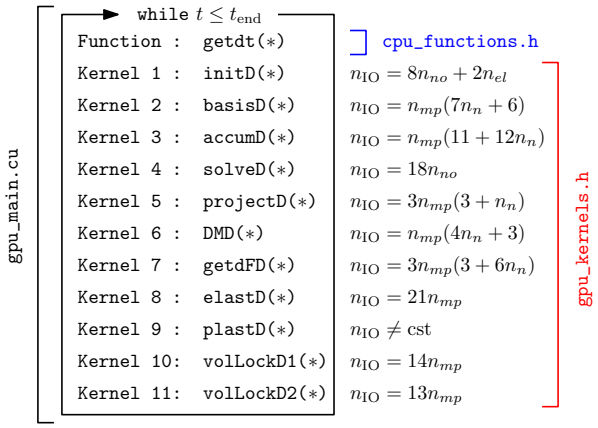


Figure 6.4. | Specific workflow for the source code running of the GPU, t_{end} is a user-defined time that controls the total time of the simulation, and the operator $*$ stands for the pointer object, as in the C language. It should be noted that a vast majority of operations within kernels are performed on pointers.

al., 2017) reads as

$$(c_x, c_y, c_z) = c_{el} + \left(\max_p(|v_{x,p}|), \max_p(|v_{y,p}|), \max_p(|v_{z,p}|) \right), \tag{6.3.1}$$

where $c_{el} = ((K + 4G/3)/\rho)^{\frac{1}{2}}$ is the elastic wave speed of the material, K and G are the bulk and shear moduli, respectively, ρ is the material density, and $v_{x,p}$, $v_{y,p}$ and $v_{z,p}$ are the material point velocity components. The time step Δt is then restricted by the CFL condition,

$$\Delta t = \alpha \min \left(\frac{\Delta x}{c_x}, \frac{\Delta y}{c_y}, \frac{\Delta z}{c_z} \right), \tag{6.3.2}$$

where $\alpha \in [0; 1]$ is the time step multiplier, and $\Delta x, \Delta y$, and Δz are the background mesh resolutions.

This requires evaluation of the maximum velocity of all material points at the beginning of each calculation cycle. We choose to sequentially find the maximum velocity using the CPU instead of a parallel implementation on the GPU. This results in systematic memory transfers between the GPU global memory and the random access memory (RAM) of the CPU. However, we report a low performance loss due to these transfers, *i.e.*, a maximal loss of 2-5 % in performance, which is acceptable.

Back-and-forth mapping between material points and their associated nodes

The GPU-based algorithm relies heavily on the use of arrays `p2e` and `e2n` (Wyser et al., 2020a). Elements are numbered with an increasing index. Associated nodes are also numbered in a similar manner. The array `e2n` of

dimension $n_{el} \times n_n$, where n_{el} is the total number of nodes and n_n is the number of nodes associated with an element e, describes the topological relation between the elements and the nodes of the mesh. Similarly, the array `p2e` describes the topological relation between the material points and the element in which they are located. These two arrays provide an intuitive definition of the relations between i) the material points and the nodes they are associated with (*i.e.,* `p2n`) and ii) the element and their nodes (*i.e.,* `e2n`). Then, it is a computationally straightforward process to identify which nodes n are associated with a material point p, which is occupying an element e.

The GPU-based implementation relies on the built-in function `atomicAdd()` in CUDA C. It performs atomic operations, which avoid the data race of multiple threads, from the same or different blocks to update the same memory location. Atomic operations are extensively used to calculate internal and external force contributions (Eqs. 6.2.6 & 6.2.7), as well as the lumped mass matrix, and to update the material point's properties such as velocities and coordinates (Eqs. 6.2.9 & 6.2.10). Dong et al., 2015; Wang et al., 2020c reported (for older GPU architectures such as Pascal or Kepler) that atomic scattering can be significantly slower compared to an optimized parallel implementation. However, atomic operations are a) intuitive to both understand and implement, and b) they avoid a complex data layout, such as recently proposed in Wang et al., 2020c. The use of built-in atomic operations considerably reduces programming efforts.

Treatment of volumetric locking for low-order elements

When low-order elements are used in a GIMP formulation, volumetric locking arises and results in spurious oscillations of the stress field (Jassim et al., 2013; Coombs et al., 2018; González Acosta et al., 2019; González Acosta et al., 2021). We implement a simple procedure to mitigate volumetric locking when considering near-incompressible behaviour for ischoric plastic flows. Cuomo et al., 2019; Lei et al., 2020 introduced an element-based averaging method, following Mast et al., 2012. Selected material point properties are reconstructed based on an average value calculated at the element's centre at the end of a time step. However, we propose averaging only the volumetric part of the stress tensor, *i.e.,* the pressure $p = -\frac{1}{3}\sigma_{kk}$, while its deviatoric part $\tau_{ij} = \sigma_{ij} - p\delta_{ij}$ remains unchanged. This results in the following:

$$p_e = \frac{\sum_{p \in e} v_p p_p}{\sum_{p \in e} v_p}, \tag{6.3.3}$$

where v_p is the material point's volume. This gives a constant distribution of the pressure field over an element because of its zero-order reconstruction (Lei et al., 2020). The Cauchy stress tensor $\sigma_{ij,p}$ of a material point p occupying an element e is corrected as

$$\sigma_{ij,p} = \tau_{ij,p} + \delta_{ij}(p_e)_p, \tag{6.3.4}$$

where δ_{ij} is the Kronecker delta and $(p_e)_p$ is the averaged pressure within an element e and assigned to a material point p.

6.3.2. Available computational resources

The CPU- and GPU-based simulations are performed on a modern workstation running on a Windows 10 operating system with the latest CUDA version v11.2. The CPU is an Intel Core i9-10900K with 10 physical cores of base clock speed (or frequency) of 3.70 GHz, which can rise up to a maximum clock speed of 5.30 GHz, supported with 64 GB DDR4 RAM. It hosts a consumer electronics Nvidia RTX 3090 GPU (the latest Ampere architecture) with 82 streaming multiprocessors (SM units) with a base frequency of 1.40 GHz. This results in 10490 CUDA cores that are supported with an on-chip memory of 24 GB GDDR6 (*i.e.,* the GPU global memory). Other GPUs installed on older desktops are also used to compare their respective GPU performances, *i.e.,* an RTX 2080 ti (workstation) and a GTX 1650 (laptop), both running on a Windows 10 operating system. Additional simulations were also ran on a workstation equipped with the latest Nvidia A100 GPU at the Lomonosov Moscow State University.

Furthermore, GPU-based simulations are also performed on the Octopus GPU supercomputer at the Swiss Geocomputing Centre, University of Lausanne, Switzerland. In particular, the GPU-based simulations are run on the Volta node, hosting an Nvidia Tesla V100 (Volta architecture) 16 GB, supported by an Intel(R) Xeon(R) E5-2620 v2 (Haswell) @ 2.1 GHz CPU. The latest CUDA version installed is v11.0, and the supercomputer Octopus is operated under a CentOS 6.9. environment. To summarize the computational resources in use, Table A.1 presents the main characteristics of the GPUs used in this study.

Table 6.1. | List of the graphical processing units (GPUs) used throughout this study. We also report the peak memory throughput, *i.e.,* $\mathsf{MTP}_{\mathsf{peak}}$, measured thanks to the routine `bandwidthTest.cu` provided by Nvidia alongside with the CUDA toolkit. When compared with the effective memory throughput $\mathsf{MTP}_{\mathsf{eff}}$, one can estimate the possible gain of an additional optimization of the algorithm. This is particularly useful when estimating the level of optimization of a GPU-based implementation.

GPU	Architecture	SM count	On-chip memory [GB]	$\mathrm{MTP}_{\mathrm{peak}}$ [GB·s^{-1}]
A100	Ampere	108	40	1127.1
RTX 3090	Ampere	82	24	774.1
RTX 2080 ti	Turing	68	11	513.1
GTX 1650	Turing	14	4	168.7
V100	Volta	80	16	732.6

6.3.3. Measuring computational performance on a GPU

Omlin, 2017; Räss et al., 2019a; Räss et al., 2019b; Alkhimenkov et al., 2021 demonstrated that a pertinent metric to quantify the performance of memory-bounded algorithms is the effective memory throughput, *i.e.,* $\mathrm{MTP}_{\mathrm{eff}}$ in $\mathrm{GB{\cdot}s^{-1}}$. It quantifies the efficiency of data transfers between the global memory (*i.e.,* the on-chip memory of the GPU) and the arithmetic logical units (ALUs) of the GPU. To determine the effective memory throughput, one must estimates (or quantifies) the overall set of memory operations (read-and-write or read only), *i.e.,* n_{IO}, which are needed to resolve a given problem. Consequently, we carefully estimate the minimum number of memory operations while considering a GIMPM-based implementation. This results in the following effective memory throughput:

$$\mathrm{MTP}_{\mathrm{eff}} = \frac{n_{\mathrm{iter}} \times n_{\mathrm{IO}} \times n_{\mathrm{p}}}{1024^3 \times t_{\mathrm{GPU}}} \ [\mathrm{GB} \cdot \mathrm{s}^{-1}], \tag{6.3.5}$$

where n_{p} is the arithmetic precision (*i.e.,* single-precision floating-point format FP32 or double-precision floating-point format FP64) and t_{GPU} is the wall-clock time in seconds to complete the n_{iter} iterations to solve for the numerical problem. For three-dimensional problems, we estimate the minimal number of memory operations for an explicit GIMP implementation as

$$n_{\mathrm{IO}} = 2n_{mp}(43 + 22n_n) + 26n_{no} + 2n_{el}, \tag{6.3.6}$$

where n_{mp} is the number of material points, n_n is the number of associated nodes for an element (*i.e.,* $n_n = 16$ in 2D and $n_n = 64$ in 3D), n_{no} is the number of nodes, and n_{el} is the number of elements. Additionally, we also report the count of calculation cycles per second of the GPU, *i.e.,* $\mathrm{it{\cdot}s^{-1}}$ as well as the wall-clock time. These two metrics give an intuitive sense of the time-to-solution, which is convenient for potential application purposes.

6.4. Results

In this section, we present two numerical models using the solver `ep2-3De v1.0`, namely,

1. Model 1, the granular collapse, which serves as
   a) a validation benchmark against the results of the widely-accepted experiment of Bui et al., 2008 under a three-dimensional configuration
   b) a demonstration of the influence of the mesh resolution on plastic strain localization under a plane strain configuration
2. Model 2, the three-dimensional earth slump (Varnes, 1958; Varnes, 1978), which serves as

a) an evaluation of the relative performances of GPU- and CPU-based implementation of the solver `ep2-3De v1.0` considering a variety of recent GPU architectures

b) a showcase of a potential application of the solver `ep2-3De v1.0` for an elastoplastic problem considering different isotropic peak cohesion fields (homogeneous and heterogeneous)

6.4.1. Model 1

Settings for Models 1a & 1b

We investigate the granular collapse of an aluminium-bar assemblage (Bui et al., 2008) under three-dimensional or plane strain configurations. The geometry of the problem is shown in Fig. 6.5, and its variables are summarized in Table 6.2 for both three-dimensional and plane strain configurations. Note that for Model 1a, we use the same number of elements along the $x-$direction $n_{el,x} = 80$ as in Huang et al., 2015. As a direct comparison for Model 1b under a plane strain configuration, Huang et al., 2015 used $n_{el} = 15360$, $\Delta x = \Delta z = 2.5$ mm and $n_{mp} = 25600$.

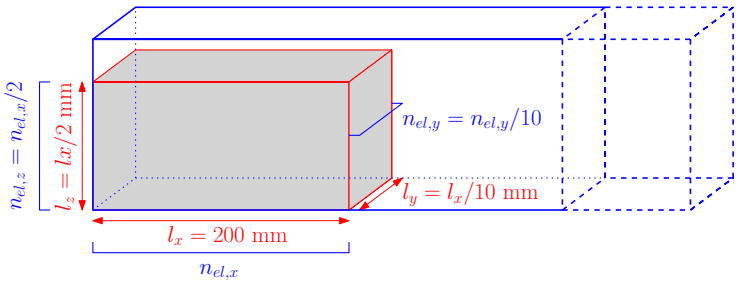


Figure 6.5. | Initial configuration for the granular collapse numerical model. The blue surrounding frame depicts the computational domain, *i.e.,* the background Eulerian mesh, and the red volume is the granular material, which is discretized by 8 material points. The total number of background elements n_{el} depends on the number of elements in the $x-$ direction $n_{el,x}$ used to discretize the granular material.

We consider a noncohesive granular material of density $\rho = 2650$ kg·m^{-3}, with a bulk modulus $K = 0.7$ MPa and a Poisson's ratio $\nu = 0.3$, as in Huang et al., 2015. The cohesion is $c = 0$ Pa, the internal friction angle is $\phi = 19.8°$ with a dilatancy angle $\psi = 0$ according to Bui et al., 2008. However, the density and stiffness properties have negligible effects on the granular flow dynamics, as reported by Nguyen et al., 2020. We introduce local damping D (see Wang et al. 2016b) to resolve numerical results that are compatible with the experimental results of Bui et al., 2008. We find that $D = 0.025$ results in the most compatible dynamics. The reasons for the introduction of local damping can be found in Appendix 6.6. Fully fixed boundary conditions (*i.e.,*

no slip) are enforced at the bottom and rollers on the sidewalls. The total simulation time is 1.0 s, considering a the time step multiplier $\alpha = 0.5$.

Table 6.2. | Parameters used in Models 1 a & b for the granular collapse. $n_{el,i}$ is the number of elements to discretize the granular material along the i-th direction, n_{el} and n_{no} are the total number of elements and nodes of the background mesh, n_{pe} is the number of material points per element and n_{mp} is the total number of material points. Note that the mesh resolution is $\Delta x = \Delta y = \Delta z = 2.5$ mm.

Experiment	$n_{el,x}$	$n_{el,y}$	$n_{el,z}$	n_{el}	n_{no}	n_{pe}	n_{mp}	Δx [mm]
1a	80	20	40	342144	365625	8	512000	2.5
1b	640	-	240	833300	836190	4	819200	0.3

Model 1a: the three-dimensional granular collapse

To validate the numerical implementation under a GPU architecture, we first compare it against the well-known granular collapse experiments initially performed by Bui et al., 2008. Here, we present and compare numerical results without focusing on the performance of the GPU-based implementation. All the simulations are performed on a consumer electronics RTX 3090 GPU with double-arithmetic precision (*i.e.*, $n_{\mathrm{p}} = 8$ bytes).

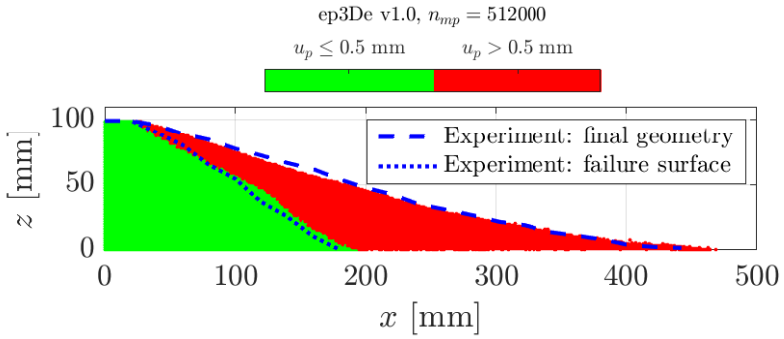


Figure 6.6. | Final geometry of the granular collapse for three-dimensional configuration of our GPU-based explicit GIMPM implementation ep3De v1.0. The green region (*i.e.*, the intact region) is defined by the L_2-norm of the material point displacement $u_p = ||\boldsymbol{u}_p||_2 \leq 0.5$ mm, whereas the red region (*i.e.*, the deformed region) is defined by $u_p = ||\boldsymbol{u}_p||_2 > 0.5$ mm. The experiment of Bui et al., 2008 is indicated by the blue dashed line (*i.e.*, the free surface) and the blue dotted line (*i.e.*, the failure surface).

The results from the numerical simulation under a three-dimensional configuration are shown in Fig. 6.6. A direct and visual comparison demonstrates excellent agreement between the numerical solver and the experiments of Bui et al., 2008. We observe a slightly higher run-out distance, but the overall

geometry of both the failure surface and the free surface is very close to the experimental data. We also report an angle of repose of $\approx 13°$. This value is also consistent with the value reported by Bui et al., 2008, *i.e.,* 14°. The good agreement between the numerical results and the experimental work of Bui et al., 2008 demonstrates that the solver `ep2-3De v1.0` is suitable to simulate large deformation elastoplastic problems such as granular collapses.

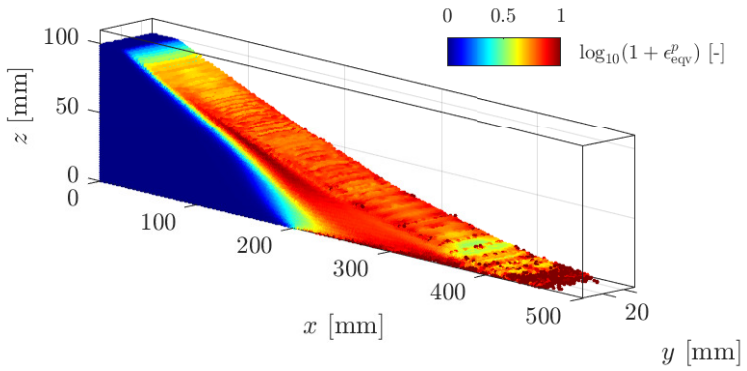


Figure 6.7. | Equivalent plastic strain ϵ^p_{eqv} for the final configuration of the granular collapse. The principal feature of a granular collapse can be observed, *i.e.,* a backward propagation of plastic deformation along a principal failure surface.

The equivalent accumulated plastic strain ϵ^p_{eqv} is shown in Fig. 6.7. We observe a coherent deformation of the granular material with a large shear zone that propagates backward from the base of the material to the top of the granular material. The mobilized granular material flows along a principal failure surface. However, the overall deformation pattern is rather coarse, *i.e.,* fine structures or local shear bands are not yet observed, even though slight deformation heterogeneities can be observed. This coarse behaviour of shear banding is also consistent with previous studies (see Huang et al. 2015; Chalk et al. 2020; Zhang et al. 2021). This is mainly due to the background mesh resolution used in the numerical simulation. We further investigate shear banding using a higher background mesh resolution under a plane strain configuration in Model 1b.

Model 1b: the plane strain granular collapse

We investigate granular collapse under a plane strain configuration, as this allows an increase in the number of elements, resulting in an even finer background mesh (see Table 6.2). For Model 1a, the numerical solution is in agreement with the experimental work of Bui et al., 2008 regarding either the free surface or the failure surface (see Fig. 6.8). This demonstrates that both the three-dimensional and plane strain configurations are in agreement with each other.

An interesting feature of granular collapse is the equivalent accumulated plastic strain (see Fig. 6.9). The GPU-based implementation allows us to in-

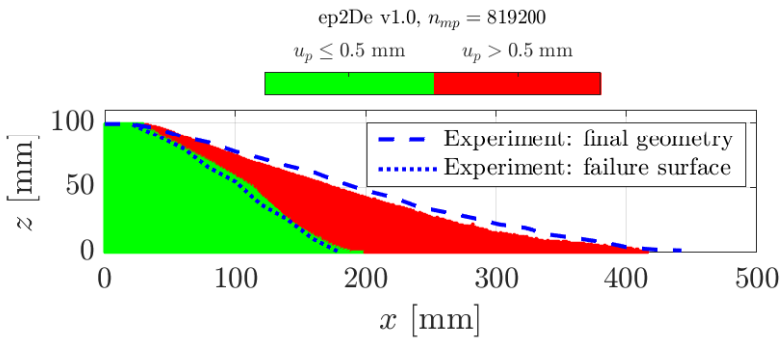


Figure 6.8. | Final geometry of the granular collapse for the plane-strain configuration for our GPU-based explicit GIMPM implementation ep2De v1.0. The numerical solution and the experimental results are in good agreement. Some differences are more pronounced when compared with the numerical results obtained under a three-dimensional configuration.

crease both the background mesh resolution and the total number of material points. This results in finer plastic strain localizations, as demonstrated by the various shear bands and their complex interactions during collapse. Such detailed shear bands are almost impossible to obtain at lower resolutions, which demonstrates the importance of a GPU-based implementation to overcome the hardware limitation of a CPU-based implementation, *i.e.,* mainly longer wall-clock times.

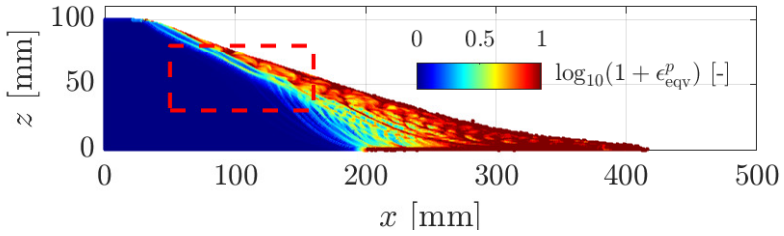


Figure 6.9. | Equivalent plastic strain ϵ^p_{eqv} for the final configuration of the granular collapse. The dashed red rectangle denotes the location of the zoomed-in region in Fig. 6.10. One can observe more complex plastic strain localizations compared to the numerical results obtained in Fig. 6.7 for a three-dimensional configuration with a coarser background mesh resolution.

6.4.2. Model 2

Settings for Models 2a & 2b

Here, we select a cohesive elastoplastic isotropic material (*i.e.,* a homogeneous or heterogeneous peak cohesion field) with no dilatancy behaviour. It

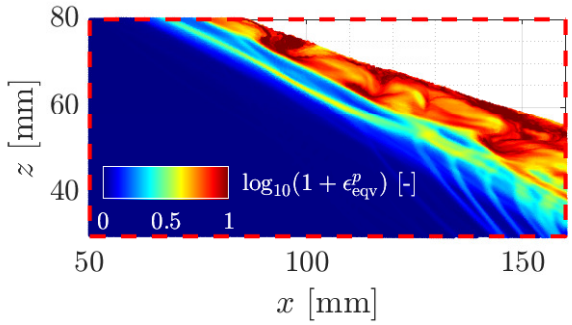


Figure 6.10. | Equivalent plastic strain ϵ^p_{eqv} for the zoomed-in area in Fig. 6.9. A shallow granular flow clearly appears, as suggested by the higher values of ϵ^p_{eqv}. This supports evidence of shallower granular avalanches during collapses. Deeper structures, which result in lower accumulated plastic strains, probably highlight slower deformation modes along well-defined and persistent shear bands.

is modelled with a pressure-sensitive Drucker-Prager model with linear strain-softening behaviour. It is well known that the numerical solutions (as in FEM) are mesh-dependent when considering the strain-softening behaviour of the material. We did not implement techniques to address this issue, but the use of nonlocal plasticity (Galavi et al., 2010; Burghardt et al., 2012) or viscoplastic formulations (Duretz et al., 2019) are possible ways to address this specific task.

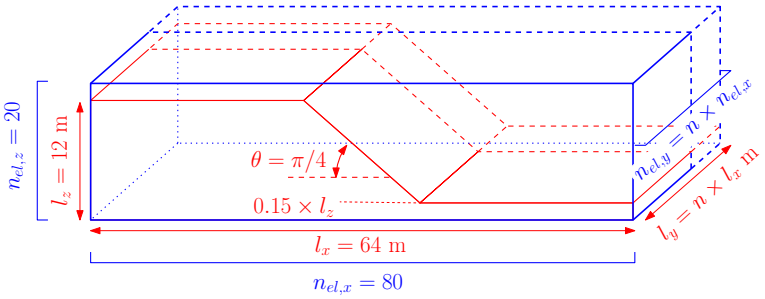


Figure 6.11. | Geometry for the earth slump. The number of elements in the y-direction $n_{el,y}$ and the width of the problem l_y are variable. This allows us to increase (or decrease) the number of both elements and material points without decreasing the mesh resolution. The parameter n controls the dimension of the domain and the number of elements along the $y-$direction. The wall-clock time depends only on the total number of elements, nodes and material points and is not influenced by the mesh resolution.

We have chosen an arbitrary geometry (see Fig. A.1 and Table 6.4), which represents an idealized three-dimensional setting, to observe elastoplastic *slumps* (*i.e.,* earth slumps according to the original classification proposed by Varnes 1958; Varnes 1978), which are now classified as rotational slides in the recent update of the Varnes classification proposed by Hungr et al., 2014. The ge-

ometrical setting differs from the one typically used in the literature, as in Zhang et al., 2021. However, it promotes the compression of the toe, which is an expected feature we want to reproduce. The size of the physical domain $l_z \times l_x \times l_y$ is, at most, 12 m × 64 m × 1024 m for Model 2a, whereas it is 12 m × 64 m × 16 m for Model 2b.

We assume this setting features the principal first-order characteristics of a typical rotational earth slump (Varnes, 1958; Varnes, 1978), *i.e.,* a complex zone of scarps (minor and major) delimiting a crown-like structure, followed by a transition (or depletion) zone in which the material flows homogeneously along internal shear zones due to severe plastic strain localizations and, finally, a compression (or accumulation) zone resulting in complex thrusting at the toe of the slump. Because of the nature of the boundary condition at the bottom of the material (*i.e.,* free-slip), an additional horizontal sliding component is introduced within the rotational part of the displacement.

Table 6.3. | Material properties shared by both Models 2a & 2b.

