University of California
Los Angeles

Simulation of Granular Media with the Material Point Method

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Gergely Klár

2016

Abstract of the Dissertation

Simulation of Granular Media with the Material Point Method

by

Gergely Klár

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2016

Professor Demetri Terzopoulos, Co-chair

Professor Joseph M. Teran, Co-chair

We propose an extension to the Material Point Method for the simulation of granular media. We model the dynamics with an elastoplastic, continuum assumption. The behavior of the granular media is captured by the Drucker-Prager yield criterion that naturally represents the frictional relationship between shear and normal stresses. We develop a stress projection algorithm that is well suited for both explicit and implicit time integration, and uses a non-associative flow rule to ensure volume preservation. Our approach is able to recreate the dynamics of granular media undergoing large deformations, topological changes, and collisions.

The dissertation of Gergely Klár is approved.

Song-Chun Zhu

Stanley Osher

Joseph M. Teran, Committee Co-chair

Demetri Terzopoulos, Committee Co-chair

University of California, Los Angeles

2016

Ez valami.

Table of Contents

1 Introduction **1**

1.1 Overview 1

1.1.1 Sand Simulation in Graphics and Engineering 2

1.1.2 The Material Point Method 3

1.2 Contributions 4

1.3 Dissertation Overview 5

2 Elastoplasticity **7**

2.1 Introduction 7

2.2 Notation 8

2.3 Large Elastoplastic Deformations 9

2.3.1 Multiplicative Decomposition 10

2.3.2 Principal Space 12

2.3.3 Measures of Strain and Stress 13

2.3.4 Governing Equations 16

2.3.5 Elastic Energy 16

2.4 Plasticity 17

2.4.1 The Mohr-Coulomb Yield Criterion 19

2.4.2 The Drucker-Prager Yield Criterion 22

2.4.3 Plastic flow 23

2.5 Work and Work Rate 30

2.5.1 Work Done by Elastic Forces 30

2.5.2 Plastic Flow Rate and Potential 33

2.5.3 The Total Work Rate Density . . . 33
2.5.4 Plastic Dissipation Rate is Non-Negative . . . 36
2.6 Conclusion . . . 37

3 The Material Point Method . . . 38
3.1 Overview . . . 39
3.2 Notation . . . 40
3.3 Push Forward and Pull Back . . . 40
3.4 Discretization . . . 42
3.4.1 Weak Form . . . 42
3.4.2 Temporal Discretization . . . 43
3.4.3 Interpolation Functions . . . 44
3.4.4 Spatial Discretization . . . 45
3.4.5 Stress Approximation . . . 48
3.5 Plasticity . . . 50
3.5.1 Return Mapping . . . 50
3.5.2 Projection . . . 53
3.5.3 Hardening . . . 58
3.6 Algorithm . . . 59
3.6.1 Transfer to Grid . . . 59
3.6.2 Force Computation . . . 61
3.6.3 Collision and Friction . . . 63
3.6.4 Transfer to Particles . . . 65
3.6.5 Particle Update . . . 66
3.6.6 Plasticity and Hardening . . . 67

3.7 Gather and Scatter . . . 67

4 Results . . . 69

4.1 Simulation Setup . . . 69

4.1.1 Friction Angle . . . 69

4.1.2 Pile from Spout . . . 69

4.1.3 Young's Modulus . . . 70

4.1.4 Castle . . . 71

4.1.5 Hourglass . . . 73

4.1.6 Sandbox: Drawing and Raking . . . 75

4.1.7 High Velocity Impact . . . 75

4.1.8 Shoveling . . . 75

4.2 Performance . . . 75

4.3 Rendering . . . 79

5 Conclusion . . . 80

A Further Derivations . . . 81

A.1 Hencky Strain Derivative Lemma . . . 81

A.2 The Relationship Between Kirchhoff Stress and Hencky Strain . . . 83

A.3 Elastic Component of the Total Work Rate Density . . . 84

A.4 Plastic Rate of Deformation . . . 85

A.5 Derivatives of Elasticity and Plasticity . . . 87

B Pseudocode . . . 91

Bibliography . . . 96

List of Figures

2.1 The material and spatial configurations, and the mapping between the two . 10

2.2 Multiplicative decomposition of a neighborhood 11

2.3 A yield surface in two dimensions 18

2.4 Normal and tangential forces 19

2.5 The Drucker-Prager yield surface in three dimensions 23

2.6 Plastic flow direction 26

3.1 Interpolation kernels 45

3.2 Connection between the standard and the updated Lagrangian views 51

3.3 Ray-cone intersection of the Drucker-Prager yield surface 55

3.4 Projection cases 56

3.5 Effect of hardening on the yield surface 58

3.6 Instability of FLIP 59

4.1 Effect of friction angle on piling. 71

4.2 Comparison to real world footage. 72

4.3 Effect of Young's modulus 72

4.4 Sand castle collapse 73

4.5 Hourglass: initial frames 74

4.6 Hourglass: complete simulation 74

4.7 Drawing in sand 76

4.8 Drawing in sand, closeup 76

4.9 Raking sand 77

4.10 Raking sand, closeup 77

4.11 High velocity impact 78

4.12 Shoveling 78

List of Tables

4.1 Simulation setup 70

4.2 Material parameters 70

4.3 Simulation performance 79

Acknowledgments

There is a set $\mathcal{P}$ of people I want to include here, and writing necessitates a *strict ordering*, but such cannot be defined. All I can do is to start and finish with a *maximal element*.

Andre Pradhana became so much more than a labmate over the years. He became a close friend, a brother-in-arms of academia, a comrade in research. Thank you, Andre, for dragging me out of the trenches of dead-end projects, helping me through the barrage of bugs, and having my back against the onslaught of equations.

I want to express my gratitude to my advisors, Professor Joseph Teran and Professor Demetri Terzopoulos. Their support and guidance made it possible for me to arrive at this definitive point in my life.

I also thank my thesis committee members, Professor Stanley Osher and Professor Song-Chun Zhu, for dedicating their time to better my work.

Of my fellow graduate students, I thank Craig Schroeder for his ruthless teaching — I wish I had the chance to work with him more to learn his genius. I thank Ted Gast and Chenfanfu Jiang for their ideas and insights; Matt Wang, Masaki Nakada, Konstantine Tsotsos, and Kresimir Petrinec for their friendship and for making the lab a cheerful place.

I thank Diana Ford, Sharath Gopal, Wenjia Huang, Daniel Ram, Tomer Weiss, and Tao Zhou for their everyday companionship.

I am grateful for my first advisor at BUTE, Hungary, László Szirmay-Kalos, and my Hungarian colleagues László Szécsi, Milán Magdics, Gábor Valasek, and Zalán Szűgyi.

Without the generous support of the Fulbright Science and Technology Program I would have never considered applying to UCLA. I hope the program will help many others to pursue their studies at the top institutions of the US.

I thank Samu and Lili for showing me all the secret emotions of love and joy that only parenthood reveals, and I am grateful to my parents, Magdi and András, whom I might understand a bit better now.

But most of all, I thank my most beloved, strong, and beautiful wife, without whom

none of this would have been possible. She was with me from my first step towards my PhD, and now she stands beside me at the last. I thank her for her hard work and sacrifice, her willingness to take on this adventure with me, and her endless support over the years-long roller coaster ride of grad school. Wherever she happens to be, I find home there.

The dissertation contains material from the paper "Drucker-Prager Elastoplasticity for Sand Animation." by G. Klár, T. Gast, A. Pradhana, C. Fu, C. Schroeder, C. Jiang, and J. Teran in *ACM Trans Graph*, **35**(4), July 2016 and the accompanying technical document "Drucker-Prager Elastoplasticity for Sand Animation: Supplementary Technical Document." by G. Klár, T. Gast, A. Pradhana, C. Fu, C. Schroeder, C. Jiang, and J. Teran in *ACM Trans Graph*, 2016.

Vita

2008	M.S. (Programmer Mathematician), Eötvös Loránd University, Hungary.
2010-2011	Assistant Lecturer, Eötvös Loránd University, Hungary.
2011-2014	Fulbright Science and Technology Program Fellow.
2012, summer	Software Engineer Intern, NVIDIA Corporation, Santa Clara.
2014-2016	Research Assistant, University of California, Los Angeles.
2015, spring	Teaching Assistant, University of California, Los Angeles.
2015, summer	Software Engineer Intern, Google, Los Angeles, California.
2016, spring	Visual Effects R&D Intern, DreamWorks Animation SKG, Glendale, California.

CHAPTER 1

Introduction

1.1 Overview

The simulation of granular material poses an ongoing challenge in both engineering and computer graphics. Granular materials are the focus of soil mechanics in engineering, and they are used for the simulation of sand, rubble, and a range of different materials in computer graphics. Depending on the circumstances, granular materials can exhibit fluid or solid behavior (Jaeger et al., 1996; Bardenhagen et al., 2000). Sand, snow, and gravel flow down in landslides and avalanches, but form piles and are able to support weight as well. The uniqueness and complexity of the phenomena arises from the myriad of grains sliding and colliding with each other. Indeed, the behavior of granular materials is so fundamentally different from both fluids and solids that the *granular state* is frequently considered as a state of matter separate from solid, liquid, or gas (Mehta and Barker, 1994).

In this dissertation, we describe our work on a continuum mechanics based approach to the simulation of dry sand. We use a hyperelastic, finite strain assumption to model the sand dynamics. The flow of sand and the permanent changes in its shape are viewed as elastoplastic behavior, and captured by the Drucker-Prager yield criterion. From the continuum-based considerations, we use the Material Point Method to derive and solve the corresponding discrete equations. We build on the work of Mast (2013) and Mast et al. (2014), where they use the Drucker-Prager yield criterion with MPM for the simulation of landslides and granular column collapse. We extend their work with an implicit elastoplastic model and adopt it for use in computer graphics.

The topic at hand has a distinct scientific beauty to it. Through complicated theory,

spanning multiple disciplines, we arrive at simple and accessible algorithms that anyone with a well founded knowledge in computer science would be able to implement, especially in the case of explicit time integration. But this simplicity does not reduce the versatility of the method, as proven by the range of related publications.

We target a dual audience: researchers and professionals interested in the intricacies of simulating granular materials with MPM, and those who are new to this robust method and want to gain a deeper understanding of the theory and modeling choices behind the implementation.

While the intended application of our work is for computer graphics, this is only reflected in the choice of experiments. Both the discussion on elastoplastic behavior and the derivation of the extended MPM algorithm are valid for engineering applications as well. Nevertheless, the validation of the modeling choices for these cases is beyond the scope of this dissertation.

1.1.1 Sand Simulation in Graphics and Engineering

Physically based simulation has a long history in computer graphics, relative to the age of the field it self at least, dating back to the pioneering work of Terzopoulos et al. for elasticity, plasticity, and fracturing (Terzopoulos et al., 1987; Terzopoulos and Fleischer, 1988a,b).

Continuum approaches have been used in a number of graphics methods for granular materials. Zhu and Bridson (2005) animate sand as a continuum with a modified Particle-In-Cell fluid solver. Narain et al. (2010) use an approximation of the Drucker-Prager yield criterion, and improve on the method of Zhu and Bridson by removing cohesion artifacts associated with incompressibility. These results inspired a number of papers that improve and generalize the previous works. Lenaerts and Dutré (2009) use a Smoothed-Particle Hydrodynamics (SPH) version to couple water with porous granular materials. Alduán and Otaduy (2011) generalize to SPH the unilateral incompressibility developed by Narain et al. Nkulikiyimfura et al. (2012) develop a GPU version of the Zhu and Bridson approach. Chang et al. (2012) use a modified Hooke's law to handle friction between grains. Ihmsen et al. (2013) show how to improve the convergence of the method of Alduán and Otaduy

(2011) and also detail refinement of base simulations by upsampling to millions of grains.

For applications where interactive simulation rates are more important than physical or visual accuracy, methods like cellular automata (Pla-Castells et al., 2006) and height fields have been proposed (Li and Moshell, 1993; Chanclou et al., 1996; Sumner et al., 1999; Onoue and Nishita, 2003; Chen and Wong, 2013).

Many methods are developed by modeling interactions between individual grains or particle idealizations of grains, rather than from a continuum. Miller and Pearce (1989) simulate interactions between particles to model sand, solid and viscous behaviors. Luciani et al. (1995) use a similar approach. Bell et al. (2005) got very impressive results by simulating many sand grains as spherical rigid bodies with friction. Milenkovic (1996) also simulated individual grains to solve for piles of rigid materials via energy minimization/optimization. Mazhar et al. (2015) use Nestov's method to simulate millions of individual grains. Yasuda et al. (2008) use the GPU to get real-time results with rigid grains. Alduán et al. (2009) use an adaptive resolution version of the method by Bell et al. (2005) to improve performance. Macklin et al. (2014) show that the extremely efficient position based dynamics methods can be applied by casting granular interactions as hard constraints.

An alternative approach to the problem is hypoplasticity, that is a more recent theory for the modeling of granular material. We point the interested reader to (Kolymbas, 1999) for an introduction, to (Kolymbas, 2000) for a treatise of related topics, and to (Sikora, 1992) for numerical examples.

1.1.2 The Material Point Method

The Material Point Method is a numerical technique well suited for simulations involving large deformations and topological changes. Proposed by Sulsky et al. in (Sulsky et al., 1994) and (Sulsky et al., 1995), the method combines the advantages of Lagrangian particles and Eulerian grids.

Since its inception, the method has been used in a wide range of engineering problems. Zhou et al. (1999) use it for geotechnical simulation of deformation and creep in landfills.

Więckowski (2004) applies it to large strain industrial problems of silo flow, silo filling, and landslides. Analysis of landslides with MPM is also the focus of Andersen and Andersen (2010). Because of the method's ability to model large displacements, Coetzee et al. (2005) found it advantageous for the simulation of structure supporting anchors. Sulsky et al. (2007) successfully matches the dynamics of ice packs as predicted by FEM models, and they propose a new model for explicit handling of leads in the ice. Fluid and thin structure interactions are investigated by York et al. (1999, 2000), and the method has been even used in biomechanics, where it is employed for the simulation of three dimensional mechanics of a vascularized scaffolds under tension by Guilkey et al. (2006).

The Material Point Method was introduced to the computer graphics community by Stomakhin et al. (2013) where they use the it for the simulation of snow. Hegemann et al. (2013) uses MPM for self collision treatment in ductile fracture. It has been used with success for the simulation of materials undergoing phase change in (Stomakhin et al., 2014), and for the simulation of foams, and viscoelastic fluids by Ram et al. (2015) and by Yue et al. (2015). Jiang (2015) provides an overview of the recent developments of MPM in graphics, and we refer the reader to the course (Jiang et al., 2016) for an introduction.

1.2 Contributions

Our contributions are as follows:

1. We develop an implicit method for the simulation of granular material with the Drucker-Prager yield condition.

2. We adopt the APIC transfer method to the simulation, and demonstrate that it allows for more stable behavior.

3. We present a complete treatise for elastoplastic simulation with the Material Point Method that can be easily extended beyond the Drucker-Prager model and granular materials.

Some of our results have been published in (Klár et al., 2016a) and its supplementary technical document (Klár et al., 2016b).

1.3 Dissertation Overview

Chapter 2 We begin by considering granular media as a continuum. We discuss the principles of continuum mechanics, and the concepts of large elastoplastic deformations, multiplicative decomposition, and yield criteria. Based on these modeling choices, we demonstrate the process of identifying the plastic flow, the quantity that defines the part of the deformation that should be regarded as permanent. As the plastic flow is the key element of the elastoplastic modeling, we provide the details of its derivation that ensures volume preservation and conformance with the second law of thermodynamics.

Chapter 3 Based on the result in continuum, next we transform the results to a discrete setting. In this chapter we provide the derivation and details of the Material Point Method for elastoplastic simulations, and the return mapping algorithm for the Drucker-Prager yield criterion. We adopt a viewpoint similar to that of the Finite Element Method, and present the derivation of the algorithm through this form. The derivation by the weak form sheds light on many important details that appear in context of elastoplasticity, and do not need to be considered for elasticity.

Chapter 4 Next we present results computed by the implementation of our MPM solver for granular media. We tested our system on a range of setups of dry sand. We include examples for parameter tuning, real world footage comparison, highly dynamic scenes, and scenes typical in animation. The material and setup parameters for each example are included here, as well as the simulation performance results.

Appendix A Both Chapter 2 and Chapter 3 rely on a number of derivations that, while essential, would prove to be a distraction from the topic at hand in the main text. These

derivations and proofs are presented here, including results on the Hencky strain and the work done in elastoplasticity. The implicit formulation of the return mapping algorithm requires the derivatives of several components whose computation we detail here.

Appendix B In closing, we include the pseudocode for the implementation of the complete simulation system described in this dissertation. Starting from the time stepping procedure of MPM, we describe the routines for both the explicit and the implicit solver, including the steps carried out during collision handling and application of friction.

CHAPTER 2

Elastoplasticity

2.1 Introduction

We model the behavior of sand using an elastoplastic, continuum assumption. We start the chapter by revisiting the fundamentals of continuum mechanics and elastoplasticity, then present the solution for identifying the change in the plastic deformation.

Elastoplasticity of solids is an easy to understand, ubiquitous phenomenon. A metal sheet or a plastic rod both demonstrate elastic behavior when bent only slightly, and subsequently they both return to their original shape when released. This behavior can be adequately modeled by elasticity. However, when the bending is severe enough that the applied forces are over a threshold, some of the deformation becomes permanent, and the sheet or rod in question will be unable to return to its original configuration. This change in the rest shape is called permanent, plastic, or inelastic change, and its modeling is the subject of elastoplasticity.

While less intuitive, granular media can be described in terms of elastoplasticity as well. When forming piles and supporting load, the material exhibits elastic behavior, and during flowing and sliding the material experiences plastic deformations.

We model the deformations through the large deformation theory, represent plasticity by a multiplicative decomposition, and model energy by hyperelasticity.

The permanent, or plastic, deformations are modeled by the choice of the yield criterion that defines the *yield function*. The choice of the yield function is entirely a modeling one. Based on physical observations and experiments, the yield function captures the limit at which further loading on the material leads to permanent deformation. Different yield

functions have been proposed in the literature to represent the behavior of different classes of material: the von Mises yield criterion is widely used to model metals and other ductile materials (Yu, 2007), while granular materials are modeled well by the Mohr-Coulomb yield criterion (Chen and Saleeb, 1994a,b) and the Drucker-Prager yield criterion (Irgens, 2008) that approximates the former to make it better suited for numerical simulations.

The *flow rule*, or plastic flow, defines what portion of the deformation will be considered permanent at yielding. In other words, while the plastic deformation encodes what should be considered permanent, the plastic flow defines the change in the plastic deformation. A distinction is made between *associative* and *non-associative* flow rules. An associative flow rule is in the direction of the plastic stain rate, that is the gradient of the yield function. As we demonstrate later on, an associative flow rule is the consequence of the principle of maximum plastic dissipation. While this principle guarantees that the system confirms to the second law of thermodynamics, it is not concerned with conservation of volume, which must be added as a separate constraint.

A non-associative flow rule gives more modeling freedom, and provides a means to avoid non-physical volume loss or gain, but extra care has to be taken to avoid adding energy to the system.

In the rest of this chapter we first introduce the notation and key theories used throughout the dissertation. We define the relevant quantities for elastoplastic modeling, but refer the reader to the texts of Gonzalez and Stuart (2008) and Bonet and Wood (2008) for more details on Lagrangian and Eulerian descriptions of the continuum. After the introduction, we focus on plasticity and the derivation of the flow rule that is the key result of this chapter. The description of work and work rate, which plays a significant role in the derivation of the flow rule, is presented after plasticity. We conclude the chapter by summarizing our findings.

2.2 Notation

Throughout this dissertation we use the following notations. Scalar quantities and scalar valued functions are typeset in italic: t, F. Vectors and tensors and functions of them are

written bold: $\mathbf{x}$, $\mathbf{X}$, $\boldsymbol{\tau}$. As a consequence, when referred in scalar components, these vectors and tensors are written as scalar: X_i, τ_{kl}.

As much as possible, we reserve uppercase letters for quantities in the reference configuration, and lowercase letters for the ones in the current configuration, but literature standards take precedence.

Temporal indices are used in superscript (Ω^t), as well as the P and E to refer to plastic and elastic quantities: $\mathbf{C}^P, \mathbf{b}^E$.

We use the convention that subscripts after a comma represent partial derivatives as in $P_{ij,k} = \frac{\partial P_{ij}}{\partial X_k}$. Unless otherwise stated, we use the Einstein summation convention where repeated indices are summed over their ranges: $A_{ki}B_{kj} = \sum_k A^T_{ik}B_{kj} = [\mathbf{A}^T\mathbf{B}]_{ij}$

Occasionally we need to consider if the problem is in two or three dimensions. In such equations we use d for the number of dimensions.

2.3 Large Elastoplastic Deformations

Large deformation theory, also known as finite stain theory, is concerned with deformations where the displacements within the material are on the same scale as the relevant dimensions of the body. Since this is clearly the case for a collapsing wall of sand, this theory represents the right choice for our modeling needs.

We denote the material in its rest configuration as Ω^0, and we will refer to this as the *material* configuration. We use $\mathbf{X} \in \Omega^0$ to address points of the material space. We use the term *material point* and *particle* interchangeably for the points of Ω^0. The material's evolution over time is described by the mapping $\boldsymbol{\phi}$, generally referred to as the flow map or the deformation map. Essentially, the location of point $\mathbf{X}$ at time t is defined by the deformation map as $\mathbf{x} = \boldsymbol{\phi}(\mathbf{X}, t)$. The deformation of the whole body is given by the image of Ω^0 under $\boldsymbol{\phi}$ at time t as Ω^t and it is referred to as the *spatial* or *current* configuration. This relationship between the material and the spatial configuration is illustrated in Figure 2.1.

The deformation gradient $\mathbf{F}(\mathbf{X}, t) = \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{X}}(\mathbf{X}, t)$ describes the local deformation of the

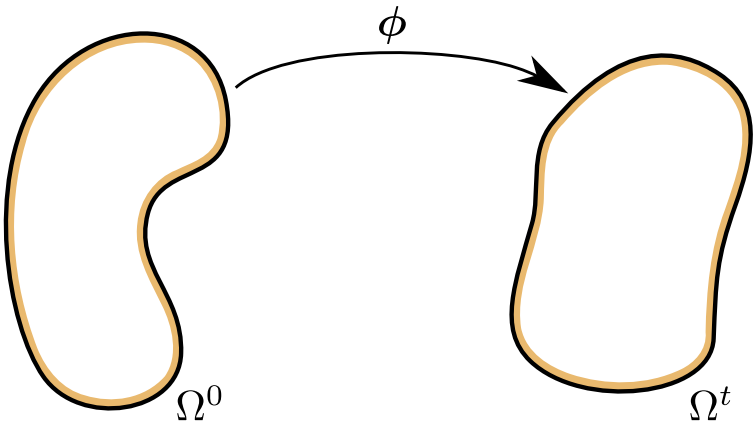

Figure 2.1: The material (left) and spatial configurations (right), and the mapping between the two

neighborhood around $\mathbf{X}$. In other words, the infinitesimal material vector $d\mathbf{X}$ in the neighborhood of $\mathbf{X}$ transforms to the spatial vector $d\mathbf{x}$ at time t as $d\mathbf{x} = \mathbf{F}(\mathbf{X}, t)d\mathbf{X}$. Consequently, the determinant of the deformation gradient $J = \det \mathbf{F}$ describes the local change in volume, and it is the ratio of the infinitesimal volume in Ω^t to the original volume in Ω^0.

In the absence of plastic deformations, if $\mathbf{F}$ is sufficient to define the *elastic potential* or *stored strain energy function* ψ then the material is said to be hyperelastic.

We write the density of the material around a material point $\mathbf{X}$ in the material configuration as $R(\mathbf{X}, t)$, that is, $\mathbf{X}$ is the center of the neighborhood in rest, and the density is evaluated at time t. We define the Lagrangian velocity of a material point $\mathbf{X}$ as

$$\mathbf{V}(\mathbf{X}, t) = \frac{\partial \boldsymbol{\phi}}{\partial t}(\mathbf{X}, t), \tag{2.1}$$

When there is no risk of ambiguity, the $\mathbf{X}$ and t arguments will be omitted in the rest of the dissertation.