Parameter	Symbol	Value	Unit
Density	ρ	2700	$\mathrm{kg{\cdot}m^{-3}}$
Poisson's ratio	ν	0.3	-
Elastic modulus	E	1	MPa
Softening modulus	H	50	kPa
Friction angles	ϕ / ϕ_{weak}	20 / 7.5	°

We select material properties (*i.e.,* bulk and shear moduli K and G, friction angle ϕ and peak and residual cohesions c_{peak} and c_{res}) that result in severe deformation processes and strain localizations. The material properties are presented in Table 6.3. They are close to the values commonly used in the literature (Wang et al., 2016b; Wang et al., 2016a; Bandara et al., 2016; Zhang et al., 2021). To increase deformations even more, we also introduce a weak layer of thickness $0.3 \times l_z$ m at the base of the material with a lower friction angle ϕ_{weak}. A time step multiplier $\alpha = 0.5$ is selected, *i.e.,* $\Delta t_{min} = 1.56{\cdot}10^{-2}$ s is obtained over the whole simulation according to the CFL condition for both Models 2a & 2b. As in Zhang et al., 2021, elastic loading dynamic relaxation is applied for a period of $t = 8$ s (*i.e.,* Models 2a & 2b), and the elastoplastic behaviour is activated for an additional 7 s, resulting in a total simulation time $t = 15$ s (*i.e.,* Model 2b only).

Table 6.4. | Geometrical and material properties for Models 2a & 2b. The correlation length vector is $\boldsymbol{\lambda} = (\lambda_x, \lambda_y, \lambda_z) = (2.5, 2.5, 2.5)$ m for both Gaussian and exponential isotropic covariance functions. The grid spacing is always constant in Models 2a & 2b, *i.e.,* $\Delta z = \Delta y = \Delta x = 0.8$ m

Model	$n_{el,y}$ [-]	n_{mp} [-]	Δx [m]	$\bar{c}_{\mathrm{peak}}$ [kPa]	σ [kPa]
2a	$\in [1; 1280]$	$\leq 3.2 \cdot 10^6$	0.8	20	0
2b	20	$\approx 10^5$	0.8	20	0 / 5

Gaussian random fields (see Appendix 6.6) are used to initialize the peak cohesion field c_{peak}, which is parametrized by an average cohesion $\bar{c}_{peak}$ and its standard deviation σ (see Table 6.4) along with the residual cohesion $c_{\text{res}} = c_{\text{peak}}/4$. This allows us to account for heterogeneities within the material, which lead to complex and heterogeneous displacement fields. We first perform preliminary simulations with a constant cohesion field and notice a homogenous solution of the displacement field along the $y-$direction. Using Gaussian fields allows us to mitigate this homogeneity.

Free-slip boundary conditions are applied on the sides and the bottom of the computational domain; only the normal component to the boundary is constrained, while the two others are free. This results in stronger deformations, which we want to highlight. Finally, and as suggested in Wang et al., 2016b, we introduce local damping, *i.e.,* $D = 0.1$.

Model 2a: relative performances

Here, we investigate the computational performances of the solver `ep2-3De v1.0` under a three-dimensional configuration on a variety of GPUs with recent architectures: Ampere, Turing and Volta. Furthermore, we restrict our performance analysis only for the elastic loading phase (*i.e.,* 8 s of simulation) because it is more complex to determine the exact number of material points that are yielding during each computational cycle (see Fig. 6.4) and to infer the exact effective memory throughput.

All the numerical simulations are performed on the computational resources and GPU hardware presented in Table A.1 under double-arithmetic precision (*i.e.,* $n_{\text{p}} = 8$ bytes in Eq. 6.3.5). As a reference baseline, we use the performance obtained for a CPU-based single-threaded implementation of `ep2-3De v1.0` on an i9-10900K CPU (*e.g.,* latest Intel CPU chip). However, this is not representative of a highly optimized multithreaded implementation under a CPU architecture.

We report the effective memory throughput MTP_{eff} of the solver `ep2-3De v1.0` on various GPUs and CPUs (see Fig. 6.12). An increase in the effective memory throughput is observed as the number of material points increases. All GPUs reach a maximum effective throughput, but the Tesla V100 scores a maximum effective throughput of ≈ 650 GB·s^{-1}. This corresponds to 88 % of its peak throughput (for the GPU's hardware limit, see Table A.1). We report a similar observation for the RTX 2080 ti, $\text{MTP}_{\text{eff}} \approx 320$ GB·s^{-1} corresponding to 62 % of its hardware limit. RTX 3090 and GTX 1650 reach $\text{MTP}_{\text{eff}} \approx 405$ GB·s^{-1} and $\text{MTP}_{\text{eff}} \approx 75$ GB·s^{-1}, respectively, which correspond to 52 % and 44 % of their respective hardware limits. Finally, we report a memory throughput of at least $\text{MTP}_{\text{eff}} \approx 5$ GB·s^{-1} for the i9-10900K CPU (10 % of its hardware limit).

The overall results suggest, as in Räss et al., 2020, that most recent GPUs, such as the data-centre Tesla V100 (Volta), offer significant performances compared to entry-level consumer electronics GPUs, such as the GTX 1650. In terms of absolute performance, the more recent the GPU is, the higher its

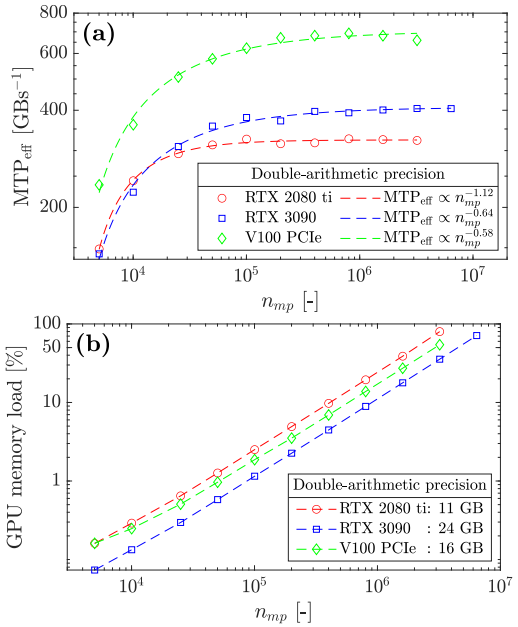


Figure 6.12. | a) Effective memory throughput $\mathrm{MTP_{eff}}$ of the solver `ep2-3De v1.0` for double-arithmetic precision. One can see the on-chip memory limit, as both the RTX 2080 ti and V100 cannot resolve the same number of material points as the RTX 3090. b) GPU on-chip memory load increases with the number of material points n_{mp}, which demonstrates, as expected, one of the GPU's hardware limits.

performance. A demonstration is given by the absolute effective throughput between the RTX 2080 ti and the RTX 3090: The latter achieves an additional 20 % throughput compared to the former. We highly suspect the hardware itself to be the main reason for this. We further investigate the performances of the most recent data-centre GPU, *i.e.,* the A100 (Ampere architecture), with its predecessor the V100 (Tesla architecture). The A100 reaches ≈ 1100 GB·s^{-1}, which yields a performance gain of 1.6× with respect to the Tesla V100. When compared to the maximum effective memory throughput in Table A.1, this correspond to 97 % of the hardware limit.

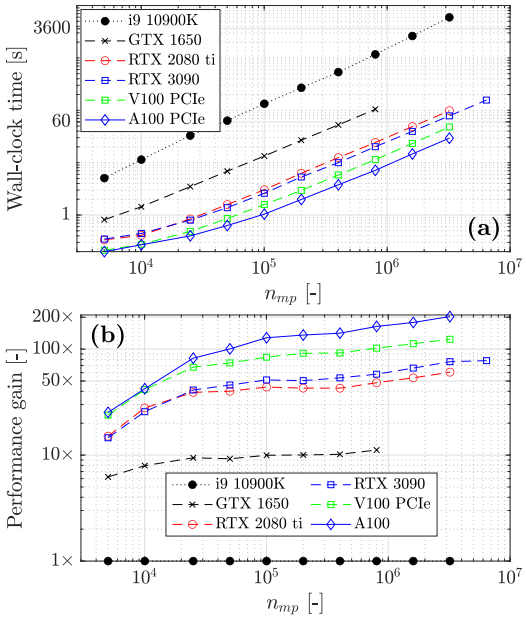


Figure 6.13. | a) Wall-clock time reported for various computing architectures (GPUs and CPU). The differences in the maximal number of material points n_{mp} are due to the on-chip memory limit. A significant difference in terms of wall-clock time is observed between the CPU and GPUs, even for the low-entry consumer electronic GTX 1650, *i.e.,* a performance gain of $\approx 10\times$. b) Performance gains of GPUs relative to the CPU, *i.e.,* $1\times$ as a baseline. We add the CPU and the GTX 1650 wall-clock time for an easier comparison.

Finally, we report the wall-clock time for various computing architectures (see Fig. 6.13a). As expected by the maximum effective memory throughput, A100 delivers the fastest solution, regardless of the number of material points n_{mp}. The A100 GPU resolves a geometry of $n_{mp} \approx 3.2 \cdot 10^6$ in less than a minute (29 seconds), whereas the i9-10900K CPU resolves the same problem in more than an hour (5949 seconds). This corresponds to a 200× performance gain (123× performance gain for the V100, see Fig. 6.13b) compared to the CPU-based implementation of `ep2-3De v1.0`. The RTX 2080 ti and the RTX 3090 reach a 60× and 77× performance gain, respectively. However, the entry-level GTX 1650 is only ten times faster than i9-10900 K. As already shown in

Fig. 6.12a, these performance gains are only expected when the different GPUs reach their maximum effective memory throughput. In terms of runtime, the performance gain (Fig. 6.13b) is in agreement with the memory throughputs reported in Fig. 6.12a.

Model 2b: homogeneous and heterogeneous slumps

As a final experiment, we show the results of the `ep2-3De v1.0` solver for a slump with homogeneous or heterogeneous cohesion fields. In this numerical model, we only show the displacement field at the end of the numerical simulation at $t = 15$ s. The interested reader is referred to Appendix 6.6 for an overview of the temporal evolution of the equivalent plastic strain ϵ_{eqv} for the slump under the three settings of the peak cohesion field. All the numerical simulations are run on a laptop equipped with GTX 1650; $t_{\text{GPU}} \approx 30$ s with the settings presented in Table 6.4. In the following, we present the main results for the three peak cohesion fields, and we discuss the main characteristics obtained for typical slumping mechanics.

Homogeneous peak cohesion field The homogeneous solution gives preliminarily interesting results (see Fig. 6.14). The first-order characteristics of a slump can be observed, even though their magnitude is relatively fair compared to the real slump. The most striking feature is the development of one major shear zone, along which the material flows (*i.e.,* depletion) towards the toe of the slump, resulting in a compression zone (*i.e.,* thrusting and folding deformations). The crown-like structure develops linearly along the $y-$direction and is highly localized at the surface of the slump (at $x \approx 20$ m in Fig. 6.14). However, the material flows homogeneously along the $x-$direction (see the vertical profile in Fig. 6.14), as shown by the displacement field. The lateral variation of the displacement field (along the $y-$direction) is almost non-existent, which is mainly due to the spatial homogeneity of the peak cohesion field.

Isotropic Gaussian covariance Considering heterogeneities with a Gaussian covariance function for the cohesion field, the displacement field starts to resolve a differential behaviour (see Fig. 6.15). Higher and/or weaker values of the peak cohesion field yield lower and/or greater displacements. This is obvious, especially in the transition zone where this differential is observable. In addition, the compression zone also starts to resolve spatial variations due to weaker and stronger cohesion values.

A striking difference is the shear zone itself (see Fig. 6.20): the shear zone exhibits a more complex spatial pattern, whereas only one major shear zone is observed in Fig. 6.20. Retrogressive shear banding appears during the time evolution of the slump, which suggests the development of a secondary shear zone within the slump. Moreover, the crown-like structure is now curved and not linear along the $y-$direction. Its spatial extent is more important and is

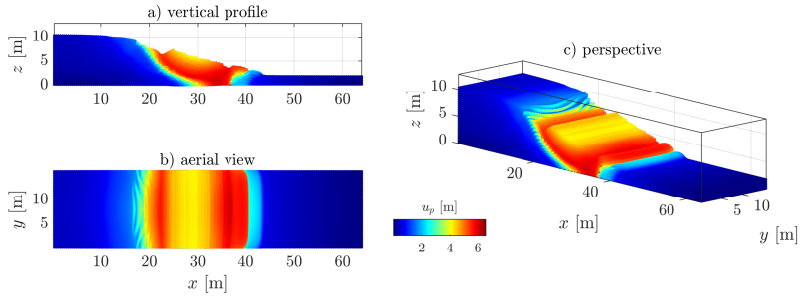


Figure 6.14. | Displacement field obtained after $t = 15$ s for a homogeneous peak cohesion field. One can see an overall homogenous displacement field with some of the first-order characteristics of a slump, *i.e.*, a rotational displacement with a compression zone at the toe, a transition zone delimited by one principal shear zone and a major scarp at the top of the material.

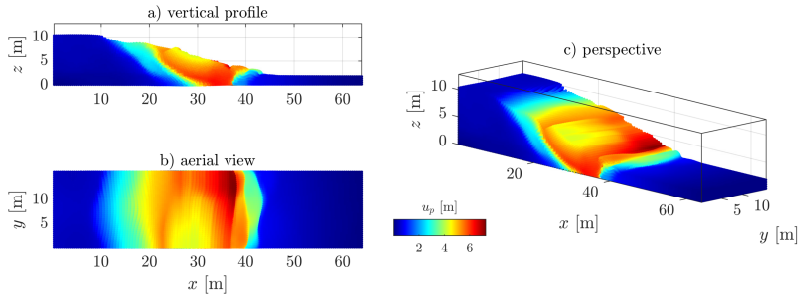


Figure 6.15. | Displacement field obtained after $t = 15$ s for a heterogeneous peak cohesion field with a Gaussian covariance function.

not as localized as in the homogeneous case. Nevertheless, a more complex arrangement of major and minor scarps within the crown-like structure has not yet been observed. Such a structure is more evident if one observes the accumulated equivalent plastic strain ϵ^p_{eqv} in Fig. 6.20 in Appendix 6.6.

The high magnitude of the displacement field in the areas $x \in [20; 40]$ m and $y \geq 8$ m is due to a weaker zone in the peak cohesion field (see Fig. 6.20). This shows a strong influence of the heterogeneous peak cohesion field on the final displacement field. A lower shear strength of the material yields faster strain-softening behaviour, promoting a faster response of shear banding.

Isotropic exponential covariance Shear banding activities become even more complex when an exponential covariance function is used, relative to Fig. 6.14 and even with Fig. 6.15 to some extent. The spatial distribution of the peak cohesion (see Fig. 6.21) resolves finer heterogeneities with a smaller length scale compared to when Gaussian covariance is used. Principal differences are observed at the top and toe of the slump, where the crown-like structure turns into a complex zone made of minor and major scarps (see Fig. 6.16). The displacement field becomes highly heterogeneous, particularly at the toe and the top of the slump. However, it is also more homogeneous when compared with Fig. 6.15, particularly in $x \in [25; 35]$. The difference is evident between Figs. 6.17 & 6.15 at this particular location.

The difference between the Gaussian and exponential covariance of the peak cohesion suggests the following. Heterogeneous displacement fields could be influenced by larger and/or coarser fluctuations of the shear strength within the material. By extrapolation, this could imply that the magnitude of the heterogeneity might be related to the fluctuation scales of the peak cohesion field. Locally rather homogeneous fluctuations of the peak cohesion (*i.e.,* Gaussian covariance) seem to promote an increasingly heterogeneous displacement field at the surface. The characteristic length scale of spatial fluctuations could have important implications for highly heterogeneous displacements within landslides. The same assumption could hold for understanding the more complex crown-like structure of slumps (see Fig. 6.21)

6.5. Discussion

6.5.1. GIMPM suitability

We investigated granular collapses in both three-dimensional and plane strain configurations. Our numerical results demonstrated the suitability of GIMPM to correctly reproduce experimental granular collapses. They also demonstrated that the results did not significantly differ between these two spatial configurations and that both approaches give similar numerical solutions.

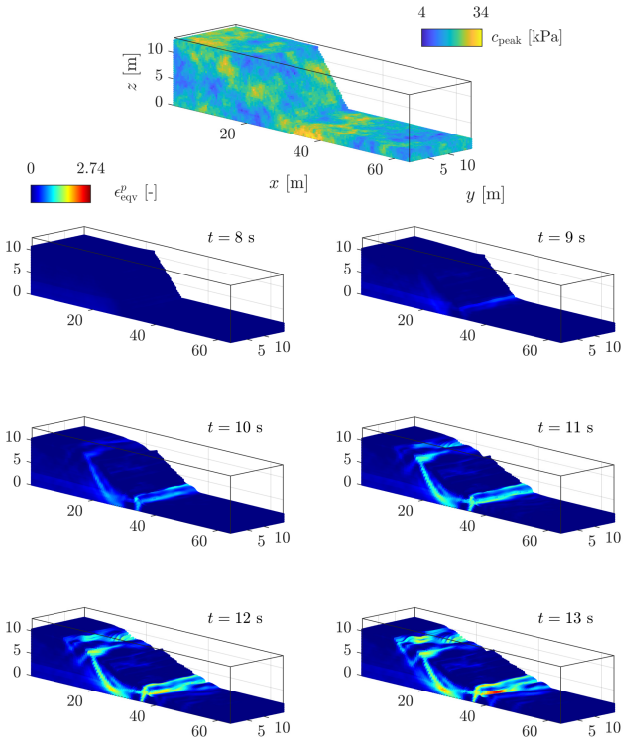


Figure 6.16. | Heterogeneous cohesion field with an exponential covariance function: time evolution of the equivalent plastic strain ϵ^p_{eqv}. Similar to Fig. 6.20, heterogeneous behaviour is observed. However, the exponential covariance function results in an even more complex pattern of strain localization, *i.e.*, minor and major scarps develop at the top. The crown-like structure of the slump becomes even more heterogeneous.

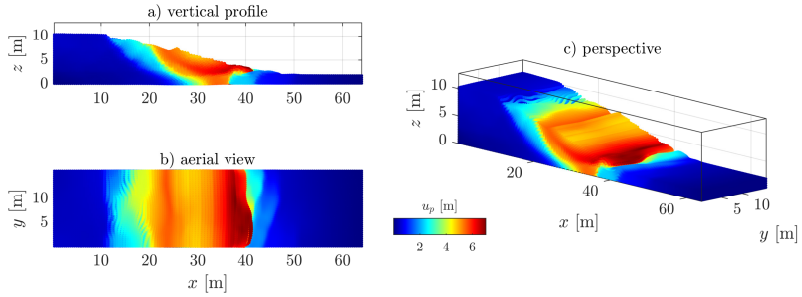


Figure 6.17. | Displacement field obtained after $t = 15$ s for a heterogeneous peak cohesion field with an exponential covariance function.

6.5.2. Collapse limitation

For Model 1a, the principal hardware limit is the on-chip memory of the GPU. Even though RTX3090 is supported by 24 GB DDR4, it is physically impossible to achieve the resolution used for plane strain granular collapse. This would require more than 24 GB of on-chip memory. However, a multi-GPU implementation using the message passing interface API (MPI) could resolve this hardware limitation when using only one GPU. As demonstrated by Räss et al., 2019a; Alkhimenkov et al., 2021, it is possible to overlap communications between multiple GPUs using asynchronous kernel executions. This allows us to achieve an optimal parallel efficiency: Räss et al., 2019a; Alkhimenkov et al., 2021 reported an efficiency of 95-98 % of the weak scaling tests involving up to 128 GPUs. Model 1b demonstrated the importance of the background mesh resolution over strain localization. Using a higher numerical resolution (*i.e.,* finer background mesh) allows full resolution plastic strain localization. Similarly, future additional development efforts towards MPI implementation could resolve highly detailed three-dimensional granular collapse simulations in the future. This will definitely benefit future studies on complex strain localization.

Regarding the performance for Model 1b, the wall-clock time is $t_{\mathrm{GPU}} = 1470.5$ s (25 min), and the number of iterations per second is 85.5 it·s^{-1} for $n_{mp} = 819200$. As a preliminary example, the same numerical model was performed by Wyser et al., 2020a, who reported 19.98 it·s^{-1} for $n_{mp} = 12800$. Proportionally, this corresponds to a performance gain factor of 275 for the GPU-based implementation (`ep2-3De v1.0`) over the MATLAB-based implementation (fMPMM-solver v1.1) (Wyser et al., 2020a).

6.5.3. Performance

The performance analysis we carried out in Model 2a demonstrated that even though the algorithm heavily relies on atomic operations to accumulate material point quantities on the nodes, the effective memory throughput reaches 88 % at most (for Tesla V100). We expected a much lower throughput due to the use of these atomic operations, since they are likely known to undermine the computational performances of an implementation under previous GPU architectures (*e.g.,* Kepler) (Dong et al., 2015; Dong et al., 2018; Gao et al., 2018). Our actual understanding (at least for a GPU-based implementation of GIMPM) is that the latest GPU architecture (Ampere and Turing) is now efficient when dealing with atomic operations and that the need to use a complex data layout for scattering is not as important as before. Furthermore, we identify the memory throughput as the main bottleneck: an additional 12 % performance improvement on the V100 before reaching the hardware limit of the memory bandwidth. The A100 shows that the solver reaches the hardware limit with an effective memory throughput which is very close (*i.e.,* 97 %) to the actual maximum memory throughput. Similarly, the true limiting factor of our implementation is the hardware limit of the GPU on-chip memory. Fur-

ther development efforts should be directed towards an MPI implementation of `ep2-3De v1.0`.

6.5.4. Slumping mechanics

We show the application of the GIMPM solver `ep2-3De v1.0` for slumping mechanics. We have presented various slump results and demonstrated the significant influence of heterogeneities within the peak cohesion field over the displacement field or the equivalent plastic strain. However, we have arbitrarily selected values that resulted in severe deformations of the material, which we wanted to highlight to demonstrate the potential of the solver. Further efforts should now be oriented towards numerical models that are closer to real and well-documented cases, such as in Tran et al., 2019a; Ying et al., 2021. Despite the simplifications we made, we have reported three-dimensional simulations that resolve all the first-order characteristics of slumps, including complex major and minor scarps, different shear zones of various activities and a complex arrangement within the compression zone. The use of three-dimensional GIMPM implementation under a GPU architecture will highly benefit future studies in the field, allowing faster and detailed numerical simulations of heterogeneous and complex strain localization problems.

6.5.5. Code portability

Our numerical models showed the efficient computing capabilities of modern GPUs under the latest Nvidia GPU architectures. An important concern is the code portability. CUDA C is only applicable for Nvidia's GPUs and is not yet compatible with other corporation's GPUs, such as AMD (ATI Technologies). As such, an extension of the `ep2-3De v1.0` solver towards an OpenCL-based implementation would ensure better code portability in the future.

6.6. Conclusion

We developed `ep2-3De v1.0`, an explicit GPU-based implementation of the generalized interpolation material point method that exploits the capabilities of the most recent GPU architectures (Ampere, Turing and Volta). We achieved fast execution times on a single GPU with a scattering approach that relies on extensive usage of atomic operations. We report, at most, an effective memory bandwidth of 88 % relative to the maximal hardware capabilities of the GPUs. We achieve, at most, a performance gain of 200× compared to a single-threaded CPU-based implementation of the solver. On entry-level customer electronics GPUs, we report a performance gain of $\approx 10\times$. We also report that the memory bandwidth is the main limiting performance factor. We validated our solver against the well-known experimental results of the granular collapse problem in a three-dimensional configuration. Furthermore, we show applications of the solver to model slumping mechanics considering different material heterogeneities.

Recent GPU architectures (Ampere, Turing and Volta) have certainly been optimized by Nvidia, increasing the efficiency of native atomic operations (*i.e.*, scattering), as was suggested before by previous studies. This is encouraging, and future implementations on GPUs might be more straightforward and could now focus on using atomic operations instead of complex parallel implementations to avoid race conditions between threads without suffering from a dramatic loss of computational performance.

The single GPU implementation we propose here should be completed in the future by taking advantage of the message passing interface (MPI) and the computing power of a multi-GPU implementation to overcome the other limiting factor of `ep2-3De v1.0`, the GPU on-chip memory. Having such an implementation ensures that one can use a computationally powerful explicit GIMP solver to address complex elastoplastic problems in the field of geomechanics.

Code availability The solver `ep2-3De v1.0` developed in this study is licenced under the GPLv3 free software licence. The solver `ep2-3De v1.0`[2] archive (v1.0) is available from a permanent DOI repository (Zenodo) at: https://doi.org/10.5281/zenodo.4966590 (Wyser et al., 2021).

[2]The latest version of the code is available for download from Bitbucket at: https://bitbucket.org/ewyser/ep2-3de/src/master/ (last access: June 16, 2021).

Appendix A: GIMPM basis functions and derivatives

One of the most important problems of any sMPM formulation is the cell-crossing instability (or error, see Steffen et al. 2008b; Wilson et al. 2021). As material points move through the mesh, they cross element boundaries. The discontinuous gradient due to the C_0 continuity of the basis functions results in spurious oscillations of the stress field and internal forces (González Acosta et al., 2020; González Acosta et al., 2019; Bardenhagen et al., 2004) when material points cross element boundaries.

To solve for this instability, Bardenhagen et al., 2004 introduced the generalized interpolation material point method (GIMPM). Whereas the material point is treated as a point in sMPM, Bardenhagen et al., 2004 assigned a *spatial extent* or a *domain* to the material point. Alternative basis functions are constructed, *i.e.,* to consider the material point domain, as follows:

$$\phi_{np} \equiv \phi_n(\boldsymbol{x}_p) = \frac{1}{v_p} \int_{\Omega_p \subset \Omega} \chi_p(\boldsymbol{x}) N_n(\boldsymbol{x}) \mathrm{d}\Omega, \tag{6.6.1}$$

where v_p is the material point volume, Ω_p denotes the material point domain, $\chi_p(\boldsymbol{x})$ is the *particle characteristic function*, $N_n(\boldsymbol{x})$ is the basis function (or shape function) for the mapping between the material point p and its associated nodes n, and $\boldsymbol{x} = \boldsymbol{x}_p - \boldsymbol{x}_n$ are the local coordinates between node n and material point p.

The particle characteristic function must satisfy the partition of unity property, *i.e.,* $\sum_p \chi_p(\boldsymbol{x}) = 1$ (Bardenhagen et al., 2004). The simplest particle characteristic function is given by the hat function, *i.e.,*

$$\chi_p(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} \subset \Omega_p, \\ 0 & \text{otherwise.} \end{cases} \tag{6.6.2}$$

The GIMPM basis functions and derivatives are constructed analytically (Coombs et al., 2020b; Charlton et al., 2017) in one dimension from a convolution of the standard finite element basis functions and the material point characteristic function (Steffen et al., 2008b), *i.e.,*

$$\phi_n(x_p) = \begin{cases} 1 - (4x^2 + l_p^2)/(4hl_p) & \text{if } |x| < l_p/2 \\ 1 - |x|/h & \text{if } l_p/2 \leq |x| < h - l_p/2 \\ (h + l_p/2 - |x|)^2/(2hl_p) & \text{if } h - l_p/2 \leq |x| < h + l_p/2 \\ 0 & \text{otherwise ,} \end{cases} \tag{6.6.3}$$

where l_p is the length of the material point domain, h is the mesh resolution, and $x = x_p - x_n$, where x_p is the coordinate of a material point and x_n is the coordinate of its associated node n. The two-dimensional basis function of a node n with its material point p is constructed as

$$\phi_{np} \equiv \phi_n(\boldsymbol{x}_p) = \phi_n(x_p)\phi_n(y_p), \tag{6.6.4}$$

for which the gradient is defined as

$$\nabla\phi_{np} \equiv \nabla\phi_n(\boldsymbol{x}_p) = (\partial_x\phi_n(x_p)\phi_n(y_p), \phi_n(x_p)\partial_y\phi_n(y_p)). \tag{6.6.5}$$

Appendix B: Gaussian random cohesion fields

In Earth Sciences, random fields (Christakos, 1992) are numerically generated predictions of a geophysical property (*i.e.,* rock- or soil-related properties) with probabilistic spatial variability. These predictions are based on i) an assumed probability density function, *i.e.,* characterized by a mean value μ with a standard deviation σ, and ii) an assumed spatial correlation function, characterised by fluctuation scales in a vector format, *i.e.,* $\boldsymbol{\lambda} = (\lambda_x, \lambda_y, \lambda_z)$. In regard to numerical modelling, the principal requirement is that both small and large scales are simultaneously resolved over the computational mesh to ensure physically meaningful solutions.

Recently, Räss et al., 2019b presented an efficient implementation based on a spectral representation of Gaussian random fields for geophysical applications using either Gaussian or exponential covariance functions. The numerical codes, named GRFS, were made available by Räss et al., 2019b in both native MATLAB and CUDA C languages [3]. However, a sufficiently large number of harmonics should be used to obtain convergent Gaussian random fields, as stated in Räss et al., 2019b.

Similar to the random material point method (RMPM, see Wang et al. 2016a; Liu et al. 2019; Remmerswaal et al. 2021) initially proposed by Fenton et al., 1990 to generate RFs for a finite element mesh (RFEM), we combined this approach with the codes proposed by Räss et al., 2019b to generate an isotropic peak cohesion field to demonstrate its influence on the mechanical behaviour.

Appendix C: Volumetric locking and damping corrections

In Huang et al., 2015, no volumetric locking mitigation strategy was introduced, even though tough low-order elements were used. This should result in severe volumetric locking issues and an overall stiffer response of the granular material. In addition, Huang et al., 2015 used the standard (or original) material point method (instead of the generalized interpolation material point method), which is well known to introduce spurious oscillations of internal forces (González Acosta et al., 2020).

When implementing the proposed volumetric locking mitigation strategy, we observed a) larger deformations of the granular material with a stronger vertical compaction (*i.e.,* stronger vertical displacement) and b) slightly longer run-out distances when compared to the experimental data. The softer mechanical response of the granular material had to be compensated somehow, which can be achieved by the introduction of a small local damping parameter.

We reproduced the numerical setting used in Huang et al., 2015 with the

[3] available at TheroutinesGRFSareavailableathttps://bitbucket.org/lraess/grfs/src/master/

same mesh resolution, *i.e.*, $\Delta x = \Delta y = 2.5$ mm, and a similar number of material points $n_{mp} = 28800$ with an initial number of material points per initially filled element $n_{pe} = 9$. The material parameters used for this preliminary investigation are presented in §6.4.1.

Figure 6.18 a) & b) shows the major differences between either a locking-free or a locking-prone solution and the experimental results. As mentioned before, a slightly longer run-out distance is obtained for the locking-free solution. As a result, the numerical prediction given by the locking-free solution of the free surface is underestimated. However, the most noticeable difference is the failure surface. Whereas the failure surface predicted by the locking-prone solution fits with the experiment of Bui et al., 2008, it diverges for a locking-free solution. In particular, the onset of the failure surface at the top of the material is underestimated by the locking-free solution compared to the experimental results. This is due to the softer response of the granular material when volumetric locking is mitigated, which promotes greater vertical compaction and stronger run-out distance at the same time.

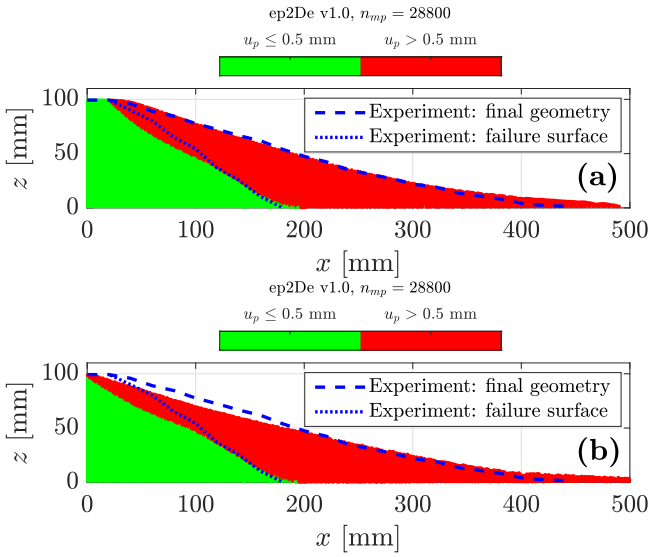


Figure 6.18. | a) Numerical solution without any volumetric locking strategy and b) numerical solution with the proposed volumetric locking strategy. For both cases, no damping is introduced.

Even though the introduction of local damping better resolves the experimental results, one can argue that the locking-free solution without the introduction of local damping still agrees with the experiment of Bui et al., 2008. The overall response of the numerical granular collapse is still very close to the actual physical experiment, and the differences between the numerical and experimental results can still be considered acceptable.

Appendix D: Heterogeneities for the peak cohesion field

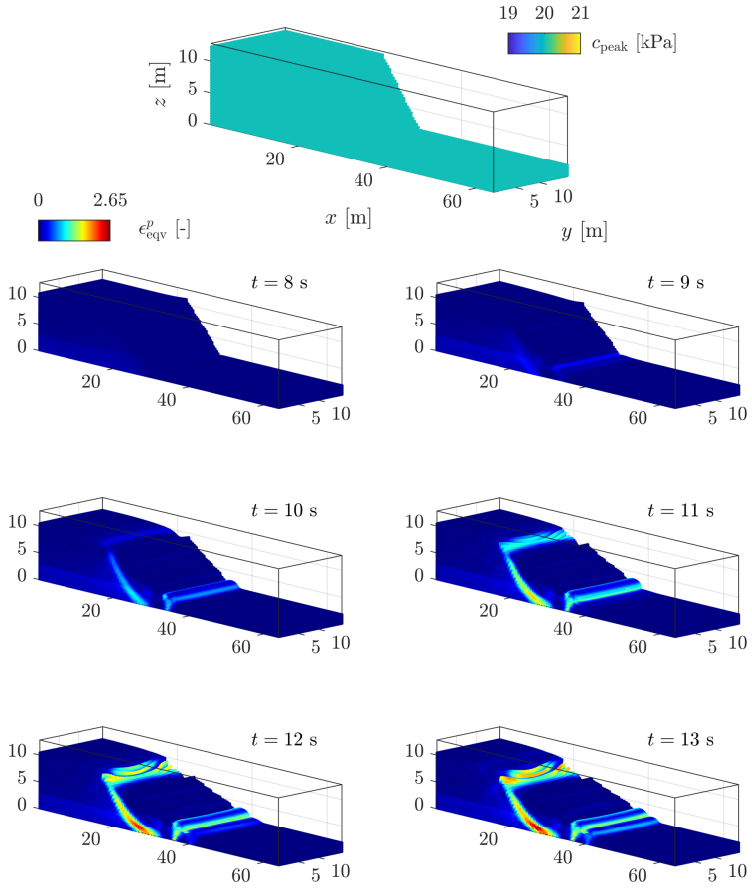


Figure 6.19. | Homogeneous cohesion field: time evolution of the equivalent plastic strain ϵ^p_{eqv}. Its evolution is rather homogeneous, and the overall plastic behaviour is free of any heterogeneities. Some of the first-order characteristics are observed, *i.e.*, a principal shear zone and a compression zone at the toe of the slump.

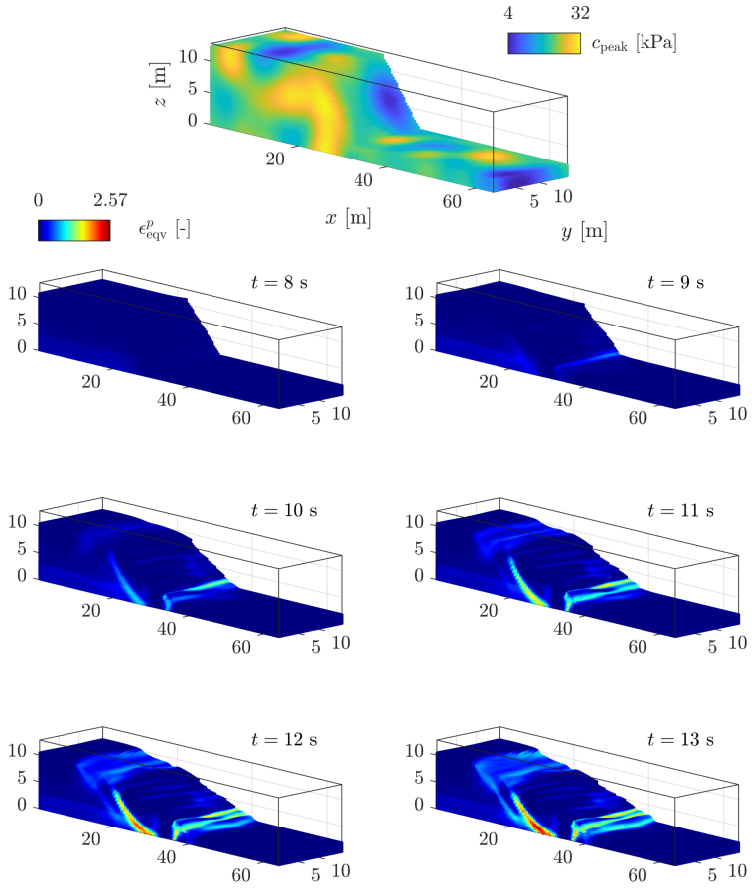


Figure 6.20. | Heterogeneous cohesion field with a Gaussian covariance function: time evolution of the equivalent plastic strain ϵ^p_{eqv}. Unlike Fig. 6.19, heterogeneous behaviour is observed, *i.e.*, the appearance of a second shear zone highlights a more complex deformation pattern. Moreover, a crown-like structure starts to develop at the top of the material, where an initial weak zone is located.

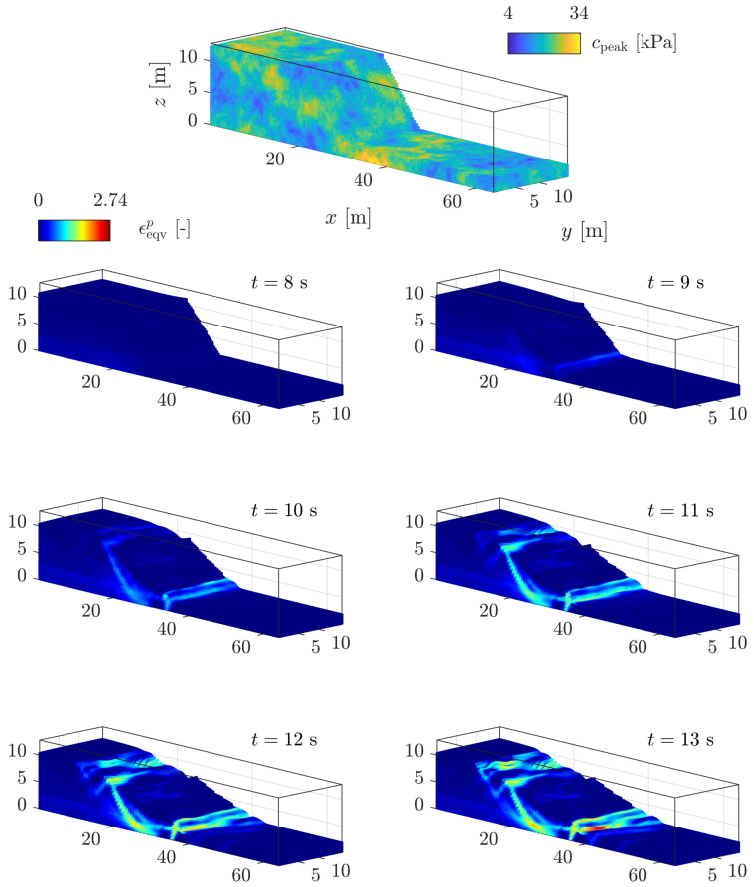


Figure 6.21. | Heterogeneous cohesion field with an exponential covariance function: time evolution of the equivalent plastic strain ϵ^p_{eqv}. Similar to Fig. 6.20, heterogeneous behaviour is observed. However, the exponential covariance function results in an even more complex pattern of strain localization, *i.e.*, minor and major scarps develop at the top. The crown-like structure of the slump becomes even more heterogeneous.

Author contributions EW and YA wrote the original manuscript and developed the first version the `ep2-3De v1.0` solver. MJ and YP supervised the early stages of the study and provided guidance. All authors have reviewed and approved the final version of the paper.

Competing interests The authors declare that they have no conflicts of interest.

Acknowledgements Yury Alkhimenkov gratefully acknowledges support from the Swiss National Science Foundation (grant no. 172691). Yury Alkhimenkov and Yury Y. Podladchikov gratefully acknowledge support from the Russian Ministry of Science and Higher Education (project No. 075-15-2019-1890).

7. Granular collapses

Analytical and numerical solutions for three-dimensional granular collapses

Emmanuel Wyser, Yury Alkhimenkov, Michel Jaboyedoff & Yury Y. Podladchikov

In consideration for submission in *Nature Communication* or *Granular Matter*

Abstract In this work, we propose a combined approach between analytical, numerical and experimental investigations of dry granular collapses considering a three-dimensional setting. Using a novel experimental apparatus, we investigate granular collapses in laboratory. We show that from a quasi-static understanding of granular collapses, an accurate prediction of final normalized run-out distances can be recovered for dynamic granular collapses. To resolve dynamic granular collapses, we relate the angle of repose to the initial aspect ratio of the granular column. We also show that the material point method is a valid candidate to model granular collapses under a three-dimensional configuration. We finally validate our in-house solver under an explicit formulation with experimental evidences. We report good agreements between numerical and experimental results. This demonstrates our in-house solver as an effective tool for the three-dimensional modelling of granular collapses..

7.1. Introduction

The angle of repose θ_c is a dominant feature of non-cohesive granular material and expresses the maximum angle above which the material starts to flow (Lowe, 1976; Barabási et al., 1999; Tsuji et al., 2018). The material is stable below this angle. It varies from 25° for smooth particles to 45° for angular particles (Carrigy, 1970; Balmforth et al., 2005; Pohlman et al., 2006). Dry sands allow for typical values around 35° whereas it is much more important under wet condition with values around 90° or even greater (Mitarai et al., 2006). Such angle is related to the friction coefficient μ (Lee et al., 1993), i.e. $\mu = \tan(\theta_c)$ (see Mitarai et al., 2006 for further details). Other parameters dictate the shape of a granular pile, such as the gravity, grain properties (e.g. roughness, sphericity and grain size (Miller et al., 1966; Buffington et al., 1992)), the number of particles involved (Grasselli et al., 2000; Kleinhans et al., 2011; Al-Hashemi et al., 2018) and external perturbations, *i.e.,* vibrations leading to fluidization toward a relaxation stage (Tsuji et al., 2018).

The collapse of a dry or wet granular column(Mangeney et al., 2010; Rondon et al., 2011; Kumar et al., 2017; Bougouin et al., 2018) is a well-known problem, for which various experimental and numerical works (Staron et al., 2005; Lube et al., 2005; Lagrée et al., 2011; Dunatunga et al., 2015) have been achieved. A fundamental metric is the initial aspect ratio of the column $\lambda_0 = h_0/r_0$, where h_0 and r_0 are the initial height and the radius of the column respectively.

The following well-established scaling law (Staron et al., 2005; Thompson et al., 2007; Warnett et al., 2014; Langlois et al., 2015) relates λ_0 with $(r_\infty - r_0)/r_0$, *i.e.,*

$$\frac{r_\infty - r_0}{r_0} = \begin{cases} \alpha\lambda_0 & , \lambda_0 < \lambda_c, \\ \alpha\lambda_0^\gamma & , \lambda_0 \geq \lambda_c, \end{cases} \tag{7.1.1}$$

where α and γ are material dependent coefficients (Balmforth et al., 2005; Mériaux, 2006), λ_c is the critical initial aspect ratio and r_∞ the final radii of the column, *i.e.,* the final run-out distance.

A transition phase exists, according to the initial aspect ratio, *i.e.,* a change from a truncated cone to a cone shape (Lo Giudice et al., 2019). Lube et al., 2005; Staron et al., 2005; Lagrée et al., 2011 proposed a power scaling law to $(r_\infty - r_0)/r_0$ for high λ_0 and a linear scaling for low λ_0. Even tough such behaviour is well observed in many studies, the value of this transition remains actively discussed.

Lajeunesse et al., 2004 proposed the following semi-empirical equation to fit to their experimental results

$$\frac{r_\infty}{r_0} = \begin{cases} \frac{1}{2\tan(\theta_c)}\left(\lambda_0 + \left(4\tan^2(\theta_c) - \frac{\lambda_0^2}{3}\right)^{1/2}\right) & , \lambda_0 < 0.74, \\ \left(\frac{3\lambda_0}{0.74}\right)^{1/2} & , \lambda_0 \geq 0.74, \end{cases} \tag{7.1.2}$$

which implies that $\theta_c = \text{cst.}$, modulated by the variation of λ_0. Such formulation was in agreements with experimental evidences, *i.e.,* a constant final height $h_\infty \approx \min(h_0, r_0 \tan(\phi_y))$ where $\phi_y \approx 36.5°$ is the internal yield angle. We refer to it as a solution to dynamic deformations of the granular column.

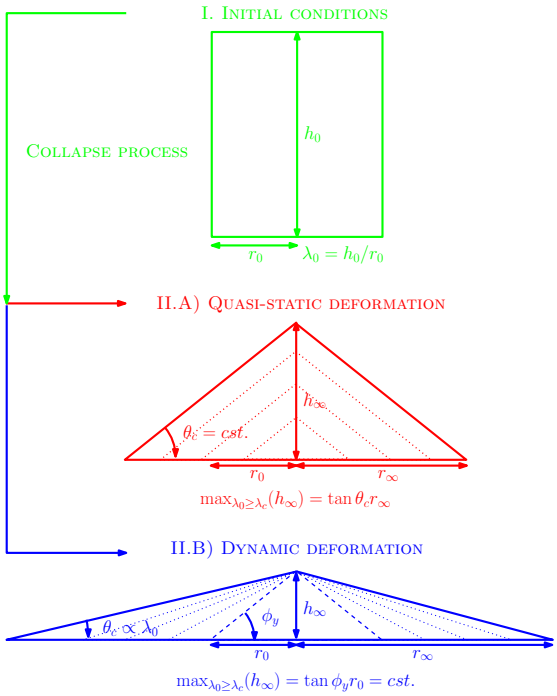


Figure 7.1. | General scheme of hypothetical quasi-static deformations and the actual experimental evidences of dynamic deformations leading to a constant h_∞. The dotted lines correspond to increasing volumes with respect to an increasing λ_0. In the case of II.B (dynamic deformations), θ_c' naturally decreases as λ_0 increases.

We propose an analytical solution to the final run-out distance of the granular continuum, for which we assume quasi-static deformations, *i.e.,* $h_\infty = r_\infty \tan(\theta_c)$ whereas Lajeunesse et al., 2004 proposed $h_\infty \approx \min(h_0, r_0 \tan(\phi_y))$, *i.e.,* dynamic deformations. These two understandings are summarized in Fig. 7.1. In the case of quasi-static deformations, the granular pile vertically grows in proportion with λ_0 and, we assume that $\theta_c \approx \phi_y$.

We further compare our analytical solution to i) an experimental data collection using a newly designed apparatus and, ii) the reference solution provided by Lajeunesse et al., 2004. Expected differences between our quasi-static solution highlight the characteristic smooth transition between quasi-static and dynamic deformations of the column. A solution to unify our quasi-static hypothesis with the dynamic understanding of Lajeunesse et al., 2004 is then provided. Finally, our experimental results are compared to numerical solutions under three-dimensional configurations throughout a material point framework, a well-suited numerical method to resolve large deformation mechanics Baumgarten et al., 2019b; Fern et al., 2019; Fern et al., 2016; Soga

et al., 2016.

7.2. Methods

7.2.1. Laboratory experiments

Experimental setup and data collection

We propose a newly designed apparatus which avoids significant influences of the container's walls over the dynamic of the flowing mass. In previous experiments, the cylinder is rapidly raised upward and boundary influences at the interface could not be fully controlled. Our experimental apparatus consists in a cylinder made of three independent shells which quickly opens radially outward, thanks to a performant pneumatic system. This ensures consistency and reproducibility over the experiments and avoids significant influences of the boundaries. The granular mass is released and freely flows on a rough wooden surface (Fig. 7.2). Measurements of the final run-out distance are proceeded once the column has fully relaxed.

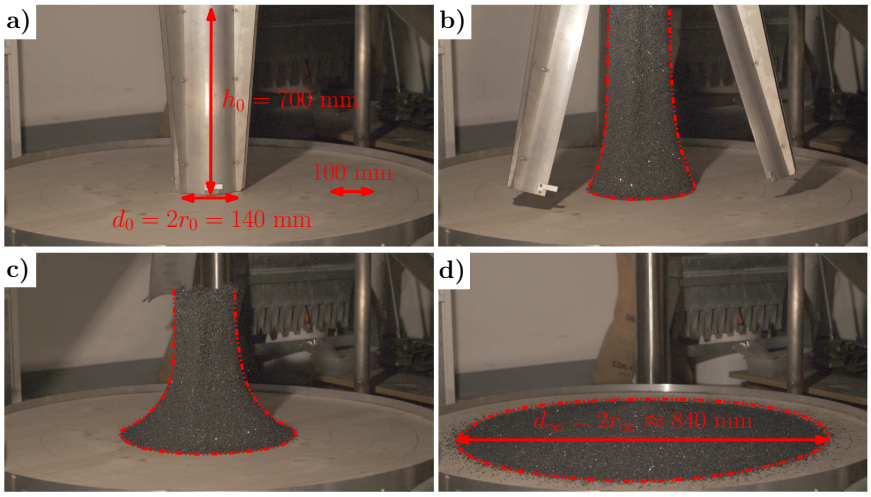


Figure 7.2. | Close-up picture of the newly designed apparatus used in this study. The cylinder of dimension 140×700 mm is filled with granular material and, (a) quickly opens radially outward from the column's centre, (b) allowing the granular mass to (c) flow freely. The final run-out distance is then measured when (d) the collapse has completely relaxed. The initial aspect ratio is typically $\lambda_0 \leq 10$.

The maximal run-out distance is defined at the dense front of the granular mass, *i.e.,* we do not consider the maximal extent of single grains because their final position results from of their previous gaseous state. Measurements are carried 6 times at different radial location with respect to the centre of the column. The final runout distance is the average of theses 6 measurements.

The granular mass is made of slighlty polydisperse and highly angular Silicon carbide beads (SiC) of a density $\rho_{\mathrm{SiC}} = 3.21$ g·cm^{-3} and an average bead

diameter $r_b \approx 0.11$ cm. This results in a rough estimation of the grain mass $m_b = 0.02$ g (*i.e.,* assuming an equivalent spherical shape), with a relatively high friction coefficient $\mu \approx 0.77$ which was determined by laboratory measurements we detail in the following subsection.

Experimental estimation of the angle of repose θ_c

To measure the angle of repose of the granular material used in this study, we filled a vertical cylinder with materials and very slowly raised the container upward. This produced a granular pile which we assumed to be representative of a quasi-static relaxation of the granular material. We further took pictures of the pile and repeated the procedure 125 times using a camera CANON EOS 450D. We further processed the pictures by measuring the angle of both sides of the granular pile to determine the angle of repose over $n = 250$ measurements. Fig. 7.3 shows the simple procedure to determine θ_c.

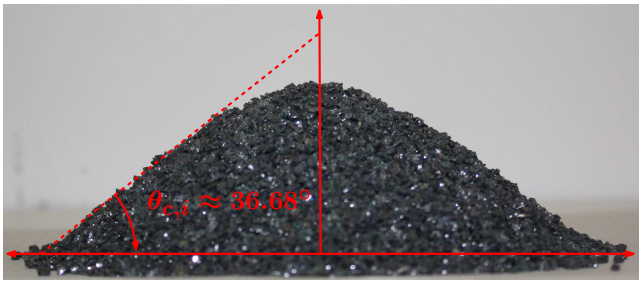


Figure 7.3. | Quasi-static relaxation of a granular material initially contained within a cylinder.