2.3.1 Multiplicative Decomposition

To adequately capture large elastoplastic deformations, we use a multiplicative decomposition of the deformation gradient as in (Bonet and Wood, 2008). This decomposition is the basic concept of elastoplasticity. The decomposition splits the deformation gradient $\mathbf{F}$ into

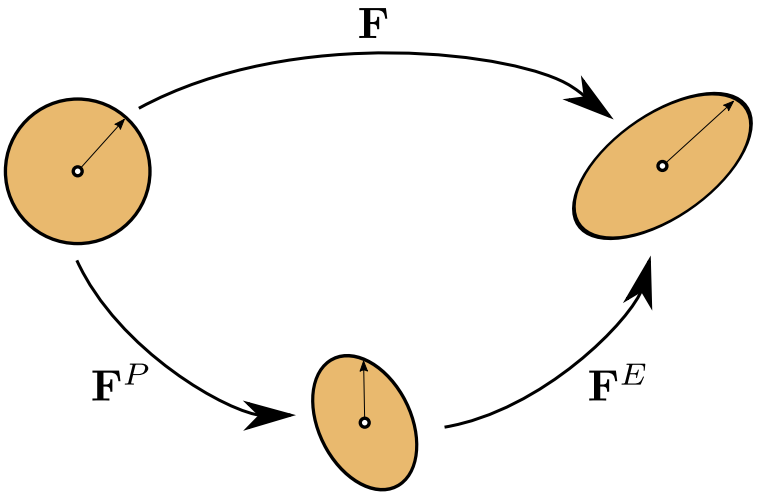

Figure 2.2: Multiplicative decomposition of a neighborhood

an elastic ($\mathbf{F}^E$), and an inelastic, or permanent, component ($\mathbf{F}^P$), such that $\mathbf{F} = \mathbf{F}^E\mathbf{F}^P$ (Figure 2.2). This does not change the definition of the deformation gradient as $\mathbf{F} = \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{X}}$, but it is used to identify the *local* configuration where the neighborhood of each point would settle, if all the external forces were removed from it. In the absence of plasticity, this unloading yields the reference configuration up to translation. At the same time, for elastoplastic materials, this locally unloaded state identifies the new permanent shape of the neighborhood. It is necessary to consider the material in locally unloaded neighborhoods. Simply removing all external forces from the object leads to a complex equilibrium of internal forces, and not to one that is absent of any forces. Finally, it must be noted, that the neighborhoods are considered locally in a conceptual isolation. Due to this isolation, a rotation would not change the intrinsic nature of the deformation of the neighborhood. This invariance to rotation has significant implications, as we will point out later, and makes the modeling of anisotropic elastoplastic materials especially complex.

With elastoplasticity, solving of the material state is a two fold problem. For an elastic material, we want to obtain the deformation map $\boldsymbol{\phi}$ at a given time. But this alone is insufficient to determine the forces acting upon the body. Elastic forces only arise from the elastic part of the deformation gradient ($\mathbf{F}^E$), thus a method is required to determine this division of the deformation gradient.

The solution process is further complicated by the fact, that it is impossible to discern

$\mathbf{F}^E$ and $\mathbf{F}^P$ merely from the current state. Only the time rate of either can be concluded, and the corresponding deformation gradients can be retrieved by integration of its time rate. This dependence on the integral of the rate is called *path-dependence.*

In light of the need to solve for the rate of the various tensors, it might come as confusing that the material is modeled with *rate-independent plasticity.* The key distinction to make is that this rate independence refers to the conditions for *yielding,* and states that the yield conditions and the flow rule are not considered to be functions of any rate variable.

2.3.2 Principal Space

As we address isotropic materials, it is sufficient for several tensor quantities, such as strain and stress, to be expressed in principal space. This greatly simplifies the equations we need to work with as second-order tensors can be represented as vectors, and fourth-order tensors, such as the stiffness tensor mapping strain to stress, as matrices. Note that is a true reduction of order, unlike the Voigt notation that only changes the representation, but keeps the original degrees of freedom.

A further advantage of working in principal space is that constitutive models and yield criteria of isotropic materials can always be written in terms of principal directions. In fact most of them can be expressed solely by the tensor invariants. The invariants of a symmetric, positive-definite second-order tensors $\mathbf{S}$ can be computed in term of its eigenvalues[1] $\lambda_1, \lambda_2, \lambda_3$ (Gonzalez and Stuart, 2008):

$$I_1 = \mathrm{tr}\mathbf{S} = \lambda_1 + \lambda_2 + \lambda_3 \tag{2.2}$$

$$I_2 = \frac{1}{2}\left[(\mathrm{tr}\mathbf{S})^2 - \mathrm{tr}(\mathbf{S}^2)\right] = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_1 \tag{2.3}$$

$$I_3 = \det \mathbf{S} = \lambda_1\lambda_2\lambda_3 \tag{2.4}$$

[1] Our methods are implemented in terms of the singular value decomposition. But since the singular value decomposition and the eigenvalue decomposition of a symmetric, positive-definite matrix are the same up to sign, this generalization poses no danger.

2.3.3 Measures of Strain and Stress

In our derivation of the solution process we employ a number of quantities that describe the state of the material. Arguably, the most important of these are the measures of strain and stress. In this section we give a summary of the ones used in the following chapters, and for completeness we include some that are not used herein, but provide the reader a reference that might prove useful.

2.3.3.1 Strain

Strain measures capture the stretching of a local neighborhood in Ω^0 to Ω^t. There are numerous ways of representing this, resulting in different measures of stress. In the context of elastoplasticity, the *elastic left Cauchy-Green deformation tensor* and the *plastic right Cauchy-Green deformation tensor* are the most significant, and we use them extensively in our derivations. We motivate their definition by first considering their simpler counterparts that ignore plasticity.

The left Cauchy-Green deformation, or strain, tensor is defined as

$$\mathbf{b} = \mathbf{F}\mathbf{F}^T, \tag{2.5}$$

and the right Cauchy-Green deformation, or strain, tensor[2] is

$$\mathbf{C} = \mathbf{F}^T\mathbf{F}. \tag{2.6}$$

Both of these strain tensors are symmetric and positive definite. The motivation for the definition is evident if we consider the polar decomposition of the deformation gradient $\mathbf{F} = \mathbf{R}\mathbf{U} = \mathbf{V}\mathbf{R}$. The deformation gradient describes both the rotation and stretch of a

[2] A useful mnemonic for remembering which one is left and which one is right is to consider the side where $\mathbf{F}$ is without the transpose.

neighborhood, but $\mathbf{b}$ and $\mathbf{C}$ only represent the stretch, since

$$\mathbf{b} = \mathbf{F}\mathbf{F}^T = \mathbf{V}^2, \tag{2.7}$$

$$\mathbf{C} = \mathbf{F}^T\mathbf{F} = \mathbf{U}^2. \tag{2.8}$$

The key difference between the two Cauchy-Green strain tensors is that $\mathbf{b}$ operates on spatial vectors and $\mathbf{C}$ operates on material vectors.

Analogous to these, the *elastic* left Cauchy-Green tensor and the *plastic* right Cauchy-Green tensor are defined as

$$\mathbf{b}^E = \mathbf{F}^E\mathbf{F}^{E^T}, \tag{2.9}$$

$$\mathbf{C}^P = \mathbf{F}^{P^T}\mathbf{F}^P. \tag{2.10}$$

Just as $\mathbf{C}$ and $\mathbf{b}$, $\mathbf{C}^P$ maps from material coordinates to material coordinates, while $\mathbf{b}^E$ maps from spatial coordinates to spatial coordinates.

$\mathbf{b}^E$ and $\mathbf{C}^P$ deserve special attention, because they satisfy the condition that they are invariant under the rotation of the locally unloaded neighborhood. As a result the Kirchhoff stress $\boldsymbol{\tau}$ and the stored energy density function ψ are both independent of these rotations.

The next measure of strain we consider is the Hencky strain, given by

$$\boldsymbol{\epsilon} = \frac{1}{2}\log(\mathbf{b}). \tag{2.11}$$

We define the *elastic* Hencky strain that only takes the elastic deformation into account.

$$\boldsymbol{\epsilon}^E = \frac{1}{2}\log(\mathbf{b}^E) \tag{2.12}$$

2.3.3.2 Stress

The force per unit area, acting on one side of an oriented surface with a unit normal $\mathbf{n}$ is called traction ($\mathbf{t}$) and we can define a second order tensor $\boldsymbol{\sigma}$ (Hashiguchi, 2009), such that

$$\mathbf{t} = \boldsymbol{\sigma}\mathbf{n}. \tag{2.13}$$

Cauchy stress: $\boldsymbol{\sigma}$ in (2.13) is the Cauchy stress if both $\mathbf{n}$ and $\mathbf{t}$ are given in the current configuration. It can be shown from the conservation of angular momentum that the Cauchy stress tensor is symmetric.

First Piola-Kirchhoff stress $\mathbf{P}$ is the non-symmetric second order tensor that maps surface normals given in the reference space to traction in the current configuration. Its relation to the Cauchy stress is given as

$$\mathbf{P} = J\boldsymbol{\sigma}\mathbf{F}^{-T}. \tag{2.14}$$

Second Piola-Kirchhoff stress $\mathbf{S}$ is symmetric, and maps from reference space normals to reference space traction. Its relation to the Cauchy stress is given as

$$\mathbf{S} = J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T}. \tag{2.15}$$

Kirchhoff stress $\boldsymbol{\tau}$, is defined as the work conjugate of the rate of deformation tensor (Bonet and Wood, 2008), and relates to the Cauchy stress and to the first Piola-Kirchhoff stress by the following:

$$\boldsymbol{\tau} = J\boldsymbol{\sigma} = \mathbf{P}\mathbf{F}^{T} \tag{2.16}$$

Since it equals the Cauchy stress scaled by the change in volume, it is also a symmetric tensor, defined over the current configuration.

2.3.4 Governing Equations

The governing equation of our problem are the conservation of mass and the conservation of linear momentum. We express these in the material configuration, in a Lagrangian view. Conservation of mass, in the Lagrangian view is

$$R(\mathbf{X}, t)J(\mathbf{X}, t) = R(\mathbf{X}, 0). \tag{2.17}$$

Conservation of linear momentum results in force density balance

$$R(\mathbf{X}, 0)\frac{\partial \mathbf{V}}{\partial t}(\mathbf{X}, t) = \nabla^{\mathbf{X}} \cdot \mathbf{P}(\mathbf{X}, t) + R(\mathbf{X}, 0)\mathbf{g}. \tag{2.18}$$

Note that this equation has units of force density, but is otherwise just Newton's second law generalized to the continuum.

2.3.5 Elastic Energy

Here we discuss the notion of energy in the context of elastoplasticity. It is important to carefully consider the effect the plastic flow will have on the rate of change of total energy. The plastic flow must obey the second law of thermodynamics and as a consequence it must not increase the rate of change of total energy. Using a hyperelastic constitutive model for the elastic stress implies that the rate of change of total energy in the absence of plasticity will be zero. Again, we take a Lagrangian view of the continuum for these derivations.

The total potential energy in the absence of plasticity is

$$E_P(t) = \int_{\Omega^0} \psi\left(\mathbf{F}(\mathbf{X}, t)\right) d\mathbf{X}. \tag{2.19}$$

By the multiplicative decomposition $\mathbf{F} = \mathbf{F}^E\mathbf{F}^P$. $\mathbf{F}^P$ identifies the inelastic portion of the deformation, therefore it does not contribute to the potential energy. Thus, the potential

energy only depends on $\mathbf{F}^E$, giving

$$E_P(t) = \int_{\Omega^0} \psi \left(\mathbf{F}^E(\mathbf{X}, t) \right) d\mathbf{X} \tag{2.20}$$

For hyperelastic materials the first Piola-Kirchhoff stress is defined as the gradient of the energy density function, but for elastoplasticity the energy loss through plasticity must be taken into account.

The energy density function that only takes $\mathbf{F}^E$ into account is

$$\hat{\psi}(\mathbf{F}) = \psi(\mathbf{F}\mathbf{F}^{P^{-1}}), \tag{2.21}$$

and with that we can write the first Piola-Kirchhoff stress as

$$\mathbf{P}(\mathbf{F}^E, \mathbf{F}^P) = \frac{\partial \hat{\psi}}{\partial \mathbf{F}} = \frac{\partial \psi}{\partial \mathbf{F}^E}(\mathbf{F}^E) : \frac{\partial \mathbf{F}^E}{\partial \mathbf{F}} = \frac{\partial \psi}{\partial \mathbf{F}^E}(\mathbf{F}^E)\mathbf{F}^{P-T}. \tag{2.22}$$

2.4 Plasticity

The maximum attainable stresses of a material are modeled by the *yield condition*. The yield condition defines the points where the material starts *yielding* under load, that is, the points at which the material is unable to exert any more resistance and any further deformation becomes permanent.

The yield condition divides the stress space into an allowable and a non-physical region. The yield function F is a scalar valued function of stress, such that $F \leq 0$ in the allowable region, and $F > 0$ in the non-physical. Figure 2.3 illustrates these regions in two dimensions for a Drucker-Prager yield function in principal stress space. The line where $\tau_1 = \tau_2$ is called the hydrostatic axis, and any stress can be decomposed to a component parallel and a component perpendicular to it. The stress component perpendicular to the hydrostatic axis is the *deviatoric stress*. Stress values in the first quadrant correspond to states in tension, and values in the third quadrant are in compression.

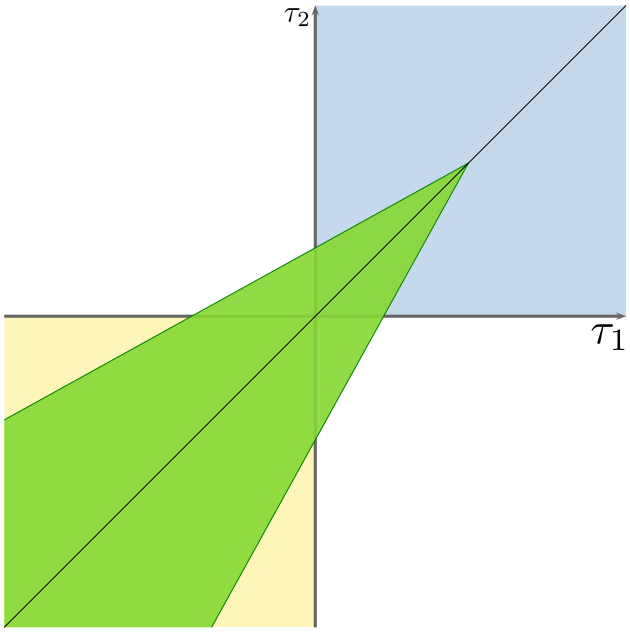

Figure 2.3: A two dimensional Drucker-Prager yield surface with cohesion. The green area is the admissible region of the stress space and its boundary is the yield surface. The material is in tension in all directions in the blue region and it is in compression in the yellow region. This particular yield surface represents a material with cohesion as it extends to the first quadrant. The tip of the Drucker-Prager cone is at the origin for cohesionless materials such as dry sand. The black diagonal line represents the hydrostatic axis.

The material is unable to support further stresses as the stress configuration reaches the yield surface, and permanent, plastic deformations will occur. This deformation prevents non-physical stresses.

Based on F and $\dot{F}$, for $\boldsymbol{\tau}$ stress there are three cases that need to be considered:

- *Case I.* If $F < 0$, then $\boldsymbol{\tau}$ is inside the yield surface, and the deformation is purely elastic.
- *Case II.* If $F = 0$ and $\dot{F} < 0$, then $\boldsymbol{\tau}$ is on the yield surface, moving inward; and the deformation is still purely elastic.
- *Case III.* If $F = 0$ and $\dot{F} = 0$, then $\boldsymbol{\tau}$ is on the yield surface, staying on it; and the deformation is plastic.

Note that the case of $F > 0$ and the case of $\dot{F} > 0$ while $F = 0$ are non-physical, therefore not permitted states.

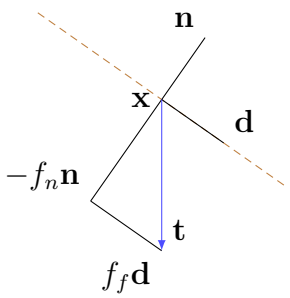

Figure 2.4: Normal and tangential forces

For the simulation of dry sand the Drucker-Prager yield condition is a common choice in engineering, as it captures well the friction between the grains, that leads to sand's characteristic pile formation. The Drucker-Prager model is an approximation of the Mohr-Coulomb model, but it is better suited for numerical simulations.

Before moving on to the derivation of plastic flow, we present the derivation of the Mohr-Coulomb model from Coulomb friction, then address the differences between the Drucker-Prager and the Mohr-Coulomb yield criterion.

2.4.1 The Mohr-Coulomb Yield Criterion

Consider a Coulomb friction interaction between two grains in contact. If $\tilde{\alpha}$ is the coefficient of friction, then the frictional force f_f can only be as large as the coefficient of friction times the normal force f_n: $f_f \leq \tilde{\alpha} f_n$. The Mohr-Coulomb model generalizes this to a continuum (Chen and Saleeb, 1994a,b). At any point in the continuum body, the Cauchy stress $\boldsymbol{\sigma}$ expresses the local mechanical interactions in the material. Specifically, at point $\mathbf{x}$, $\boldsymbol{\sigma}(\mathbf{x})$ relates the force per area, or traction, $\mathbf{t}$ that material on one side of an imaginary plane with normal $\mathbf{n}$ exerts on material on the other side, as $\mathbf{t} = \boldsymbol{\sigma}(\mathbf{x})\mathbf{n}$. If we consider this interaction to be from friction, we can use the Coulomb model to relate the frictional force (per area) $f_f = \mathbf{d}^T\mathbf{t}$ to the normal force (per area) $f_n = -\mathbf{n}^T\mathbf{t}$ as $\mathbf{d}^T\mathbf{t} \leq -\tilde{\alpha}\mathbf{n}^T\mathbf{t}$. Here, $\mathbf{d}$ is the normalized projection of the traction $\mathbf{t}$ into the plane orthogonal to $\mathbf{n}$. This relation is illustrated in Figure 2.4. In terms of $\boldsymbol{\sigma}$, this is expressed as $\mathbf{d}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n} \leq -\tilde{\alpha}\mathbf{n}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n}$.

The frictional force (per area) $f_f = \mathbf{d}^T\mathbf{t}$ is often referred to as the shear stress (at $\mathbf{x}$, in

direction $\mathbf{n}$) and the normal force (per area) is often referred to as the normal stress (at $\mathbf{x}$, in direction $\mathbf{n}$). If we consider all shear stresses to arise from friction, then we get a notion of states of stress consistent with the Coulomb model of frictional interaction. That is, we consider the stress field $\boldsymbol{\sigma}(\mathbf{x})$ as admissible (or consistent with the Coulomb model) if

$$\mathbf{d}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n} \leq -\tilde{\alpha}\mathbf{n}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n} \tag{2.23}$$

for all $\mathbf{x}$ in the material and for arbitrary directions $\mathbf{d}$ and $\mathbf{n}$ with $\mathbf{d}^T\mathbf{n} = 0$.

When the normal stress $\mathbf{n}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n}$ is positive, the material on one side of the imaginary plane is pulling on the material on the other side. This does not arise from a contact/frictional interaction and is a cohesive interaction. Note that Inequality (2.23) implies that in the presence of a positive normal stress, the shear stress would have to be zero. In fact, it can be shown that it is not possible to be consistent with Inequality (2.23) (for all $\mathbf{d}$ and $\mathbf{n}$) with a positive normal stress, and thus cohesion is not possible with this model.

2.4.1.1 Reformulation of Stress Admissibility

Consider the two dimensional case and states of stress consistent with Inequality (2.23). In this case, given normal $\mathbf{n}$, there are only two directions $\mathbf{d}$ orthogonal to it, namely $\mathbf{d} = \pm\mathbf{R}\mathbf{n}$ where

$$\mathbf{R} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \tag{2.24}$$

In this case, satisfaction of Inequality (2.23) is achieved when

$$\pm\mathbf{n}^T\mathbf{R}\boldsymbol{\sigma}(\mathbf{x})\mathbf{n} + \tilde{\alpha}\mathbf{n}^T\boldsymbol{\sigma}(\mathbf{x})\mathbf{n} \leq 0 \tag{2.25}$$

for all directions $\mathbf{n}$. Since the Cauchy stress must be symmetric (by conservation of angular momentum), it has an eigendecomposition

$$\boldsymbol{\sigma} = \mathbf{Q}\mathbf{D}\mathbf{Q}^T = \mathbf{Q}\begin{pmatrix} s_1 & 0 \\ 0 & s_2 \end{pmatrix}\mathbf{Q}^T \tag{2.26}$$

where $\mathbf{Q}$ is a rotation matrix. Rewriting Inequality (2.25) in terms of the eigendecomposition gives

$$\pm\mathbf{n}^T\mathbf{R}\mathbf{Q}\mathbf{D}\mathbf{Q}^T\mathbf{n} + \tilde{\alpha}\mathbf{n}^T\mathbf{Q}\mathbf{D}\mathbf{Q}^T\mathbf{n} \leq 0 \tag{2.27}$$

and since $\mathbf{R}$ and $\mathbf{Q}$ commute (2D rotations commute), satisfaction of Inequality (2.25) is the same as

$$\tilde{\mathbf{n}}^T\left(\pm\mathbf{R}\mathbf{D} + \tilde{\alpha}\mathbf{D}\right)\tilde{\mathbf{n}} \leq 0 \tag{2.28}$$

where $\tilde{\mathbf{n}} = \mathbf{Q}\mathbf{n}$ and

$$\mathbf{R}\mathbf{D} = \begin{pmatrix} 0 & -s_2 \\ s_1 & 0 \end{pmatrix}. \tag{2.29}$$

Since Inequality (2.28) must be true for all $\tilde{\mathbf{n}}$ and choice of sign, it is equivalent to require that the maximum of

$$F(\tilde{\mathbf{n}}, h) = \tilde{\mathbf{n}}^T\left(h\mathbf{R}\mathbf{D} + \tilde{\alpha}\mathbf{D}\right)\tilde{\mathbf{n}} \tag{2.30}$$

subject to $\|\tilde{\mathbf{n}}\|^2 = 1$ and $h^2 = 1$, is less than 0. Using the method of Lagrange multipliers it can be shown that this maximum is given by

$$\frac{s_1 + s_2}{2}\tilde{\alpha} + \frac{|s_1 - s_2|}{2}\sqrt{1 + \tilde{\alpha}^2}. \tag{2.31}$$

Dividing by $\frac{\sqrt{1+\tilde{\alpha}^2}}{\sqrt{2}}$ we obtain that

$$\begin{aligned} (s_1 + s_2)\frac{\tilde{\alpha}}{\sqrt{2}\sqrt{1+\tilde{\alpha}^2}} + \frac{|s_1 - s_2|}{\sqrt{2}} &\leq 0 \\ \operatorname{tr}(\boldsymbol{\sigma}(\mathbf{x}))\alpha + \left\|\boldsymbol{\sigma}(\mathbf{x}) - \frac{\operatorname{tr}(\boldsymbol{\sigma}(\mathbf{x}))}{2}\mathbf{I}\right\|_F &\leq 0 \end{aligned} \tag{2.32}$$

Where $\|\cdot\|_F$ is the Frobenius norm and $\alpha = \frac{\tilde{\alpha}}{\sqrt{2}\sqrt{1+\tilde{\alpha}^2}}$.

If we solve the analogous maximization problem in three dimensions we obtain the Mohr-Coulomb yield surface (Mast, 2013).

2.4.2 The Drucker-Prager Yield Criterion

There is a simple generalization of Inequality (2.32) that works for both two and three dimensions given by

$$\operatorname{tr}(\boldsymbol{\sigma}(\mathbf{x}))\alpha + \left\|\boldsymbol{\sigma}(\mathbf{x}) - \frac{\operatorname{tr}(\boldsymbol{\sigma}(\mathbf{x}))}{d}\mathbf{I}\right\|_F \leq 0. \tag{2.33}$$

where d is the number of space dimensions. The Drucker-Prager model uses Inequality (2.33) in both two and three dimensions, because it is easier to work with than the Mohr-Coulomb model in 3D and it is a decent approximation of Mohr-Coulomb in that case (Irgens, 2008).

In summary, the Drucker-Prager model for the stress field $\boldsymbol{\sigma}$ requires that

$$F(\boldsymbol{\sigma}(\mathbf{x})) \leq 0 \tag{2.34}$$

for all points $\mathbf{x}$ in the domain occupied by the material, where $F(\boldsymbol{\sigma}) = \operatorname{tr}(\boldsymbol{\sigma})\alpha + \|\boldsymbol{\sigma} - \frac{\operatorname{tr}(\boldsymbol{\sigma})}{d}\mathbf{I}\|_F$ and d is the number of space dimensions. Note that this function is actually defined in terms of the eigenvalues of $\boldsymbol{\sigma}$ as $F(\boldsymbol{\sigma}) = \operatorname{tr}(\mathbf{D})\alpha + \|\mathbf{D} - \frac{\operatorname{tr}(\mathbf{D})}{d}\mathbf{I}\|_F$.

It is often mathematically convenient to express the Drucker-Prager yield condition in terms of the Kirchhoff stress. We will find this useful when deriving and analyzing properties of the plastic flow. Expressing the Drucker-Prager condition in terms of $\boldsymbol{\tau}$ is simply the requirement that $F(\boldsymbol{\tau}(\mathbf{x})) \leq 0$ for all $\mathbf{x}$ in the domain.