Using the commercial software Agisoft Lens, we give a rough estimation of the maximal radial distortion to be 0.18 % at the edge of images, *i.e.,* corresponding to a metric distortion of 10^{-4} mm considering an average distance of the sensor $d = 600$ mm. Such distortion appears reasonably low enough to be negligible.

We further calculate a cumulated average angle of repose given by

$$\langle \theta_c \rangle_n = \frac{1}{n} \sum_{i=1}^{n} \theta_{c,i}, \tag{7.2.1}$$

where $n = 250$ angles of repose. This allows to identify the average value for which an equilibrium is reached, *i.e.,* $\partial_n \langle \theta_c \rangle_n \to 0$. The Figure 7.4 shows overall results of the experimental measurements of the angle of repose. This yields an average angle of repose $\theta_c = 37.55° \pm 0.29°$ and thus, $\mu = 0.77$.

7.2.2. A continuously smooth piece-wise analytical solution

To define a solution $\mathcal{R}(\theta_c, \lambda_0) \equiv (r_\infty - r_0)/r_0$ resulting from the quasi-static relaxation of a granular column as a functional of $\tan(\theta_c)$ and λ_0, let us assume,

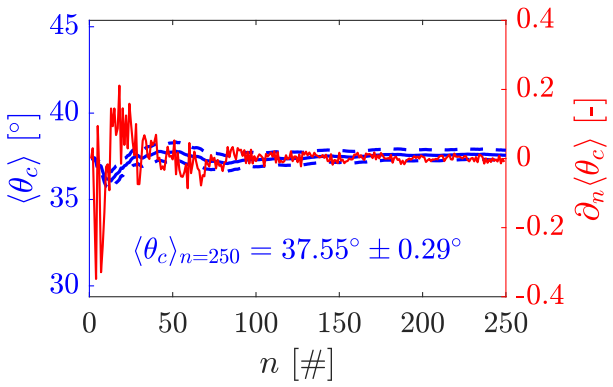


Figure 7.4. | The cumulative average value of the angle of repose is shown by the blue solid line (with the dashed-dotted blue lines indicating the standard deviation of the cumulated average value) and the derivative of the cumulative average value $\langle\theta_c\rangle$ as a function of the number of measurements n.

based on Fig. 7.5, that i) the mass is conserved, and ii) both the angle of repose θ_c and initial aspect ratio λ_0 govern the relaxation of the granular column, *i.e.*, a granular pile due to quasi-static deformations during which the internal energy is dissipated by slow frictional interactions only.

The initial volume of the cylinder is given by $V_0 = \pi r_0^2 h_0$ and the final volume is given by $V_\infty = 1/3\pi r_\infty^2 h_\infty$ for a cone or $V_\infty = 1/3\pi(r_\infty^2 + r_\infty \Delta r + \Delta r^2)$ for a truncated cone. As demonstrated in Fig. 7.5, a transition occurs at $\lambda_c := \tan(\theta_c)\sqrt{3}$, *i.e.*, when $\lambda_0 \to \lambda_c$.

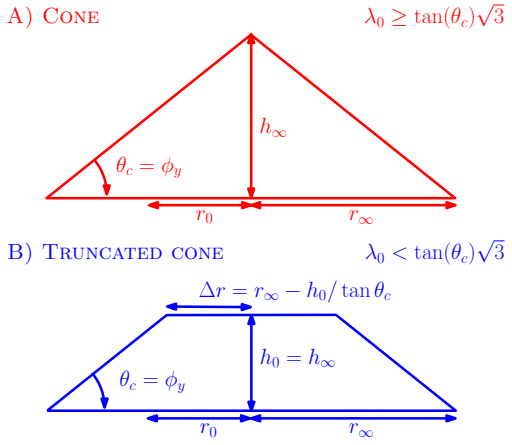


Figure 7.5. | Scheme of the relaxed granular column governed by the angle of repose θ_c and the initial aspect ratio λ_0, under the quasi-static deformation assumption.

Considering the relevant dimensions of the problem (Fig. 7.5), the mass conservation implies $V_i = V_\infty$ (*i.e.*, the granular material is assumed to be incompressible). Equating volumes, substituting and collecting terms leads to

the following

$$\mathcal{R}(\theta_c, \lambda_0) = \begin{cases} \frac{1}{2}\left(\frac{\lambda_0}{\tan(\theta_c)} + \left(4 - \frac{\lambda_0^2}{3\tan^2(\theta_c)}\right)^{1/2}\right) - 1 & , \lambda_0 < \tan(\theta_c)\sqrt{3}, \\ \left(\frac{3\lambda_0}{\tan(\theta_c)}\right)^{1/3} - 1 & , \lambda_0 \geq \tan(\theta_c)\sqrt{3}. \end{cases} \tag{7.2.2}$$

Such analytical solution assumes quasi-static deformations of the column. To consider dynamic deformation of the column, we relate the angle of repose θ_c with the initial aspect ratio λ_0, *i.e.,* $\theta_c = f(\lambda_0)$. The latter is no longer constant, but depends on the initial aspect ratio of the column λ_0, and is given by

$$\theta_c(\lambda_0) = \tan^{-1}\left((\tan^3(\phi_y)/(3\lambda_0))^{1/2}\right). \tag{7.2.3}$$

Inserting Eq. 7.2.3 within Eq. 7.2.2 in a similar formulation with respect to Eq. 7.1.2 (Lajeunesse et al., 2004).

7.2.3. Numerical simulation

The Material Point Method (MPM), originally proposed by Sulsky et al., 1994, is an extension of the particle-in-cell method. The weak form of momentum equations are solved on an eulerian background mesh and updated nodal solutions are further mapped to the material points. They can be regarded as Gauss points that are allowed to move. State variables, *e.g.,* stresses or displacements, are transported by the material points. This allows MPM to handle large deformations, such as in granular flows.

In order to resolve the three-dimensional dynamics of the granular collapse, we performed numerical simulation using the material point method, *i.e.,* its variant called generalized interpolation material point method (Bardenhagen et al., 2004). We implement an explicit MPM solver which takes advantages of modern graphics processing unit (GPU) architectures, *i.e.,* `ep2-3De v1.0`[1]. Further details of the implementation of the solver can be found in Wyser et al., 2020c; Wyser et al., 2021. Because of the large deformations involved during the collapse, we selected the uGIMP variant, *i.e.,* the material point's spatial extent is constant (see Coombs et al., 2020a for limitations of this variant). A non-associated Drucker-Prager elasto-plastic model is selected to reproduce the granular collapse.

7.2.4. Numerical parameters and geometry

Recently, Nguyen et al., 2020 showed that density and stiffness properties have a negligible effect on the morphology and the run-out distance of granular collapses. We used the same values as in Nguyen et al., 2020, *i.e.,* a density $\rho = 2000$ kg·m^{-3}, with the Young's modulus $E = 5.84$ MPa, a cohesion $c = 0$

[1] The latest version of the solver is available for download from Github at: https://github.com/ewyser/ep2-3De (last access: August 10, 2021).

kPa and a Poisson's ratio $\nu = 0.3$. The friction angle is $\phi = 37.55°$, *i.e.,* $\phi = \langle\theta_c\rangle$. We also introduce a local damping D, for which $D \in [0.05; 0.1]$ is a commonly accepted range for explicit formulations (Wang et al., 2016b). This local damping is proportional to the magnitude of the out-of-balance forces calculated on the background mesh (Wang et al., 2016b).

Even tough GPU programming enables performant (in terms of wall-clock time) numerical solvers, one of the major limitation is the required memory (Wyser et al., 2021), which can exceeds the hardware limit by far. As such, we consider only one quarter of the granular column assuming that horizontal momentum transfers are sufficiently small to be neglected. This allows to spare an important amount of memory utilization on the GPU during computation.

Table 7.1. | The granular column is discretized by 40 elements along x and y directions and $n_{pe} = 8$ are assigned per initially filled element.

λ_0	n_{el}	n_{no}	n_{mp}	Δx [m]
0.5	27104	30375	25120	0.05
1.0	169344	180625	50240	0.05
2.0	1183424	1225125	100480	0.05
4.0	8817984	8978125	200960	0.05

Since the initial aspect ratio λ_0 of the column strongly governs the run-out, we assume that the numerical geometry could differs from the experimental setting. Consequently, we consider a column of radius $r_0 = 1$ m, to artificially increase numerical time steps that are restricted by the CFL condition, *i.e.,* adaptative time steps are implemented in `ep2-3De v1.0`. The three-dimensional background mesh is made of regular tri-linear elements. Roller boundary conditions, *i.e.,* free-slip boundary conditions, are enforced on the side boundaries of the background mesh, whereas a no-slip boundary condition is enforced on the bottom. The granular column is discretized by 40 elements along x and y directions and $n_{pe} = 8$ are regularly assigned per initially filled element (see Table 7.1). The background mesh depends on the initial height of the column and, it is defined to be sufficiently larger to fully enclose the collapse without influences of side boundaries.

7.3. Results

7.3.1. Analytical solutions and experimental collapses

We first present (see Fig. 7.6(a)) the experimental results from the laboratory experiments fitted by i) our analytical solution under the quasi-static deformation hypothesis of the column and, ii) the solution given by Lajeunesse et al., 2004. The latter is in agreement with the experimental granular collapses, whereas the former (our quasi-static understanding of the deformation) rapidly diverges from the experimental data. In this case, we consider $\theta_c = \phi_y$, where the internal yield angle is $\phi_y = \tan^{-1}(\mu)$ with $\mu = 0.77$. The latter was

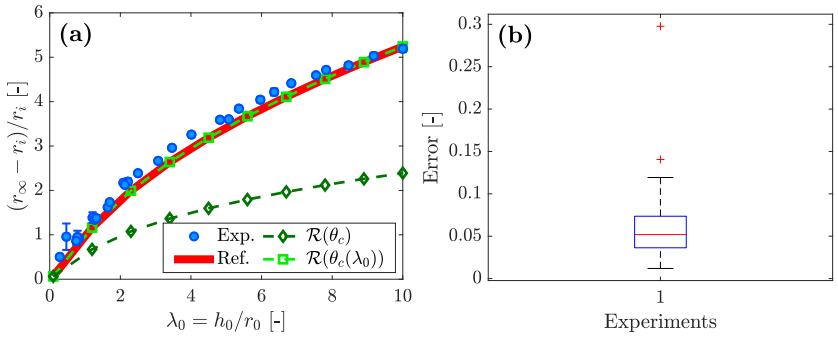


Figure 7.6. | (a) Normalized final run-out distances $(r_\infty - r_0)/r_0$ with respect to the initial aspect ratio $\lambda_0 = h_0/r_0$. Our analytical solution $g(\theta_c)$ (dark green line) predicts lower normalized run-out distances when considering a quasi-static deformation of the column. When relating λ_0 to θ_c, the solution (green line) is in agreement with both the experimental results (blue circles) and the solution proposed by Lajeunesse et al., 2004 (thick red line). (b) Box plot of errors for the experimental data. It demonstrate a rather skewed distribution of errors with few outliers (red crosses). This explains the overall small amplitude of error bars in (a).

inferred with the experimental protocol previously presented. When considering $\theta_c = f(\lambda_0)$, the analytical solution (green line in Fig. 7.6(a)) is then in agreement with both the experimental results and Eq. 7.1.2 (Lajeunesse et al., 2004).

Regarding the experimental data, we observe a small amplitude of error measurements (see Fig. 7.6(b)). This indicates our observations to be consistent and reliable.

To quantify the goodness of fit of Eq. 7.2.2, we apply a power-law fit to the experimental data. It is given by

$$\frac{(r_\infty - r_0)}{r_0} = 0.63\lambda_0^{1.30}, \tag{7.3.1}$$

for which RMSE $= 0.13$ ($R^2 = 0.99$), which is lower than our analytical solution. We calculate an RMSE $= 0.23$ for $\phi_y = \tan^{-1}(\mu)$. We further investigate the optimal ϕ_y (see Fig. 7.7) for which the RMSE for Eq. 7.2.2 is the smallest when fitted to the experimental data.

The difference observed for the parameter μ (*i.e.,* 0.71 and 0.77 accordingly to Fig. 7.7) appears reasonable. In one hand, the value inferred from a robust fitting shows the smallest RMSE but, on the other hand, the value inferred from laboratory measurements has a physical meaning (*i.e.,* the friction) despite its greater error. Since the error difference is acceptable, we consider $\mu = 0.77$.

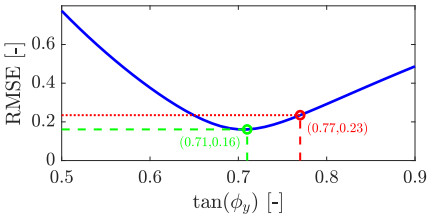


Figure 7.7. | Optimization of the parameter $\mu = \tan(\phi_y)$. The smallest RMSE is resolved for $\mu = 0.71$ whereas the physical value $\mu = 0.77$, derived from laboratory measurements, is similar in magnitude. However, the smallest RMSE achieved is still greater than the one obtained by a robust power-law fitting.

7.3.2. Experimental and numerical granular collapses

A typical numerical solution obtained with uGIMP variant in `ep2-3De v1.0` is showed in Fig. 7.8. The initial aspect ratio is $\lambda_0 = 2$ and the local damping is $D = 0.05$. The granular mass spreads in a realistic fashion on the bottom surface. The equivalent plastic strain ϵ^p_{eqv} highlights intense zones of shearing. However, successive shear-bands are roughly resolve because of the numerical

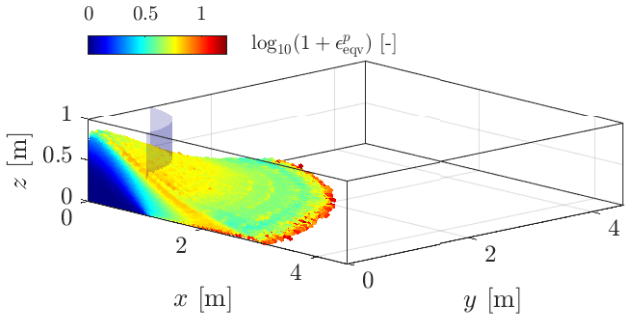


Figure 7.8. | Final morphology of the granular deposit for $\lambda_0 = 2$ and $D = 0.05$. Colour denotes the equivalent plastic strain ϵ^p_{eqv}. The transparent blue shell indicates the initial maximum extent of the granular column.

resolution, *i.e.*, $\Delta x = 0.05$ m. One can observe that most of plastic deformation is superficial. In addition to overall elasto-plastic deformations of the granular column, shallower granular avalanches occurred. We also report few material points which can be considered as fully disconnected from the main body, *i.e.*, the gaseous state observed during the experiments. Similarly, the proportion of material points in a disconnected state increases as the local damping is reduced.

We select different values for the local damping, *i.e.*, $D = \{0.0, 0.05, 0.1\}$. For simplicity, we report the maximal radial distance of the farthest material point to determine r_∞. We observe that for $D = 0.05$, the numerical solution agrees well with the experimental granular collapses. Shall the damping

be smaller or greater than 0.05, the numerical model either overestimates or underestimates the normalized run-out distance, respectively. As such, the local damping is an important parameter and requires an iterative calibration process.

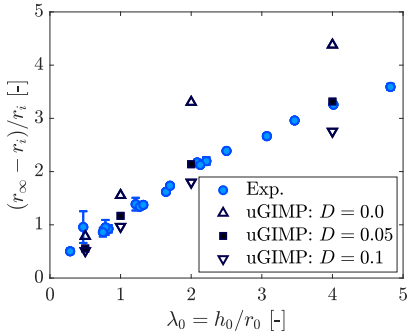


Figure 7.9. | Direct comparison between the experimental results and the numerical solutions using ep2-3De `v1.0`. The local damping D strongly influences the final normalized run-out distance. This is expected since it damps the out-of-balance forces calculated on the background mesh. For $D = 0.05$, the numerical solution given by ep2-3De `v1.0` is in agreement with the experiments, even tough some discrepancies exist.

Increasing the local damping even more (*i.e.*, $D = 0.4$) leads to a numerical solution closer to our analytical solution assuming quasi-static deformation. This makes sense since most of out-of-balance forces are damped out during calculation, yielding to final run-out distances close to the quasi-static state we assumed previously.

7.4. Discussion

The difference between our analytical solutions (quasi-static and dynamic hypotheses, see Fig. 7.10) expresses an important principle under the following hypothesis: the kinetic energy loss nearly asymptotically increases during the collapse because of an increase of elasto-plastic collision rate. This suggests that the final angle of repose θ_c of a granular collapse expresses the amount of energy lost during the process, comparatively to a purely quasi-static deformation of the column: The greater the difference between θ_c and ϕ_y, the greater the kinetic energy loss.

We further interpret and propose the following: the near-asymptotic behaviour of $\tan(\theta_c)$ (see Fig. 7.10) could be understood as a consequence of a microscopic steady state collision rate during the granular collapse. The collision rate rapidly increases, as the initial aspect ratio increases. When the grains reach their free-fall velocity, their collision rate becomes steady. When such equilibrium is resolved, the energy dissipation rate within the system can no longer changer.

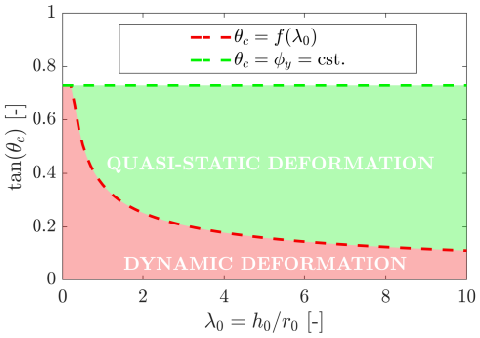


Figure 7.10. | Difference between quasi-static (sand pile, green area) and dynamic deformations (collapse, red area) of the granular column: $\tan(\theta_c)$ expresses the ratio h_∞/r_∞ and shows a significant decrease when $\theta_c = f(\lambda_0)$. It also indirectly expresses the increase of energy loss due to elasto-plastic collisions during the inertial deformation of the continuum.

Within the material point framework under an explicit formulation, the numerical solutions agree with the experimental results of analogue granular collapses. This confirms the solver `ep2-3De v1.0` to be an appropriate tool for the numerical modelling of dynamic granular collapses. However, a proper calibration procedure for the local damping must be carried. An increase of this local damping yields the numerical solution closer to the quasi-static analytical solution. However, the explicit formulation of the solver is not well suited to further investigate such numerical transition. An implicit formulation (Charlton et al., 2017; Coombs et al., 2020a) should be prefer to fully resolve quasi-static granular collapses.

Because of hardware limitations, a full three-dimensional model of the granular collapse is not yet possible. This should be the focus of future studies, by proposing a multi-GPU implementation of the solver `ep2-3De v1.0` using a message passing interface standard (MPI), such as Open MPI.

As suggested by Nguyen et al., 2020, some material properties (*i.e.,* stiffness and density) have little influence over the behaviour of the collapse. The only common material parameter between experiments and the numerical model is the friction angle. Still, the numerical solutions agree well with the experimental data. This also demonstrates that the only geometrical parameter that really matters is the initial aspect ratio λ_0 of the column.

7.5. Conclusion

We present an analytical solution for the normalized run-out distance of three-dimensional quasi-static granular collapses. We further propose a correction to consider dynamic collapse. This yields an analytical solution which agreed with experimental results. In addition, we benchmark our in-house MPM solver `ep2-3De v1.0` with experimental granular collapses. Good agreements

are found. Future works should be directed toward 1) a multi-GPU implementation of the solver to overcome the memory limit and, 2) an implicit formulation of the solver to fully resolve quasi-static granular collapses.

Author contributions EW performed the laboratory experiment. EW and YA wrote the first draft of the manuscript and developed the first version the ep2-3De v1.0 solver. MJ and YP supervised the early stages of the study and provided guidance. All authors have reviewed and approved the final version of the paper.

Competing interests The authors declare that they have no conflicts of interest.

Acknowledgements Authors greatfully acknowledge Pierre-Etienne Cherix (Mécanix Sàrl, 1880 Bex, Switzerland), who created the whole experimental device and thus made the experimental part of this research possible. Yury Alkhimenkov gratefully acknowledges support from the Swiss National Science Foundation (grant no. 172691). Yury Alkhimenkov and Yury Y. Podladchikov gratefully acknowledge support from the Russian Ministry of Science and Higher Education (project No. 075-15-2019-1890).

Supplementary materials

Local damping and quasi-static convergence

We present the supplementary materials regarding the transition from dynamic to quasi-static granular collapses. As mentioned in §7.3.2, we report the influence of the local damping (see Fig. 7.11) over the behaviour of the granular collapse, *i.e.,* an increase of damping yields a more pronounced quasi-static deformation of the granular column. We select different values for the local damping, *i.e.,* $D = \{0.0, 0.05, 0.1, 0.2, 0.4\}$.

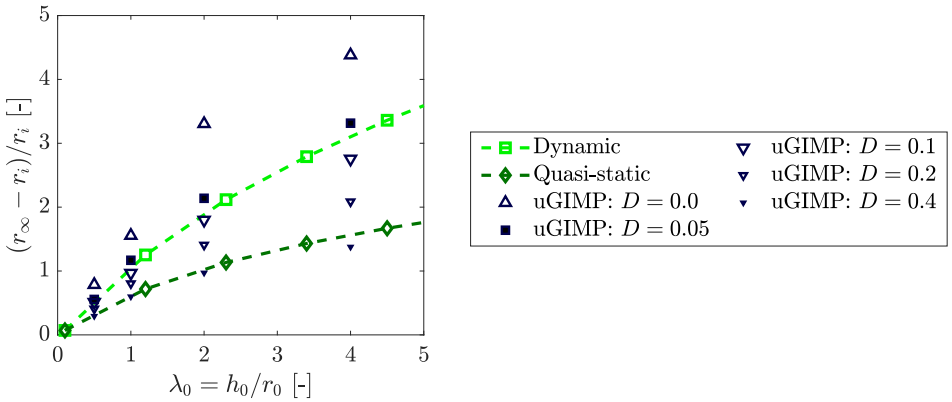


Figure 7.11. | Normalized run-out distances with respect to an initial aspect ratio of the column λ_0 for a variety of local damping coefficients D. Analytical solutions under quasi-static and dynamic hypotheses are also reported.

We can observe that as the local damping coefficient increases, the numerical solution gets closer to the quasi-static analytical solution we propose. For $D = 0.4$, the numerical solution is the closest to a quasi-static relaxation of the granular column.

8. Perspective

8.1. Discussion

8.1.1. Cratering in a critical state place

This research work was carried at the early beginning of the thesis. At that time, research objectives were "*blurry*" in my mind, but, my interest was the study of the behaviour of complex granular matter. In addition, the discrete element method (DEM) was seriously considered as a numerical tool to gain insights about the internal micromechanical processes of cratering. In this subsection, I explain the major outcomes and the reasons why cratering yielded my research interest toward landslides.

The cratering response is an elasto-plastic problem. The crater itself is a perfect example of an irreversible process. Moreover, the micromechanical processes involved in the differential uplifts of granular materials is closely related to shear banding. While the crater is forming, the granular material is compacted, *i.e.,* probably along compaction-bands, until a critical packing state is reached. Beyond this limit, the material can only dilate along plastic strain localization such as shear bands. As such, the observed morphological features, *i.e.,* rims and uplifts, are resulting from 1) this transition from compaction to dilation, 2) promoted by displacements along shear bands. This is the logical course of action if the initial packing state is below the critical packing state. This also implies that an important part of kinetic energy is dissipated in compressive-type deformations, *i.e.,* a larger crater is obtained. When the initial granular packing is equal or higher than the critical packing state, the material can only dilates along shear localizations. As a result, most of the impact energy is transferred to dilation, which is demonstrated by more significant uplifts of the granular bed. Hence, the crater is smaller.

Even tough the compaction-dilation transition is a well-known and well-documented phenomenon (see *e.g.,* Nedderman 1992; Umbanhowar et al. 2010; Zhao et al. 2015b), this is what triggered my interest for elasto-plasticity and shear bands. One of the take home message of this first research paper is that the granular packing (or porosity), and probably its spatial variability for poly-disperse granular systems, contributes to the crater formation, alongside with the kinetic energy released at impact.

After this first research work, a large amount of time was invested in the code development of a two-dimensional solver based on the discrete element method (DEM). It was written in the C programming language. This new programming experience showed that: 1) the C language is tedious for pre- and post-processing activities (*e.g.,* data initialization) and, 2) DEM is com-

putationally very expansive.

Beside and as mentioned in §3.1, there are important limitations, such as the particle shape and the necessary calibration parameters. Moreover, it can suffer from is advantage, *i.e.,* the micro-scale description of physics is not well suited to our human meso-scale understanding and observation of phenomena.

Finally, the necessary coupling between granular mechanics and fluid dynamics also requires the coupling of two distinct numerical methods. Here comes the interest for the material point method, which could be seen, to some extent, as a multi-phase numerical method. In addition, recent research topics propose a multi-scale coupling between DEM and MPM in order to consider micromechanics in a meso-scale constitutive framework (Yang et al., 2017; Liu et al., 2018b; Zhao et al., 2018).

8.1.2. Code prototyping in MATLAB

As demonstrated in Chapter 5, MATLAB is an effective high-level programming language to prototype numerical codes. In particular, this work demonstrated that vectorization activities can significantly improved the MATLAB's computational performances. This was already demonstrated by Dabrowski et al., 2008; Bird et al., 2017; Räss et al., 2017; O'Sullivan et al., 2019 for MATLAB-based Finite Element implementations. This work demonstrated their findings (*i.e.,* vectorization or optimal RAM-to-cache communication) can highly benefit to the numerical implementation of MPM codes. At most, a 28× performance gain is reported between an iterative solver and a vectorized solver. This is consistent with other vectorized implementation O'Sullivan et al., 2019.

This MATLAB-based solver is important, because it delivers a fast prototyping environment to test and validate numerical algorithms. Similarly, the extensive usage of the built-in function `accumarray()` is the precursor to the CUDA-C built-in function `atomicAdd()`. However, one can question the application for production. Its is widely accepted that MATLAB is less efficient than lower-level languages, *i.e.,* C/C++ and Fortran programming languages (Coombs et al., 2020a). This concern was also raised by one of the reviewer during the first submission stage of the paper. Despite the good performances achieved, the focus of this work was restricted to plane strain conditions. Resolving three-dimensional problems requires much more computational power (Gerya, 2010). Nevertheless and relatively from MATLAB, this still provides a fast prototyping environment.

These comments gives a nuanced answer to the following question:

"*Can a high-level programming language (e.g., MATLAB, R or Python) be performant and efficient ? Yes, relatively to MATLAB's serial performances.*"

The MATLAB-based solver also demonstrated that the material point framework is a well-suited method to investigate elasto-dynamic or elasto-plastic problems. In particular, large elasto-plastic deformations of granular collapses

were correctly resolved Bui et al., 2008. Similarly, selected elastodynamic problems were also correctly reproduced by the solver. This demonstrates that the proposed MPM implementation is reliable. It could be translated toward a more efficient programming language for production activities. Moreover and as suggested by preliminary investigations in §3.2.7, the rate-dependent formulation is suitable for large elasto-plastic deformations. This is important since such formulation is straightforward to implement. Within a MATLAB environment, the expansive computation of eigenvalues and eigenvectors of the left Cauchy-Green strain tensor is avoided. These additional computations undermine the computational efficiency of a finite deformation formulation within the material point framework.

From an a posteriori point of view, these prototyping activities designed the hybrid numerical framework chosen in the GPU-based solver `ep2-3De v1.0`, *i.e.,* the coupling between MATLAB and CUDA C. As an example, the overall workflow is really similar between `fMPMM-solver` and `ep2-3De v1.0`. This also demonstrates that MATLAB is an effective language when it comes to pre- or post-process geometry and numerical results.

8.1.3. GPU support for super[b]computing

As demonstrated in Chapter 6, the GPU-based solver `ep2-3De v1.0` allows to consider three-dimensional elasto-plastic problems while benefiting from extremely decent wall-clock times. The main results showed it was possible to accurately simulate three-dimensional granular collapses, provided that the geometry setting is simple. This restriction is due to the on-chip memory limit, which can not be overcome with a single GPU implementation. I will address this concern later.

The GPU-based solver also demonstrated significant performance gains with respect to a CPU-based solver[1]. When using the most recent GPU architecture (*i.e.,* Ampere), the performance gain is roughly 200×. However, such gain is only achieved once the GPU reaches its maximum effective memory throughput. Again, this confirms that the memory (*i.e.,* throughput and on-chip memory limit of the GPU) is the main bottleneck of modern GPU-based codes. Using Nvidia V100's or A100's is an expensive computational activity. Considering fairer performances and much cheaper devices, a low-entry consumer electronics GPU such as the GTX 1650 still delivers a 10× performance gain. Here again, the total amount of memory restricts the simulation to small and moderate dimensions.

As a closing remark regarding performances, the next step is to consider multi-GPU implementations, using the message passing interface standard (*i.e.,* Open MPI). All of this considerations give an answer to one of the main research question of this research work, which is:

"*Are thee-dimensional elasto-plastic simulations (i.e., landslides) affordable in a decent amount of time ? Yes, they definitely are.*"

[1]The CPU-based version of `ep2-3De v1.0` does not support multi-threaded executions

Moreover, `ep2-3De v1.0` demonstrated a great potential to resolve complex three-dimensional landslides. A future perspective is an application toward well-known landslide case studies to extensively validate the solver. However, a strain-softening behaviour of the material was considered. Another perspective would be to overcome the usage of strain-softening laws to trigger strain localization using much higher resolutions. This resolution influence was suggested by De Borst et al., 1992; De Borst et al., 1993, at least for the finite element method. Regarding GPU performances, the numerical resolution or the computational domain's dimensions can no longer be considered as limitations. This should be the focus of future studies. However, this also necessitate to resolve the on-chip memory concern.

Chapters 5 and 6 both demonstrated an important limitation of current computer's hardware: the memory capacity and the memory throughput. For a MATLAB-based implementation, the optimal RAM-to-cache communication and the L2 cache capacity of the CPU limit the computational performances of the material point method. Regarding GPU-based implementations, the main limit is the on-chip memory available. As such, these considerations give an answer to the following question:

"*What is the actual limitation of modern computer hardware ? The on-chip memory currently limits the computational performances.*"

Finally, the influence of heterogeneous material properties (*i.e.,* the peak cohesion) was demonstrated. The spatial variability of the cohesion field contributes significantly on strain localization. Using a heterogeneous cohesion field resolves all the first-order features of landslides, *i.e.,* a complex crown with multiple major and minor scarps or the toe bulging. A future perspective would be a deeper investigation of the influences of these fluctuation scales over the mechanical behaviour of landslides. This could shed lights on the genesis of the different landslide types (Varnes, 1958; Varnes, 1978). GPU's computational power gives the opportunity to resolve three-dimensional geometries in a decent amount of time. Such power should serve deeper physical investigations of these fluctuation scales. The novelty of this research is that a very limited number of studies (Feng et al., 2021) use GPU computing to investigate three-dimensional configurations. This is promising for future three-dimensional geomechanical investigations.

To conclude, the material point framework coupled with GPU-accelerated numerical implementations offer a performant, efficient and reliable framework to investigate elasto-plastic deformations of various magnitudes. This answers the first research question: "*How elasto-plastic deformations can be investigated ?*"

8.1.4. Three-dimensional granular collapses

At long last, an application of `ep2-3De v1.0` was proposed in Chapter 7. Because of the on-chip memory limitation, this works was restricted to a reduced geometry of the granular column, *i.e.,* only a quarter of it is modelled. The

numerical solutions agree well with a wider range of experimental data obtained with a novel apparatus. The latter allowed a variety of initial aspect ratio λ_0 of the column, as opposed to the work of Bui et al., 2008 which was restricted to a unique aspect ratio, *i.e.,* $\lambda_0 = 0.5$. To some extent, this last work generalized the validity of `ep2-3De v1.0` for the numerical modelling of granular collapses.

A significant influence of the local damping coefficient was demonstrated, *i.e.,* a higher damping promotes a greater quasi-static response of the granular collapse. This also confirmed the quasi-static hypothesis, which agrees with quasi-static numerical solutions.

There is a two-fold perspective here. A multi-GPU implementation should be investigated, in order to resolve a thee-dimensional configuration without the introduced restriction. Moreover, additional experimental works should be directed toward quasi-static granular collapses, in order to confront the quasi-static hypothesis to experimental evidences.

8.2. Outlook

The figure 8.1 perfectly illustrates the major outcome of this research work: three-dimensional large elasto-plastic deformations in less than 30 s on a modern laptop. This gives the opportunity for researchers to test new ideas and/or

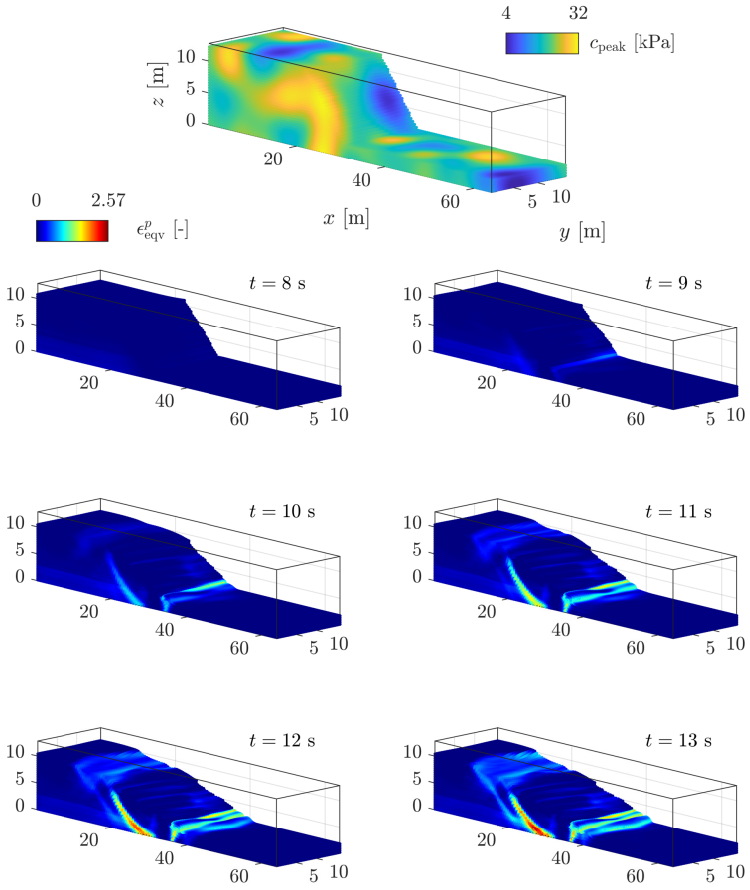


Figure 8.1. | Three-dimensional complex slump, with $n_{mp} \approx 10^5$ material points, obtained in a few seconds when running on a Nvidia A100 GPU.

more complex rheological laws under a three-dimensional configuration, without restricting the initial problems to plane strain conditions. Furthermore, Fig. 8.1 demonstrates that, when considering both the spatial variability of material properties and a three-dimensional configuration, a complex elasto-plastic behaviour is resolved. The whole argues for a wider acceptation of three-dimensional modelling and, a progressive shift towards three-dimensional numerical models, even for well-known computationally expansive numerical

methods. GPUs are now increasingly supporting scientific computing, ranging from laptops to supercomputers. This research demonstrated the enormous power of GPUs.

By the end, the material point framework has proven itself well suited for large elasto-plastic deformations, natural free-surface problems and performant numerical implementations on modern GPUs. Even tough much of this research work was dedicated to numerical implementation strategies, the upcoming perspective should now be directed toward the analysis of three-dimensional numerical models. This works resulted in the hybrid, versatile and performant numerical solver `ep2-3De v1.0`, which now should be extensively used for scientific purpose.

In addition, a related concern is the evolution of `ep2-3De v1.0` toward a hydro-mechanical formulation. It is well-known that landslides are very sensitive to pore pressure, which could trigger strain localization. This pore pressure concern was neither raised nor mentioned during the writing of the dissertation. But, a poro-elasto-plastic solver under plane strain condition was addressed in a prototyping attempt in MATLAB (see Fig. 8.2 for a typical result for a CPDI2q implementation). Further implementation effort should

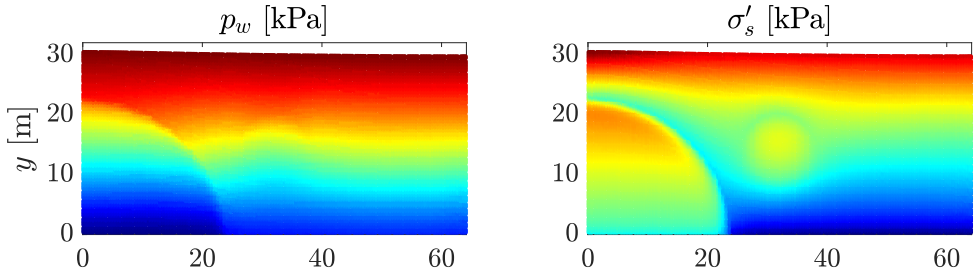


Figure 8.2. | Pore pressure p_w and effective pressure σ'_s for an heterogeneous soil subjected to a progressive self-weight elastic loading for 8 seconds.

now be devoted to a three-dimensional hydro-mechanical solver, benefiting from the performant environment allowed by GPU computing.

Still, one challenge remains regarding the multi-GPU implementation: a parallel implementation that hides communication latencies and allows to efficiently transfer material points between GPUs. This is the main concern in the supplementary materials presented in Appendix A. These two points should be address to fully benefit from the massive computational power of GPUs. Only then a material point framework will be available for massive numerical simulations of complex elasto-plastic problems.

A. Message passing interface and multi-GPU computing

Multi-GPU implementation of `ep2-3De v1.0`

Emmanuel Wyser, Yury Alkhimenkov, Michel Jaboyedoff & Yury Y. Podladchikov

In consideration as additional materials for the second submission stage in *Geoscientific Model Development Discussions*

Foreword This additional work provides a multi-GPU implementation of `ep2-3De v1.0` using a message passing interface standard (*e.g.,* Open MPI). This is to overcome the on-chip memory limitation highlighted in Chapter 6. These supplementary materials will be added during the revision process of the manuscript submitted to *Geoscientific Model Development Discussions*, as it partially addresses one of the major concern of the research work.

A.1. Introduction

One of the major limitation of `ep2-3De v1.0` is the on-chip memory (see Chapter 6). We demonstrated that an implementation of the material point framework quickly reaches the hardware limit of GPUs, even on modern architectures. It is then essential to overcome this limit in order to resolve larger computational domain with a greater amount of material points.

Here, we address this concern by implementing a distributed memory parallelisation using the message passing interface (MPI) standard. However, we limit our implementation efforts by considering 1) a one-dimensional GPU topology, 2) no computation/communication overlaps, and 3) only mesh-related quantities are shared amongst GPUs, *i.e.,* the material points are not transferred between GPUs during a simulation. We also selected a non-adaptative time step to avoid the collection of the material point's velocities located in different GPUs at the beginning of each calculation cycle.

A.2. Available computational resources

The CPU- and GPU-based simulations are performed on a modern workstation running on a Windows 10 operating system with the latest CUDA version v11.2. The CPU is an Intel Core i9-10900K with 10 physical cores of base clock speed (or frequency) of 3.70 GHz, which can rise up to a maximum clock speed of 5.30 GHz, supported with 64 GB DDR4 RAM. It hosts a consumer

Table A.1. | List of the graphical processing units (GPUs) used throughout this study. We also report the peak memory throughput, *i.e.,* $\mathsf{MTP}_{\mathsf{peak}}$, measured thanks to the routine `bandwidthTest.cu` provided by Nvidia alongside with the CUDA toolkit. When compared with the effective memory throughput $\mathsf{MTP}_{\mathsf{eff}}$, one can estimate the possible gain of an additional optimization of the algorithm. This is particularly useful when estimating the level of optimization of a GPU-based implementation.

GPU	Architecture	SM count	On-chip memory [GB]	$\mathrm{MTP}_{\mathrm{peak}}$ [GB·s^{-1}]
A100	Ampere	108	40	1127.1
RTX 3090	Ampere	82	24	774.1
RTX 2080 ti	Turing	68	11	513.1
GTX 1650	Turing	14	4	168.7
V100	Volta	80	16/32	732.6
GTX Titan X	Maxwell	28	12	260.1

electronics Nvidia RTX 3090 GPU (the latest Ampere architecture) with 82 streaming multiprocessors (SM units) with a base frequency of 1.40 GHz. This results in 10490 CUDA cores that are supported with an on-chip memory of 24 GB GDDR6 (*i.e.,* the GPU global memory). Other GPUs installed on older desktops are also used to compare their respective GPU performances, *i.e.,* an RTX 2080 ti (workstation) and a GTX 1650 (laptop), both running

on a Windows 10 operating system. Additional simulations were also ran on a workstation equipped with the latest Nvidia A100 GPU at the Lomonosov Moscow State University.

Single- and multi-GPU simulations are also performed on the Octopus GPU supercomputer at the Swiss Geocomputing Centre, University of Lausanne, Switzerland. In particular, the single-GPU simulations are run on a first computing system, hosting an Nvidia Tesla V100 (16 GB), supported by an Intel(R) Xeon E5-2620 v2 (2.1 GHz) CPU. Most of the multi-GPU simulations were performed on a computing system hosting 8 Nvidia Tesla V100 Nvlink (32 GB), supported by 2 Intel(R) Xeon Silver 4112 350 (2.6GHz) CPUs. The last computing system is composed of 32 nodes, each hosting 4 GTX Titan X (12 GB), 2 Intel(R) Xeon E5-2620 v3 (2.4GHz) CPUs.

The latest CUDA version installed is v11.0, and the supercomputer Octopus is operated under a CentOS 6.9 environment. To summarize the computational resources in use, Table A.1 presents the main characteristics of the GPUs used in this study.

A.3. Model 2a: initial setting

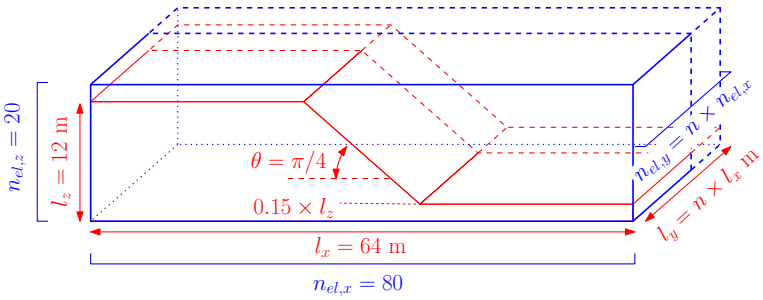


Figure A.1. | Geometry for the earth slump. For the multi-GPU implementation, the number of element along the $y-$direction can be largely increased, *i.e.,* $n = 2048$.

To avoid frequent material point's transfers amongst the GPUs, we consider an overlap of 8 elements between neighbouring meshes, *i.e.,* 9 nodes. This results in a one-dimensional GPU topology, for which both material points and meshes are distributed along the $y-$direction of the global computational domain (see Figs. A.1 & A.2). Arranging GPUs along this direction allows to overcome the need to transfer material points amongst GPUs, provided that the material point's displacement is not greater than the buffer zone, *i.e.,* the element overlap.

The evaluation of the multi-GPU implementation is based on the Model 2a in Chapter 6, with slight modifications, *i.e.,* the number of element along the y-direction is largely increased. The size of the physical domain $l_z \times l_x \times l_y$ is, at most, 12 m $\times$ 64 m $\times$ (64$\times$2048) m for Model 2a for the multi-GPU implementation.

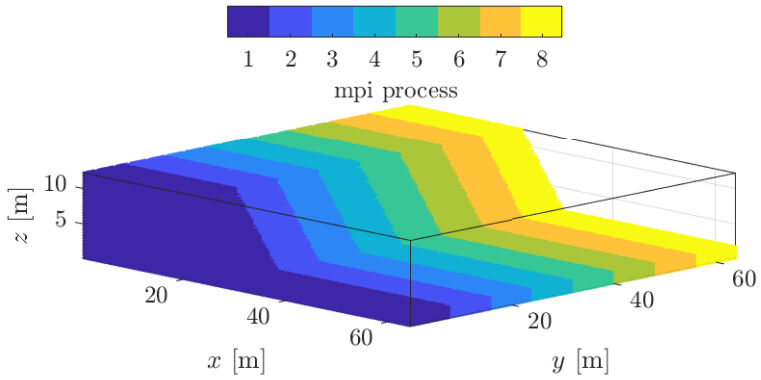


Figure A.2. | Domain partition of the material points amongst 8 GPUs. Combined with an overlap of 8 elements along the y-direction, material points can moderately move while still residing within the same GPU during the whole simulation.

A.4. Model 2a: multi-GPU performances

Here, we consider two distributed computing systems for parallel GPU computation, using up to 8 Tesla V100 (Volta architecture) or 128 Geforce GTX Titan X (Maxwell architecture). All numerical simulations are performed using a single-arithmetic precision (*i.e.,* $n_\mathrm{p} = 4$ bytes). This allows to increase the maximum number of material points and mesh dimensions. In addition, our GPU implementation relies on the usage of the built-in function `atomicAdd()`. It does not support the double-precision floating-point format FP64 for GPUs with compute capabilities lower than 6.0, *i.e.,* the Maxwell architecture amongst others.

We have to mention that, unlike the Tesla V100, the Geforce GTX Titan X only delivers an effective memory throughput of $\mathrm{MTP}_\mathrm{eff} \approx 100$ GBs^{-1}. This corresponds to 38 % of its hardware limit. This was already reported by Alkhimenkov et al., 2021 and, it could be attributed to its older architecture Gao et al., 2018; Wang et al., 2020c, *i.e.,* Maxwell. This performance drop is even more severe, probably due to the use of built-in functions, *i.e.,* `atomicAdd()`.

A.4.1. Computing system: up to 8 Tesla V100

We first performed parallel simulations with a moderate number of GPUs, *i.e.,* up to 8 Tesla V100 NVlink (32 GB). The respective wall-clock times are

reported in Fig. A.3. We report a wall-clock time of ≈ 110 s for $n_{mp} \approx 10^8$.

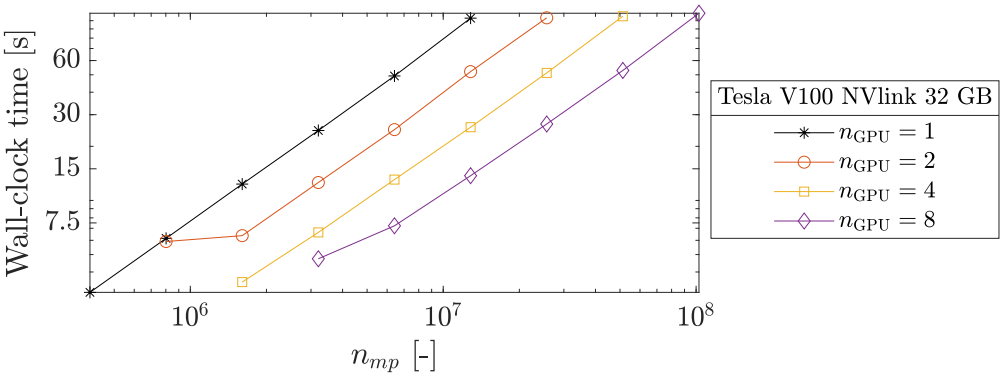


Figure A.3. | Wall-clock time for 1, 2, 4 and 8 Tesla V100 GPUs.

For the same amount of material points, we report a roughly weak scaling between the number of GPUs and the wall-clock time. If n_{mp} is increased by a factor 2, 4 or 8, the wall-clock time is roughly similar to the baseline, *i.e.*, $n_{\text{GPU}} = 1$.

Such weak scaling is more obvious when inspecting the MTP_{eff} measured (see Fig. A.4), *i.e.*, the total sum of MTP_{eff} across all the GPUs. Based on the memory throughput of 1 GPU, an estimation of a perfect weak scaling is possible. For 8 GPUs, it should correspond to $\text{MTP}_{\text{eff}} = 4824$ GBs^{-1}, whereas we report $\text{MTP}_{\text{eff}} = 4538$ GBs^{-1}. This gives a parallel efficiency of $\approx 94\%$ and, an effective speed-up of $7.5\times$. Similar observations are made for $n_{\text{GPU}} = 2$ and $n_{\text{GPU}} = 4$.

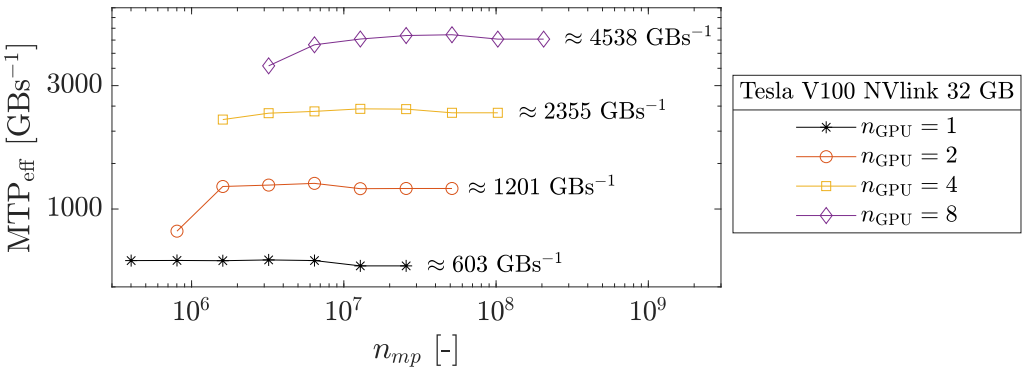


Figure A.4. | Sum across the GPUs involved of the MTP_{eff}. We roughly report a weak scaling between the number of GPUs and the overall effective memory throughput.

A.4.2. Computing system: up to 128 Geforce GTX Titan X

Here, we investigate parallel GPU computing using up to 128 Geforce GTX Titan X. This allows to address even larger geometries, as showed in Fig A.5

where a geometry of nearly $n_{mp} \approx 8{\cdot}10^8$ is resolved in less than 8 minutes. The first observation is that, for parallel computing up to 64 GPUs, the wall-clock time evolution is smooth. This is in contrast when 128 GPUs are used. For

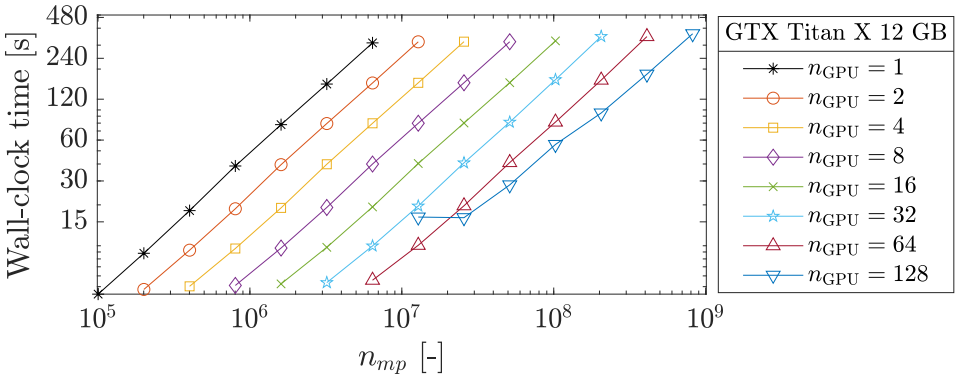


Figure A.5. | Wall-clock time reported for up to 128 Geforce GTX Titan X GPUs and up to $n_{mp} \approx 8 \cdot 10^8$.

fewer material points, the wall-clock time is chaotic whereas it stabilizes as the number of material points increases. We suspect the absence of computation/-communication overlaps to be the main reason of this erratic behaviour. The time spent of communication between GPUs is proportional to the number of GPUs. Hence, we should observe increasing performance losses. We explain this contradiction as followed. As the number of material points increases, the time spent on communication becomes smaller with respect to the time spent on communications and exchanges between GPUs. The total size of the overlap is constant, regardless of the $y-$dimension. This is not the case for the total number of material points per GPU, which proportionally increases with the $y-$dimension. The latency between computation and communication reduces as the $y-$dimension increases, because the time spent on computations growths faster than the time spent on communications.