We can think of the condition $F(\boldsymbol{\tau}) = \operatorname{tr}(\boldsymbol{\tau})\alpha + \|\boldsymbol{\tau} - \frac{\operatorname{tr}(\boldsymbol{\tau})}{d}\mathbf{I}\|_F \leq 0$ as defining a feasible region in stress space. Since the constraint can be evaluated as a function of the principal stresses, we can visualize it as the cone

$$(\tau_1 + \tau_2)\alpha + \frac{|\tau_1 - \tau_2|}{\sqrt{2}} \leq 0 \tag{2.35}$$

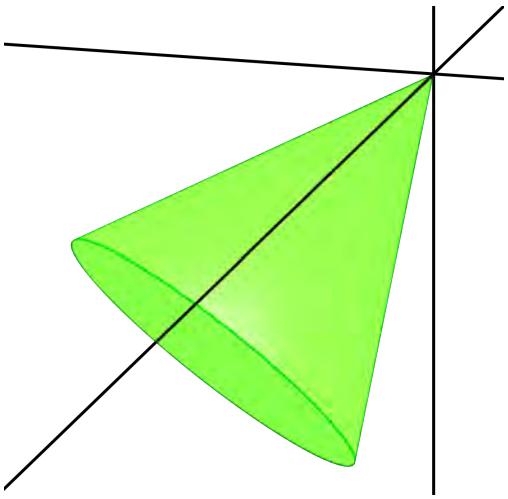

Figure 2.5: The Drucker-Prager yield surface in three dimensions

for 2D problems, or the cone

$$(\tau_1 + \tau_2 + \tau_3)\alpha + \sqrt{\sum_{j=1}^{3}\left(\tau_j - \sum_{i=1}^{3}\frac{\tau_i}{3}\right)^2} \leq 0 \tag{2.36}$$

for 3D problems (Figure 2.5). The plastic flow will be chosen as a means of satisfying this constraint. When the stress is in the feasible region, there is no plastic flow. However, when the stress reaches the boundary of this region, the plastic flow will be chosen in a manner that prevents the stress from leaving the feasible region. For this reason, the boundary of the feasible region is called the yield surface, since plastic "yield" occurs when the state of stress reaches it.

2.4.3 Plastic flow

While F is a function of the stress, and specifically we write the Drucker-Prager yield condition in terms of the Kirchhoff stress, for the following $\mathbf{b}^E$ is the natural choice addressing the yield conditions. Since the Kirchhoff stress $\boldsymbol{\tau}$ is a function of the elastic Hencky strain $\boldsymbol{\epsilon}^E$, and that is in turn a function of $\mathbf{b}^E$, the choice of considering F as a function of $\mathbf{b}^E$ creates no contradiction.

In the presence of plastic deformations, the evolution of $\mathbf{b}^E$, namely $\dot{\mathbf{b}}^E$, is dictated by

$\dot{F}$. We detail this connection between $\dot{\mathbf{b}}^E$ and $\dot{F}$ in the rest of the section.

The plastic flow needs to satisfy the following conditions:

1. Yield conditions
2. Principle of maximum plastic dissipation
3. Second law of thermodynamics
4. Volume preservation

The yield conditions are of primary concern when determining the plastic flow, while the other conditions will be used to arrive at a concrete solution.

For the purpose of analysis and constitutive modeling, we will base our equations on the elastic left Cauchy-Green tensor ($\mathbf{b}^E$), and the plastic right Cauchy-Green tensor ($\mathbf{C}^P$), as defined in (2.9) and (2.10). The connection between the two is expressed by the following equation:

$$\begin{aligned} \mathbf{b}^E &= \mathbf{F}^E\mathbf{F}^{E^T} \\ &= \mathbf{F}\mathbf{F}^{P^{-1}}\mathbf{F}^{P^{-T}}\mathbf{F}^T \\ &= \mathbf{F}\mathbf{C}^{P^{-1}}\mathbf{F}^T \end{aligned} \tag{2.37}$$

Using this identity, we can express the evolution of $\mathbf{b}^E$ as

$$\begin{aligned} \frac{D\mathbf{b}^E}{Dt} &= \frac{D\mathbf{F}}{Dt}\mathbf{C}^{P^{-1}}\mathbf{F}^T + \mathbf{F}\frac{D\mathbf{C}^{P^{-1}}}{Dt}\mathbf{F}^T + \mathbf{F}\mathbf{C}^{P^{-1}}\frac{D\mathbf{F}^T}{Dt} \\ &= (\nabla\mathbf{v})\mathbf{F}\mathbf{C}^{P^{-1}}\mathbf{F}^T + \mathbf{F}\frac{D\mathbf{C}^{P^{-1}}}{Dt}\mathbf{F}^T + \mathbf{F}\mathbf{C}^{P^{-1}}((\nabla\mathbf{v})\mathbf{F})^T \\ &= (\nabla\mathbf{v})\mathbf{b}^E + \mathbf{F}\frac{D\mathbf{C}^{P^{-1}}}{Dt}\mathbf{F}^T + \mathbf{b}^E(\nabla\mathbf{v})^T, \end{aligned} \tag{2.38}$$

where we used the fact that the deformation gradient evolves as

$$\frac{D\mathbf{F}}{Dt} = (\nabla\mathbf{v})\mathbf{F}. \tag{2.39}$$

The term $\mathbf{F}\frac{D\mathbf{C}^{P^{-1}}}{Dt}\mathbf{F}^T$ is the Lie derivative of $\mathbf{b}^E$ with respect to $\mathbf{v}$. We denote this by $\mathcal{L}_\mathbf{v}\mathbf{b}^E$. The Lie derivative of $\mathbf{b}^E$ is its rate of change independent of the deformation in the flow. $\mathcal{L}_\mathbf{v}\mathbf{b}^E$ defines the plastic flow, therefore we treat the Lie derivative as the free variable of our equation.

When the stress is within the feasible region ($F < 0$), the deformation is purely elastic and the plastic deformation stays unchanged. This implies that no plastic flow takes place, therefore $\mathcal{L}_\mathbf{v}\mathbf{b}^E = \mathbf{0}$. From (2.38) it follows that in this case

$$\frac{D\mathbf{b}^E}{Dt} = (\nabla\mathbf{v})\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T, \tag{2.40}$$

meaning that the evolution of $\mathbf{b}^E$ is defined by $\nabla\mathbf{v}$.

On the other hand, when the stress is on the yield surface ($F = 0$), we chose $\mathcal{L}_\mathbf{v}\mathbf{b}^E$ to guarantee that $\dot{F} \leq 0$ is satisfied, thus preventing any future elastic stresses attaining values outside the feasible region.

When $F = 0$ then the problem to be solved can be view as a constrained optimization of a scalar valued matrix function, that is the yield function. At the yield surface, any direction perpendicular to the surface normal will guarantee that the solution remains on the yield surface. This is illustrated in Figure 2.6. The additional constrains are used to define the appropriate direction.

To better capture the possible choices of $\mathcal{L}_\mathbf{v}\mathbf{b}^E$, we denote it as $\mathcal{L}_\mathbf{v}\mathbf{b}^E = -\gamma\mathbf{L}$, where $\mathbf{L}$ is an arbitrary matrix. With this view, $\mathbf{L}$ is the direction of the Lie derivative and γ is its magnitude. Given a direction $\mathbf{L}$, we will choose magnitude γ to guarantee that $\dot{F} \leq 0$.

In general, the yield function is considered to be a function of stress $\boldsymbol{\tau}$. At the same time the stress itself is a function of the elastic state $\mathbf{b}^E$, that in turn is a function of time. With this in mind, we can write the yield function as $F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right)$, and regard it as a function of time alone. By the chain rule, we get the connection between the evolution of F and the

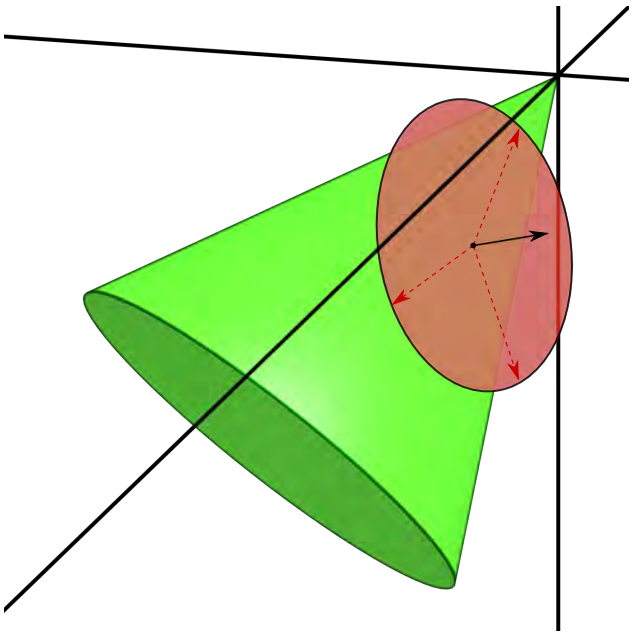

Figure 2.6: Finding the plastic flow direction. The red circle is the tangent plane around the point on the yield surface marked by the black dot. The black arrow is the normal at that point, and red arrows illustrate possible choices for the plastic flow direction. As the tangent plane is perpendicular to the normal, any direction chosen from it would keep the stress on the yield surface.

plastic flow.

$$\begin{aligned}\dot{F}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) &= \frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \frac{D\mathbf{b}^E}{Dt}(t) \\ &= \frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \left((\nabla \mathbf{v})\mathbf{b}^E + \mathcal{L}_{\mathbf{v}}\mathbf{b}^E + \mathbf{b}^E(\nabla \mathbf{v})^T\right)\end{aligned} \tag{2.41}$$

The : operator denotes a generalized dot product to express the chain rule when differentiating the composition of scalar and matrix valued functions of matrix argument.

The material derivative $\frac{D}{Dt}$ appears since we are considering the evolution of F in the Lagrangian frame, that is, how F evolves over time for one particle of the continuum.

Recall that in the absence of plasticity $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \mathbf{0}$, therefore the rate of change of F for purely elastic behavior can be defined as β:

$$\beta = \frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \left((\nabla \mathbf{v})\mathbf{b}^E + \mathbf{b}^E(\nabla \mathbf{v})^T\right) \tag{2.42}$$

Combining this with the convention that $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = -\gamma\mathbf{L}$, we write (2.41) as

$$\dot{F}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = \beta - \gamma\frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \mathbf{L} \tag{2.43}$$

For determining $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E$ the three cases listed in Section 2.4 must be considered.

In Case I, $F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) < 0$, therefore no plastic deformation occurs, so as before we get $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \mathbf{0}$.

In Case II, the stress state is on the yield surface, $F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0$, and $\dot{F} < 0$. Here the purely elastic evolution is sufficient to take the stress back to the inside with $\beta < 0$, if $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \mathbf{0}$. Therefore this case is absent of plastic deformations again.

In Case III, $F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0$ and $\dot{F} = 0$, the stress is on the yield surface again, but also staying on the surface. This is the only case when plastic flow may occur. The purely elastic evolution can either keep the stress on the yield surface with $\beta = 0$, or take the stress into the unfeasible region when $\beta > 0$. In the former case, no plastic deformation should occur, giving $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \mathbf{0}$ once again. In the latter case we chose γ to balance β, so that $\dot{F}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0$, that is

$$\gamma = \frac{\beta}{\frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \mathbf{L}}. \tag{2.44}$$

These cases can be summarized to define the plastic flow:

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \begin{cases} \mathbf{0}, & \text{if } F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) < 0 \text{ or } \left[F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0 \text{ and } \beta \leq 0\right] \\ -\gamma\mathbf{L}, & \text{if } F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0 \text{ and } \beta > 0 \end{cases} \tag{2.45}$$

Notice how the sign of β captures the difference between Cases II. and III. When $\beta \leq 0$ then no plastic flow is required, and when $\beta > 0$ then the choice of γ guarantees the yield conditions.

When plastic flow occurs, (2.45) defines only the magnitude of $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E$, but not its direction. To determine the direction, and in turn close the system of equations, we employ the

aforementioned constraints for volume preservation, and enforce the second law of thermodynamics.

If $\frac{\partial F}{\partial \boldsymbol{\tau}}\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E(t)) : \mathbf{L} \neq 0$, it is always possible to choose γ according to (2.44). Thus, for a given value of $\mathbf{b}^E$, there are infinitely many choices of $\mathbf{L}$ that will ensure that the stress remains in the feasible region. However, by the second law of thermodynamics, the plastic flow must not decrease the entropy of the system. More specifically, it must not instantaneously increase the rate of change of the total energy (Gonzalez and Stuart, 2008). Notably, the rate of change of total energy is zero in absence of plasticity. Physically, plastic deformations should only cause dissipation of energy, therefore the total energy of the system should decrease due to plastic flow. With these considerations in mind, we refine the choice of $\mathcal{L}_{\mathbf{v}}\mathbf{b}^E$

To avoid distraction from the key ideas, details of the following derivations have been deferred to Section 2.5. At each step we refer to the corresponding section.

The total energy in the system is the sum of the elastic potential energy and the kinetic energy:

$$E(t) = E_P(t) + E_K(t), \tag{2.46}$$

and its change over time satisfies

$$E(t+\Delta t) - E(t) = W^{\mathbf{t}}(t, \Delta t) - \int_t^{t+\Delta t} \int_{\Omega^0} \dot{w}^P(\mathbf{X}, t) d\mathbf{X} ds, \tag{2.47}$$

which we show in Section 2.5 with (2.83).

Here $W^{\mathbf{t}}(t, \Delta t)$ is the work done by external traction $\mathbf{t}$ boundary conditions and $\dot{w}^P(\mathbf{X}, t)$ is *rate of plastic dissipation* from (2.79). The rate of plastic dissipation is given in term of the Kirchhoff stress and the *plastic rate of deformation* as

$$\dot{w}^P = \boldsymbol{\tau} : \mathbf{l}^P, \tag{2.48}$$

where

$$\mathbf{l}^P = -\frac{1}{2}\mathcal{L}_{\mathbf{v}}\mathbf{b}^E\mathbf{b}^{E^{-1}}. \tag{2.49}$$

In absence of plasticity the change in energy equals to the work done by the boundary conditions, or the energy is exactly conserved in the lack of boundary forces, as discussed in Section 2.5.1. In case of plasticity, theoretically the total energy may go up or down from the work done by the mechanical stress. The integral term of (2.47) quantifies this. The increase of total energy would be in violation of the second law of thermodynamics, therefore the direction of the plastic flow must be chosen so that it ensures a non-negative $\dot{w}^P$.

The principle of maximum plastic dissipation (Hill, 1948, 1998; Bonet and Wood, 2008) seeks to design the plastic flow in a way that maximizes $\dot{w}^P$ to respect this concern. This leads to a so called *associative flow rule*, where

$$\mathbf{l}^P = \gamma\frac{\partial F}{\partial \boldsymbol{\tau}} \tag{2.50}$$

or equivalently

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = -2\gamma\frac{\partial F}{\partial \boldsymbol{\tau}}\mathbf{b}^E. \tag{2.51}$$

That is, with an associative flow rule, the plastic flow is in the direction of the gradient of the yield function.

However, the choice of matrix $\mathbf{L}$ will affect the volume change in the plastic flow. Specifically, it can be shown that if $\mathrm{tr}\mathbf{L} = 0$, then the plastic flow will be volume preserving with $J^P = \det\mathbf{F}^P = 1$. Since the elastic potential seeks to preserve $\det\mathbf{F}^E = 1$ by design, a volume preserving plastic flow will produce an overall deformation that tends to preserve volume. However, without $\mathrm{tr}\mathbf{L} = 0$ there is a potential for excessive volume loss or gain in the model. Indeed, simply using the associative flow rule from (2.51) will tend to cause excessive volume gain during sheering, as noted in (Mast, 2013).

This can be alleviated by the use of a *non-associative flow rule*, where the plastic flow is in the direction of the deviatoric portion of the yield function's gradient. Defining this

direction as

$$\mathbf{G} = \operatorname{dev}\left(\frac{\partial F}{\partial \boldsymbol{\tau}}\right) = \frac{\partial F}{\partial \boldsymbol{\tau}} - \frac{1}{d}\operatorname{tr}\left(\frac{\partial F}{\partial \boldsymbol{\tau}}\right)\mathbf{I}, \tag{2.52}$$

we can write the plastic flow

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = -\gamma \mathbf{G}\mathbf{b}^E. \tag{2.53}$$

We demonstrate in Section 2.5.4 that the flow rule defined above guarantees that $\dot{w}^P$ is non-negative and thus satisfies the second law of thermodynamics.

With these refinements, we arrive at the flow rule, that adheres to the yield conditions, obeys the second law of thermodynamics, and facilitates volume preservation:

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \begin{cases} \mathbf{0}, & \text{if } F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) < 0 \text{ or } \left[F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0 \text{ and } \beta \leq 0\right] \\ -\gamma \mathbf{G}\mathbf{b}^E, & \text{if } F\left(\boldsymbol{\tau}(\mathbf{b}^E(t))\right) = 0 \text{ and } \beta > 0 \end{cases} \tag{2.54}$$

where γ is as in (2.44) and $\mathbf{G}$ is as in (2.52). Note that this matches our previous definition of the plastic flow in (2.45) with $\mathbf{L} = \mathbf{G}\mathbf{b}^E$.

2.5 Work and Work Rate

2.5.1 Work Done by Elastic Forces

First we demonstrate that in the absence of plasticity, the change in energy equals to the work done by the boundary forces.

To this end, the work done by the elastic forces is defined as

$$W^E(t, \Delta t) = \int_t^{t+\Delta t} \int_{\Omega^0} \left(\nabla^{\mathbf{X}} \cdot \mathbf{P}\right)^T \mathbf{V}\, d\mathbf{X} ds \tag{2.55}$$

Applying integration by parts and the divergence theorem, the equation above yields:

$$\int_t^{t+\Delta t} \int_{\Omega^0} \nabla \cdot (\mathbf{P}^T \mathbf{V}) - \mathbf{P} : \nabla \mathbf{V} d\mathbf{X} ds = \int_t^{t+\Delta t} \int_{\partial\Omega^0} \mathbf{t}^T \mathbf{V} dS(\mathbf{X}) - \int_{\Omega^0} \mathbf{P} : \nabla \mathbf{V} d\mathbf{X} ds, \quad (2.56)$$

where $\mathbf{t} = \mathbf{P}\mathbf{n}$ is the applied traction boundary condition, and $\mathbf{n}$ is the outward pointing normal.

With defining the first term of the right hand side as the work done by the boundary conditions

$$W^{\mathbf{t}}(t, \Delta t) = \int_t^{t+\Delta t} \int_{\partial\Omega^0} \mathbf{t}^T \mathbf{V} dS(\mathbf{X}) ds, \quad (2.57)$$

we rewrite (2.55) as

$$W^E(t, \Delta t) = W^{\mathbf{t}}(t, \Delta t) - \int_t^{t+\Delta t} \int_{\Omega^0} \mathbf{P} : \nabla \mathbf{V} d\mathbf{X} ds \quad (2.58)$$

Using the definition of total potential energy from (2.19), its rate of change is

$$\frac{d}{dt} E_P(t) = \frac{\partial}{\partial t} \int_{\Omega^0} \psi \left(\mathbf{F}(\mathbf{X}, t) \right) d\mathbf{X} = \int_{\Omega^0} \frac{\partial \psi}{\partial \mathbf{F}} : \nabla \mathbf{V} d\mathbf{X} = \int_{\Omega^0} \mathbf{P} : \nabla \mathbf{V} d\mathbf{X} \quad (2.59)$$

In other words:

$$\int_t^{t+\Delta t} \int_{\Omega^0} \mathbf{P} : \nabla \mathbf{V} d\mathbf{X} ds = E_P(t + \Delta t) - E_P(t) \quad (2.60)$$

Using this directly in (2.58), we get

$$W^E(t, \Delta t) = W^{\mathbf{t}}(t, \Delta t) - E_P(t + \Delta t) + E_P(t) \quad (2.61)$$

This gives us the work done by elastic deformation in terms of the traction boundary conditions and the change in potential energy.

Next we look at the kinetic energy to be able to write the change in total energy in terms of work. The kinetic energy is defined as

$$E_K(t) = \int_{\Omega^0} \frac{1}{2} \mathbf{V}(\mathbf{X}, t)^T R(\mathbf{X}, 0) \mathbf{V}(\mathbf{X}, t) d\mathbf{X}, \quad (2.62)$$

and the rate of change of kinetic energy density is

$$\frac{\partial}{\partial t}\left[\frac{1}{2}\mathbf{V}(\mathbf{X},t)^T R(\mathbf{X},0)\mathbf{V}(\mathbf{X},t)\right] = \left(R(\mathbf{X},0)\frac{\partial \mathbf{V}(\mathbf{X},t)}{\partial t}\right)^T \mathbf{V}(\mathbf{X},t). \tag{2.63}$$

Assuming no external forces in (2.18) we have

$$R(\mathbf{X},0)\frac{\partial \mathbf{V}}{\partial t}(\mathbf{X},t) = \nabla^{\mathbf{X}} \cdot \mathbf{P}(\mathbf{X},t). \tag{2.64}$$

We integrate both sides of (2.63) over Δt and the material domain:

$$\int_t^{t+\Delta t}\int_{\Omega^0} \frac{\partial}{\partial t}\left[\frac{1}{2}\mathbf{V}(\mathbf{X},t)^T R(\mathbf{X},0)\mathbf{V}(\mathbf{X},t)\right] d\mathbf{X}ds = \int_t^{t+\Delta t}\int_{\Omega^0}\left(R(\mathbf{X},0)\frac{\partial \mathbf{V}(\mathbf{X},t)}{\partial t}\right)^T \mathbf{V}(\mathbf{X},t)d\mathbf{X}bs. \tag{2.65}$$

The left hand side is trivially the change in kinetic energy from t to $t+\Delta t$. Substituting (2.64) into the right hand side we get

$$E_K(t+\Delta t) - E_K(t) = \int_t^{t+\Delta t}\int_{\Omega^0}\left(\nabla^{\mathbf{X}} \cdot \mathbf{P}\right)^T \mathbf{V}(\mathbf{X},t)d\mathbf{X}bs. \tag{2.66}$$

Here the right hand side is exactly the definition of elastic work from (2.55), therefore the change in kinetic energy equals to the work done by the elastic deformation:

$$E_K(t+\Delta t) - E_K(t) = W^E(t,\Delta t), \tag{2.67}$$

and by (2.61) we ultimately arrive at

$$E_K(t+\Delta t) - E_K(t) + E_P(t+\Delta t) - E_P(t) = E(t+\Delta t) - E(t) = W^{\mathbf{t}}(t,\Delta t). \tag{2.68}$$

That is, if only elastic forces are considered, the change in energy is balanced by boundary traction forces.

2.5.2 Plastic Flow Rate and Potential

So far we have only considered purely elastic deformations. In the next sections we investigate how the definition of work changes when plasticity is taken into account as well.

With plasticity we have $\mathbf{F} = \mathbf{F}^E\mathbf{F}^P$, therefore

$$\dot{\mathbf{F}} = \dot{\mathbf{F}}^E\mathbf{F}^P + \mathbf{F}^E\dot{\mathbf{F}}^P, \text{ and} \tag{2.69}$$

$$\dot{\mathbf{F}}^E = \dot{\mathbf{F}}\mathbf{F}^{P-1} - \mathbf{F}^E\dot{\mathbf{F}}^P\mathbf{F}^{P-1} \tag{2.70}$$

The rate of elastic potential energy defined in (2.20) is

$$\begin{aligned}\frac{d}{dt}E_P(t) &= \int_{\Omega_0} \frac{\partial\psi}{\partial F^E_{ij}}(\mathbf{F}^E(\mathbf{X},t))\dot{F}^E_{ij}(\mathbf{X},t)d\mathbf{X} \\ &= \int_{\Omega_0} \frac{\partial\psi}{\partial F^E_{ij}}(\mathbf{F}^E)F^{P-T}_{jk}\dot{F}_{ik} - F^{E^T}_{ki}\frac{\partial\psi}{\partial F^E_{ij}}(\mathbf{F}^E)F^{P-T}_{jl}\dot{F}^P_{kl}d\mathbf{X}.\end{aligned} \tag{2.71}$$

Recall that when we account for plasticity, then the first Piola-Kirchhoff stress is

$$\mathbf{P}(\mathbf{F}^E,\mathbf{F}^P) = \frac{\partial\psi}{\partial\mathbf{F}^E}(\mathbf{F}^E)\mathbf{F}^{P-T}. \tag{2.72}$$

With this definition, the work done by the mechanical forces is

$$\begin{aligned}W^E(t,\Delta t) &= \int_t^{t+\Delta t}\int_{\Omega_0} P_{ij,j}V_i d\mathbf{X}ds \\ &= W^{\mathbf{t}}(t,\Delta t) - \int_t^{t+\Delta t}\int_{\Omega_0}\frac{\partial\psi}{\partial F^E_{ij}}(\mathbf{F}^E)F^{P-T}_{jk}\dot{F}_{ik}d\mathbf{X}ds\end{aligned} \tag{2.73}$$

where we used the results from (2.58) and the fact that $\nabla\mathbf{V} = \dot{\mathbf{F}}$.

2.5.3 The Total Work Rate Density

The total work rate per unit initial volume, or stress power density is defined as

$$\dot{w}(\mathbf{X},t) = \tau_{ij}(\mathbf{X},t)l_{ij}(\mathbf{X},t) = P_{ij}(\mathbf{X},t)\dot{F}_{ij}(\mathbf{X},t) \tag{2.74}$$

with $\mathbf{l} = \dot{\mathbf{F}}\mathbf{F}^{-1}$, the velocity gradient, and $\boldsymbol{\tau} = J\boldsymbol{\sigma} = \mathbf{P}\mathbf{F}^T$. Physically its reasonable to assume, that the work rate can be decomposed to an elastic recoverable and a permanent, nonrecoverable component. In this section we seek to identify these components.

Using the definition of P_{ij} from (2.72) this equals

$$\dot{w}(\mathbf{X},t) = P_{ij}(\mathbf{X},t)\dot{F}_{ij}(\mathbf{X},t) = \frac{\partial \psi}{\partial F^E_{ij}}\left(\mathbf{F}^E(\mathbf{X},t)\right) F^{P^{-T}}_{jk}(\mathbf{X},t)\dot{F}_{ik}(\mathbf{X},t) \tag{2.75}$$

We also define the elastic component of the total work rate

$$\dot{w}^E(\mathbf{X},t) = \tau_{ij}(\mathbf{X},t)l^E_{ij}(\mathbf{X},t) = \frac{\partial \psi}{F^E_{ij}}(\mathbf{F}^E(\mathbf{X},t))\dot{F}^E_{ij}(\mathbf{X},t) \tag{2.76}$$

with $\mathbf{l}^E = \dot{\mathbf{F}}^E\mathbf{F}^{E^{-1}}$. We prove the equality in Section A.3. When compared with the integrand in the middle of (2.71), it is clear, that $\dot{w}^E(\mathbf{X},t)$ is the rate of change in elastic potential density, that is

$$\frac{d}{dt}E_P(t) = \int_{\Omega_0} \frac{\partial \psi}{\partial F^E_{ij}}(\mathbf{F}^E(\mathbf{X},t))\dot{F}^E_{ij}(\mathbf{X},t)d\mathbf{X} = \int_{\Omega_0} \dot{w}^E(\mathbf{X},t)d\mathbf{X} \tag{2.77}$$

Next, looking at the second line of (2.71), and using the definition of $\dot{w}^E(\mathbf{X},t)$ and $\dot{w}(\mathbf{X},t)$, we can write

$$\frac{d}{dt}E_P(t) = \int_{\Omega_0} \dot{w}^E(\mathbf{X},t)d\mathbf{X} = \int_{\Omega_0} \dot{w}(\mathbf{X},t) - F^{E^T}_{ki}\frac{\partial \psi}{\partial F^E_{ij}}(\mathbf{F}^E)F^{P^{-T}}_{jl}\dot{F}^P_{kl}d\mathbf{X}. \tag{2.78}$$

The last term in the integral is the *rate of plastic dissipation*:

$$\dot{w}^P(\mathbf{X},t) = F^{E^T}_{ki}\frac{\partial \psi}{\partial F^E_{ij}}(\mathbf{F}^E)F^{P^{-T}}_{jl}\dot{F}^P_{kl} = P_{il}F^E_{ik}\dot{F}^P_{kl}. \tag{2.79}$$

With these definitions we can finally decompose the total stress power density to a term due to elastic and a term due to plastic deformations:

$$\dot{w}(\mathbf{X},t) = \dot{w}^E(\mathbf{X},t) + \dot{w}^P(\mathbf{X},t). \tag{2.80}$$

The term $\mathbf{F}^E\dot{\mathbf{F}^P}$ in (2.79) is referred to as the plastic rate of deformation (Bonet and Wood, 2008).

The work then can be expressed as

$$\begin{aligned} W^E(t,\Delta t) &= W^{\mathbf{t}}(t,\Delta t) - \int_t^{t+\Delta t}\int_{\Omega^0} P_{ij}V_{i,j}d\mathbf{X}ds \\ &= W^{\mathbf{t}}(t,\Delta t) - \int_t^{t+\Delta t}\int_{\Omega_0} \dot{w}^E(\mathbf{X},t) + \dot{w}^P(\mathbf{X},t)d\mathbf{X}ds \end{aligned} \tag{2.81}$$

and then also

$$\begin{aligned} E_K(t+\Delta t) - E_K(t) &= W^{\mathbf{t}}(t,\Delta t) - \int_t^{t+\Delta t}\int_{\Omega_0} \dot{w}^E(\mathbf{X},t) + \dot{w}^P(\mathbf{X},t)d\mathbf{X}ds \\ &= W^{\mathbf{t}}(t,\Delta t) - E_P(t+\Delta t) + E_P(t) - \int_t^{t+\Delta t}\int_{\Omega_0} \dot{w}^P(\mathbf{X},t)d\mathbf{X}ds \end{aligned} \tag{2.82}$$

and thus

$$\begin{aligned} &E_K(t+\Delta t) - E_K(t) + E_P(t+\Delta t) - E_P(t) = \\ &\qquad E(t+\Delta t) - E(t) = W^{\mathbf{t}}(t,\Delta t) - \int_t^{t+\Delta t}\int_{\Omega_0} \dot{w}^P(\mathbf{X},t)d\mathbf{X}ds \end{aligned} \tag{2.83}$$

That is, the change in total energy is due to the traction boundary condition and the loss due to plasticity. This motivates why $\dot{w}^P(\mathbf{X},t)$ is often referred to as *plastic dissipation rate*, and, as we demonstrate in Section 2.5.4, the term indeed represents loss of energy.

The *plastic rate of deformation* is defined as

$$\mathbf{l}^P = -\frac{1}{2}\mathcal{L}_{\mathbf{v}}\mathbf{b}^E\mathbf{b}^{E^{-1}}, \tag{2.84}$$

and we prove in Section A.4 that with this definition $\mathbf{l}^P$ satisfies

$$\boldsymbol{\tau} : \mathbf{l}^P = \boldsymbol{\tau} : \mathbf{l} - \boldsymbol{\tau} : \mathbf{l}^E \tag{2.85}$$

from (2.80).

2.5.4 Plastic Dissipation Rate is Non-Negative

With the results so far, and using the non-associative flow rule we arrived at in (2.52) we can show that the plastic dissipation rate is always non-negative, therefore it can only increase entropy.

Recall that we have previously defined $\mathbf{s} = \boldsymbol{\tau} - \frac{1}{d}\text{tr}(\boldsymbol{\tau})$, and that $\mathbf{G} = \frac{\partial F}{\partial \boldsymbol{\tau}} - \frac{1}{d}\text{tr}(\frac{\partial F}{\partial \boldsymbol{\tau}})\mathbf{I}$, i.e. it satisfies $\mathbf{G} = -\gamma \mathcal{L}_{\mathbf{v}} \mathbf{b}^E \mathbf{b}^{E^{-1}}$. Therefore, recalling $\mathbf{l}^P = -\frac{1}{2}\mathcal{L}_{\mathbf{v}} \mathbf{b}^E \mathbf{b}^{E^{-1}}$,

$$\begin{aligned} \dot{w}^P &= \boldsymbol{\tau} : \mathbf{l}^P \\ &= -\boldsymbol{\tau} : \frac{1}{2}\mathcal{L}_{\mathbf{v}} \mathbf{b}^E \mathbf{b}^{E^{-1}} \\ &= \gamma \boldsymbol{\tau} : \mathbf{G} \\ &= \frac{\gamma}{\|\mathbf{s}\|_F} \boldsymbol{\tau} : \mathbf{s} \\ &= \frac{\gamma}{\|\mathbf{s}\|_F} \left(\mathbf{s} + \frac{1}{d}\text{tr}(\boldsymbol{\tau})\mathbf{I} \right) : \mathbf{s} \\ &= \gamma \|\mathbf{s}\|_F. \end{aligned} \tag{2.86}$$

Thus all that remains to prove is that $\gamma \geq 0$. To do this we use the constraint that $\frac{\partial F}{\partial t} \leq 0$ when $F = 0$.

$$\begin{aligned} \frac{\partial F}{\partial t} &= \frac{\partial F}{\partial \boldsymbol{\tau}} : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E} : \dot{\mathbf{b}}^E \\ &= \frac{\partial F}{\partial \boldsymbol{\tau}} : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E} : \left(\mathbf{l}\mathbf{b}^E + \mathbf{b}^E \mathbf{l}^T - 2\gamma \frac{\partial F}{\partial \boldsymbol{\tau}} \mathbf{b}^E \right) \\ &= \underbrace{\frac{\partial F}{\partial \boldsymbol{\tau}} : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E} : \left(\mathbf{l}\mathbf{b}^E + \mathbf{b}^E \mathbf{l}^T \right)}_{\eta} - 2\gamma \underbrace{\frac{\partial F}{\partial \boldsymbol{\tau}} : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E} : \left(\frac{\partial F}{\partial \boldsymbol{\tau}} \mathbf{b}^E \right)}_{\nu}. \end{aligned} \tag{2.87}$$

So

$$0 = \eta - 2\gamma\nu \Longrightarrow \gamma = \frac{\eta}{2\nu} \tag{2.88}$$

Note that η is what $\frac{\partial F}{\partial t}$ would be in the absence of plastic flow. Thus if $\eta \leq 0$ the material is deforming in such a way that the yield function is going down, and therefore is undergoing

elastic deformation which means $\gamma = 0$. Otherwise $\eta > 0$ and

$$\begin{aligned}\nu &= \frac{\partial F}{\partial \mathbf{b}^E} : \left(2\frac{\mathbf{s}}{\|\mathbf{s}\|_F}\mathbf{b}^E\right) \\ &= 2\frac{\partial F}{\partial \mathbf{b}^E}\mathbf{b}^E : \frac{\mathbf{s}}{\|\mathbf{s}\|_F}.\end{aligned} \tag{2.89}$$

Applying the Hencky strain derivative lemma to F from Appendix A.1 we have

$$\begin{aligned}\nu &= \frac{\partial F}{\partial \boldsymbol{\epsilon}^E} : \frac{\mathbf{s}}{\|\mathbf{s}\|_F} \\ &= \frac{\partial F}{\partial \boldsymbol{\tau}} : \frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{\epsilon}^E} : \frac{\mathbf{s}}{\|\mathbf{s}\|_F} \\ &= \left(\frac{\mathbf{s}}{\|\mathbf{s}\|} + \tilde{\eta}\mathbf{I}\right) : \mathbb{C} : \frac{\mathbf{s}}{\|\mathbf{s}\|_F} \\ &= 2\mu\|\mathbf{s}\|_F.\end{aligned} \tag{2.90}$$

2.6 Conclusion

In summary, when determining the elastic and plastic portion of the deformation at some time t we arrive at the following set of equations:

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \mathbf{F}\frac{D\mathbf{C}^{P^{-1}}}{Dt}\mathbf{F}^T \tag{2.91}$$

$$\mathcal{L}_{\mathbf{v}}\mathbf{b}^E = \begin{cases}\mathbf{0}, & \text{if } F < 0 \text{ or } [F = 0 \text{ and } \beta \leq 0] \\ -\gamma \text{dev}\left(\frac{\partial F}{\partial \boldsymbol{\tau}}\right)\mathbf{b}^E, & \text{if } F = 0 \text{ and } \beta > 0\end{cases} \tag{2.92}$$

where

$$\gamma = \frac{\beta}{\frac{\partial F}{\partial \boldsymbol{\tau}}(\boldsymbol{\tau}) : \frac{\partial \boldsymbol{\tau}}{\partial \mathbf{b}^E}(\mathbf{b}^E) : \text{dev}\left(\frac{\partial F}{\partial \boldsymbol{\tau}}\right)\mathbf{b}^E} \tag{2.93}$$

With these, the rate of plastic deformation can be computed, and, by the integration of that, the plastic deformation gradient.

CHAPTER 3

The Material Point Method

The Material Point Method combines the advantages of grid based and particle based methods. It has been invented by Sulsky et al. (1994, 1995), and gained popularity in computer graphics by the snow simulation method of Stomakhin et al. (2013).

The Material Point Method is a hybrid Lagrangian-Eulerian numerical technique, where Lagrangian particles keep track of history dependent variables, while the Eulerian grid allows for simple computation of spatial gradients. Advection is naturally handled in the Lagrangian view, while interaction between the particles and self collision are resolved in the Eulerian view. Since there is no need for explicitly maintained connectivity information, the method efficiently handles large topological changes. Being a hybrid method MPM is similar to the Particle-in-Cell (PIC) (Harlow, 1964; Harlow and Welch, 1965) and the Fluid-Implicit-Particle (FLIP) (Brackbill and Ruppel, 1986) methods, but can be considered as a extension of the two to solid mechanics.

In this chapter we present the derivation of the Material Point Method for elastoplastic simulations. We adopt a Finite Element Method inspired view, where we view the particles as quadrature points and the grid as the Finite Element mesh. With this view, we use the weak formulation to discretize the governing equations presented in Section 2.3.4. Section 3.5 explains the additional considerations needed in MPM for the simulation of granular material modeled by the Drucker-Prager yield criterion. Finally Section 3.6 details the operations of each step of the simulation.

3.1 Overview

MPM proceeds from time step to time step by executing a series of steps. Each time step begins and ends with the particles carrying all the simulation data. No grid information is retained between time steps, and the grids can be freely discarded. This allows for the creation of new grid data structures at each time step, although in practice it is often more efficient to reuse them.

The steps carried out over a time step are the following:

1. Transfer to grid (Section 3.6.1): A mass and a momentum grid are created by interpolating mass and momentum from the particles. The velocity grid is established by division of momentum grid nodes by mass grid nodes.

2. Force computation on the grid (Section 3.6.2): Forces acting on each grid node are computed from the deformation gradients stored on the particles, and the change in velocities are computed from the forces.

3. Collision and friction (Section 3.6.3): Collision handling is responsible for coupling the simulated material with rigid bodies. Collision handling is performed during force computation for implicit solvers, and performed as a separate step for explicit solvers. Friction is computed afterwards in both cases, and it is calculated from the change of velocity.

4. Transfer back to particles (Section 3.6.4): Grid velocities are transferred back to the particles, and, in case of APIC transfers, each particle's affine momentum matrix is updated.

5. Particle update (Section 3.6.5): Particle positions are updated by interpolating the grid velocities before friction has been applied. The deformation gradient on each particle is updated by the gradient of the same velocities used for position update.

6. Plasticity and hardening (Section 3.6.6): Finally, the projection to the Drucker-Prager yield surface is applied to the elastic deformation gradient of each particle, and the

hardening variables are updated based on the projection.

Each step is described in detail in their corresponding section, and their pseudocode implementations are provided in Appendix B.

3.2 Notation

In addition to the notation introduced in Section 2.2, we introduce the following for variables on particles and grid nodes. We index the particles with subscript p and the grid nodes with $\mathbf{i}$ or $\mathbf{j}$. For example $\mathbf{x}_p$ refers to the position of a particle, $\mathbf{x}_\mathbf{i}$ to the position of a grid node, and $b_{k\mathbf{i}}$ is used for the kth component of the grid node position. Grid indices are tuples of two or three elements in two or three dimensions, correspondingly.

Procedures of the algorithm at times require an array of all the values of some variable. We use the $\langle\cdot\rangle$ notation to denote the array of the values.

3.3 Push Forward and Pull Back

Before the derivation of the equations solved by MPM, the concepts *push forward* and *pull back* must be addressed as they will be used repeatedly.

Quantities represented in material coordinates can be transformed to spatial coordinates, and vice versa. This is equivalent to shifting from the Lagrangian framework to the Eulerian, or the other way around. This transformation is possible due to the assumption that $\boldsymbol{\phi}$ is smooth and bijective. With these properties the sets Ω^0 and Ω^t are homeomorphic under $\boldsymbol{\phi}$. These assumptions stem from the physical requirement that no particles should be at the same position at the same time. Formally this means that in $\mathbf{x} = \boldsymbol{\phi}(\mathbf{X}, t)$ $\mathbf{X}$ exists and unique for all $\mathbf{x} \in \Omega^t$, therefore the inverse mapping $\boldsymbol{\phi}^{-1}(\cdot, t)\colon \Omega^t \to \Omega^0$ exists. These two mappings, $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{-1}$, allow us to define the *push forward* and the *pull back* of a function.

Push forward moves functions from material coordinates to spatial coordinates. More formally, given a function G defined over Ω^0, its *push forward* is the function g over Ω^t, such

that

$$g(\mathbf{x}) = G(\boldsymbol{\phi}^{-1}(\mathbf{x}, t)) \tag{3.1}$$

Pull back moves functions from spatial coordinates to material coordinates. More formally, given a function g defined over Ω^t, its *pull back* is the function G over Ω^0, such that

$$G(\mathbf{X}) = g(\boldsymbol{\phi}(\mathbf{X}, t)) \tag{3.2}$$

As Lagrangian quantities are defined in material coordinates, and Eulerian quantities are in spatial coordinates, the push forward of a Lagrangian function is transformed to the Eulerian frame, and conversely, the pull back of an Eulerian function is transformed to the Lagrangian frame.

Change of Variables

The push forward and pull back of a variable is commonly encountered in the weak form during the discretization of the governing equations. In practice this amounts to a change of variables in the integrals over subsets of Ω^0 or Ω^t.

Let $S^0 \subseteq \Omega^0$ and $S^t \subseteq \Omega^t$, such that S^0 is the preimage of S^t under $\boldsymbol{\phi}(\cdot, t)$. Furthermore, let $G \in \Omega^0 \to \mathbb{R}$ be the pull back of g, then

$$\int_{S^t} g(\mathbf{x}) d\mathbf{x} = \int_{S^0} G(\mathbf{X}) J(\mathbf{X}, t) d\mathbf{X}, \tag{3.3}$$

where $J(\mathbf{X}, t)$ is the determinant of the deformation gradient.

For surface integrals we arrive at the slightly more complicated formula: let $\mathbf{H} \in \Omega^0 \to \mathbb{R}^d$ be the pull back of $\mathbf{h}$, $\mathbf{n}(\mathbf{x})$ the outward unit normal on ∂S^t at $\mathbf{x}$, $\mathbf{N}(\mathbf{X})$ the outward unit normal on ∂S^0 at $\mathbf{X}$, then

$$\int_{\partial S^t} \mathbf{h}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) ds(\mathbf{x}) = \int_{\partial S^0} \mathbf{H}(\mathbf{X}) \cdot \mathbf{F}^{-T}(\mathbf{X}, t) \mathbf{N}(\mathbf{X}) J(\mathbf{X}, t) d\mathbf{X}. \tag{3.4}$$

3.4 Discretization

The equations to be discretized are the same as in Chapter 2, namely the conservation of mass and the conservation of momentum:

$$RJ = R^0 \tag{3.5}$$

$$R^0 \frac{\partial \mathbf{V}}{\partial t} = \nabla^{\mathbf{x}} \cdot \mathbf{P} + R^0 \mathbf{g} \tag{3.6}$$

We formulate our equations in the Lagrangian frame as that is the natural space for the particles. The equations are pushed forward to the Eulerian frame, where the grid based computations take place, then they are discretized through the weak form, viewing the particles as quadrature points.

3.4.1 Weak Form

The MPM discretization of the equations to the Eulerian grid can be viewed as a Finite Element Method discretization through the weak form. Ignoring gravity in (3.6), and rewriting it in index notation we get

$$R^0(\mathbf{X}) \frac{\partial V_i}{\partial t}(\mathbf{X}, t) = P_{ij,j}(\mathbf{X}, t). \tag{3.7}$$

Let $\mathbf{Q}(\cdot, t) \colon \Omega^0 \to \mathbb{R}^d$ be an arbitrary function. Then the weak form of (3.7) is

$$\begin{aligned} \int_{\Omega^0} Q_i(\mathbf{X}, t) R^0(\mathbf{X}) \frac{\partial V_i}{\partial t}(\mathbf{X}, t) d\mathbf{X} &= \int_{\Omega^0} Q_i(\mathbf{X}, t) P_{ij,j} d\mathbf{X} \\ &= \int_{\Omega^0} \left(Q_i(\mathbf{X}, t) P_{ij}(\mathbf{X}, t)\right)_{,j} - Q_{i,j}(\mathbf{X}, t) P_{ij}(\mathbf{X}, t) d\mathbf{X} \\ = \int_{\partial\Omega^0} Q_i(\mathbf{X}, t) P_{ij}(\mathbf{X}, t) N_j(\mathbf{X}, t) ds(\mathbf{X}) &- \int_{\Omega^0} Q_{i,j}(\mathbf{X}, t) P_{ij}(\mathbf{X}, t) d\mathbf{X} \end{aligned} \tag{3.8}$$

The traction in material space, $T_i(\mathbf{X}, t) = P_{ij}(\mathbf{X}, t) N_j(\mathbf{X}, t),\ \mathbf{X} \in \partial\Omega^0$, is given as the boundary condition. This gives the weak form:

$\forall \mathbf{Q}(\cdot, t) \colon \Omega^0 \to \mathbb{R}^d$

$$\int_{\Omega^0} Q_i R^0 \frac{\partial V_i}{\partial t} d\mathbf{X} = \int_{\partial\Omega^0} Q_i T_i ds(\mathbf{X}) - \int_{\Omega^0} Q_{i,j} P_{ij} d\mathbf{X} \tag{3.9}$$

3.4.2 Temporal Discretization

For time discretization, first we approximate $\frac{\partial \mathbf{V}}{\partial t}$ as

$$\frac{\partial V_i}{\partial t} \approx \frac{1}{\Delta t}(V_i^{n+1} - V_i^n), \tag{3.10}$$

yielding

$$\frac{1}{\Delta t} \int_{\Omega^0} Q_i R^0 (V_i^{n+1} - V_i^n) d\mathbf{X} = \int_{\partial\Omega^0} Q_i T_i ds(\mathbf{X}) - \int_{\Omega^0} Q_{i,j} P_{ij} d\mathbf{X} \tag{3.11}$$

The grid based part of MPM operates in Eulerian view, therefore we need to reformulate the above equation in this frame. This is done by applying push forward to the $\mathbf{Q}, \mathbf{R}^0, \mathbf{V}, \mathbf{T}$, and $\mathbf{P}$ functions.

$$\mathbf{q}(\mathbf{x}, t) = \mathbf{Q}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t) \tag{3.12}$$

$$Q_{i,j} = \frac{\partial Q_i}{\partial X_j} = \frac{\partial q_i}{\partial x_k} \frac{\partial x_k}{\partial X_j} = q_{i,k} F_{kj} \tag{3.13}$$

$$\boldsymbol{\sigma} = \frac{1}{J} \mathbf{P} \mathbf{F}^T \tag{3.14}$$

We are doing the discretization by freezing time at t^n, and, because of this, v^n and v^{n+1} have the following definition:

$$\mathbf{v}^n(\mathbf{x}) = \mathbf{V}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t^n), t^n) \tag{3.15}$$

$$\mathbf{v}^{n+1}(\mathbf{x}) = \mathbf{V}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t^n), t^{n+1}) \tag{3.16}$$

Note that, since $\mathbf{x} \in \Omega^{t^n}$, the definition of $\mathbf{v}^{n+1}$ uses $\boldsymbol{\phi}^{-1}(\mathbf{x}, t^n)$ and not $\boldsymbol{\phi}^{-1}(\mathbf{x}, t^{n+1})$. At the same time $\mathbf{v}^{n+1}$ represents the velocity at t^{n+1}, therefore $\mathbf{V}$ must be evaluated at that time.

We arrive at:

$$\frac{1}{\Delta t}\int_{\Omega^{t^n}} q_i(\mathbf{x},t^n)\varrho(\mathbf{x},t^n)\left(v_i^{n+1}(\mathbf{x})-v_i^n(\mathbf{x})\right)d\mathbf{x} = \int_{\partial\Omega^{t^n}} q_i(\mathbf{x},t^n)t_i(\mathbf{x},t^n)ds(\mathbf{x}) - \int_{\Omega^{t^n}} q_{i,j}(\mathbf{x},t^n)\sigma_{ij}(\mathbf{x},t^n)d\mathbf{X} \tag{3.17}$$

3.4.3 Interpolation Functions

We define our interpolation functions as the dyadic product of second and third order splines (Steffen et al., 2008). For a given one dimensional spline, we define the interpolation function $N_\mathbf{i}$ as

$$N_\mathbf{i}(\mathbf{x}) = N(\frac{1}{\Delta x}x - i)N(\frac{1}{\Delta x}y - j), \tag{3.18}$$

with $\mathbf{x} = (x, y)$ and $\mathbf{i} = (i, j)$ in two dimensions, and

$$N_\mathbf{i}(\mathbf{x}) = N(\frac{1}{\Delta x}x - i)N(\frac{1}{\Delta x}y - j)N(\frac{1}{\Delta x}z - k), \tag{3.19}$$

with $\mathbf{x} = (x, y, z)$ and $\mathbf{i} = (i, j, k)$ in three dimensions, where Δx is the grid spacing in both cases.

The quadratic spline is

$$N(x) = \begin{cases} \frac{1}{2}|x|^3 - |x|^2 + \frac{2}{3} & 0 \le |x| < 1 \\ \frac{1}{6}(2-|x|)^3 & 1 \le |x| < 2 \\ 0 & 2 \le |x| \end{cases} \tag{3.20}$$

and the cubic spline is

$$N(x) = \begin{cases} \frac{3}{4} - |x|^2 & 0 \le |x| < \frac{1}{2} \\ \frac{1}{2}\left(\frac{3}{2} - |x|\right)^2 & \frac{1}{2} \le |x| < \frac{3}{2} \\ 0 & \frac{3}{2} \le |x| \end{cases} \tag{3.21}$$

The quadratic and cubic kernels are shown in Figure 3.1. In either case, it is a requirement of the splines to form a partition of unity.