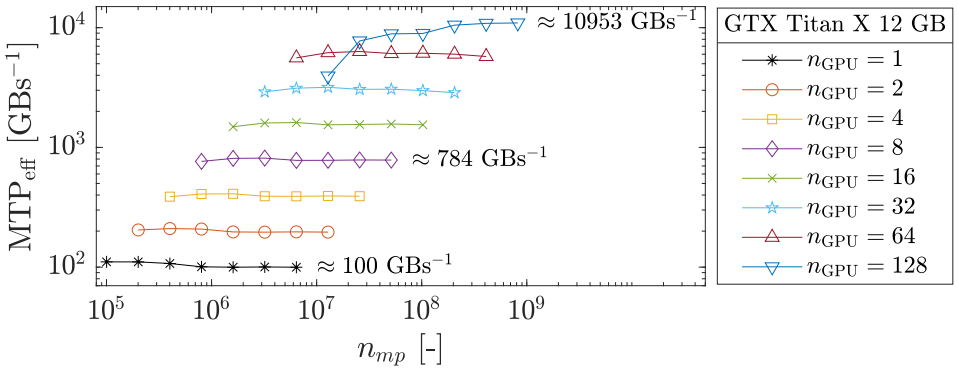


Figure A.6. | MTP_{eff} sum across the GPUs involved.

Another observation is the effective memory throughput (see Fig. A.6).

When considering a perfect weak scaling, one should measure an effective memory throughput $\text{MTP}_{\text{eff}} = 12800$ GBs^{-1} for 128 GPUs whereas we report only $\text{MTP}_{\text{eff}} = 10953$ GBs^{-1}. This gives a parallel efficiency of $\approx 85\%$ and, an effective speed-up of $\approx 110\times$. When using less GPUs, the parallel efficiency is higher, *i.e.,* 98 % for 8 GPUs.

A.5. Discussion

Even tough the simplifications made alleviate the on-chip memory limitation, the type of problem, which can be addressed, is reduced. As an example, investigating high-resolution three-dimensional granular collapses is not possible under the assumptions made, because of small displacement required along the $y-$direction. This is incompatible with three-dimensional granular collapses. Hence, this motivates future deeper investigations toward a more versatile multi-GPU implementation. In addition, we report a slight drop of the parallel efficiency, as the number of GPUs increases. Future works should be directed toward a parallel strategy that hides communication latency, as proposed in Räss et al., 2020; Alkhimenkov et al., 2021.

However, such multi-GPU implementation is particularly well-suited to resolve highly-detailed three-dimensional shear-banding, as first investigated in §3.2.10. We also reported decent wall-clock times (less than 8 minutes) for simulations with nearly a billion material points. One could argue that limiting the material point method to small displacement is a non-sense. Essentially, finite element codes are better suited for small strain analysis. However, this gives interesting insights on a multi-GPU implementation of the material point framework on a GPU supercomputer.

References

Abe, K., K. Soga, and S. Bandara (2014). "Material Point Method for Coupled Hydromechanical Problems". In: *Journal of Geotechnical and Geoenvironmental Engineering* 140.3, p. 04013033. DOI: https://doi.org/10.1061/(ASCE)GT.1943-5606.0001011.

Abed Zadeh, A. et al. (2019). "Enlightening force chains: a review of photoelasticimetry in granular matter". In: *Granular Matter* 21.4, p. 83. DOI: 10.1007/s10035-019-0942-2.

Ahn, S., S. H. Doerr, P. Douglas, R. Bryant, C. A. E. Hamlett, G. McHale, M. I. Newton, and N. J. Shirtcliffe (2013). "Effects of hydrophobicity on splash erosion of model soil particles by a single water drop impact". In: *Earth Surface Processes and Landforms* 38.11, pp. 1225–1233. DOI: https://doi.org/10.1002/esp.3364.

Al-Hashemi, H. M. B. and O. S. B. Al-Amoudi (2018). "A review on the angle of repose of granular materials". In: *Powder Technology* 330, pp. 397–417.

Alejano, L. R. and A. Bobet (2012). "Drucker–Prager Criterion". In: *Rock Mechanics and Rock Engineering* 45.6, pp. 995–999. DOI: https://doi.org/10.1007/s00603-012-0278-2.

Alkhimenkov, Y., L. Räss, L. Khakimova, B. Quintal, and Y. Podladchikov (2021). "Resolving wave propagation in anisotropic poroelastic media using graphical processing units (GPUs)". In: *Journal of Geophysical Research: Solid Earth* n/a.n/a. e2020JB021175 2020JB021175, e2020JB021175. DOI: https://doi.org/10.1029/2020JB021175.

Amato, J. C. and R. E. Williams (1998). "Crater formation in the laboratory: An introductory experiment in error analysis". In: *American Journal of Physics* 66.2, pp. 141–143. DOI: 10.1119/1.18832.

Ambroso, M. A., R. D. Kamien, and D. J. Durian (2005). "Dynamics of shallow impact cratering". In: *Phys. Rev. E* 72 (4), p. 041305. DOI: 10.1103/PhysRevE.72.041305.

Anderson, C. E. (1987). "An overview of the theory of hydrocodes". In: *International Journal of Impact Engineering* 5.1. Hypervelocity Impact Proceedings of the 1986 Symposium, pp. 33–59. DOI: https://doi.org/10.1016/0734-743X(87)90029-7.

Anderson, R. S. and S. P. Anderson (2010). *Geomorphology: The Mechanics and Chemistry of Landscapes*. Cambridge University Press.

Andreotti, B., Y. Forterre, and O. Pouliquen (2013). *Granular Media : Between Fluid and Solid*. Cambridge University Press.

Bagi, K. (2006). "Analysis of microstructural strain tensors for granular assemblies". In: *International Journal of Solids and Structures* 43.10, pp. 3166–3184. DOI: https://doi.org/10.1016/j.ijsolstr.2005.07.016.

Balmforth, N. J. and R. R. Kerswell (2005). "Granular collapse in two dimensions". In: *Journal of Fluid Mechanics* 538, 399–428. DOI: 10.1017/S0022112005005537.

Bandara, S., A. Ferrari, and L. Laloui (2016). "Modelling landslides in unsaturated slopes subjected to rainfall infiltration using material point method". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 40.9, pp. 1358–1380. DOI: https://doi.org/10.1002/nag.2499.

Bandara, S. and K. Soga (2015). "Coupling of soil deformation and pore fluid flow using material point method". In: *Computers and Geotechnics* 63, pp. 199–214. DOI: https://doi.org/10.1016/j.compgeo.2014.09.009.

Barabási, A.-L., R. Albert, and P. Schiffer (1999). "The physics of sand castles: maximum angle of stability in wet and dry granular media". In: *Physica A: Statistical Mechanics and its Applications* 266.1-4, pp. 366–371.

Bardenhagen, S. G., J. U. Brackbill, and D. Sulsky (2000). "The material-point method for granular materials". In: *Computer Methods in Applied Mechanics and Engineering* 187.3, pp. 529–541. DOI: https://doi.org/10.1016/S0045-7825(99)00338-2.

Bardenhagen, S. G. and E. M. Kober (2004). "The Generalized Interpolation Material Point Method". In: *Computer Modeling in Engineering & Sciences* 5.6, pp. 477–496. DOI: https://doi.org/10.3970/cmes.2004.005.477.

Baumgarten, A. S. and K. Kamrin (2019a). "A general fluid–sediment mixture model and constitutive theory validated in many flow regimes". In: *Journal of Fluid Mechanics* 861, 721–764. DOI: 10.1017/jfm.2018.914.

– (2019b). "A general fluid–sediment mixture model and constitutive theory validated in many flow regimes". In: *Journal of Fluid Mechanics* 861, 721–764. DOI: 10.1017/jfm.2018.914.

Bažant, Z. P. and M. Jirásek (2002). "Nonlocal Integral Formulations of Plasticity and Damage: Survey of Progress". In: *Journal of Engineering Mechanics* 128.11, pp. 1119–1149. DOI: https://doi.org/10.1061/(ASCE)0733-9399(2002)128:11(1119).

Behringer, R. P. (2015). "Jamming in granular materials". In: *Comptes Rendus Physique* 16.1. Granular physics / Physique des milieux granulaires, pp. 10–25. DOI: https://doi.org/10.1016/j.crhy.2015.02.001.

Belytschko, T., W. K. Liu, B. Moran, and K. Elkhodary (2013). *Nonlinear finite elements for continua and structures*. John wiley & sons.

Beuth, L., T. Benz, P. A. Vermeer, and Z. Więckowski (2008). "Large deformation analysis using a quasi-static material point method". In: *Journal of Theoretical and Applied Mechanics* 38.2, pp. 45–60.

Birch, S. P. D., M. Manga, B. Delbridge, and M. Chamberlain (2014). "Penetration of spherical projectiles into wet granular media". In: *Phys. Rev. E* 90 (3), p. 032208. DOI: 10.1103/PhysRevE.90.032208.

Bird, R. E., W. M. Coombs, and S. Giani (2017). "Fast native-MATLAB stiffness assembly for SIPG linear elasticity". In: *Computers & Mathematics with Applications* 74.12, pp. 3209–3230. DOI: https://doi.org/10.1016/j.camwa.2017.08.022.

Bisht, V., R. Salgado, and M. Prezzi (2021). "Simulating penetration problems in incompressible materials using the material point method". In: *Computers and Geotechnics* 133, p. 103593. DOI: https://doi.org/10.1016/j.compgeo.2020.103593.

Boac, J. M., R. P. K. Ambrose, M. E. Casada, R. G. Maghirang, and D. E. Maier (2014). "Applications of Discrete Element Method in Modeling of Grain Postharvest Operations". In: *Food Engineering Reviews* 6.4, pp. 128–149. DOI: https://doi.org/10.1007/s12393-014-9090-y.

Borja, R. (2013). *Plasticity: Modeling & Computation*. SpringerLink : Bücher. Springer Berlin Heidelberg.

Bougouin, A. and L. Lacaze (2018). "Granular collapse in a fluid: Different flow regimes for an initially dense-packing". In: *Physical Review Fluids* 3.6, p. 064305.

Bower, A. F. (2009). *Applied mechanics of solids*. CRC press.

Brilliantov, N. V., F. Spahn, J.-M. Hertzsch, and T. Pöschel (1996). "Model for collisions in granular gases". In: *Phys. Rev. E* 53 (5), pp. 5382–5392. DOI: https://doi.org/10.1103/PhysRevE.53.5382.

Brodowski, R. (2013). "Soil detachment caused by divided rain power from raindrop parts splashed downward on a sloping surface". In: *CATENA* 105, pp. 52–61. DOI: https://doi.org/10.1016/j.catena.2013.01.006.

Buffington, J. M., W. E. Dietrich, and J. W. Kirchner (1992). "Friction angle measurements on a naturally formed gravel streambed: Implications for critical boundary shear stress". In: *Water Resources Research* 28.2, pp. 411–425.

Bui, H. H., R. Fukagawa, K. Sako, and S. Ohno (2008). "Lagrangian meshfree particles method (SPH) for large deformation and failure flows of geomaterial using elastic–plastic soil constitutive model". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 32.12, pp. 1537–1570. DOI: https://doi.org/10.1002/nag.688.

Burghardt, J., R. Brannon, and J. Guilkey (2012). "A nonlocal plasticity formulation for the material point method". In: *Computer Methods in Applied Mechanics and Engineering* 225-228, pp. 55–64. DOI: https://doi.org/10.1016/j.cma.2012.03.007.

Buzzi, O., D. M. Pedroso, and A. Giacomini (2008). "Caveats on the Implementation of the Generalized Material Point Method". In: *Computer Modeling in Engineering & Sciences* 31.2, pp. 85–106. DOI: 10.3970/cmes.2008.031.085.

Carrea, D., A. Abellan, J. Bechet, M. H. Derron, J. Duc, M. Jaboyedoff, and A. Pedrazzini (2011). "Sandbox simulation of slope failure mechanisms with 3D digitizer". In: *Slope Stability 2011, September 18-21, Vancouver*.

Carrea, D., A. Abellan, M.-H. Derron, N. Gauvin, and M. Jaboyedoff (2012). "Using 3D surface datasets to understand landslide evolution: From analogue models to real case study". In: *Landslides and engineered slopes: Protecting society through improved understanding*. Ed. by E. Eberhardt, C. Froese, K. Turner, and S. Leroueil. CRC Press, pp. 575 –579.

Carrigy, M. A. (1970). "Experiments on the angles of repose of granular materials 1". In: *Sedimentology* 14.3-4, pp. 147–158.

Chalk, C. M., M. Pastor, J. Peakall, D. J. Borman, P. A. Sleigh, W. Murphy, and R. Fuentes (2020). "Stress-Particle Smoothed Particle Hydrodynamics: An application to the failure and post-failure behaviour of slopes". In: *Computer Methods in Applied Mechanics and Engineering* 366, p. 113034. DOI: https://doi.org/10.1016/j.cma.2020.113034.

Charlton, T. J., W. M. Coombs, and C. E. Augarde (2017). "iGIMP: An implicit generalised interpolation material point method for large deformations". In: *Computers & Structures* 190, pp. 108–125. DOI: https://doi.org/10.1016/j.compstruc.2017.05.004.

Chen, J.-S., M. Hillman, and S.-W. Chi (2017). "Meshfree Methods: Progress Made after 20 Years". In: *Journal of Engineering Mechanics* 143.4, p. 04017001. DOI: 10.1061/(ASCE)EM.1943-7889.0001176.

Chen, X., D. Li, X. Tang, and Y. Liu (2021). "A three-dimensional large-deformation random finite-element study of landslide runout considering spatially varying soil". In: *Landslides*. DOI: 10.1007/s10346-021-01699-1.

Cho, S. E. (2010). "Probabilistic Assessment of Slope Stability That Considers the Spatial Variability of Soil Properties". In: *Journal of Geotechnical and Geoenvironmental Engineering* 136.7, pp. 975–984. DOI: 10.1061/(ASCE)GT.1943-5606.0000309.

Christakos, G. (1992). "2 - The Spatial Random Field Model". In: *Random Field Models in Earth Sciences*. Ed. by G. Christakos. Boston: Academic Press, pp. 21–106. DOI: https://doi.org/10.1016/B978-0-12-174230-0.50007-X.

Christoffersen, J., M. M. Mehrabadi, and S. Nemat-Nasser (June 1981). "A Micromechanical Description of Granular Material Behavior". In: *Journal of Applied Mechanics* 48.2, pp. 339–344. DOI: https://doi.org/10.1115/1.3157619.

Clark, A. and P. Behringer (2013). "Granular Impact Model as an Energy-Depth Relation". In: *Europhysics Letters* 101, p. 64001. DOI: 10.1209/0295-5075/101/64001.

Clark, A. H., L. Kondic, and R. P. Behringer (2012). "Particle Scale Dynamics in Granular Impact". In: *Phys. Rev. Lett.* 109 (23), p. 238302. DOI: 10.1103/PhysRevLett.109.238302.

Clark, A. H., A. J. Petersen, and R. P. Behringer (2014). "Collisional model for granular impact dynamics". In: *Phys. Rev. E* 89 (1), p. 012201. DOI: 10.1103/PhysRevE.89.012201.

Coombs, W. M. and C. E. Augarde (2020a). "AMPLE: A Material Point Learning Environment". In: *Advances in Engineering Software* 139, p. 102748. DOI: https://doi.org/10.1016/j.advengsoft.2019.102748.

Coombs, W. M., C. E. Augarde, A. J. Brennan, M. J. Brown, T. J. Charlton, J. A. Knappett, Y. Ghaffari Motlagh, and L. Wang (2020b). "On Lagrangian mechanics and the implicit material point method for large deformation elasto-plasticity". In: *Computer Methods in Applied Mechanics and Engineering* 358, p. 112622. DOI: https://doi.org/10.1016/j.cma.2019.112622.

Coombs, W. M., T. J. Charlton, M. Cortis, and C. E. Augarde (2018). "Overcoming volumetric locking in material point methods". In: *Computer Methods in Applied Mechanics and Engineering* 333, pp. 1–21. DOI: https://doi.org/10.1016/j.cma.2018.01.010.

Cooper, J. R., J. Wainwright, A. J. Parsons, Y. Onda, T. Fukuwara, E. Obana, B. Kitchener, E. J. Long, and G. H. Hargrave (2012). "A new approach for simulating the redistribution of soil particles by water erosion: A marker-in-cell model". In: *Journal of Geophysical Research: Earth Surface* 117.F4. DOI: https://doi.org/10.1029/2012JF002499.

Cortis, M., W. Coombs, C. Augarde, M. Brown, A. Brennan, and S. Robinson (2018). "Imposition of essential boundary conditions in the material point method". In: *International Journal for Numerical Methods in Engineering* 113.1, pp. 130–152. DOI: https://doi.org/10.1002/nme.5606.

Cruden, D. M. and D. J. Varnes (1996). "Landslide types and processes". In: *Landslides: investigation and mitigation*. Ed. by T. editor. 247. Washington: National academy Press. Chap. 3, pp. 36–75.

Cui, Y., D. Chan, and A. Nouri (2017). "Coupling of solid deformation and pore pressure for undrained deformation—a discrete element method approach". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 41.18, pp. 1943–1961. DOI: https://doi.org/10.1002/nag.2708.

Cundall, P. (1971). "A computer model for simulating progressive, large-scale movements in blocky rock systems". In: *Proc. Symp. Int. Rock Mech.* Vol. 2, pp. 132–150.

Cundall, P. A. and O. D. L. Strack (1979). "A discrete numerical model for granular assemblies". In: *Géotechnique* 29.1, pp. 47–65. DOI: https://doi.org/10.1680/geot.1979.29.1.47.

Cuomo, S., P. Ghasemi, M. Martinelli, and M. Calvello (2019). "Simulation of Liquefaction and Retrogressive Slope Failure in Loose Coarse-Grained Material". In: *International Journal of Geomechanics* 19.10, p. 04019116. DOI: https://doi.org/10.1061/(ASCE)GM.1943-5622.0001500.

Dabrowski, M., M. Krotkiewski, and D. W. Schmid (2008). "MILAMIN: MATLAB-based finite element method solver for large problems". In: *Geochemistry, Geophysics, Geosystems* 9.4. DOI: https://doi.org/10.1029/2007GC001719.

Davis, T. A. (2013). *Suite Sparse*. URL: https://people.engr.tamu.edu/davis/research.html (visited on 08/25/2020).

De Blasio, F. V. (2011). *Introduction to the physics of landslides: lecture notes on the dynamics of mass wasting*. Springer Science & Business Media.

De Borst, R. (1988). "Bifurcations in finite element models with a non-associated flow law". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 12.1, pp. 99–116. DOI: https://doi.org/10.1002/nag.1610120107.

De Borst, R., L. J. Sluys, H. Muhlhaus, and J. Pamin (1993). "Fundamental issues in finite element analyses of localization of deformation". In: *Engineering Computations* 10.2, pp. 99–121. DOI: https://doi.org/10.1108/eb023897.

De Borst, R., M. A. Crisfield, J. J. C. Remmers, and C. V. Verhoosel (2012). *Nonlinear finite element analysis of solids and structures*. John Wiley & Sons.

De Borst, R. and H.-B. Mühlhaus (1992). "Gradient-dependent plasticity: Formulation and algorithmic aspects". In: *International Journal for Numerical Methods in Engineering* 35.3, pp. 521–539. DOI: https://doi.org/10.1002/nme.1620350307.

De Borst, R. and T. Duretz (2020). "On viscoplastic regularisation of strain-softening rocks and soils". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 44.6, pp. 890–903. DOI: https://doi.org/10.1002/nag.3046.

de Vaucorbeil, A., V. P. Nguyen, and C. R. Hutchinson (2020). "A Total-Lagrangian Material Point Method for solid mechanics problems involving large deformations". In: *Computer Methods in Applied Mechanics and Engineering* 360, p. 112783. DOI: https://doi.org/10.1016/j.cma.2019.112783.

Deboeuf, S., P. Gondret, and M. Rabaud (2009). "Dynamics of grain ejection by sphere impact on a granular bed". In: *Phys. Rev. E* 79 (4), p. 041306. DOI: 10.1103/PhysRevE.79.041306.

Dey, R., B. Hawlader, R. Phillips, and K. Soga (2015). "Large deformation finite-element modelling of progressive failure leading to spread in sensitive clay slopes". In: *Géotechnique* 65.8, pp. 657–668. DOI: 10.1680/geot.14.P.193.

Doghri, I. (2013). *Mechanics of Deformable Solids: Linear, Nonlinear, Analytical and Computational Aspects*. Springer Berlin Heidelberg.

Donev, A., F. H. Stillinger, P. M. Chaikin, and S. Torquato (2004). "Unusually Dense Crystal Packings of Ellipsoids". In: *Phys. Rev. Lett.* 92 (25), p. 255506. DOI: 10.1103/PhysRevLett.92.255506.

Dong, Y. and J. Grabe (2018). "Large scale parallelisation of the material point method with multiple GPUs". In: *Computers and Geotechnics* 101, pp. 149–158. DOI: https://doi.org/10.1016/j.compgeo.2018.04.001.

Dong, Y., D. Wang, and M. F. Randolph (2015). "A GPU parallel computing strategy for the material point method". In: *Computers and Geotechnics* 66, pp. 31–38. DOI: https://doi.org/10.1016/j.compgeo.2015.01.009.

Drucker, D. C. and W. Prager (1952). "SOIL MECHANICS AND PLASTIC ANALYSIS OR LIMIT DESIGN". In: *Quarterly of Applied Mathematics* 10.2, pp. 157–165.

Dubois, F., V. Acary, and M. Jean (2018). "The Contact Dynamics method: A nonsmooth story". In: *Comptes Rendus Mécanique* 346.3. The legacy of Jean-Jacques Moreau in mechanics / L'héritage de Jean-Jacques Moreau en mécanique, pp. 247–262. DOI: https://doi.org/10.1016/j.crme.2017.12.009.

Dufour, L. (Jan. 1962). "Microphysique des nuages". In: *Ciel et Terre* 78, p. 314.

Dunatunga, S. and K. Kamrin (2015). "Continuum modelling and simulation of granular flows through their many phases". In: *Journal of Fluid Mechanics* 779, 483–513. DOI: https://doi.org/10.1017/jfm.2015.383.

– (2017). "Continuum modeling of projectile impact and penetration in dry granular media". In: *Journal of the Mechanics and Physics of Solids* 100, pp. 45–60. DOI: https://doi.org/10.1016/j.jmps.2016.12.002.

Dunne, T., D. V. Malmon, and S. M. Mudd (2010). "A rain splash transport equation assimilating field and laboratory measurements". In: *Journal of Geophysical Research: Earth Surface* 115.F1. DOI: https://doi.org/10.1029/2009JF001302.

Duretz, T., R. de Borst, and L. Le Pourhiet (2019). "Finite Thickness of Shear Bands in Frictional Viscoplasticity and Implications for Lithosphere Dynamics". In: *Geochemistry, Geophysics, Geosystems* 20.11, pp. 5598–5616. DOI: https://doi.org/10.1029/2019GC008531.

Duretz, T., A. Souche, R. De Borst, and L. Le Pourhiet (2018). "The Benefits of Using a Consistent Tangent Operator for Viscoelastoplastic Computations in Geodynamics". In: *Geochemistry, Geophysics, Geosystems* 19.12, pp. 4904–4924. DOI: https://doi.org/10.1029/2018GC007877.

Edwards, B. F., J. W. Wilder, and E. E. Scime (2001). "Dynamics of falling raindrops". In: *European Journal of Physics* 22.2. DOI: 10.1088/0143-0807/22/2/302.

Farr, R. S. and R. D. Groot (2009). "Close packing density of polydisperse hard spheres". In: *The Journal of Chemical Physics* 131.24, p. 244104. DOI: 10.1063/1.3276799.

Feng, Z.-K. and W.-J. Xu (2021). "GPU material point method (MPM) and its application on slope stability analysis". In: *Bulletin of Engineering Geology and the Environment* 80.7, pp. 5437–5449. DOI: 10.1007/s10064-021-02265-8.

Fenton, G. A. and E. H. Vanmarcke (1990). "Simulation of Random Fields via Local Average Subdivision". In: *Journal of Engineering Mechanics* 116.8, pp. 1733–1749. DOI: https://doi.org/10.1061/(ASCE)0733-9399(1990)116:8(1733).

Fern, E. J. and K. Soga (2016). "The role of constitutive models in MPM simulations of granular column collapses". In: *Acta Geotechnica* 11.3, pp. 659–678.

Fern, J., A. Rohe, K. Soga, and E. Alonso (2019). *The material point method for geotechnical engineering: a practical guide*. CRC Press.

Foot, K. and R. P. C. Morgan (2005). "The role of leaf inclination, leaf orientation and plant canopy architecture in soil particle detachment by raindrops". In: *Earth Surface Processes and Landforms* 30.12, pp. 1509–1520. DOI: https://doi.org/10.1002/esp.1207.

Foote, G. B. and P. S. D. Toit (1Apr. 1969). "Terminal Velocity of Raindrops Aloft". In: *Journal of Applied Meteorology and Climatology* 8.2, pp. 249 – 253. DOI: 10.1175/1520-0450(1969)008<0249:TVORA>2.0.CO;2.

Furbish, D. J., P. K. Haff, W. E. Dietrich, and A. M. Heimsath (2009). "Statistical description of slope-dependent soil transport and the diffusion-like coefficient". In: *Journal of Geophysical Research: Earth Surface* 114.F3. DOI: https://doi.org/10.1029/2009JF001267.

Furbish, D. J., K. K. Hamner, M. Schmeeckle, M. N. Borosund, and S. M. Mudd (2007). "Rain splash of dry sand revealed by high-speed imaging and sticky paper splash targets". In: *Journal of Geophysical Research: Earth Surface* 112.F1. DOI: https://doi.org/10.1029/2006JF000498.

Galavi, V. and H. F. Schweiger (2010). "Nonlocal Multilaminate Model for Strain Softening Analysis". In: *International Journal of Geomechanics* 10.1, pp. 30–44. DOI: 10.1061/(ASCE)1532-3641(2010)10:1(30).

Gan, Y., Z. Sun, Z. Chen, X. Zhang, and Y. Liu (2018). "Enhancement of the material point method using B-spline basis functions". In: *International Journal for Numerical Methods in Engineering* 113.3, pp. 411–431. DOI: https://doi.org/10.1002/nme.5620.

Gao, M., X. Wang, K. Wu, A. Pradhana, E. Sifakis, C. Yuksel, and C. Jiang (2018). "GPU Optimization of Material Point Methods". In: *ACM Trans. Graph.* 37.6. DOI: 10.1145/3272127.3275044.

Gaume, J., T. Gast, J. Teran, A. van Herwijnen, and C. Jiang (2018). "Dynamic anticrack propagation in snow". In: *Nature Communications* 9.1, p. 3047. DOI: https://doi.org/10.1038/s41467-018-05181-w.

Gaume, J., A. van Herwijnen, T. Gast, J. Teran, and C. Jiang (2019). "Investigating the release and flow of snow avalanches at the slope-scale using a unified model based on the material point method". In: *Cold Regions Science and Technology* 168, p. 102847. DOI: https://doi.org/10.1016/j.coldregions.2019.102847.

Gerya, T. (2010). *Introduction to numerical geodynamic modelling*. Cambridge University Press.

Ghadiri, H. and D. Payne (1979). "Raindrop impact and soil splashs". In: *Soil Physical Properties and Crop Production in the Tropics*. John Wiley and Sons.

Ghadiri, H. (2004). "Crater formation in soils by raindrop impact". In: *Earth Surface Processes and Landforms* 29.1, pp. 77–89. DOI: https://doi.org/10.1002/esp.1014.

Goldman, D. I. and P. Umbanhowar (2008). "Scaling and dynamics of sphere and disk impact into granular media". In: *Phys. Rev. E* 77 (2), p. 021308. DOI: 10.1103/PhysRevE.77.021308.

González Acosta, J. L., X. Zheng, P. J. Vardon, M. A. Hicks, and F. Pisano (2019). "On stress oscillation in MPM simulations involving one or two phases". In: *Proceedings of the Second International Conference on the Material Point Method for Modelling Soil-Water-Structure Interaction*.

González Acosta, J. L., P. J. Vardon, G. Remmerswaal, and M. A. Hicks (2020). "An investigation of stress inaccuracies and proposed solution in the material point method". In: *Computational Mechanics* 65.2, pp. 555–581. DOI: https://doi.org/10.1007/s00466-019-01783-3.

González Acosta, J. L., P. J. Vardon, and M. A. Hicks (2021). "Development of an implicit contact technique for the material point method". In: *Computers and Geotechnics* 130, p. 103859. DOI: https://doi.org/10.1016/j.compgeo.2020.103859.

Gracia, F., P. Villard, and V. Richefeu (2019). "Comparison of two numerical approaches (DEM and MPM) applied to unsteady flow". In: *Computational Particle Mechanics* 6.4, pp. 591–609. DOI: https://doi.org/10.1007/s40571-019-00236-1.

Grasselli, Y., H. J. Herrmann, G. Oron, and S. Zapperi (2000). "Effect of impact energy on the shape of granular heaps". In: *Granular Matter* 2, pp. 97–100.

Gravish, N., P. B. Umbanhowar, and D. I. Goldman (2010). "Force and Flow Transition in Plowed Granular Media". In: *Phys. Rev. Lett.* 105 (12), p. 128301. DOI: 10.1103/PhysRevLett.105.128301.

Griffiths, D. V. and G. A. Fenton (2004). "Probabilistic Slope Stability Analysis by Finite Elements". In: *Journal of Geotechnical and Geoenvironmental Engineering* 130.5, pp. 507–518. DOI: 10.1061/(ASCE)1090-0241(2004)130:5(507).

Guilkey, J. E. and J. A. Weiss (2003). "Implicit time integration for the material point method: Quantitative and algorithmic comparisons with the finite element method". In: *International Journal for Numerical Methods in Engineering* 57.9, pp. 1323–1338. DOI: https://doi.org/10.1002/nme.729.

Hapgood, K. P., J. D. Litster, S. R. Biggs, and T. Howes (2002). "Drop Penetration into Porous Powder Beds". In: *Journal of Colloid and Interface Science* 253.2, pp. 353–366. DOI: https://doi.org/10.1006/jcis.2002.8527.

Hashiguchi, K. and Y. Yamakawa (2012). *Introduction to Finite Strain Theory for Continuum Elasto-Plasticity*. Wiley Series in Computational Mechanics. Wiley.

Heyman, J., R. Delannay, H. Tabuteau, and A. Valance (2017). "Compressibility regularizes the mu(I)-rheology for dense granular flows". In: *Journal of Fluid Mechanics* 830, 553–568. DOI: 10.1017/jfm.2017.612.

Homel, M. A., R. M. Brannon, and J. Guilkey (2016). "Controlling the onset of numerical fracture in parallelized implementations of the material point method (MPM) with convective particle domain interpolation (CPDI) do-

main scaling". In: *International Journal for Numerical Methods in Engineering* 107.1, pp. 31–48. DOI: https://doi.org/10.1002/nme.5151.

Hu, Y., T.-M. Li, L. Anderson, J. Ragan-Kelley, and F. Durand (Nov. 2019). "Taichi: A Language for High-Performance Computation on Spatially Sparse Data Structures". In: *ACM Trans. Graph.* 38.6. DOI: 10.1145/3355089.3356506.

Huang, J. and D. V. Griffiths (2008). "Observations on Return Mapping Algorithms for Piecewise Linear Yield Criteria". In: *International Journal of Geomechanics* 8.4, pp. 253–265. DOI: https://doi.org/10.1061/(ASCE)1532-3641(2008)8:4(253).

– (2009). "Return Mapping Algorithms and Stress Predictors for Failure Analysis in Geomechanics". In: *Journal of Engineering Mechanics* 135.4, pp. 276–284. DOI: https://doi.org/10.1061/(ASCE)0733-9399(2009)135:4(276).

Huang, P., S.-l. Li, H. Guo, and Z.-m. Hao (2015). "Large deformation failure analysis of the soil slope based on the material point method". In: *Computational Geosciences* 19.4, pp. 951–963. DOI: https://doi.org/10.1007/s10596-015-9512-9.

Hungr, O., S. Leroueil, and L. Picarelli (2014). "The Varnes classification of landslide types, an update". In: *Landslides* 11.2, pp. 167–194. DOI: 10.1007/s10346-013-0436-y.

Iaconeta, I., A. Larese, R. Rossi, and E. Oñate (2019). "A stabilized mixed implicit Material Point Method for non-linear incompressible solid mechanics". In: *Computational Mechanics* 63.6, pp. 1243–1260. DOI: https://doi.org/10.1007/s00466-018-1647-9.

Iaconeta, I., A. Larese, R. Rossi, and Z. Guo (2017). "Comparison of a Material Point Method and a Galerkin Meshfree Method for the Simulation of Cohesive-Frictional Materials". eng. In: *Materials (Basel, Switzerland)* 10.10. 28974013[pmid], p. 1150. DOI: https://doi.org/10.3390/ma10101150.

Islam, N., B. Hawlader, C. Wang, and K. Soga (2019). "Large-deformation finite-element modelling of earthquake-induced landslides considering strain-softening behaviour of sensitive clay". In: *Canadian Geotechnical Journal* 56.7, pp. 1003–1018. DOI: 10.1139/cgj-2018-0250.

Jassim, I., D. Stolle, and P. Vermeer (2013). "Two-phase dynamic analysis by material point method". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 37.15, pp. 2502–2522. DOI: https://doi.org/10.1002/nag.2146.

Jean, M. (1999). "The non-smooth contact dynamics method". In: *Computer Methods in Applied Mechanics and Engineering* 177.3, pp. 235–257. DOI: https://doi.org/10.1016/S0045-7825(98)00383-1.

Jiang, H. and Y. Xie (2011). "A note on the Mohr–Coulomb and Drucker–Prager strength criteria". In: *Mechanics Research Communications* 38.4, pp. 309–314. DOI: https://doi.org/10.1016/j.mechrescom.2011.04.001.

Jiang, Z., J. Du, C. Rieck, A. Bück, and E. Tsotsas (2020). "PTV experiments and DEM simulations of the coefficient of restitution for irregular particles

impacting on horizontal substrates". In: *Powder Technology* 360, pp. 352–365. DOI: https://doi.org/10.1016/j.powtec.2019.10.072.

Jin, W., Y. Jiao, L. Liu, Y. Yuan, and S. Li (2017). "Dense crystalline packings of ellipsoids". In: *Phys. Rev. E* 95 (3), p. 033003. DOI: 10.1103/PhysRevE.95.033003.

Jong, R. de, S.-C. Zhao, and D. van der Meer (2017). "Crater formation during raindrop impact on sand". In: *Phys. Rev. E* 95 (4), p. 042901. DOI: 10.1103/PhysRevE.95.042901.

Katsuragi, H. (2010). "Morphology Scaling of Drop Impact onto a Granular Layer". In: *Phys. Rev. Lett.* 104 (21), p. 218001. DOI: 10.1103/PhysRevLett.104.218001.

– (2011). "Length and time scales of a liquid drop impact and penetration into a granular layer". In: *Journal of Fluid Mechanics* 675, 552–573. DOI: 10.1017/jfm.2011.31.

Katsuragi, H. and D. J. Durian (2007). "Unified force law for granular impact cratering". In: *Nature Physics* 3.6, pp. 420–423. DOI: 10.1038/nphys583.

Kaus, B. J. P. (2010). "Factors that control the angle of shear bands in geodynamic numerical models of brittle deformation". In: *Tectonophysics* 484.1. Quantitative modelling of geological processes, pp. 36–47. DOI: https://doi.org/10.1016/j.tecto.2009.08.042.

Kawamoto, R., E. Andò, G. Viggiani, and J. E. Andrade (2018). "All you need is shape: Predicting shear banding in sand with LS-DEM". In: *Journal of the Mechanics and Physics of Solids* 111, pp. 375–392. DOI: https://doi.org/10.1016/j.jmps.2017.10.003.

Kleinhans, M. G., H. Markies, S. J. de Vet, A. C. in 't Veld, and F. N. Postema (2011). "Static and dynamic angles of repose in loose granular materials under reduced gravity". In: *Journal of Geophysical Research: Planets* 116.E11. DOI: 10.1029/2011JE003865.

KONICA (2004). *Instruction Manual (HARDWARE)*. URL: http://www.konicaminolta.com/instruments/download/instruction_manual/3d/pdf/vivid-9i_vi-9i_instruction_eng.pdf.

Koster, P. de, R. Tielen, E. Wobbes, and M. Möller (2021). "Extension of B-spline Material Point Method for unstructured triangular grids using Powell–Sabin splines". In: *Computational Particle Mechanics* 8.2, pp. 273–288. DOI: https://doi.org/10.1007/s40571-020-00328-3.

Krabbenhoft, K., M. R. Karim, A. V. Lyamin, and S. W. Sloan (2012). "Associated computational plasticity schemes for nonassociated frictional materials". In: *International Journal for Numerical Methods in Engineering* 90.9, pp. 1089–1117. DOI: https://doi.org/10.1002/nme.3358.

Kumar, K., J.-Y. Delenne, and K. Soga (2017). "Mechanics of granular column collapse in fluid at varying slope angles". In: *Journal of Hydrodynamics* 29.4, pp. 529–541.

Lagrée, P.-Y., L. Staron, and S. Popinet (2011). "The granular column collapse as a continuum: validity of a two-dimensional Navier-Stokes model with a [mu](I)-rheology". In: *Journal of Fluid Mechanics* 686, p. 378.

Lajeunesse, E, A Mangeney-Castelnau, and J. Vilotte (2004). "Spreading of a granular mass on a horizontal plane". In: *Physics of fluids* 16.7, pp. 2371–2381.

Langlois, V. J., A. Quiquerez, and P. Allemand (2015). "Collapse of a two-dimensional brittle granular column: Implications for understanding dynamic rock fragmentation in a landslide". In: *Journal of Geophysical Research: Earth Surface* 120.9, pp. 1866–1880. DOI: https://doi.org/10.1002/2014JF003330.

Lätzel, M., S. Luding, and H. J. Herrmann (2000). "Macroscopic material properties from quasi-static, microscopic simulations of a two-dimensional shear-cell". In: *Granular Matter* 2.3, pp. 123–135. DOI: https://doi.org/10.1007/s100350000048.

Le Pourhiet, L. (July 2013). "Strain localization due to structural softening during pressure sensitive rate independent yielding". In: *Bulletin de la Société Géologique de France* 184.4-5, pp. 357–371. DOI: https://doi.org/10.2113/gssgfbull.184.4-5.357.

Leavy, R. B., J. E. Guilkey, B. R. Phung, A. D. Spear, and R. M. Brannon (2019). "A convected-particle tetrahedron interpolation technique in the material-point method for the mesoscale modeling of ceramics". In: *Computational Mechanics* 64.3, pp. 563–583. DOI: https://doi.org/10.1007/s00466-019-01670-x.

Lee, J and H. J. Herrmann (1993). "Angle of repose and angle of marginal stability: molecular dynamics of granular particles". In: *Journal of Physics A: Mathematical and General* 26.2, 373–383. DOI: 10.1088/0305-4470/26/2/021.

Lei, X., S. He, and L. Wu (2020). "Stabilized generalized interpolation material point method for coupled hydro-mechanical problems". In: *Computational Particle Mechanics*. DOI: https://doi.org/10.1007/s40571-020-00365-y.

Leroueil, S., J. Vaunat, L. Picarelli, J. Locat, and K. Senneset (1996). "Geotechnical characterization of slope movements, International symposium; 7th, Landslides". In: *Landslides*. Vol. 1. A.A. Balkema; pp. 53–74.

Li, L., E. Marteau, and J. E. Andrade (2019). "Capturing the inter-particle force distribution in granular material using LS-DEM". In: *Granular Matter* 21.3, p. 43. DOI: https://doi.org/10.1007/s10035-019-0893-7.

Li, W. L., N. Guo, Z. X. Yang, and T. Helfer (2021). "Large-deformation geomechanical problems studied by a shear-transformation-zone model using the material point method". In: *Computers and Geotechnics* 135, p. 104153. DOI: https://doi.org/10.1016/j.compgeo.2021.104153.

Li, Y., Y. Xu, and C. Thornton (2005). "A comparison of discrete element simulations and experiments for 'sandpiles' composed of spherical particles". In: *Powder Technology* 160.3, pp. 219–228. DOI: https://doi.org/10.1016/j.powtec.2005.09.002.

Lim, K.-W. and J. E. Andrade (2014a). "Granular element method for three-dimensional discrete element calculations". In: *International Journal for Nu-*

merical and Analytical Methods in Geomechanics 38.2, pp. 167–188. DOI: https://doi.org/10.1002/nag.2203.

Lim, K.-W., K. Krabbenhoft, and J. E. Andrade (2014b). "On the contact treatment of non-convex particles in the granular element method". In: *Computational Particle Mechanics* 1.3, pp. 257–275. DOI: https://doi.org/10.1007/s40571-014-0019-2.

Lim, K.-W., K. Krabbenhoft, and J. E. Andrade (2014c). "A contact dynamics approach to the Granular Element Method". In: *Computer Methods in Applied Mechanics and Engineering* 268, pp. 557–573. DOI: https://doi.org/10.1016/j.cma.2013.10.004.

Liu, C., Q. Sun, and G. G. Zhou (2018a). "Coupling of material point method and discrete element method for granular flows impacting simulations". In: *International Journal for Numerical Methods in Engineering* 115.2, pp. 172–188. DOI: https://doi.org/10.1002/nme.5800.

– (2018b). "Coupling of material point method and discrete element method for granular flows impacting simulations". In: *International Journal for Numerical Methods in Engineering* 115.2, pp. 172–188. DOI: https://doi.org/10.1002/nme.5800.

Liu, X., Y. Wang, and D.-Q. Li (2019). "Investigation of slope failure mode evolution during large deformation in spatially variable soils by random limit equilibrium and material point methods". In: *Computers and Geotechnics* 111, pp. 301–312. DOI: https://doi.org/10.1016/j.compgeo.2019.03.022.

– (2020). "Numerical simulation of the 1995 rainfall-induced Fei Tsui Road landslide in Hong Kong: new insights from hydro-mechanically coupled material point method". In: *Landslides* 17.12, pp. 2755–2775. DOI: https://doi.org/10.1007/s10346-020-01442-2.

Liu, Y., F.-H. Lee, S.-T. Quek, E. J. Chen, and J.-T. Yi (2015). "Effect of spatial variation of strength and modulus on the lateral compression response of cement-admixed clay slab". In: *Géotechnique* 65.10, pp. 851–865. DOI: 10.1680/jgeot.14.P.254.

Lo Giudice, A, G Giammanco, D Fransos, and L Preziosi (2019). "Modeling sand slides by a mechanics-based degenerate parabolic equation". In: *Mathematics and Mechanics of Solids* 24.8, pp. 2558–2575.

Long, E. J., G. K. Hargrave, J. R. Cooper, B. G. B. Kitchener, A. J. Parsons, C. J. M. Hewett, and J. Wainwright (2014). "Experimental investigation into the impact of a liquid droplet onto a granular bed using three-dimensional, time-resolved, particle tracking". In: *Phys. Rev. E* 89 (3), p. 032201. DOI: 10.1103/PhysRevE.89.032201.

Lowe, D. R. (1976). "Grain flow and grain flow deposits". In: *Journal of Sedimentary Research* 46, pp. 188–199.

Lube, G., H. E. Huppert, R. S. J. Sparks, and A. Freundt (2005). "Collapses of two-dimensional granular columns". In: *Physical Review E* 72.4, p. 041301.

Lubliner, J. (2008). *Plasticity Theory*. Dover books on engineering. Dover Publications.

Luding, S. (2008). "Cohesive, frictional powders: contact models for tension". In: *Granular Matter* 10.4, p. 235. DOI: https://doi.org/10.1007/s10035-008-0099-x.

Malone, K. F. and B. H. Xu (2008). "Determination of contact parameters for discrete element method simulations of granular systems". In: *Particuology* 6.6. Simulation and Modeling of Particulate Systems, pp. 521–528. DOI: https://doi.org/10.1016/j.partic.2008.07.012.

Mangeney, A, O. Roche, O Hungr, N Mangold, G. Faccanoni, and A Lucas (2010). "Erosion and mobility in granular collapse over sloping beds". In: *Journal of Geophysical Research: Earth Surface* 115.F3.

Marshall, J. S. and W. Mck. Palmer (1948). "The distribution of raindrops with size". In: *Journal of Meteorology* 5, pp. 165–166.

Marston, J. O., E. Q. Li, and S. T. Thoroddsen (2012). "Evolution of fluid-like granular ejecta generated by sphere impact". In: *Journal of Fluid Mechanics* 704, 5–36. DOI: 10.1017/jfm.2012.141.

Marston, J. O. and S. T. Thoroddsen (2015). "Investigation of granular impact using positron emission particle tracking". In: *Powder Technology* 274, pp. 284–288. DOI: https://doi.org/10.1016/j.powtec.2015.01.033.

Marston, J. O., S. T. Thoroddsen, W. K. Ng, and R. B. H. Tan (2010). "Experimental study of liquid drop impact onto a powder surface". In: *Powder Technology* 203.2, pp. 223–236. DOI: https://doi.org/10.1016/j.powtec.2010.05.012.

Mason, B. J. (1971). *The Physics of Clouds*. Oxford University Press.

Mast, C. M., P. Mackenzie-Helnwein, P. Arduino, G. R. Miller, and W. Shin (2012). "Mitigating kinematic locking in the material point method". In: *Journal of Computational Physics* 231.16, pp. 5351–5373. DOI: https://doi.org/10.1016/j.jcp.2012.04.032.

Mériaux, C. (2006). "Two dimensional fall of granular columns controlled by slow horizontal withdrawal of a retaining wall". In: *Physics of Fluids* 18.9, p. 093301.

Métayer, J.-F., I. I. I. D. J. Suntrup, C. Radin, H. L. Swinney, and M. Schröter (2011). "Shearing of frictional sphere packings". In: *EPL (Europhysics Letters)* 93.6, p. 64003. DOI: 10.1209/0295-5075/93/64003.

Middleton, G. V. and P. R. Wilcock (1994). *Mechanics in the Earth and Environmental Sciences*. Cambridge University Press.

Miller, R. L. and R. J. Byrne (1966). "The angle of repose for a single grain on a fixed rough bed". In: *Sedimentology* 6.4, pp. 303–314.

Mitarai, N. and F. Nori (2006). "Wet granular materials". In: *Advances in Physics* 55.1-2, pp. 1–45.

Moler, C. (2000). *MATLAB Incorporates LAPACK*. URL: https://ch.mathworks.com/de/company/newsletters/articles/matlab-incorporates-lapack.html?refresh=true (visited on 08/25/2020).

Moreau, J. J. (2000). "Contact et frottement en dynamique des systèmes de corps rigides". In: *Revue Européenne des Éléments Finis* 9.1-3, pp. 9–28. DOI: https://doi.org/10.1080/12506559.2000.10511427.

Morris, A. B., S. Pannala, Z. Ma, and C. M. Hrenya (2016). "Development of soft-sphere contact models for thermal heat conduction in granular flows". In: *AIChE Journal* 62.12, pp. 4526–4535. DOI: https://doi.org/10.1002/aic.15331.

Müller, A. and E. A. Vargas (2019). "Stability analysis of a slope under impact of a rock block using the generalized interpolation material point method (GIMP)". In: *Landslides* 16.4, pp. 751–764. DOI: https://doi.org/10.1007/s10346-018-01131-1.

Mühlhaus, H. B. and I. Vardoulakis (1987). "The thickness of shear bands in granular materials". In: *Géotechnique* 37.3, pp. 271–283. DOI: 10.1680/geot.1987.37.3.271.

Nagtegaal, J. C., D. M. Parks, and J. R. Rice (1974). "On numerically accurate finite element solutions in the fully plastic range". In: *Computer Methods in Applied Mechanics and Engineering* 4.2, pp. 153–177. DOI: https://doi.org/10.1016/0045-7825(74)90032-2.

Nairn, J. A. (2003). "Material Point Method Calculations with Explicit Cracks". In: *Computer Modeling in Engineering & Sciences* 4.6, pp. 649–664. DOI: https://doi.org/10.3970/cmes.2003.004.649.

Nayak, G. C. and O. C. Zienkiewicz (1972). "Elasto-plastic stress analysis. A generalization for various contitutive relations including strain softening". In: *International Journal for Numerical Methods in Engineering* 5.1, pp. 113–135. DOI: https://doi.org/10.1002/nme.1620050111.

Nedderman, R. M. (1992). *Statics and Kinematics of Granular Materials*. Cambridge University Press.

Nefzaoui, E. and O. Skurtys (2012a). "Impact of a liquid drop on a granular medium: Inertia, viscosity and surface tension effects on the drop deformation". In: *Experimental Thermal and Fluid Science* 41, pp. 43–50. DOI: https://doi.org/10.1016/j.expthermflusci.2012.03.007.

– (2012b). "Impact of a liquid drop on a granular medium: Inertia, viscosity and surface tension effects on the drop deformation". In: *Experimental Thermal and Fluid Science* 41, pp. 43–50. DOI: https://doi.org/10.1016/j.expthermflusci.2012.03.007.

Nguyen, N. H. T., H. H. Bui, and G. D. Nguyen (2020). "Effects of material properties on the mobility of granular flow". In: *Granular Matter* 22.3, p. 59. DOI: https://doi.org/10.1007/s10035-020-01024-y.

Ni, R. and X. Zhang (2020). "A precise critical time step formula for the explicit material point method". In: *International Journal for Numerical Methods in Engineering* 121.22, pp. 4989–5016. DOI: https://doi.org/10.1002/nme.6506.

Nicot, F., N. Hadda, M. Guessasma, J. Fortin, and O. Millet (2013). "On the definition of the stress tensor in granular media". In: *International Journal of Solids and Structures* 50.14, pp. 2508–2517. DOI: https://doi.org/10.1016/j.ijsolstr.2013.04.001.

Nordstrom, K. N., E. Lim, M. Harrington, and W. Losert (2014). "Granular Dynamics During Impact". In: *Phys. Rev. Lett.* 112 (22), p. 228002. DOI: 10.1103/PhysRevLett.112.228002.

Nvidia (Mar. 20, 2021). *CUDA Programming Guide Version 1.0*. URL: http://developer.download.nvidia.com/compute/cuda/1.0/NVIDIA_CUDA_Programming_Guide_1.0.pdf.

Omlin, S. (2017). "Development of massively parallel near peak performance solvers for three-dimensional geodynamic modelling". PhD thesis. University of Lausanne (UNIL).