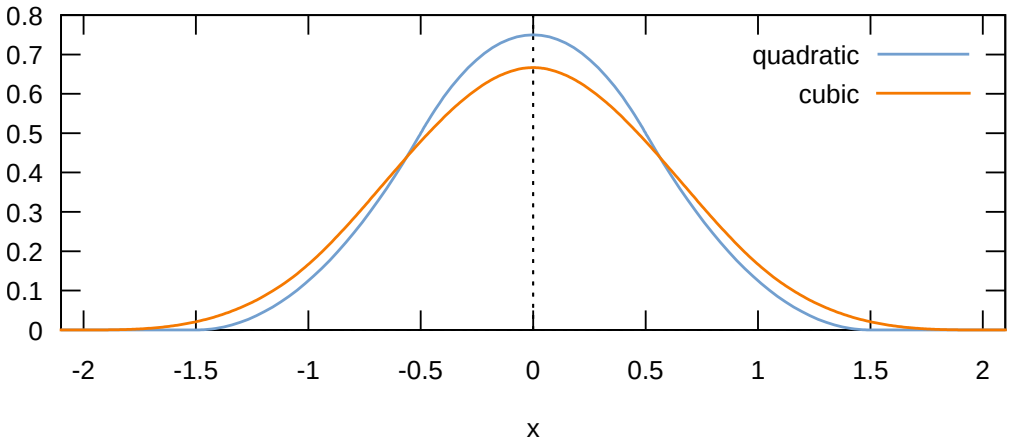

Figure 3.1: Interpolation kernels

3.4.4 Spatial Discretization

For discretization in space, we approximate q_i, v_i^n, and v_i^{n+1} with their grid based interpolants $q_{i\mathbf{i}}, v_{i\mathbf{i}}^n$, and $v_{i\mathbf{i}}^{n+1}$. The bold index refers to grid indices, and either have the form $\mathbf{i} = (x, y)$ or $\mathbf{i} = (x, y, z)$ in two or three dimensions, correspondingly. For a given $N_\mathbf{i}$ interpolation function, the approximates are

$$q_i(\mathbf{x}, t^n) \approx \sum_\mathbf{i} q_{i\mathbf{i}}^n N_\mathbf{i}(\mathbf{x}) \tag{3.22}$$

$$v_i^n(\mathbf{x}) \approx \sum_\mathbf{i} v_{i\mathbf{i}}^n N_\mathbf{i}(\mathbf{x}) \tag{3.23}$$

$$v_i^{n+1}(\mathbf{x}) \approx \sum_\mathbf{i} v_{i\mathbf{i}}^{n+1} N_\mathbf{i}(\mathbf{x}) \tag{3.24}$$

Note that, with this definition, the derivatives of the continuous variables are simply the discrete variables weighted by the derivative of the interpolation function, as in

$$q_{i,j}(\mathbf{x}, t^n) \approx \sum_\mathbf{i} q_{i\mathbf{i}}^n N_{\mathbf{i},j}(\mathbf{x}). \tag{3.25}$$

With this, the spatial discretization of (3.17) becomes

$$\frac{1}{\Delta t}\int_{\Omega^{t^n}} q^n_{i\mathbf{i}} N_{\mathbf{i}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n+1}_{i\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} - \frac{1}{\Delta t}\int_{\Omega^{t^n}} q^n_{i\mathbf{i}} N_{\mathbf{i}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n}_{i\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} = \int_{\partial\Omega^{t^n}} q^n_{i\mathbf{i}} N_{\mathbf{i}}(\mathbf{x})t_i(\mathbf{x},t^n)ds(\mathbf{x}) - \int_{\Omega^{t^n}} q^n_{i\mathbf{i}} N_{\mathbf{i},j}(\mathbf{x})\sigma_{ij}(\mathbf{x},t^n)d\mathbf{X} \tag{3.26}$$

where we use the Einstein summation convention.

Because of the weak form requirement, this must be true for all choices of $q^n_{i\mathbf{i}}$. In particular, we can select a class of function such that for each spatial dimension $\hat{\imath}$, and each grid node $\hat{\mathbf{i}}$, we define a $q^n_{i\mathbf{i}}$ in the set as

$$q^n_{i\mathbf{i}} = \begin{cases} 1, & \text{if } i = \hat{\imath} \text{ and } \mathbf{i} = \hat{\mathbf{i}} \\ 0, & \text{otherwise} \end{cases} \tag{3.27}$$

Intuitively, there are two or three unknowns at each grid node for $\mathbf{v}^{n+1}$, depending on the number of dimensions. We define the $q_{i\mathbf{i}}$ functions such that each choice of $q_{i\mathbf{i}}$ isolates a single node and dimension. With this we get a system of equations with the same number of equations and unknowns.

With the choice of $q^n_{i\mathbf{i}}$ as above, (3.26) becomes a set of equations for $\hat{\imath}$ and $\hat{\mathbf{i}}$:

$$\frac{1}{\Delta t}\int_{\Omega^{t^n}} N_{\hat{\mathbf{i}}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n+1}_{\hat{\imath}\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} - \frac{1}{\Delta t}\int_{\Omega^{t^n}} N_{\hat{\mathbf{i}}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n}_{\hat{\imath}\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} = \int_{\partial\Omega^{t^n}} N_{\hat{\mathbf{i}}}(\mathbf{x})t_{\hat{\imath}}(\mathbf{x},t^n)ds(\mathbf{x}) - \int_{\Omega^{t^n}} N_{\hat{\mathbf{i}},j}(\mathbf{x})\sigma_{\hat{\imath}j}(\mathbf{x},t^n)d\mathbf{X}, \tag{3.28}$$

in which we immediately replace $\hat{\imath}$ with i, and $\hat{\mathbf{i}}$ with $\mathbf{i}$ for sanity's sake:

$$\frac{1}{\Delta t}\int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n+1}_{i\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} - \frac{1}{\Delta t}\int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x})\varrho(\mathbf{x},t^n)v^{n}_{i\mathbf{j}}N_{\mathbf{j}}(\mathbf{x})d\mathbf{x} = \int_{\partial\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x})t_i(\mathbf{x},t^n)ds(\mathbf{x}) - \int_{\Omega^{t^n}} N_{\mathbf{i},j}(\mathbf{x})\sigma_{ij}(\mathbf{x},t^n)d\mathbf{X}. \tag{3.29}$$

The quantities appearing on the left hand side,

$$m^n_{\mathbf{ij}} = \int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x})\varrho(\mathbf{x},t^n)N_{\mathbf{j}}(\mathbf{x})d\mathbf{x}, \tag{3.30}$$

are the elements of what often referred to as the mass matrix in Finite Element Method literature. Using this notation, we get the more compact form:

$$\begin{aligned} \frac{1}{\Delta t} m^n_{\mathbf{ij}} v^{n+1}_{\mathbf{ij}} - \frac{1}{\Delta t} m^n_{\mathbf{ij}} v^n_{\mathbf{ij}} = \\ \int_{\partial\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) t_i(\mathbf{x}, t^n) ds(\mathbf{x}) - \int_{\Omega^{t^n}} N_{\mathbf{i},j}(\mathbf{x}) \sigma_{ij}(\mathbf{x}, t^n) d\mathbf{X} \end{aligned} \tag{3.31}$$

The mass matrix can be further approximated by *mass lumping*. The mass lumped matrix $\widehat{\mathbf{m}}$ is a diagonal matrix, with elements that are the row sums of the mass matrix. Using a single index to address elements along the diagonal, the lumped mass matrix is

$$\begin{aligned} \widehat{m}_{\mathbf{i}} &= \sum_{\mathbf{j}} \int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) \varrho(\mathbf{x}, t^n) N_{\mathbf{j}}(\mathbf{x}) d\mathbf{x} \\ &= \int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) \varrho(\mathbf{x}, t^n) \sum_{\mathbf{j}} N_{\mathbf{j}}(\mathbf{x}) d\mathbf{x} \\ &= \int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) \varrho(\mathbf{x}, t^n) d\mathbf{x}, \end{aligned} \tag{3.32}$$

Where we used the fact, that $N_{\mathbf{j}}$ is a partition of unity, therefore $\sum_{\mathbf{j}} N_{\mathbf{j}}(\mathbf{x}) = 1$. Furthermore, after changing the integration domain to Ω^0, $\hat{m}_{\mathbf{i}}$ is well approximated by the particle masses:

$$\begin{aligned} \widehat{m}_{\mathbf{i}} &= \int_{\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) \varrho(\mathbf{x}, t^n) d\mathbf{x} \\ &= \int_{\Omega^0} N_{\mathbf{i}}(\boldsymbol{\phi}(\mathbf{X})) \varrho(\boldsymbol{\phi}(\mathbf{X}), t^n) J(\mathbf{X}, t^n) d\mathbf{X} \\ &= \int_{\Omega^0} N_{\mathbf{i}}(\boldsymbol{\phi}(\mathbf{X})) R(\mathbf{X}, t^n) J(\mathbf{X}, t^n) d\mathbf{X} \\ &= \int_{\Omega^0} N_{\mathbf{i}}(\boldsymbol{\phi}(\mathbf{X})) R(\mathbf{X}, 0) d\mathbf{X} \\ &\approx \sum_p m_p N_{\mathbf{i}}(\mathbf{x}_p) \end{aligned} \tag{3.33}$$

The final sum corresponds to the masses defined on the Eulerian grid[1] by rasterizing the masses of the particles. The particle to grid mass transfer is the same across the possible

[1] We would like to draw the attention of the reader to the subtle difference in notation of two very distinct quantities: $m_{\mathbf{ij}}$ is an element of the full mass matrix, while $m_{\mathbf{i}}$ is the mass defined at the $\mathbf{i}^{\text{th}}$grid node.

transfer options, and the mass at each grid node is defined as

$$m_{\mathbf{i}} = \sum_p m_p N_{\mathbf{i}}(\mathbf{x}_p). \tag{3.34}$$

Using this approximation of $m_{\mathbf{ii}} = m_{\mathbf{i}}$, and $m_{\mathbf{ij}} = 0, \mathbf{i} \neq \mathbf{j}$, (3.31) becomes

$$\frac{m^n_{\mathbf{i}} v^{n+1}_{i\mathbf{i}} - m^n_{\mathbf{i}} v^n_{i\mathbf{i}}}{\Delta t} = \int_{\partial\Omega^{t^n}} N_{\mathbf{i}}(\mathbf{x}) t_i(\mathbf{x}, t^n) ds(\mathbf{x}) - \int_{\Omega^{t^n}} N_{\mathbf{i},j}(\mathbf{x}) \sigma_{ij}(\mathbf{x}, t^n) d\mathbf{X} \tag{3.35}$$

The left hand side of the above can be viewed as the change in momentum from t^n to t^{n+1}, since $m^n_{\mathbf{i}} v^n_{i\mathbf{i}} = (mv_i)^n_{\mathbf{i}}$ by construction, and $m^n_{\mathbf{i}} v^{n+1}_{i\mathbf{i}} = (mv_i)^{n+1}_{\mathbf{i}}$, because the particle positions are kept fixed within a time step.

In all of our simulations the traction boundary conditions correspond to free surface boundary conditions, meaning $\mathbf{t} = \mathbf{0}$. This leaves only the forces arising from internal stress to be resolved.

3.4.5 Stress Approximation

First, to derive an approximation of $\int_{\Omega^{t^n}} N_{\mathbf{i},j}(\mathbf{x}) \sigma_{ij}(\mathbf{x}, t^n) d\mathbf{X}$, it is more natural to think of the stress in terms of the elastic energy density function ψ. In the absence of plasticity, the definition of $\boldsymbol{\sigma}$ is simply

$$\boldsymbol{\sigma} = \frac{1}{J} \mathbf{P} \mathbf{F}^T = \frac{1}{J} \frac{\partial \psi}{\partial \mathbf{F}} \mathbf{F}^T \tag{3.36}$$

At the same time, in the presence of plasticity, we have to use the definition of $\mathbf{P}$ from Section 2.4, leading to

$$\boldsymbol{\sigma} = \frac{1}{J} \frac{\partial \psi}{\partial \mathbf{F}^E} \mathbf{F}^{P-T} \mathbf{F}^T = \frac{1}{J} \frac{\partial \psi}{\partial \mathbf{F}^E} \mathbf{F}^{E T} \tag{3.37}$$

Secondly, since the particles are viewed as quadrature points for the spatial discretization,

we can use the following approximation:

$$\begin{aligned}\int_{\Omega^{t^n}} \sigma_{ij}(\mathbf{x}, t^n) N_{\mathbf{i},j}(\mathbf{x}) d\mathbf{X} &\approx \sum_p V_p^n \sigma_{p\,ij}^n N_{\mathbf{i},j}(\mathbf{x}_p^n) \\ &= \sum_p V_p^n \frac{1}{J_p^n} \frac{\partial \psi}{\partial \mathbf{F}^E} \left(\mathbf{F}_p^{E,n}\right) \mathbf{F}_p^{E,n^T} N_{\mathbf{i},j}(\mathbf{x}_p^n),\end{aligned} \tag{3.38}$$

where V_p^n is the volume associated with the particle,[2] and we use the deformation gradient stored by particles. Each particle also maintains its initial volume as V_p^0, and the volume at t^n can be approximated from the determinant of the deformation gradient:

$$V_p^n \approx J_p^n V_p^0, \tag{3.39}$$

where V_p^0 is the initial volume associated with particle p. If Δx is the grid spacing in each direction, and n is the seeding parameter that adds n particles to each grid cell fully contained by the material, then the initial volume can be trivially defined as:

$$V_p^0 = \frac{\Delta x^d}{n}. \tag{3.40}$$

Using the approximation for the integral in (3.35) gives the forces due to internal elastic stress:

$$\mathbf{f_i} = -\sum_p V_p^0 \frac{\partial \psi}{\partial \mathbf{F}^E} \left(\mathbf{F}_p^{E,n}\right) \mathbf{F}_p^{E,n^T} N_{\mathbf{i},j}(\mathbf{x}_p^n) \tag{3.41}$$

Combining (3.35), (3.41), and the assumption that $\mathbf{t} = \mathbf{0}$, we get the final discrete equations:

$$\frac{m_{\mathbf{i}}^n v_{i\mathbf{i}}^{n+1} - m_{\mathbf{i}}^n v_{i\mathbf{i}}^n}{\Delta t} = -\sum_p V_p^0 \frac{\partial \psi}{\partial \mathbf{F}^E} \left(\mathbf{F}_p^{E,n}\right) \mathbf{F}_p^{E,n^T} N_{\mathbf{i},j}(\mathbf{x}_p^n) \tag{3.42}$$

[2]While the symbols are dangerously close, the notation is without ambiguity: volume, being a scalar, is always V, and velocity is either $\mathbf{V}$ when written as a vector, or V_i when written in components.

3.5 Plasticity

The plasticity theory presented in Section 2.4.3 defines the direction of the plastic flow in a continuous setting. These results have to be translated to a discrete time, or incremental, setting. In this section we present our return mapping algorithm that is the discrete equivalent of finding the flow direction. With the return mapping we can define the projection of the deformation gradient that projects a trial $\mathbf{F}$ to one that satisfies the Drucker-Prager yield condition. Finally, we discuss our handling of hardening.

3.5.1 Return Mapping

The return mapping algorithm is the discrete equivalent to solving for a strain that satisfies the plastic flow rule in (2.54) and that lies on or inside the Drucker-Prager yield surface. We follow the work of Simo and Meschke (1993) to derive the discrete equations from their continuous versions, and then we show how they can be solved leading to a procedure that computes the projection $\mathbf{Z}(\mathbf{F}^E, \alpha)$. This procedure starts by assuming there is no plastic flow, and the return mapping algorithm shows how to project back to the surface, if the assumption of no plastic flow is invalid.

For the rest of the section we consider the elastoplastic behavior per particle, and accordingly we adopt a Lagrangian view.

Considering the evolution of $\mathbf{b}^E$ from time t^n to time $t^{n+1} = t^n + \Delta t$, we define a new, intermediate reference configuration at t^n as

$$\Omega^{t^n} = \left\{ \tilde{\mathbf{x}} | \exists \mathbf{X} \in \Omega^0 : \tilde{\mathbf{x}} = \boldsymbol{\phi}(\mathbf{X}, t^n) \right\}, \tag{3.43}$$

and a new deformation map from this configuration onward:

$$\tilde{\boldsymbol{\phi}} : \Omega^{t^n} \times [t^n, T] \to \mathbb{R}^d, \tag{3.44}$$

$$\tilde{\boldsymbol{\phi}}(\tilde{\mathbf{x}}, t) = \boldsymbol{\phi}\left(\boldsymbol{\phi}^{-1}(\tilde{\mathbf{x}}, t^n), t \right). \tag{3.45}$$

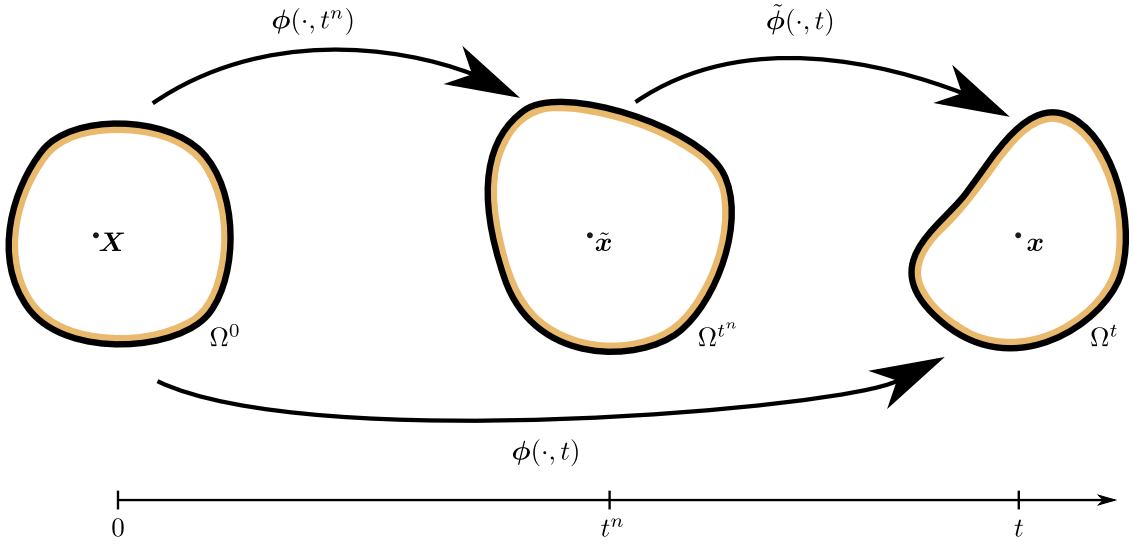

Figure 3.2: Connection between the standard and the updated Lagrangian views

Intuitively, $\tilde{\boldsymbol{\phi}}$ defines the deformation regarding Ω^{t^n} as its reference configuration. In this sense, as Ω^{t^n} replaces Ω^0 of the standard Lagrangian view, $\tilde{\boldsymbol{\phi}}$ replaces $\boldsymbol{\phi}$ the same way. This is often referred to as an updated Lagrangian view (Figure 3.2).

Similarly to the deformation gradient $\mathbf{F}(\mathbf{X}, t)$ that defines the deformation from the initial configuration (Ω^0) to the configuration Ω^t, we define $\tilde{\mathbf{F}}$ from the configuration Ω^{t^n} to the configuration Ω^t for some $t \geq t^n$:

$$\tilde{\mathbf{F}} = \frac{\partial \tilde{\boldsymbol{\phi}}}{\partial \tilde{\mathbf{x}}}, \tag{3.46}$$

$$\tilde{\mathbf{F}}\left(\boldsymbol{\phi}(\mathbf{X}, t^n), t\right) = \mathbf{F}(\mathbf{X}, t)\mathbf{F}^{-1}(\mathbf{X}, t^n), \quad \forall \mathbf{X} \in \Omega^0, t \in [t^n, T]. \tag{3.47}$$

Let us define

$$\breve{\mathbf{b}}^E = \tilde{\mathbf{F}}^{-1}\mathbf{b}^E\tilde{\mathbf{F}}^{-T} \tag{3.48}$$

to remove the elastic effects after t^n. Using (2.37), it immediately follows from the definition

$$\begin{aligned} \breve{\mathbf{b}}^E(\tilde{\mathbf{x}}, t) &= \tilde{\mathbf{F}}^{-1}(\tilde{\mathbf{x}}, t)\mathbf{F}(\mathbf{X}, t)\mathbf{C}^{P^{-1}}(\mathbf{X}, t)\mathbf{F}^T(\mathbf{X}, t)\tilde{\mathbf{F}}^{-T}(\tilde{\mathbf{x}}, t) \\ &= \mathbf{F}(\mathbf{X}, t^n)\mathbf{C}^{P^{-1}}(\mathbf{X}, t)\mathbf{F}^T(\mathbf{X}, t^n) \end{aligned} \tag{3.49}$$

where $\tilde{\mathbf{x}} = \boldsymbol{\phi}(\mathbf{X}, t^n)$, and the evolution of $\breve{\mathbf{b}}^E$ is

$$\frac{D\breve{\mathbf{b}}^E}{Dt} = -2\gamma\tilde{\mathbf{F}}^{-1}\mathbf{G}\tilde{\mathbf{F}}\breve{\mathbf{b}}^E. \tag{3.50}$$

With this, we consider the difference between the evolution of $\breve{\mathbf{b}}^E$ and $\mathbf{b}^E$ in the absence of plasticity at time $t^n < t < t^{n+1}$. In this case $\breve{\mathbf{b}}^E$ is constant, since $\frac{D\breve{\mathbf{b}}^E}{Dt} = 0$ by (3.49) and (3.50). In contrast, we can express $\mathbf{b}^E$ as

$$\mathbf{b}^E(\mathbf{x}, t) = \tilde{\mathbf{F}}(\mathbf{x}, t)\mathbf{b}^E(\tilde{\boldsymbol{\phi}}^{-1}(\mathbf{x}, t), t^n)\tilde{\mathbf{F}}^T(\mathbf{x}, t), \tag{3.51}$$

and that clearly changes in time with $\tilde{\mathbf{F}}$. In other words, $\breve{\mathbf{b}}^E$ is constant along characteristics except for the effect of plasticity, but at the same time $\mathbf{b}^E$ is also stretched by the flow. This isolation of the plastic deformation allows for a more intuitive discretization. Specifically, combined with the initial value $\breve{\mathbf{b}}^E(\mathbf{x}, t^n) = \mathbf{b}^E(\mathbf{x}, t^n)$, we can use the exponential approximation

$$\breve{\mathbf{b}}^E\Big|_{t^{n+1}} \approx \exp\left(-2\delta\gamma\tilde{\mathbf{F}}^{-1}\mathbf{G}\tilde{\mathbf{F}}\right)\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^n}, \tag{3.52}$$

where $\delta\gamma \geq 0$ will be used to enforce the constraint $F(\boldsymbol{\tau}(\mathbf{b}^E\Big|_{t^{n+1}})) \leq 0$. Multiplying the approximation by $\tilde{\mathbf{F}}\Big|_{t^{n+1}}$ on the left and $\tilde{\mathbf{F}}^T\Big|_{t^{n+1}}$ on the right, and recalling the definition of $\breve{\mathbf{b}}^E$, we obtain

$$\begin{aligned}
\mathbf{b}^E\Big|_{t^{n+1}} &= \tilde{\mathbf{F}}\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^{n+1}}\tilde{\mathbf{F}}^T\Big|_{t^{n+1}} \\
&\approx \tilde{\mathbf{F}}\Big|_{t^{n+1}}\exp\left(-2\delta\gamma\tilde{\mathbf{F}}^{-1}\mathbf{G}\tilde{\mathbf{F}}\right)\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^n}\tilde{\mathbf{F}}^T\Big|_{t^{n+1}} \\
&= \tilde{\mathbf{F}}\Big|_{t^{n+1}}\tilde{\mathbf{F}}^{-1}\Big|_{t^{n+1}}\exp\left(-2\delta\gamma\mathbf{G}\right)\Big|_{t^{n+1}}\tilde{\mathbf{F}}\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^n}\tilde{\mathbf{F}}^T\Big|_{t^{n+1}} \\
&= \exp\left(-2\delta\gamma\mathbf{G}\right)\Big|_{t^{n+1}}\tilde{\mathbf{F}}\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^n}\tilde{\mathbf{F}}^T\Big|_{t^{n+1}}.
\end{aligned} \tag{3.53}$$

Using the notation $\hat{\mathbf{b}}^E = \tilde{\mathbf{F}}\Big|_{t^{n+1}}\breve{\mathbf{b}}^E\Big|_{t^n}\tilde{\mathbf{F}}^T\Big|_{t^{n+1}}$, we are looking for a solution pair $\delta\gamma$ and

$\mathbf{b}^E\Big|_{t^{n+1}}$ such that

$$\mathbf{b}^E\Big|_{t^{n+1}} = \exp\left(-2\delta\gamma \mathbf{G}\left(\boldsymbol{\tau}(\mathbf{b}^E\Big|_{t^{n+1}})\right)\right)\hat{\mathbf{b}}^E, \quad \text{and} \tag{3.54}$$

$$F(\boldsymbol{\tau}(\mathbf{b}^E\Big|_{t^{n+1}})) \leq 0. \tag{3.55}$$

Note that $\hat{\mathbf{b}}^E$ is the elastic strain we would get without the effect of plasticity. For example, if $F(\boldsymbol{\tau}(\hat{\mathbf{b}}^E)) \leq 0$, then $\delta\gamma = 0$ and $\mathbf{b}^E\Big|_{t^{n+1}} = \hat{\mathbf{b}}^E$ is the trivial solution pair, and there is no plastic flow. In this sense, we can see that $\hat{\mathbf{b}}^E$ can be considered as the trial elastic state obtained without any plastic flow. If this does not satisfy the constraint, then $\delta\gamma$ and $\mathbf{b}^E\Big|_{t^{n+1}}$ must be defined to project $\hat{\mathbf{b}}^E$ to $\mathbf{b}^E\Big|_{t^{n+1}}$.

3.5.2 Projection

We use this process to define the projection $\mathbf{Z}(\mathbf{F}^E, \alpha)$. $\mathbf{F}^E$ is considered the trial elastic state, one obtained in the absence of plastic flow. Thus, $\hat{\mathbf{b}}^E = \mathbf{F}^E\mathbf{F}^{E^T}$ and we seek the solution of (3.54) to define the projection to $\mathbf{b}^E\Big|_{t^{n+1}}$, from which we can determine $\mathbf{Z}(\mathbf{F}^E, \alpha)$. This can be done most easily by considering the singular value decomposition of $\mathbf{F}^E$.

If the singular value decomposition of $\mathbf{F}^E$ is given by

$$\mathbf{F}^E = \mathbf{U}^E\boldsymbol{\Sigma}^E\mathbf{V}^{E^T}, \tag{3.56}$$

then

$$\hat{\mathbf{b}}^E = \mathbf{F}^E\mathbf{F}^{E^T} = \mathbf{U}^E\boldsymbol{\Sigma}^{E^2}\mathbf{U}^{E^T} \tag{3.57}$$

It can be shown, that $\mathbf{U}$ diagonalizes $\mathbf{G}\left(\boldsymbol{\tau}(\mathbf{b}^E\Big|_{t^{n+1}})\right)$ and $\mathbf{b}^E\Big|_{t^{n+1}}$, that is

$$\mathbf{G}\left(\boldsymbol{\tau}(\mathbf{b}^E\Big|_{t^{n+1}})\right) = \mathbf{U}^E\hat{\mathbf{G}}(\boldsymbol{\Sigma}^{E,n+1})\mathbf{U}^{E^T}, \tag{3.58}$$

$$\mathbf{b}^E\Big|_{t^{n+1}} = \mathbf{U}^E(\boldsymbol{\Sigma}^{E,n+1})^2\mathbf{U}^{E^T}. \tag{3.59}$$

Then we may write (3.54) as

$$\begin{aligned}\mathbf{U}^E(\boldsymbol{\Sigma}^{E,n+1})^2\mathbf{U}^{E^T} &= \exp\left(-2\delta\gamma\mathbf{U}^E\hat{\mathbf{G}}(\boldsymbol{\Sigma}^{E,n+1})\mathbf{U}^{E^T}\right)\mathbf{U}^E\boldsymbol{\Sigma}^{E^2}\mathbf{U}^{E^T} \\ &= \mathbf{U}^E\exp\left(-2\delta\gamma\hat{\mathbf{G}}(\boldsymbol{\Sigma}^{E,n+1})\right)\boldsymbol{\Sigma}^{E^2}\mathbf{U}^{E^T}\end{aligned} \tag{3.60}$$

Multiplying both sides of (3.60) by $\mathbf{U}^{E^T}$ on the left and by $\mathbf{U}^E$ on the right, and taking the logarithm of the results in

$$2\ln\boldsymbol{\Sigma}^{E,n+1} = -2\delta\gamma\hat{\mathbf{G}}(\boldsymbol{\Sigma}^E) + 2\ln\boldsymbol{\Sigma}^E. \tag{3.61}$$

The model that we choose uses the Hencky strain as the measure of deformation. By defining

$$\boldsymbol{\epsilon}^E = \ln\boldsymbol{\Sigma}^E \tag{3.62}$$

$$\boldsymbol{\epsilon}^{E,n+1} = \ln\boldsymbol{\Sigma}^{E,n+1} \tag{3.63}$$

we may simplify and rearrange (3.61) to

$$\boldsymbol{\epsilon}^E - \boldsymbol{\epsilon}^{E,n+1} = \delta\gamma\hat{\mathbf{G}} \tag{3.64}$$

This is our discrete flow rule. In the return mapping algorithm, we want to solve for an $\boldsymbol{\epsilon}^{E,n+1}$ that satisfies (3.64) subject to the constraint

$$F(\boldsymbol{\tau}(\boldsymbol{\epsilon}^{E,n+1})) \leq 0. \tag{3.65}$$

Projection to the Drucker-Prager Yield Surface

In case of the Drucker-Prager yield condition, solving (3.64) and (3.65) can be seen as a ray-cone intersection problem, as illustrated in Figure 3.3. The rays are cast from the trial stress perpendicular to the hydrostatic axis. There are three possible cases that must be considered. In Case I the trial stress in on or within the yield surface, consequently there is

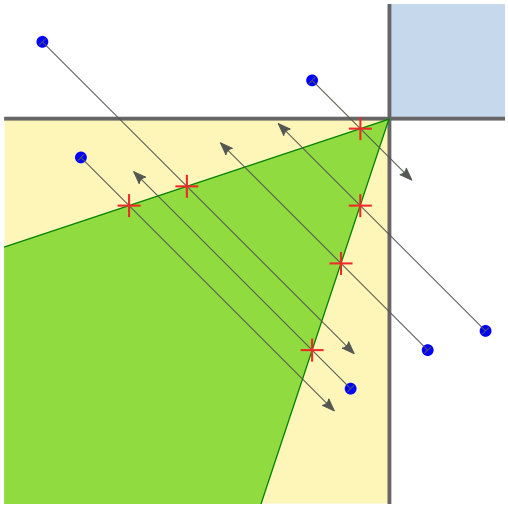

Figure 3.3: Ray-cone intersection of the Drucker-Prager yield surface

no need for projection, and the trial state is accepted. In Case II the material is in tension, and there is no resistance to motion. Accordingly, the stress is projected to the tip of the cone. In Case III there is dynamic friction, and the stress must be projected onto the yield surface. These cases are depicted in Figure 3.4.

Recall the deviatoric component of a tensors is defined as:

$$\text{dev}(\mathbf{A}) := \mathbf{A} - \frac{1}{d}\text{tr}(\mathbf{A})\mathbf{I}. \tag{3.66}$$

In essence, $\text{dev}(\mathbf{A})$ gives the deviatoric part of any arbitrary square matrix $\mathbf{A}$ of size $d \times d$.

Matching its definition from (2.52), we can write

$$\mathbf{G} = \text{dev}\left(\frac{\partial F}{\partial \boldsymbol{\tau}}\right), \tag{3.67}$$

and for the Drucker-Prager cone we have

$$\frac{\partial F}{\partial \boldsymbol{\tau}} = \alpha\mathbf{I} + \frac{\text{dev}(\boldsymbol{\tau})}{\|\text{dev}(\boldsymbol{\tau})\|_F}, \tag{3.68}$$

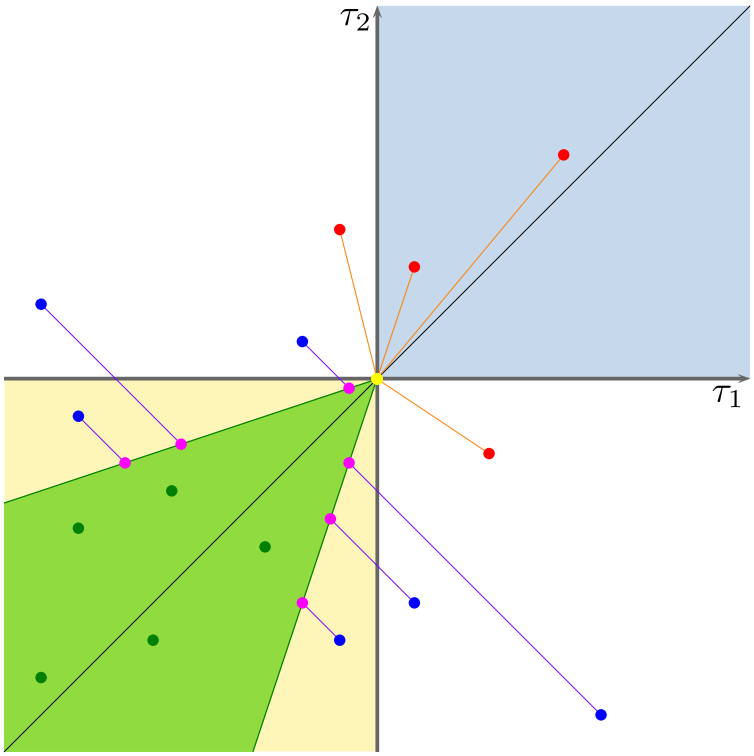

Figure 3.4: Projection cases in two dimensional principal stress space. The green points are within the Drucker-Prager cone, and no projection is needed (Case I). Red points correspond to states of expansion, and they are projected to the tip of the cone (Case II). Blue points experience more shear than friction could resist and must be projected back to the surface (Case III).

thus $\mathbf{G}$ is simply $\frac{\text{dev}(\boldsymbol{\tau})}{\|\text{dev}(\boldsymbol{\tau})\|_F}$. In principal space this becomes

$$\hat{\mathbf{G}} = \frac{\text{dev}(\hat{\boldsymbol{\tau}})}{\|\text{dev}(\hat{\boldsymbol{\tau}})\|_F}, \tag{3.69}$$

where $\hat{\boldsymbol{\tau}}$ and $\hat{\mathbf{G}}$ are diagonal. From Section A.2 we have

$$\hat{\boldsymbol{\tau}} = \frac{\partial \psi}{\partial \boldsymbol{\epsilon}^E} = 2\mu \boldsymbol{\epsilon}^{E,n+1} + \lambda \text{tr}(\boldsymbol{\epsilon}^{E,n+1})\mathbf{I} \tag{3.70}$$

because we use the energy density

$$\psi(\boldsymbol{\epsilon}^E) = \mu \text{tr}((\boldsymbol{\epsilon}^E)^2) + \frac{1}{2}\lambda \text{tr}(\boldsymbol{\epsilon}^E)^2. \tag{3.71}$$

Thus

$$\hat{\mathbf{G}} = \frac{\operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1})}{\|\operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1})\|_F}. \tag{3.72}$$

Using (3.64), we can see that $\operatorname{tr}(\boldsymbol{\epsilon}) = \operatorname{tr}(\boldsymbol{\epsilon}^{E,n+1})$, since $\operatorname{tr}(\hat{\mathbf{G}}) = 0$. Therefore

$$\operatorname{dev}(\boldsymbol{\epsilon}^{E}) - \operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1}) = \delta\gamma \frac{\operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1})}{\|\operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1})\|_F}, \tag{3.73}$$

and collecting like terms we have

$$\operatorname{dev}(\boldsymbol{\epsilon}^{E}) = \left(1 + \frac{\delta\gamma}{\|\operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1})\|_F}\right) \operatorname{dev}(\boldsymbol{\epsilon}^{E,n+1}). \tag{3.74}$$

Thus $\hat{\mathbf{G}} = \frac{\operatorname{dev}(\boldsymbol{\epsilon}^{E})}{\|\operatorname{dev}(\boldsymbol{\epsilon}^{E})\|_F}$. Then plugging the equation for the ray

$$\boldsymbol{\epsilon}^{E,n+1} = \boldsymbol{\epsilon}^{E} - \delta\gamma \frac{\operatorname{dev}(\boldsymbol{\epsilon}^{E})}{\|\operatorname{dev}(\boldsymbol{\epsilon}^{E})\|_F}, \tag{3.75}$$

into the equation for the cone $F(\boldsymbol{\tau}(\boldsymbol{\epsilon}^{E,n+1})) = \mathbf{0}$, and solving for $\delta\gamma$, we obtain

$$\delta\gamma = \|\operatorname{dev}(\boldsymbol{\epsilon}^{E})\|_F + \left(\frac{d\lambda + 2\mu}{2\mu}\right) \operatorname{tr}(\boldsymbol{\epsilon}^{E})\alpha. \tag{3.76}$$

With this we can identify the aforementioned three cases.

Case I: if $\delta\gamma \leq 0$ we intersect the cone from the inside and thus do not need to project and have $\boldsymbol{\epsilon}^{E,n+1} = \boldsymbol{\epsilon}^{E}$.

Case II: if $\operatorname{tr}(\boldsymbol{\epsilon}^{E}) \geq 0$, then the sand is in extension and we project to the tip, setting $\boldsymbol{\epsilon}^{E,n+1} = \mathbf{0}$.

Case III: otherwise we project to the cone and $\boldsymbol{\epsilon}^{E,n+1} = \boldsymbol{\epsilon}^{E} - \delta\gamma\hat{\boldsymbol{\epsilon}}^{E}$.

Finally, we return

$$\mathbf{Z}(\mathbf{F}^{E}, \alpha) = \mathbf{U}e^{\boldsymbol{\epsilon}^{E,n+1}}\mathbf{V}^{T}. \tag{3.77}$$

The pseudocode for the algorithm that carries out these steps is listed in Algorithm 2.

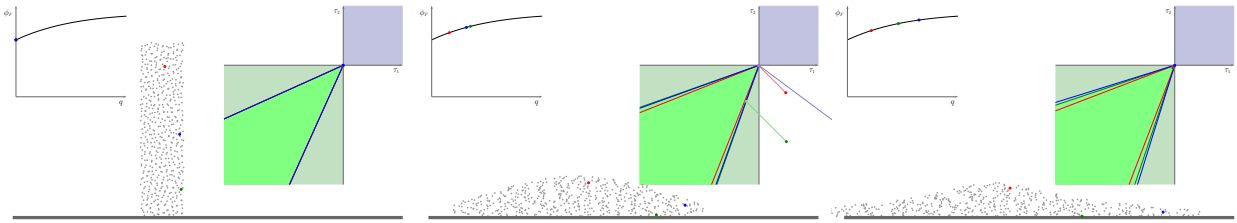

Figure 3.5: Effect of hardening on the yield surface: Three particles in a collapsing pile of sand are colored for reference. As these particles deform plasticly, their yield surface changes as they undergo hardening, resulting in a wider cone for projection.

3.5.3 Hardening

In modeling hardening, we adopt the model of Mast et al. (2014), where plastic deformation can increase the friction between sand particles. The amount of hardening depends on the amount of correction that occurred due to plasticity. In Case I, no plasticity occurred, so $\delta q_p = 0$. In Case II, all of the stress was removed, so $\delta q_p = \|\boldsymbol{\epsilon}_p^{E,n+1}\|_F$. In Case III, the amount of plasticity that occurred was $\delta q_p = \delta\gamma_p$. In each case, $\delta q_p \geq 0$. We define our hardening update using

$$q_p^{n+1} = q_p^n + \delta q_p \tag{3.78}$$

$$\phi_{F_p} = h_0 + (h_1 q_p^{n+1} - h_3)e^{-h_2 q_p^{n+1}} \tag{3.79}$$

$$\alpha_p^{n+1} = \sqrt{\frac{2}{3}}\frac{2\sin\phi_{F_p}}{3-\sin\phi_{F_p}} \tag{3.80}$$

The quantity q_p^n is the hardening state, ϕ_{F_p} is often referred to as the friction angle, the *internal coefficient of friction* is $\tan\phi_{F_p}$, and (3.79) models a curve with a maximum and an asymptote. Plausible values of ϕ_{F_p} lie in $[0, \frac{\pi}{2})$, with $\phi_{F_p} = 0$ behaving as a fluid. Feasible hardening parameters satisfy $h_0 > h_3 \geq 0$ and $h_1, h_2 \geq 0$. The values we use are listed in Table 4.2. Figure 3.5 illustrates the change in yield surface as particles undergo hardening in 2D.

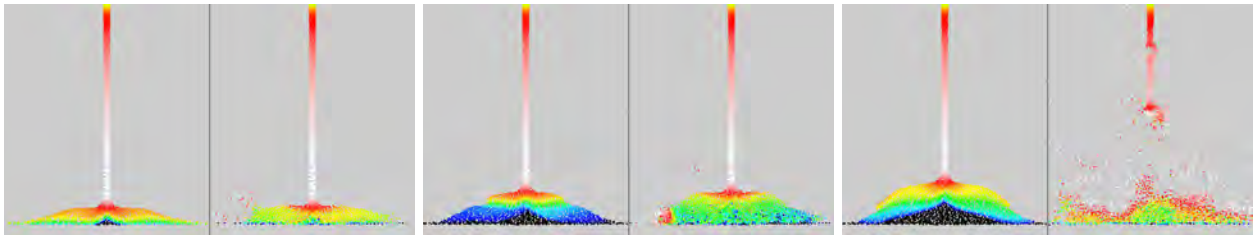

Figure 3.6: Instability of FLIP transfers: the APIC transfer (left of each frame) is able to keep the simulation stable even for stiff problems, where simulations using FLIP transfers would explode (right). The particles are colored by their vertical velocity.

3.6 Algorithm

3.6.1 Transfer to Grid

For both grid-to-particle and particle-to-grid transfer we employ the Affine Particle-In-Cell (APIC) method. APIC transfers are favorable to PIC and FLIP transfers in hybrid methods for its ability to conserve linear and angular momentum (Jiang et al., 2015; Jiang et al., 2016). PIC transfers suffer from excessive dissipation (Brackbill and Ruppel, 1986; Brackbill et al., 1988), and FLIP transfers exhibit ringing instabilities caused by particle velocity modes invisible to the grid (Brackbill, 1988; Gritton, 2014). The instabilities of FLIP transfers are particularly troublesome when used with MPM for solid mechanics. At the same time, APIC is able to conserve both linear and angular momentum without introducing artifacts.

More importantly, in the simulation of elastoplastic materials, FLIP transfers can lead to severe instability issues, as demonstrated in Figure 3.6.

Disregarding the actual transfer method used, each time steps begins by the distribution of particle masses to the Eulerian grid.

$$m_{\mathbf{i}}^n = \sum_p m_p N_{\mathbf{i}}(\mathbf{x}_p^n) \tag{3.81}$$

This is followed by the transfer of momentum. With PIC and FLIP it is performed by

$$(m\mathbf{v})_{\mathbf{i}}^n = \sum_p m_p \mathbf{v}_p^n N_{\mathbf{i}}(\mathbf{x}_p^n) \tag{3.82}$$

Both PIC and FLIP are based on the simplification, that the volume represented by particle p moves with a uniform $\mathbf{v}_p$ velocity. APIC discards this simplification, and approximates the local velocity field surrounding each particle with an affine function

$$\mathbf{v}_p^n(\mathbf{x}) = \mathbf{v}_p^n + \mathbf{C}_p^n(\mathbf{x} - \mathbf{x}_p^n). \tag{3.83}$$

The $\mathbf{C}_p^n$ matrix is expressed by the affine momentum matrix $\mathbf{B}_p^n$ and the inertia like tensor $\mathbf{D}_p^n$,

$$\mathbf{C}_p^n = \mathbf{B}_p^n(\mathbf{D}_p^n)^{-1}, \tag{3.84}$$

where

$$\mathbf{D}_p^n = \sum_{\mathbf{i}}(\mathbf{x}_\mathbf{i}^n - \mathbf{x}_p^n)(\mathbf{x}_\mathbf{i}^n - \mathbf{x}_p^n)^T N_\mathbf{i}(\mathbf{x}_p). \tag{3.85}$$

While this $\mathbf{D}_p^n$ tensor depends on the relative position of the influenced grid cells and the particle, as well as the interpolation kernel itself, fortunately it simplifies to constant tensors in case of the presented quadratic and cubic kernels as

$$\mathbf{D}_p^n = \begin{cases} \frac{\Delta x^2}{4}\mathbf{I} & \text{for quadratic } N \\ \frac{\Delta x^2}{3}\mathbf{I} & \text{for cubic } N \end{cases}, \tag{3.86}$$

where Δx is the grid spacing.

Because of the simple form of $\mathbf{D}_p$, only the $\mathbf{B}_p$ matrix is stored and tracked by each particle.

We rasterize the particle momentum to the grid using the definition of affine velocity from (3.83):

$$(m\mathbf{v})_\mathbf{i}^n = \sum_p m_p \mathbf{v}_p^n(\mathbf{x}_\mathbf{i}) N_\mathbf{i}(\mathbf{x}_p^n) = \sum_p m_p \left(\mathbf{v}_p^n + \mathbf{C}_p^n(\mathbf{x}_\mathbf{i} - \mathbf{x}_p^n)\right) N_\mathbf{i}(\mathbf{x}_p^n) \tag{3.87}$$

Then the grid velocities are simply determined by dividing the momentum by mass:

$$\mathbf{v}_\mathbf{i}^n = \frac{(m\mathbf{v})_\mathbf{i}^n}{m_\mathbf{i}^n} \tag{3.88}$$

3.6.2 Force Computation

As the next step of the algorithm, forces are computed per grid node, and in turn, these forces will be used for updating the grid velocities. The velocity update can happen through either explicit or implicit integration. The explicit integration works well with problems of low stiffness, but requires smaller time steps. The implicit integration allows for larger time steps, that could get prohibitively small for stiff problems with explicit integration. However, the implementation of an implicit integrator is significantly more complex.

Since the forces in (3.41) depend on the $\mathbf{F}_p^{E,n}$ of the particles, and that is the only term in $\mathbf{f_i}$ that depends on time, we can write the equation as a function of all the elastic deformation gradients, using the $\langle\cdot\rangle$ symbol to include all values of a variable.

$$\mathbf{f_i}(\langle\mathbf{F}_p^E\rangle) = -\sum_p V_p^0 \frac{\partial\psi}{\partial\mathbf{F}^E}\left(\langle\mathbf{F}_p^E\rangle\right)\mathbf{F}_p^{E^T}\nabla N_\mathbf{i}(\mathbf{x}_p^n) \tag{3.89}$$

3.6.2.1 Explicit Integration

With explicit integration we compute the forces based on the current state, and use those to update the velocities.

$$\mathbf{v}_\mathbf{i}^\star = \mathbf{v}_\mathbf{i}^n + \frac{\Delta t}{m_i^n}\mathbf{f_i}(\langle\mathbf{F}_p^{E,n}\rangle) \tag{3.90}$$

In case of explicit integration, it is more convenient to compute the forces separately, including all external forces, then apply them per grid node.

$$\mathbf{f}_\mathbf{i}^n = -\sum_p V_p^0 \frac{\partial\psi}{\partial\mathbf{F}^E}\left(\mathbf{F}_p^{E,n}\right)\mathbf{F}_p^{E,n^T}\nabla N_\mathbf{i}(\mathbf{x}_p^n) + \mathbf{f}^{\text{ext}}(\mathbf{x_i}) \tag{3.91}$$

$$\mathbf{v}_\mathbf{i}^\star = \mathbf{v}_\mathbf{i}^n + \frac{\Delta t}{m_i^n}\mathbf{f}_\mathbf{i}^n \tag{3.92}$$

Collision handling is performed on these velocities and friction is applied as a final step to arrive at the final velocities of the time step. Denoting the grid velocities after collision

response as $\bar{\mathbf{v}}_\mathbf{i}$, we can outline the different velocity computations during a time step as

$$\mathbf{v_i} \xrightarrow{\text{explicit update}} \mathbf{v}_\mathbf{i}^\star \xrightarrow{\text{collision}} \bar{\mathbf{v}}_\mathbf{i} \xrightarrow{\text{friction}} \mathbf{v}_\mathbf{i}^{n+1} \tag{3.93}$$

3.6.2.2 Implicit Integration

The key difference between the implicit and explicit integrations is that with implicit the updated velocities are used to compute the forces acting on the grid nodes. Furthermore, the collision handling has a profound effect on the velocities, and therefore must be accounted for. We enforce collision response through a set of G_k constraints.

With all this considered, the implicit velocity update is

$$\bar{\mathbf{v}}_\mathbf{i} = \mathbf{v}_\mathbf{i}^n + \frac{\Delta t}{m_i^n}\mathbf{f_i}(\langle \mathbf{F}_p^{E,n+1} \rangle) + \sum_k \nabla G_{k\mathbf{i}} \lambda_k \tag{3.94}$$

$$G_k \geq 0 \tag{3.95}$$

$$\lambda_k \geq 0 \tag{3.96}$$

$$G_k \lambda_k = 0 \tag{3.97}$$

A collision object-grid node pair is considered collision-free, if $G_k(\langle \bar{\mathbf{x}}_\mathbf{i} \rangle) \geq 0$, where $\bar{\mathbf{x}}_\mathbf{i} = \mathbf{x}_\mathbf{i}^n + \Delta t \bar{\mathbf{v}}_\mathbf{i}$.

This approach is implicit in plasticity, but not in hardening or friction. In the absence of plasticity, the update rules above are the Karush-Kuhn-Tucker (KKT) conditions for minimization (Nocedal and Wright, 2006).

The resulting system is nonlinear, that we solve using Newton's method. The collision constraints are solved by projection, and each $\mathbf{F}_p^{E,n+1}$ is updated by (3.114) and (3.77).

Because of plasticity the system is asymmetric, and we solve it using GMRES. We observed that the solution converges adequately within three iterations, in less than 1% of the occasions takes more than four. Based on this, we limit GMRES to 15 iterations and allow more Newton iterations.

In contrast, in the absence of plasticity, either the Conjugate Gradient method can be used in each Newton step, or the optimization solver presented in (Gast and Schroeder, 2014; Gast et al., 2015).

We present the details for the computation of the derivatives required by Newton's method in Section A.5.

After the velocity update friction needs to be applied to get the final velocities, giving the following outline:

$$\mathbf{v_i} \xrightarrow{\text{implicit update}} \bar{\mathbf{v}}_\mathbf{i} \xrightarrow{\text{friction}} \mathbf{v}_\mathbf{i}^{n+1} \tag{3.98}$$

3.6.3 Collision and Friction

Interaction with the rigid object in the simulated environment is achieved by collision handling and application of friction.