Ortiz, M. and J. C. Simo (1986). "An analysis of a new class of integration algorithms for elastoplastic constitutive relations". In: *International Journal for Numerical Methods in Engineering* 23.3, pp. 353–366. DOI: https://doi.org/10.1002/nme.1620230303.

Ostojic, S., E. Somfai, and B. Nienhuis (2006). "Scale invariance and universality of force networks in static granular matter". In: *Nature* 439.7078, pp. 828–830. DOI: 10.1038/nature04549.

O'Sullivan, S., R. E. Bird, W. M. Coombs, and S. Giani (2019). "Rapid nonlinear finite element analysis of continuous and discontinuous Galerkin methods in MATLAB". In: *Computers & Mathematics with Applications* 78.9. Applications of Partial Differential Equations in Science and Engineering, pp. 3007–3026. DOI: https://doi.org/10.1016/j.camwa.2019.03.012.

Parteli, E. J. R., J. Schmidt, C. Blümel, K.-E. Wirth, W. Peukert, and T. Pöschel (2014). "Attractive particle interaction forces and packing density of fine glass powders". In: *Scientific Reports* 4.1, p. 6227. DOI: https://doi.org/10.1038/srep06227.

Pohlman, N. A., B. L. Severson, J. M. Ottino, and R. M. Lueptow (2006). "Surface roughness effects in granular matter: Influence on angle of repose and the absence of segregation". In: *Physical Review E* 73.3, p. 031304.

Poirier, J.-P. (1985). *Creep of Crystals: High-Temperature Deformation Processes in Metals, Ceramics and Minerals*. Cambridge Earth Science Series. Cambridge University Press. DOI: https://doi.org/10.1017/CBO9780511564451.

Poschel, T. and T. Schwager (2005). *Computational Granular Dynamics: Model and Algorithms*. Springer.

Pudasaini, S. P. and K. Hutter (2007). *Avalanche Dynamics: Dynamics of Rapid Flows of Dense Granular Avalanches*. Springer.

Räss, L., A. Licul, F. Herman, Y. Y. Podladchikov, and J. Suckale (2020). "Modelling thermomechanical ice deformation using an implicit pseudo-transient method (FastICE v1.0) based on graphical processing units (GPUs)". In: *Geoscientific Model Development* 13.3, pp. 955–976. DOI: 10.5194/gmd-13-955-2020.

Remmerswaal, G., P. J. Vardon, and M. A. Hicks (2021). "Evaluating residual dyke resistance using the Random Material Point Method". In: *Computers and Geotechnics* 133, p. 104034. DOI: https://doi.org/10.1016/j.compgeo.2021.104034.

Remy, B., J. G. Khinast, and B. J. Glasser (2009). "Discrete element simulation of free flowing grains in a four-bladed mixer". In: *AIChE Journal* 55.8, pp. 2035–2048. DOI: https://doi.org/10.1002/aic.11876.

Roessler, T. and A. Katterfeld (2019). "DEM parameter calibration of cohesive bulk materials using a simple angle of repose test". In: *Particuology* 45, pp. 105–115. DOI: https://doi.org/10.1016/j.partic.2018.08.005.

Rojek, J., C. Labra, O. Su, and E. Oñate (2012). "Comparative study of different discrete element models and evaluation of equivalent micromechanical parameters". In: *International Journal of Solids and Structures* 49.13, pp. 1497–1517. DOI: https://doi.org/10.1016/j.ijsolstr.2012.02.032.

Rondon, L., O. Pouliquen, and P. Aussillous (2011). "Granular collapse in a fluid: role of the initial volume fraction". In: *Physics of Fluids* 23.7, p. 073301.

Royer, J. R., B. Conyers, E. I. Corwin, P. J. Eng, and H. M. Jaeger (2011). "The role of interstitial gas in determining the impact response of granular beds". In: *EPL (Europhysics Letters)* 93.2, p. 28008. DOI: 10.1209/0295-5075/93/28008.

Royer, J. R., E. I. Corwin, P. J. Eng, and H. M. Jaeger (2007). "Gas-Mediated Impact Dynamics in Fine-Grained Granular Materials". In: *Phys. Rev. Lett.* 99 (3), p. 038003. DOI: 10.1103/PhysRevLett.99.038003.

Rycroft, C. H., K. Kamrin, and M. Z. Bazant (2009). "Assessing continuum postulates in simulations of granular flow". In: *Journal of the Mechanics and Physics of Solids* 57.5, pp. 828–839. DOI: https://doi.org/10.1016/j.jmps.2009.01.009.

Räss, L., T. Duretz, and Y. Y. Podladchikov (May 2019a). "Resolving hydromechanical coupling in two and three dimensions: spontaneous channelling of porous fluids owing to decompaction weakening". In: *Geophysical Journal International* 218.3, pp. 1591–1616. DOI: 10.1093/gji/ggz239.

Räss, L., T. Duretz, Y. Y. Podladchikov, and S. M. Schmalholz (2017). "M2Di: Concise and efficient MATLAB 2-D Stokes solvers using the Finite Difference Method". In: *Geochemistry, Geophysics, Geosystems* 18.2, pp. 755–768. DOI: https://doi.org/10.1002/2016GC006727.

Räss, L., D. Kolyukhin, and A. Minakov (2019b). "Efficient parallel random field generator for large 3-D geophysical problems". In: *Computers & Geosciences* 131, pp. 158–169. DOI: https://doi.org/10.1016/j.cageo.2019.06.007.

Sabet, S. A. and R. De Borst (2019). "Structural softening, mesh dependence, and regularisation in non-associated plastic flow". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 43.13, pp. 2170–2183. DOI: https://doi.org/10.1002/nag.2973.

Sabuwala, T., C. Butcher, G. Gioia, and P. Chakraborty (2018). "Ray Systems in Granular Cratering". In: *Phys. Rev. Lett.* 120 (26), p. 264501. DOI: 10.1103/PhysRevLett.120.264501.

Sadeghirad, A., R. M. Brannon, and J. Burghardt (2011). "A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations". In: *International Journal for Numerical Methods in Engineering* 86.12, pp. 1435–1456. DOI: https://doi.org/10.1002/nme.3110.

Sadeghirad, A., R. M. Brannon, and J. E. Guilkey (2013). "Second-order convected particle domain interpolation (CPDI2) with enrichment for weak discontinuities at material interfaces". In: *International Journal for Numerical Methods in Engineering* 95.11, pp. 928–952. DOI: https://doi.org/10.1002/nme.4526.

Schröter, M., S. Nägle, C. Radin, and H. L. Swinney (2007). "Phase transition in a static granular system". In: *Europhysics Letters (EPL)* 78.4, p. 44004. DOI: 10.1209/0295-5075/78/44004.

Seguin, A., Y. Bertho, and P. Gondret (2008). "Influence of confinement on granular penetration by impact". In: *Phys. Rev. E* 78 (1), p. 010301. DOI: 10.1103/PhysRevE.78.010301.

Shan, Z., W. Zhang, D. Wang, and L. Wang (2021). "Numerical investigations of retrogressive failure in sensitive clays: revisiting 1994 Sainte-Monique slide, Quebec". In: *Landslides* 18.4, pp. 1327–1336. DOI: 10.1007/s10346-020-01567-4.

Simo, J. C. and T. J. R. Hughes (1998). *Computational inelasticity*. Interdisciplinary applied mathematics Mechanics and materials. New York: Springer.

Simo, J. C. and M. Ortiz (1985a). "A unified approach to finite deformation elastoplastic analysis based on the use of hyperelastic constitutive equations". In: *Computer Methods in Applied Mechanics and Engineering* 49.2, pp. 221–245. DOI: https://doi.org/10.1016/0045-7825(85)90061-1.

Simo, J. C. and R. L. Taylor (1985b). "Consistent tangent operators for rate-independent elastoplasticity". In: *Computer Methods in Applied Mechanics and Engineering* 48.1, pp. 101–118. DOI: https://doi.org/10.1016/0045-7825(85)90070-2.

Simpson, G. (2017). *Practical finite element modeling in earth science using matlab*. Wiley Online Library.

Sinaie, S., V. P. Nguyen, C. T. Nguyen, and S. Bordas (2017). "Programming the material point method in Julia". In: *Advances in Engineering Software* 105, pp. 17–29. DOI: https://doi.org/10.1016/j.advengsoft.2017.01.008.

Skempton, A. and J. Hutchinson (1969). "Stability of natural slopes and embankment foundations". In: *Soil Mech & Fdn Eng Conf Proc/Mexico/*.

Soga, K., E. Alonso, A. Yerro, K. Kumar, and S. Bandara (2016). "Trends in large-deformation analysis of landslide mass movements with particular emphasis on the material point method". In: *Géotechnique* 66.3, pp. 248–273. DOI: https://doi.org/10.1680/jgeot.15.LM.005.

Souza Neto, E. A. de, D. Peric, and D. R. J. Owen (2011). *Computational methods for plasticity: theory and applications*. John Wiley & Sons.

Spencer, A. J. M. (2012). *Continuum Mechanics*. Dover Books on Physics. Dover Publications.

Starman, B., M. Halilovič, M. Vrh, and B. Štok (2014). "Consistent tangent operator for cutting-plane algorithm of elasto-plasticity". In: *Computer Methods in Applied Mechanics and Engineering* 272, pp. 214–232. DOI: https://doi.org/10.1016/j.cma.2013.12.012.

Staron, L. and E. J. Hinch (2005). "Study of the collapse of granular columns using two-dimensional discrete-grain simulation". In: *Journal of Fluid Mechanics* 545, 1–27. DOI: https://doi.org/10.1017/S0022112005006415.

Steffen, M., P. C. Wallstedt, J. E. Guilkey, R. M. Kirby, and M. Berzins (2008a). "Examination and Analysis of Implementation Choices within the Material Point Method (MPM)". In: *Computer Modeling in Engineering & Sciences* 31.2, pp. 107–128. DOI: https://doi.org/10.3970/cmes.2008.031.107.

Steffen, M., R. M. Kirby, and M. Berzins (2008b). "Analysis and reduction of quadrature errors in the material point method (MPM)". In: *International Journal for Numerical Methods in Engineering* 76.6, pp. 922–948. DOI: https://doi.org/10.1002/nme.2360.

Stomakhin, A., C. Schroeder, L. Chai, J. Teran, and A. Selle (July 2013). "A Material Point Method for Snow Simulation". In: *ACM Trans. Graph.* 32.4. DOI: https://doi.org/10.1145/2461912.2461948.

Sulsky, D., Z. Chen, and H. L. Schreyer (1994). "A particle method for history-dependent materials". In: *Computer Methods in Applied Mechanics and Engineering* 118.1, pp. 179–196. DOI: https://doi.org/10.1016/0045-7825(94)90112-0.

Sulsky, D. and H. L. Schreyer (1996). "Axisymmetric form of the material point method with applications to upsetting and Taylor impact problems". In: *Computer Methods in Applied Mechanics and Engineering* 139.1, pp. 409–429. DOI: https://doi.org/10.1016/S0045-7825(96)01091-2.

Sulsky, D., S.-J. Zhou, and H. L. Schreyer (1995). "Application of a particle-in-cell method to solid mechanics". In: *Computer Physics Communications* 87.1. Particle Simulation Methods, pp. 236–252. DOI: https://doi.org/10.1016/0010-4655(94)00170-7.

Szabó, L. and A. Kossa (2012). "A new exact integration method for the Drucker–Prager elastoplastic model with linear isotropic hardening". In: *International Journal of Solids and Structures* 49.1, pp. 170–190. DOI: https://doi.org/10.1016/j.ijsolstr.2011.09.021.

Terzaghi, K. (Jan. 1950). "Mechanism of Landslides". In: *Application of Geology to Engineering Practice*. Geological Society of America. DOI: 10.1130/Berkey.1950.83.

Thompson, E. L. and H. E. Huppert (2007). "Granular column collapses: further experimental results". In: *Journal of Fluid Mechanics* 575, p. 177.

Thompson, P. A. and G. S. Grest (1991). "Granular flow: Friction and the dilatancy transition". In: *Phys. Rev. Lett.* 67 (13), pp. 1751–1754. DOI: 10.1103/PhysRevLett.67.1751.

Torquato, S., T. M. Truskett, and P. G. Debenedetti (2000). "Is Random Close Packing of Spheres Well Defined?" In: *Phys. Rev. Lett.* 84 (10), pp. 2064–2067. DOI: 10.1103/PhysRevLett.84.2064.

Tran, Q.-A. and W. Sołowski (2019a). "Generalized Interpolation Material Point Method modelling of large deformation problems including strain-rate effects – Application to penetration and progressive failure problems". In: *Computers and Geotechnics* 106, pp. 249–265. DOI: https://doi.org/10.1016/j.compgeo.2018.10.020.

– (2019b). "Temporal and null-space filter for the material point method". In: *International Journal for Numerical Methods in Engineering* 120.3, pp. 328–360. DOI: https://doi.org/10.1002/nme.6138.

Tsuji, D., M. Otsuki, and H. Katsuragi (2018). "Relaxation Dynamics of a Granular Pile on a Vertically Vibrating Plate". In: *Phys. Rev. Lett.* 120 (12), p. 128001. DOI: https://doi.org/10.1103/PhysRevLett.120.128001.

Tsuji, Y., T. Tanaka, and T. Ishida (1992). "Lagrangian numerical simulation of plug flow of cohesionless particles in a horizontal pipe". In: *Powder Technology* 71.3, pp. 239–250. DOI: https://doi.org/10.1016/0032-5910(92)88030-L.

Uehara, J. S., M. A. Ambroso, R. P. Ojha, and D. J. Durian (2003). "Low-Speed Impact Craters in Loose Granular Media". In: *Phys. Rev. Lett.* 90 (19), p. 194301. DOI: 10.1103/PhysRevLett.90.194301.

Umbanhowar, P. and D. I. Goldman (2010). "Granular impact and the critical packing state". In: *Phys. Rev. E* 82 (1), p. 010301. DOI: 10.1103/PhysRevE.82.010301.

Vardon, P. J., B. Wang, and M. A. Hicks (2017). "Slope Failure Simulations with MPM". In: *Procedia Engineering* 175. Proceedings of the 1st International Conference on the Material Point Method (MPM 2017), pp. 258–264. DOI: https://doi.org/10.1016/j.proeng.2017.01.021.

Vardoulakis, I. (1980). "Shear band inclination and shear modulus of sand in biaxial tests". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 4.2, pp. 103–119. DOI: https://doi.org/10.1002/nag.1610040202.

Varnes, D. J. (1958). "Landslide types and processes". In: *Landslides and engineering practice* 24, pp. 20–47.

– (1978). "Slope movement types and processes". In: *Special Report 176: Landslides: Analysis and Control*. Ed. by R. L. Schuster and R. J. Krizek. Transportation and Road Research Board, National Academy of Science, Washington D.C., pp. 11–33.

Vermeer, P. A. (1990). "The orientation of shear bands in biaxial tests". In: *Géotechnique* 40.2, pp. 223–236. DOI: https://doi.org/10.1680/geot.1990.40.2.223.

Vermeer, P. A. and R. De Borst (1984). "Non-associated plasticity for soils, concrete and rock". In: *HERON, 29 (3), 1984*.

Vet, S. J. de and J. R. de Bruyn (2007). "Shape of impact craters in granular media". In: *Phys. Rev. E* 76 (4), p. 041306. DOI: 10.1103/PhysRevE.76.041306.

Villermaux, E. and B. Bossa (2009). "Single-drop fragmentation determines size distribution of raindrops". In: *Nature Physics* 5.9, pp. 697–702. DOI: 10.1038/nphys1340.

Vivanco, F., S. Rica, and F. Melo (2012). "Dynamical arching in a two dimensional granular flow". In: *Granular Matter* 14.5, pp. 563–576. DOI: https://doi.org/10.1007/s10035-012-0359-7.

Wallstedt, P. C. and J. E. Guilkey (2008). "An evaluation of explicit time integration schemes for use with the generalized interpolation material point method". In: *Journal of Computational Physics* 227.22, pp. 9628–9642. DOI: https://doi.org/10.1016/j.jcp.2008.07.019.

Walsh, A. M., K. E. Holloway, P. Habdas, and J. R. de Bruyn (2003). "Morphology and Scaling of Impact Craters in Granular Media". In: *Phys. Rev. Lett.* 91 (10), p. 104301. DOI: 10.1103/PhysRevLett.91.104301.

Wang, B., M. A. Hicks, and P. J. Vardon (2016a). "Slope failure analysis using the random material point method". In: *Géotechnique Letters* 6.2, pp. 113–118. DOI: 10.1680/jgele.16.00019.

Wang, B., P. J. Vardon, and M. A. Hicks (2016b). "Investigation of retrogressive and progressive slope failure mechanisms using the material point method". In: *Computers and Geotechnics* 78, pp. 88–98. DOI: https://doi.org/10.1016/j.compgeo.2016.04.016.

Wang, B., P. J. Vardon, M. A. Hicks, and Z. Chen (2016c). "Development of an implicit material point method for geotechnical applications". In: *Computers and Geotechnics* 71, pp. 159–167. DOI: https://doi.org/10.1016/j.compgeo.2015.08.008.

Wang, C., B. Hawlader, D. Perret, and K. Soga (2020a). "Effects of geometry and soil properties on type and retrogression of landslides in sensitive clays". In: *Géotechnique* 0.0, pp. 1–15. DOI: 10.1680/jgeot.20.P.046.

Wang, L. et al. (2019). "On the use of domain-based material point methods for problems involving large distortion". In: *Computer Methods in Applied Mechanics and Engineering* 355, pp. 1003–1025. DOI: https://doi.org/10.1016/j.cma.2019.07.011.

Wang, L., X. Zhang, and S. Tinti (2021). "Large deformation dynamic analysis of progressive failure in layered clayey slopes under seismic loading using the particle finite element method". In: *Acta Geotechnica*. DOI: 10.1007/s11440-021-01142-8.

Wang, L., Z. Zheng, Y. Yu, T. Liu, and Z. Zhang (2020b). "Determination of the energetic coefficient of restitution of maize grain based on laboratory experiments and DEM simulations". In: *Powder Technology* 362, pp. 645–658. DOI: https://doi.org/10.1016/j.powtec.2019.12.024.

Wang, X. et al. (July 2020c). "A Massively Parallel and Scalable Multi-GPU Material Point Method". In: *ACM Trans. Graph.* 39.4. DOI: 10.1145/3386569.3392442.

Warnett, J., P Denissenko, P. Thomas, E Kiraci, and M. Williams (2014). "Scalings of axisymmetric granular column collapse". In: *Granular Matter* 16.1, pp. 115–124.

Weinhart, T., C. Labra, S. Luding, and J. Y. Ooi (2016). "Influence of coarse-graining parameters on the analysis of DEM simulations of silo flow". In: *Powder Technology* 293. Particle Modelling with the Discrete Element Method A success story of PARDEM (www.pardem.eu), pp. 138–148. DOI: https://doi.org/10.1016/j.powtec.2015.11.052.

Wieczorek, G. F. (1996). "Landslide triggering mechanisms". In: *Landslides: investigation and mitigation*. Ed. by T. editor. 247. Washington: National academy Press. Chap. 4, pp. 76–90.

Wilson, P., R. Wüchner, and D. Fernando (2021). "Distillation of the material point method cell crossing error leading to a novel quadrature-based C0 remedy". In: *International Journal for Numerical Methods in Engineering* 122.6, pp. 1513–1537. DOI: https://doi.org/10.1002/nme.6588.

Więckowski, Z. (2004). "The material point method in large strain engineering problems". In: *Computer Methods in Applied Mechanics and Engineering* 193.39. The Arbitrary Lagrangian-Eulerian Formulation, pp. 4417–4438. DOI: https://doi.org/10.1016/j.cma.2004.01.035.

Wobbes, E., M. Möller, V. Galavi, and C. Vuik (2019). "Conservative Taylor least squares reconstruction with application to material point methods". In: *International Journal for Numerical Methods in Engineering* 117.3, pp. 271–290. DOI: https://doi.org/10.1002/nme.5956.

Wyser, E., Y. Alkhimenkov, M. Jaboyedoff, and Y. Y. Podladchikov (2020a). "A fast and efficient MATLAB-based MPM solver: fMPMM-solver v1.1". In: *Geoscientific Model Development* 13.12, pp. 6265–6284. DOI: https://doi.org/10.5194/gmd-13-6265-2020.

– (Oct. 2020b). *fMPMM-solver*. Version v1.1. DOI: https://doi.org/10.5281/zenodo.4068585.

Wyser, E., Y. Alkhimenkov, M. Jaboyedoff, and Y. Y. Podladchikov (June 2021). *ep2-3De v1.0*. Version 1.0. DOI: 10.5281/zenodo.4966590.

Wyser, E., Y. Alkhimenkov, M. Jayboyedoff, and Y. Podladchikov (Oct. 2020c). *fMPMM-solver*. Version v1.1. DOI: 10.5281/zenodo.4068585.

Yan, Z., S. K. Wilkinson, E. H. Stitt, and M. Marigo (2015). "Discrete element modelling (DEM) input parameters: understanding their impact on model predictions using statistical analysis". In: *Computational Particle Mechanics* 2.3, pp. 283–299. DOI: https://doi.org/10.1007/s40571-015-0056-5.

Yang, Y., P. Sun, and Z. Chen (2017). "Combined MPM-DEM for Simulating the Interaction Between Solid Elements and Fluid Particles". In: *Communications in Computational Physics* 21.5, 1258–1281. DOI: 10.4208/cicp.OA-2016-0050.

Yerro, A., K. Soga, and J. Bray (2019). "Runout evaluation of Oso landslide with the material point method". In: *Canadian Geotechnical Journal* 56.9, pp. 1304–1317. DOI: 10.1139/cgj-2017-0630.

Ying, C., K. Zhang, Z.-N. Wang, S. Siddiqua, G. M. H. Makeen, and L. Wang (2021). "Analysis of the run-out processes of the Xinlu Village landslide using the generalized interpolation material point method". In: *Landslides* 18.4, pp. 1519–1529. DOI: 10.1007/s10346-020-01581-6.

York II, A. R., D. Sulsky, and H. L. Schreyer (1999). "The material point method for simulation of thin membranes". In: *International Journal for Numerical Methods in Engineering* 44.10, pp. 1429–1456. DOI: https://doi.org/10.1002/(SICI)1097-0207(19990410)44:10<1429::AID-NME536>3.0.CO;2-4.

Yuan, W.-H., K. Liu, W. Zhang, B. Dai, and Y. Wang (2020). "Dynamic modeling of large deformation slope failure using smoothed particle finite element method". In: *Landslides* 17.7, pp. 1591–1603. DOI: 10.1007/s10346-020-01375-w.

Zhang, F., X. Zhang, K. Y. Sze, Y. Lian, and Y. Liu (2017). "Incompressible material point method for free surface flow". In: *Journal of Computational Physics* 330, pp. 92–110. DOI: https://doi.org/10.1016/j.jcp.2016.10.064.

Zhang, Q., M. Gao, R. Zhao, and X. Cheng (2015). "Scaling of liquid-drop impact craters in wet granular media". In: *Phys. Rev. E* 92 (4), p. 042205. DOI: 10.1103/PhysRevE.92.042205.

Zhang, W., Z. hao Zhong, C. Peng, W. hai Yuan, and W. Wu (2021). "GPU-accelerated smoothed particle finite element method for large deformation analysis in geomechanics". In: *Computers and Geotechnics* 129, p. 103856. DOI: https://doi.org/10.1016/j.compgeo.2020.103856.

Zhang, X., Z. Chen, and Y. Liu (2016). *The material point method: a continuum-based particle method for extreme loading cases*. Academic Press.

Zhang, X., E. Oñate, S. A. G. Torres, J. Bleyer, and K. Krabbenhoft (2019). "A unified Lagrangian formulation for solid and fluid dynamics and its possibility for modelling submarine landslides and their consequences". In: *Computer Methods in Applied Mechanics and Engineering* 343, pp. 314–338. DOI: https://doi.org/10.1016/j.cma.2018.07.043.

Zhang, X., S. W. Sloan, and E. Oñate (2018). "Dynamic modelling of retrogressive landslides with emphasis on the role of clay sensitivity". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 42.15, pp. 1806–1822. DOI: https://doi.org/10.1002/nag.2815.

Zhao, J. and W. Liang (2018). "Multiscale Modeling of Large Deformation in Geomechanics: A Coupled MPM-DEM Approach". In: *Proceedings of China-Europe Conference on Geotechnical Engineering*. Ed. by W. Wu and H.-S. Yu. Cham: Springer International Publishing, pp. 449–452.

Zhao, R., Q. Zhang, H. Tjugito, and X. Cheng (2015a). "Granular impact cratering by liquid drops: Understanding raindrop imprints through an analogy to asteroid strikes". In: vol. 112. 2. National Academy of Sciences, pp. 342–347. DOI: 10.1073/pnas.1419271112.

Zhao, S.-C., R. de Jong, and D. van der Meer (2015b). "Raindrop impact on sand: a dynamic explanation of crater morphologies". In: *Soft Matter* 11 (33), pp. 6562–6568. DOI: 10.1039/C5SM00957J.

Zhao, Y. and J. Choo (2020). "Stabilized material point methods for coupled large deformation and fluid flow in porous materials". In: *Computer Methods in Applied Mechanics and Engineering* 362, p. 112742. DOI: https://doi.org/10.1016/j.cma.2019.112742.

Zienkiewicz, O. C., S. Valliappan, and I. P. King (1969). "Elasto-plastic solutions of engineering problems initial stress, finite element approach". In: *International Journal for Numerical Methods in Engineering* 1.1, pp. 75–100. DOI: https://doi.org/10.1002/nme.1620010107.

Zuo, Z., S. Gong, and G. Xie (2020). "Numerical simulation of granular mixing in a rotary drum using a generalized interpolation material point method". In: *Asia-Pacific Journal of Chemical Engineering* 15.2. e2426 APJ-19-0407.R1, e2426. DOI: https://doi.org/10.1002/apj.2426.