We represent each collision object as signed distance function $\phi(\mathbf{x})$, where $\phi(\mathbf{x}) = 0$ represents the surface, and $\phi(\mathbf{x}) < 0$ the inside of the obstacle.

Conceptually, we want to enforce $\phi(\mathbf{x}_p) \geq 0$ for each particle. While it is tempting in its simplicity to perform collision handling between the particles and collision objects directly, it would lead to severe artifacts. The direct update of particle positions creates deformations that are not captured by the deformation gradient, therefore lead material states, where the actual shape and the internally represented state drift further and further apart.

This drift can be avoided by updating the grid velocities instead, to get the desired particle positions. Since $\mathbf{F}_p$ is updated from $\mathbf{v}_\mathbf{i}^{n+1}$, any change in the positions driven by $\mathbf{v}_\mathbf{i}^{n+1}$ will be reflected correctly on the deformation gradient.

In its most scrupulous form, we would want to adjust $\mathbf{v}_\mathbf{i}^{n+1}$ such that the updated particle positions satisfy $\phi(\mathbf{x}_p) \geq 0$. At the same time, doing so would significantly complicate the collision handling process, therefore we adopt the approach of (Gast et al., 2015) instead.

With this approach, we set constraints on the deformed grid. We define the deformed

grid node locations as

$$\bar{\mathbf{x}}_\mathbf{i} = \mathbf{x}_\mathbf{i}^n + \Delta t \bar{\mathbf{v}}_\mathbf{i}. \tag{3.99}$$

Since a particle close, but outside of an obstacle would activate grid nodes within, therefore the enforcing $\phi(\bar{\mathbf{x}}_\mathbf{i})$ would not be reasonable. Instead, we distinguish three kinds of collision behaviors, *no-slip, slip*, and *separating*. The constraints for each kind in the form of $G_k(\langle\bar{\mathbf{x}}_\mathbf{i}\rangle) \geq 0$ or $G_k(\langle\bar{\mathbf{x}}_\mathbf{i}\rangle) = 0$ are set, and can be applied independently for each grid node.

No-slip constraints imply that any particles colliding with the obstacle should remain fixed at that relative location. This is enforced by setting the velocities of the colliding grid nodes to

$$\mathbf{v}_\mathbf{i}^{n+1} = \mathbf{v}_o(\bar{\mathbf{x}}_\mathbf{i}, t^{n+1}), \tag{3.100}$$

where $\mathbf{v}_o$ is the obstacle's velocity. This gives the constraint for velocity

$$G_k(\bar{\mathbf{x}}_\mathbf{i}) = \bar{\mathbf{x}}_\mathbf{i} - \mathbf{x}_\mathbf{i}^n - \Delta t \mathbf{v}_o(\bar{\mathbf{x}}_\mathbf{i}, t^{n+1}) = 0. \tag{3.101}$$

The constraint can be enforced by directly setting the velocity according to (3.100).

Slip constraints restrict the motion to be tangential to the surface, therefore sliding on it, but never separating or going deeper. If a grid node is already inside an obstacle ($\phi(\mathbf{x}_\mathbf{i}^n) < 0$), then it must remain at its current depth:

$$\phi(\bar{\mathbf{x}}_\mathbf{i}) = \phi(\mathbf{x}_\mathbf{i}^n). \tag{3.102}$$

If a grid node is outside of the obstacle at its original location, then no collision constraint is enforced. By allowing for this, some penetration is allowed, and the collision will be addressed once it happens.

An unsatisfied constraint in the form

$$\phi(\mathbf{x}) \geq a, \text{ or} \tag{3.103}$$

$$\phi(\mathbf{x}) = a \tag{3.104}$$

can be enforced using the normal direction of the signed distance function, $\nabla\phi$, by the update rule

$$\mathbf{x} \leftarrow \mathbf{x} - (\phi(\mathbf{x}) - a)\,\nabla\phi(\mathbf{x}). \tag{3.105}$$

Separating constraints allow movement along or away from the obstacle. The constraints encompass two cases. If a node is already inside the obstacle, then it should not move any further in:

$$\phi(\bar{\mathbf{x}}_\mathbf{i}) \geq \phi(\mathbf{x}_\mathbf{i}^n). \tag{3.106}$$

If a node is originally outside ($\phi(\mathbf{x}_\mathbf{i}^n) > 0$), then it should not penetrate:

$$\phi(\bar{\mathbf{x}}_\mathbf{i}) \geq 0. \tag{3.107}$$

The two cases can be succinctly expressed as

$$\phi(\bar{\mathbf{x}}_\mathbf{i}) \geq \max(0, \phi(\mathbf{x}_\mathbf{i}^n)). \tag{3.108}$$

The constraints are enforced as in the case of slipping constraints.

3.6.4 Transfer to Particles

With the final grid velocities of the time step, $\mathbf{v}_\mathbf{i}^{n+1}$, obtained, particle velocities can be updated. The PIC and FLIP velocities at each grid node are

$$\mathbf{v}_p^{\text{PIC}} = \sum_\mathbf{i} \mathbf{v}_\mathbf{i}^{n+1} N_\mathbf{i}(\mathbf{x}_p^n), \tag{3.109}$$

$$\mathbf{v}_p^{\text{FLIP}} = \sum_\mathbf{i} \left(\mathbf{v}_\mathbf{i}^{n+1} - \mathbf{v}_\mathbf{i}^n\right) N_\mathbf{i}(\mathbf{x}_p^n). \tag{3.110}$$

With FLIP transfers, the particle velocities are set as the linear combination of the two by the α *FLIP ratio*:

$$\mathbf{v}_p^{n+1} = \alpha(\mathbf{v}_p^n + \mathbf{v}_p^{\text{FLIP}}) + (1-\alpha)\mathbf{v}_p^{\text{PIC}}. \tag{3.111}$$

With PIC and APIC, the particle velocities are simply the PIC velocities:

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^{\text{PIC}}. \tag{3.112}$$

APIC needs to maintain the affine momentum matrix $\mathbf{B}_p$ as well, which is updated by

$$\mathbf{B}_p^{n+1} = \sum_{\mathbf{i}} \mathbf{v}_{\mathbf{i}}^{n+1} (\mathbf{x}_{\mathbf{i}}^n - \mathbf{x}_p^n)^T. \tag{3.113}$$

3.6.5 Particle Update

Next, particle positions and deformation gradients are updated. As the deformation gradient is the source for all elastic stresses, it is crucial that it accurately reflects the changes of the body. To this end, the update schemes of position and deformation gradient have to match each other.

The implicit integration solves for the updated velocities by taking into account the deformation gradients defined by the new velocities. Therefore, the update of $\mathbf{F}_p^E$ needs to follow the results of the integration, and those are the velocities before friction has been applied. That is, both the deformation gradient and the position update is written in terms of $\bar{\mathbf{v}}_{\mathbf{i}}$.

The deformation gradient evolves according to (2.39). Since the particles are advected by the flow, the material derivative can be computed as a standard forward Euler step:

$$\hat{\mathbf{F}}_p^{E,n+1} = \mathbf{F}_p^{E,n} + \Delta t (\nabla \bar{\mathbf{v}})_p \mathbf{F}_p^{E,n}, \tag{3.114}$$

where $(\nabla \bar{\mathbf{v}})_p$ is computed as

$$(\nabla \bar{\mathbf{v}})_p = \sum_{\mathbf{i}} \bar{\mathbf{v}}_{\mathbf{i}}^{n+1} \left(\nabla N_{\mathbf{i}}(\mathbf{x}_p^n) \right)^T. \tag{3.115}$$

The particle positions are updated by interpolating the deformed grid node locations defined in (3.99). For each particle this results in an update similar to the PIC velocity

update, but it based on $\bar{\mathbf{v}}_{\mathbf{i}}^{n+1}$ instead of $\mathbf{v}_{\mathbf{i}}^{n+1}$:

$$\mathbf{x}_p^{n+1} = \sum_{\mathbf{i}} \bar{\mathbf{x}}_{\mathbf{i}}^{n+1} N_{\mathbf{i}}(\mathbf{x}_p^n) = \sum_{\mathbf{i}} (\mathbf{x}_{\mathbf{i}} + \Delta t \bar{\mathbf{v}}_{\mathbf{i}}^{n+1}) N_{\mathbf{i}}(\mathbf{x}_p^n) = \mathbf{x}_p^n + \Delta t \sum_{\mathbf{i}} \bar{\mathbf{v}}_{\mathbf{i}}^{n+1} N_{\mathbf{i}}(\mathbf{x}_p^n). \tag{3.116}$$

Note that only the elastic deformation gradient $\mathbf{F}_p^E$ is used in the algorithm, therefore neither the plastic deformation gradient $\mathbf{F}_p^P$ nor the total deformation gradient $\mathbf{F}_p$ is stored. For purely elastic materials $\mathbf{F}_p = \mathbf{F}_p^E$, and $\mathbf{F}_p^E$ can be used for $\mathbf{F}_p$ everywhere.

3.6.6 Plasticity and Hardening

The plastic projection of the deformation gradient is performed at the end of each simulation step. The projection steps guarantees that the stress state associated with the final elastic deformation gradient is on or within the material's yield surface. Each particle of an elastoplastic material must undergo projection according to:

$$\mathbf{F}_p^{E,n+1} = \mathbf{Z}(\hat{\mathbf{F}}_p^{E,n+1}), \tag{3.117}$$

where $\mathbf{Z}$ is the projection operator defined by (3.77), and $\mathbf{F}_p^{E,n+1}$ is the final elastic deformation gradient.

The hardening variable is updated based on the projection according to Section 3.5.3.

For elastic materials, $\mathbf{F}_p^{E,n+1} = \hat{\mathbf{F}}_p^{E,n+1}$.

3.7 Gather and Scatter

Several computations involve summation over the grid cells to get values on particles, and vice versa. In practice these are never implemented as an iteration over the whole grid or all of the particles. The interpolation kernel defines the support of each particle on the grid, and the influence between grid nodes and particle are zero outside of that.

For a given particle, the affected nodes in can be identified with ease, but the same is not true starting from the grid nodes. This poses interesting challenges for efficient parallel

implementations.

Update of particle variables from grid nodes can be done in parallel, as it is a typical *gather* operation, where each concurrent thread reads from the same resource, but writes to distinct memory locations.

At the same time, care must be taken for implementations of particle-to-grid transfers, as different particles will have to access the same grid nodes, marking this as a *scatter* operation. In CPU implementation the two most common approaches are dividing the domain into disjoint bands, and the use of individual grids for each parallel thread. For an efficient GPU implementation we point the reader to the talk by the author (Klár, 2014).

CHAPTER 4

Results

We demonstrate the efficacy of our method by simulating several setups of dry sand. The setups have been chosen to demonstrate key behaviors of dry sand and effects of different model parameters. Table 4.1 lists the simulation setup details, and Table 4.2 lists the material parameters for each test case.

4.1 Simulation Setup

4.1.1 Friction Angle

The test demonstrates the piling behavior of a collapsing sand column. We used a fixed height/radius ratio of 4. The ground plane is set to have no-slip velocity boundary conditions to best match the experiments by Lajeunesse et al. (2004). We run tests with $20°, 25°, 35°$, and $40°$ friction angles, and as illustrated in Figure 4.1, the higher the angle, the more more weight can the pile support due to the resulting higher friction angle.

4.1.2 Pile from Spout

In this test we compare our simulation results to real world footage. In both cases sand in poured on a high frictional surface with a constant flow rate. Our model is able to capture the avalanche instability typical for similar setups (Yoshioka, 2003).

	Frame rate	Scheme	Δt	Particle #	Δx	Grid resolution	Particles/cell
Castle	72	Implicit	1×10^{-3}	7.95×10^5	0.01	$300 \times 140 \times 200$	8
Friction angle	120	Explicit	1×10^{-4}	1.20×10^6	0.001	$432 \times 144 \times 432$	32
Hourglass	48	Explicit	1×10^{-4}	4.60×10^5	0.0025	$160 \times 360 \times 160$	8
Butterfly	24	Explicit	2.5×10^{-4}	3.84×10^6	0.0045	$220 \times 55 \times 220$	8
Butterfly close	48	Explicit	1×10^{-4}	4.1×10^6	0.0014	$280 \times 140 \times 210$	2
Raking	24	Explicit	2.5×10^{-4}	3.84×10^6	0.0045	$220 \times 55 \times 220$	8
Raking close	24	Explicit	1×10^{-4}	4.3×10^6	0.0021	$336 \times 96 \times 288$	2
Pile from spout	120	Implicit	1.5×10^{-4}	9.94×10^5	0.00083	$240 \times 96 \times 240$	32
Impact	240	Explicit	5×10^{-5}	6.6×10^6	0.0078	$256 \times 256 \times 256$	9
Shovel	24	Explicit	1×10^{-4}	1.96×10^6	0.005	$160 \times 100 \times 100$	8
Young's modulus	24	Implicit	7.5×10^{-4}	*see below*	0.016	256×64	4

Table 4.1: Simulation setup. Young's modulus test particle counts: 742/739/746/745. Note that Δt denotes the maximum allowed time step size. The actual Δt is adaptive and may be restricted by CFL condition when the particle velocities are high. In all of our simulations we use a CFL number of 1, i.e., we do not allow particles to move further than Δx in a time step.

	ϱ	E	ν	Friction angle	$h_0/h_1/h_2/h_3$
Castle	2200	3.537×10^5	0.3	—	35/0/0.2/10
Friction angle	2200	3.537×10^5	0.3	20/25/30/35/40	—
Hourglass	2200	3.537×10^5	0.3	—	35/9/0.3/10
Butterfly	2200	3.537×10^5	0.3	—	35/9/0.2/10
Butterfly close	2200	3.537×10^5	0.3	—	35/9/0.2/10
Raking	2200	3.537×10^5	0.3	—	35/9/0.2/10
Raking close	2200	3.537×10^5	0.3	—	35/9/0.2/10
Pile from spout	2200	3.537×10^5	0.3	30	—
Impact	1582	3.537×10^6	0.3	22	—
Shovel	2200	3.537×10^5	0.3	—	35/9/0.2/10
Young's modulus	2200	$10^{3,4,5,6}$	0.3	—	35/9/0.3/10

Table 4.2: Material parameters are provided for all of test cases. Friction angle ϕ_F and hardening parameters h_0, h_1, and h_3 are listed in degrees for convenience.

4.1.3 Young's Modulus

The test demonstrates the effects of different Young's modulus values. Measured data for different kinds of sand are accessible, but simulations that focus on visual accuracy might opt to use lower values that allow for higher Δt. The Young's modulus for medium dense sand is in the range of 30-50 MPa (Kézdi and Rétháti, 1974), however we have found that a Young's modulus even as low as 100 kPa leads to visually adequate dynamics. Values in the range of 10 kPa or lower lead to "bouncy", gelatin-like behavior. Figure 4.3 illustrates the effect on a number of selected values.

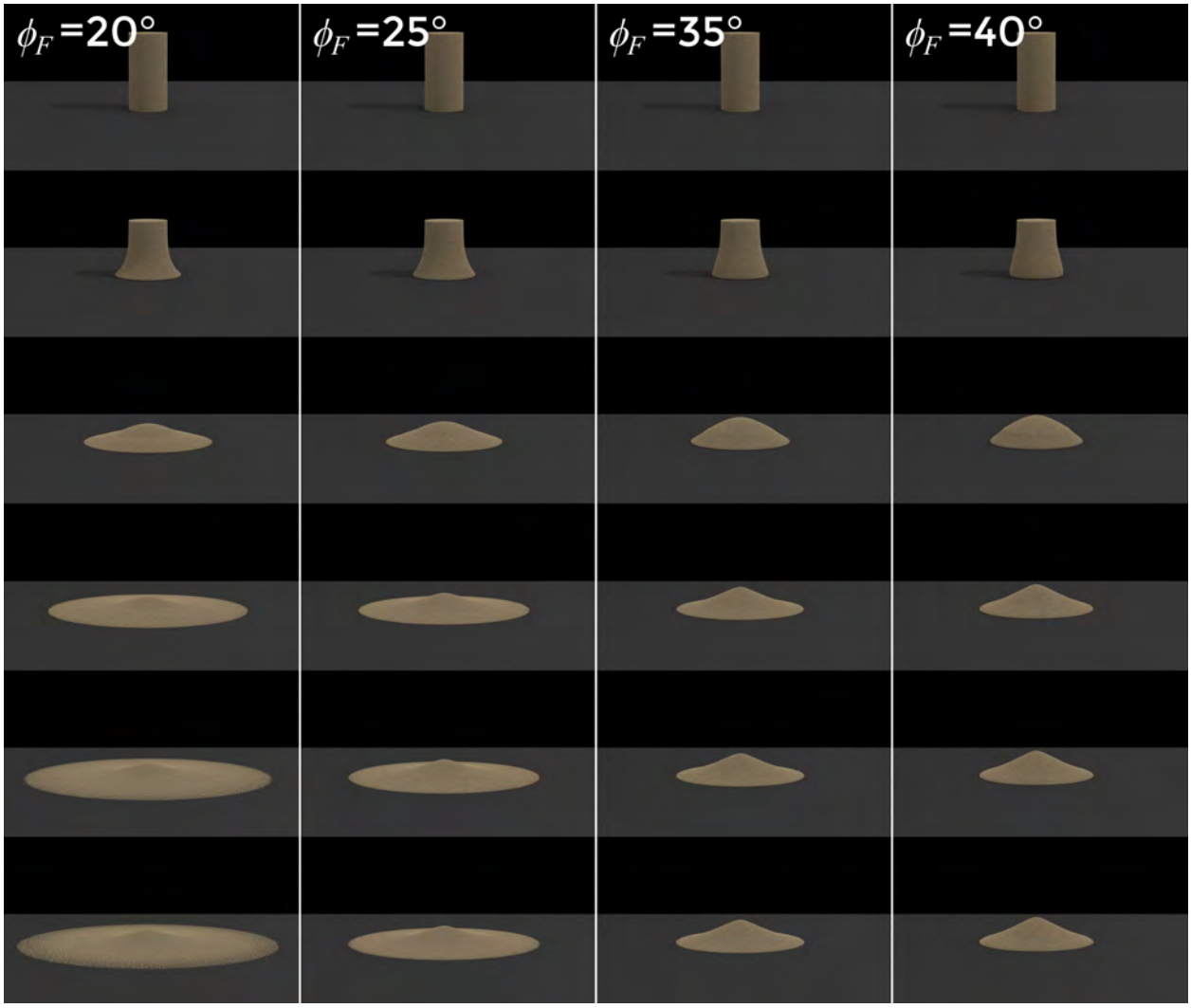

Figure 4.1: Effect of friction angle on piling.

4.1.4 Castle

Two way coupling with other bodies simulated by MPM can be done readily with our framework. Different particles can have different constitutive models associated with them, allowing for the interaction of multiple materials. The constitutive model needs to be able to compute stress values and their derivatives for a given configuration and the rest of the system is oblivious to the underling computations.

This is demonstrated in our "sand castle" test case, where an elastic ball hits a collapsing sand castle. Frames of the simulation are included in Figure 4.4

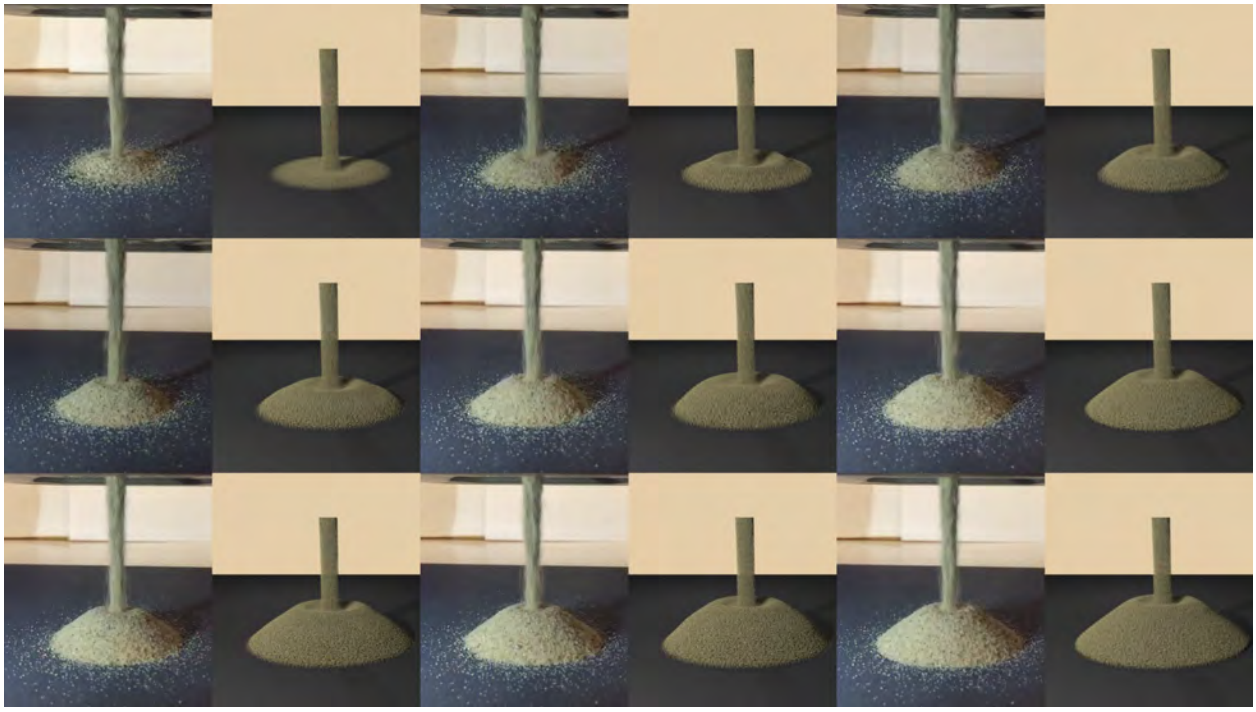

Figure 4.2: Comparison to real world footage.

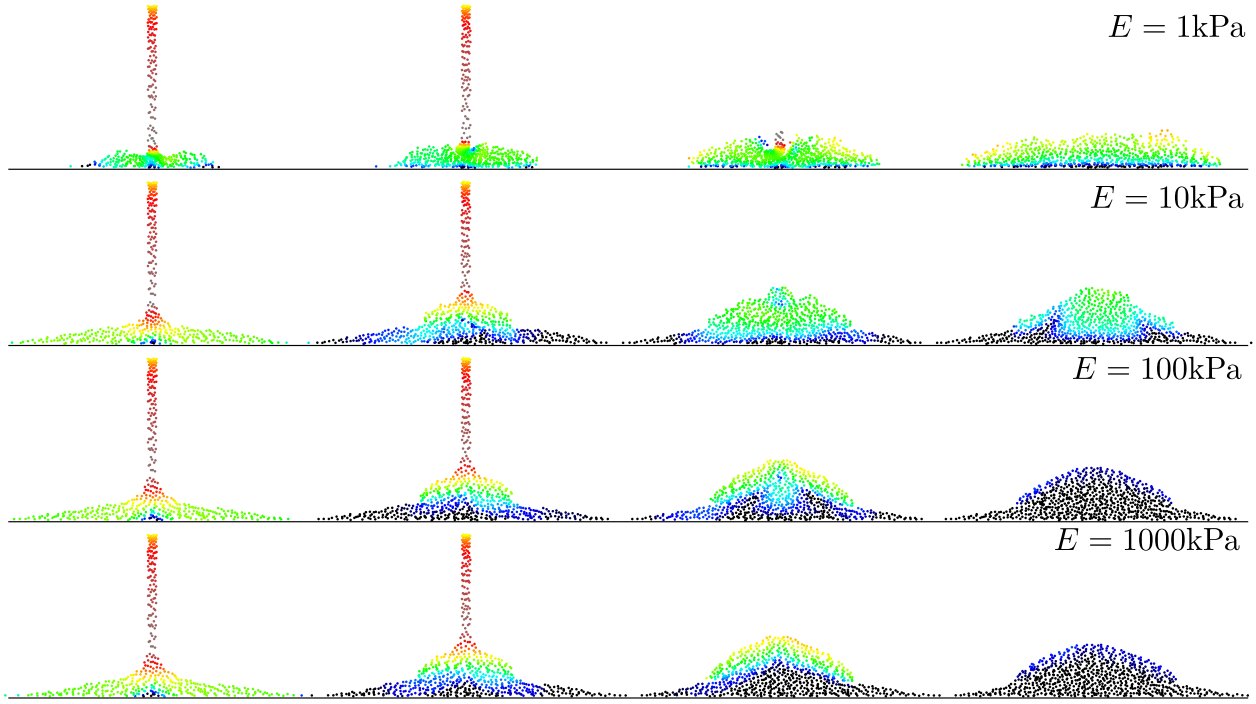

Figure 4.3: Effect of Young's modulus: Sand with a very low Young's modulus tends to be bouncy. The behavior is more like sand as the Young's modulus approaches its physical value.

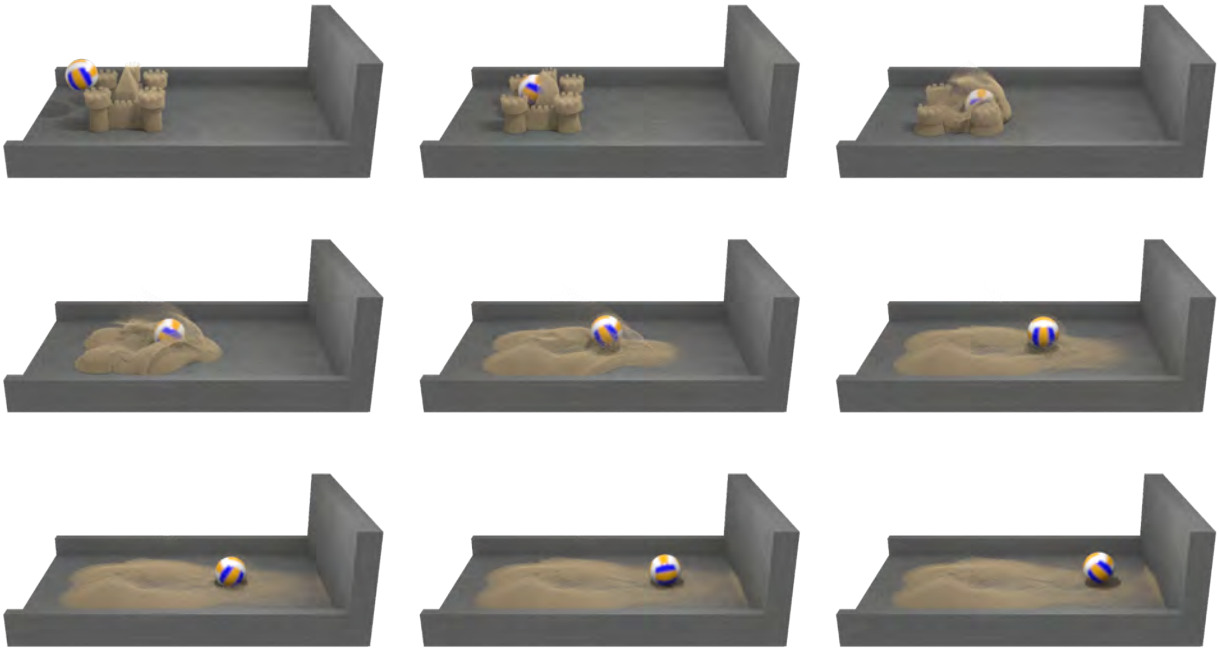

Figure 4.4: Sand castle collapse

4.1.5 Hourglass

Our hourglass example tests the quality of our results in multiple ways. The image of sand flowing through an hourglass is so prevalent, that everyone[1] has a solid intuition as to how the dynamics should look. In the top section, the majority of the grains are stationary, exhibiting an almost solid like behavior. Through to the neck, the sand is pouring like a liquid, and in the bottom portion the grains settle, forming a pile and supporting their own weight.

Our simulation is able to capture all these behaviors, as illustrated in Figure 4.5 for the initial frames, and in Figure 4.6 for the complete simulation.

[1] Admittedly, *almost* everyone.

Figure 4.5: Hourglass: initial frames

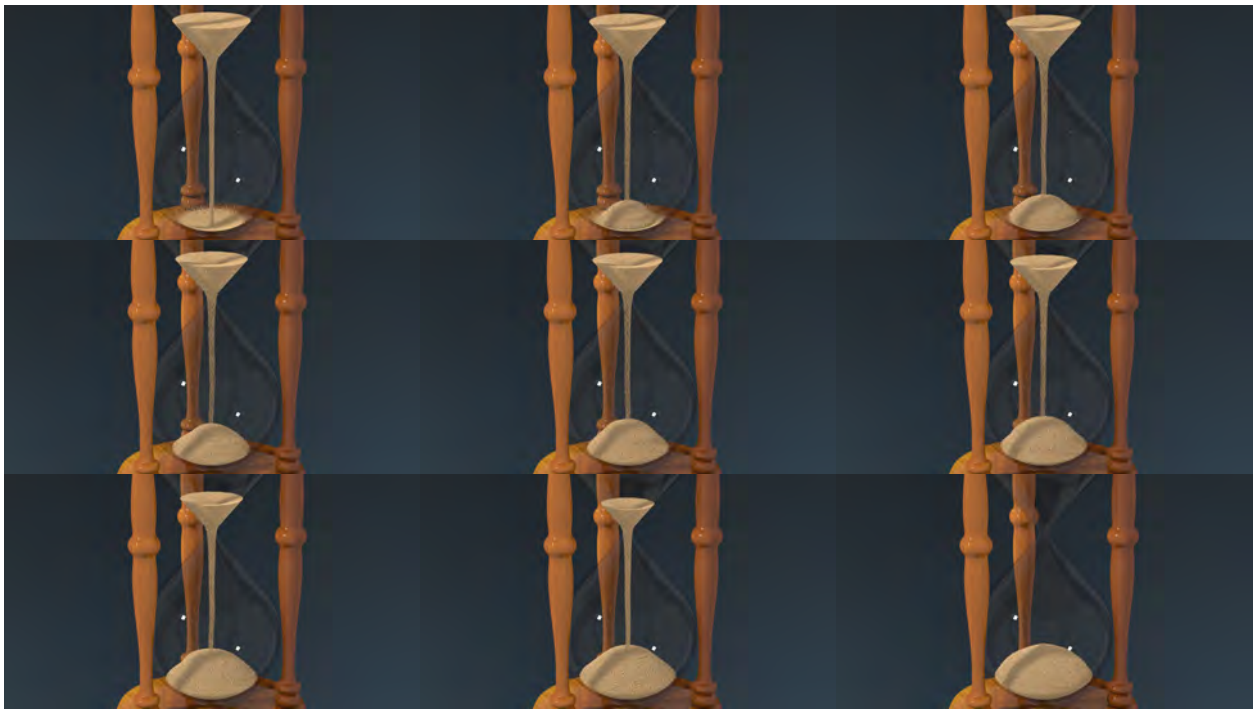

Figure 4.6: Hourglass: complete simulation

4.1.6 Sandbox: Drawing and Raking

These four tests demonstrate the interaction with collision objects. The drawing in the sand and the raking of the sand in style of a Japanese rock garden illustrates features of the simulation that can be used in animated scenes.

Both cases have been run for the whole sand box (Figures 4.7 and 4.9), and have been rendered from a top-down vantage point, and both have been rerun for a close-up view (Figures 4.8 and 4.10). The close-up version only uses a portion of the original domain, but that is simulated at a higher resolution. The increased resolution results in higher number of simulated particles for the subdomain than the total number of particles in the original simulations.

4.1.7 High Velocity Impact

The test simulates a steel ball hitting a sand bed at $6m/s$ velocity. The setup creates a stiff system, with energetic effects on the sand. The robustness and stability of our method is demonstrated by the stable and almost noise-free impact dynamics. The impact results in a smooth and symmetric crown splash, free of directional artifacts.

Figure 4.11 shows frames of the simulation.

4.1.8 Shoveling

Our shoveling test case is again an example simulation that might come up in a typical animated scene. Unlike the sandbox examples, shoveling creates larger deformations in the sand (Figure 4.12).

4.2 Performance

In explicit simulations the particle to grid operations are the most costly. These are classical *scatter* operations, and their parallelization requires either intricate synchronization

Figure 4.7: Drawing in sand

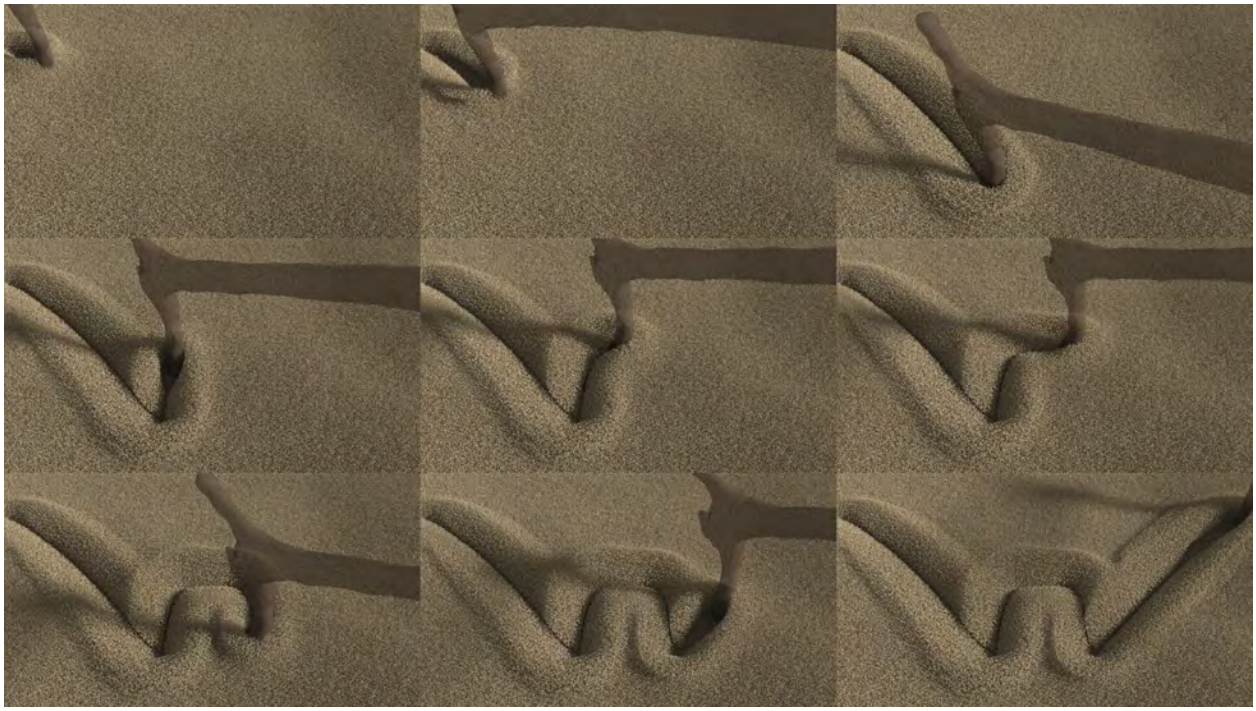

Figure 4.8: Drawing in sand, closeup

Figure 4.9: Raking sand

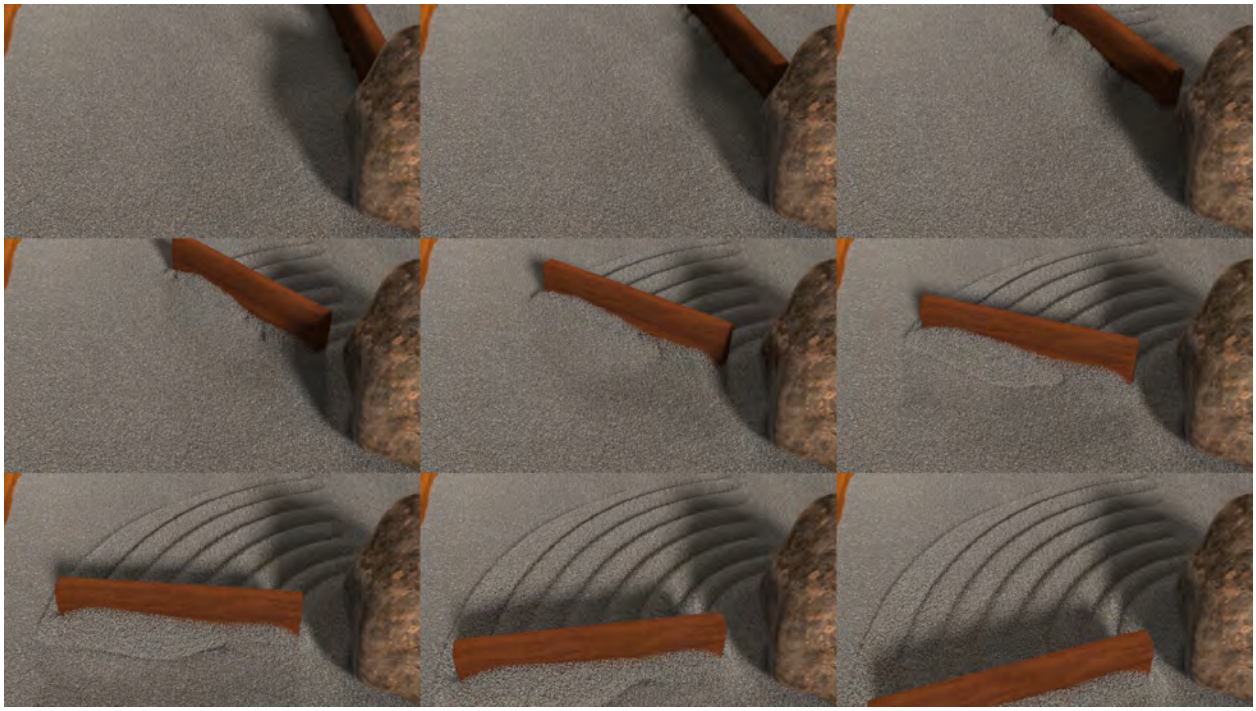

Figure 4.10: Raking sand, closeup

Figure 4.11: High velocity impact

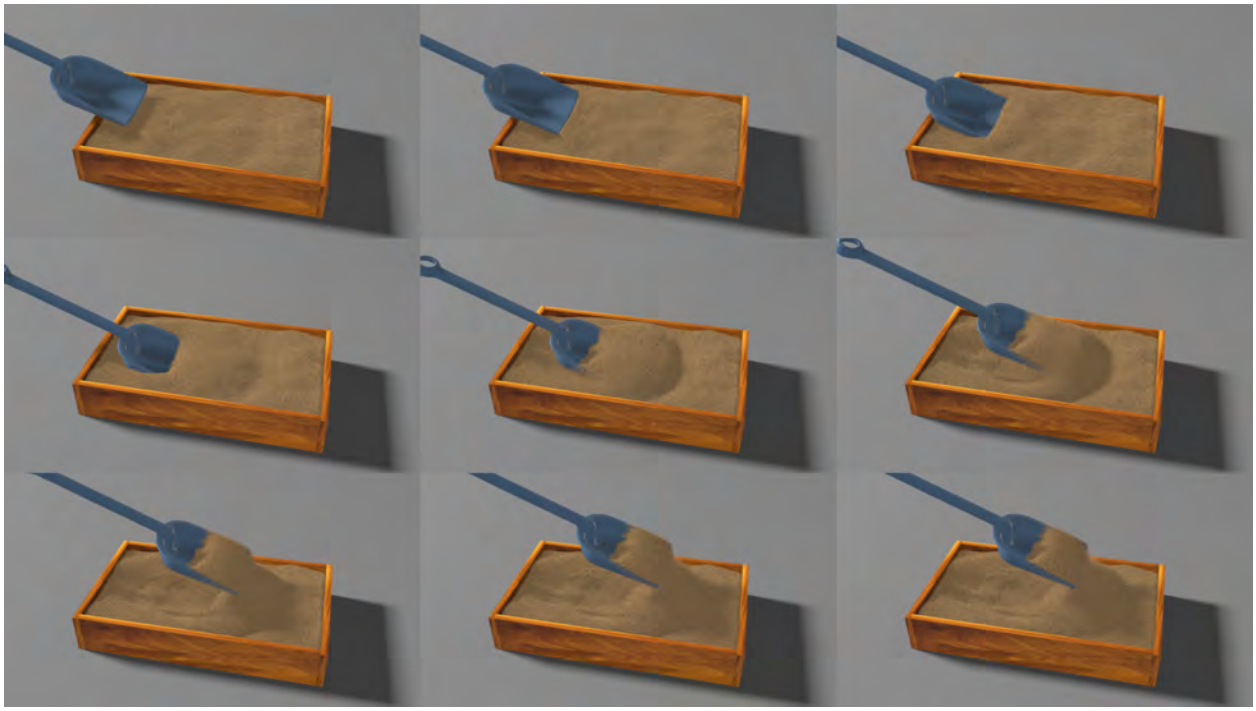

Figure 4.12: Shoveling

	Min/frame	Threads	CPU
Castle	6.80	4	Intel Xeon X5650 2.67GHz
Friction angle	4.10	4	Intel Xeon W3680 3.33GHz
Hourglass	2.00	12	Intel Xeon E5-2690 v2 3.00GHz
Butterfly	10.31	10	Intel Xeon E5-2690 v2 3.00GHz
Butterfly close	21.23	8	Intel Xeon E5-2690 v2 3.00GHz
Raking	9.78	10	Intel Xeon E5-2690 v2 3.00GHz
Raking close	32.84	8	Intel Xeon E5-2690 2.90GHz
Pile from spout	4.34	8	Intel Xeon X5650 2.67GHz
Impact	9.27	8	Intel Xeon X5690 3.47GHz
Shovel	24.84	4	Intel Xeon X5690 3.47GHz
Young's modulus	49×10^{-4}	1	Intel Xeon X5650 2.67GHz

Table 4.3: Simulation performance.

techniques or a significant memory overhead. The former can be alleviated in GPGPU implementation by taking advantage of the special memory architecture of the hardware and its fast atomic instructions (Klár, 2014), and the overhead of the latter can be diminished by the use of sparse data structures, such as OpenVDB (Museth, 2013).

Table 4.3 collects the timing results for the test cases presented in this chapter. The reported times reflect the wall-clock timing for completing a frame at the given frame rate, including the time spent on logging and I/O operations.

4.3 Rendering

We are able to run simulations with a large enough number of particles that can be rendered directly. Each particle is represented by a sphere so it can receive and cast shadows on its environment. The sphere radii are uniform in a simulation, and the colors are chosen randomly from brown tones.

Secondary render particles were not employed in the examples, but it is certainly possible to add such techniques to our framework. It is up to the given application to decide, if a lower resolution simulation augmented with passive particles would give beneficial quality-performance trade-offs.

CHAPTER 5

Conclusion

In this document we described a simulation framework for granular materials. We based our modeling on a continuum assumption, and used elastoplasticity theory to formulate the problem. We extended the Material Point Method for elastoplastic simulations. The Material Point Method is well suited for the simulation of history dependent materials with large deformations, topological changes and self collisions. The Drucker-Prager yield condition is used for the modeling of the permanent deformations of the material, and it allows us to simulate the dynamics with high visual accuracy. We demonstrate this efficacy over several examples of dry sand in various setting. Our main contributions are the derivation of an implicit integration method for elastoplasticity, the addition of APIC transfers for improved stability, and an end-to-end description of the theory of elastoplasticity, solution for the plastic flow, and the return mapping algorithm.

The focus of our discussion was on using the Drucker-Prager yield condition for cohesionless materials, and creating results worthy of the attention of the sand pile enthusiasts around the world. At the same time much of the results are applicable to other yield conditions and materials. We are excited about the prospect of future work that opens the way for new materials based on our framework.

APPENDIX A

Further Derivations

A.1 Hencky Strain Derivative Lemma

Consider symmetric positive definite matrix $\mathbf{B} \in \mathbb{R}^{d\times d}$ with $d = 2$ or 3. Use $\mathbf{B} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ to denote the eigenvalue decomposition of $\mathbf{B}$ where $\mathbf{\Lambda}$ is diagonal and positive definite. Define $\boldsymbol{\epsilon}(\mathbf{B}) = \frac{1}{2}\ln(\mathbf{B}) = \mathbf{U}\ln(\mathbf{\Lambda}^{\frac{1}{2}})\mathbf{U}^T$. In particular, if $\mathbf{B} = \mathbf{F}\mathbf{F}^T$, then $\boldsymbol{\epsilon}$ is the Hencky strain. We can also write $\mathbf{B}(\boldsymbol{\epsilon}) = e^{2\boldsymbol{\epsilon}}$.

Lemma: Suppose that $f(\mathbf{B})$ is a scalar function of $\mathbf{B}$ which is invariant under coordinate changes, and $\hat{f}(\boldsymbol{\epsilon}) = f(\mathbf{B}(\boldsymbol{\epsilon}))$, then

$$\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}} = 2\frac{\partial f}{\partial \mathbf{B}}\mathbf{B}. \tag{A.1}$$

Proof: First note that $\hat{f}$ will also be invariant under coordinate change and therefore can be written as a function of the invariants of $\boldsymbol{\epsilon}$. That is $\hat{f}(\boldsymbol{\epsilon}) = \tilde{f}(I_1, I_2, I_3)$. This means we have

$$\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}} = \frac{\partial \tilde{f}}{\partial I_1}\frac{\partial I_1}{\partial \boldsymbol{\epsilon}} + \frac{\partial \tilde{f}}{\partial I_2}\frac{\partial I_2}{\partial \boldsymbol{\epsilon}} + \frac{\partial \tilde{f}}{\partial J}\frac{\partial J}{\partial \boldsymbol{\epsilon}} \tag{A.2}$$

where $\frac{\partial I_1}{\partial \boldsymbol{\epsilon}} = \mathbf{I}$, $\frac{\partial I_2}{\partial \boldsymbol{\epsilon}} = I_1\mathbf{I} - \boldsymbol{\epsilon}^T$, $\frac{\partial J}{\partial \boldsymbol{\epsilon}} = J\boldsymbol{\epsilon}^{-T}$. Therefore, if $\boldsymbol{\epsilon}$ is diagonal then $\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}$ will be as well. Note that from frame invariance we have $\hat{f}(\boldsymbol{\epsilon}) = \hat{f}(\mathbf{R}^T\boldsymbol{\epsilon}\mathbf{R})$ for any rotation $\mathbf{R}$. Which

means

$$\begin{aligned}
\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}} : \delta\boldsymbol{\epsilon} &= \left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\mathbf{R}^T\boldsymbol{\epsilon}\mathbf{R}} : \mathbf{R}^T\delta\boldsymbol{\epsilon}\mathbf{R} \\
\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}} : \delta\boldsymbol{\epsilon} &= \mathbf{R}\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\mathbf{R}^T\boldsymbol{\epsilon}\mathbf{R}} \mathbf{R}^T : \delta\boldsymbol{\epsilon} \\
\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}} &= \mathbf{R}\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\mathbf{R}^T\boldsymbol{\epsilon}\mathbf{R}} \mathbf{R}^T \\
\mathbf{R}\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}} \mathbf{R}^T &= \left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\mathbf{R}^T\boldsymbol{\epsilon}\mathbf{R}}
\end{aligned} \tag{A.3}$$

Plugging in $\mathbf{R} = \mathbf{U}$ we have $\mathbf{U}\left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}} \mathbf{U}^T = \left.\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\right|_{\mathbf{U}^T\boldsymbol{\epsilon}\mathbf{U}}$ and is therefore diagonal.

Deriving

$$\begin{aligned}
\frac{\partial f}{\partial \mathbf{B}} : \delta\mathbf{B} &= \frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}} : \delta\boldsymbol{\epsilon} \\
&= \mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U} : \mathbf{U}^T\delta\boldsymbol{\epsilon}\mathbf{U} \\
&= \mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U} : \mathrm{diag}(\mathbf{U}^T\delta\boldsymbol{\epsilon}\mathbf{U})
\end{aligned} \tag{A.4}$$

From $\boldsymbol{\epsilon} = \frac{1}{2}\mathbf{U}\log(\boldsymbol{\Lambda})\mathbf{U}^T$ we have

$$\begin{aligned}
\delta\boldsymbol{\epsilon} &= \frac{1}{2}(\delta\mathbf{U}\log(\boldsymbol{\Lambda})\mathbf{U}^T + \mathbf{U}\delta\boldsymbol{\Lambda}\boldsymbol{\Lambda}^{-1}\mathbf{U}^T + \mathbf{U}\log(\boldsymbol{\Lambda})\delta\mathbf{U}^T) \\
\mathbf{U}^T\delta\boldsymbol{\epsilon}\mathbf{U} &= \frac{1}{2}(\mathbf{U}^T\delta\mathbf{U}\log(\boldsymbol{\Lambda}) + \delta\boldsymbol{\Lambda}\boldsymbol{\Lambda}^{-1} + \log(\boldsymbol{\Lambda})\delta\mathbf{U}^T\mathbf{U})
\end{aligned} \tag{A.5}$$

Since $\mathbf{U}$ is orthonormal $\mathbf{U}^T\delta\mathbf{U}$ is skew and therefore $\mathrm{diag}(\mathbf{U}^T\delta\boldsymbol{\epsilon}\mathbf{U}) = \mathrm{diag}(\delta\boldsymbol{\Lambda})\boldsymbol{\Lambda}^{-1}$. Similarly from $\mathbf{B} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ we have $\mathrm{diag}(\mathbf{U}^T\delta\mathbf{B}\mathbf{U}) = \frac{1}{2}\mathrm{diag}(\delta\boldsymbol{\Lambda})$. Continuing the derivation we have

$$
\begin{aligned}
\frac{\partial f}{\partial \mathbf{B}} : \delta\mathbf{B} &= \frac{1}{2}\mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U} : \mathrm{diag}(\delta\boldsymbol{\Lambda})\boldsymbol{\Lambda}^{-1} \\
&= \frac{1}{2}\mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U} : \mathrm{diag}(\mathbf{U}^T\delta\mathbf{B}\mathbf{U})\boldsymbol{\Lambda}^{-1} \\
&= \frac{1}{2}\mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U}\boldsymbol{\Lambda}^{-1} : \mathrm{diag}(\mathbf{U}^T\delta\mathbf{B}\mathbf{U}) \\
&= \frac{1}{2}\mathbf{U}^T\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U}\boldsymbol{\Lambda}^{-1} : \mathbf{U}^T\delta\mathbf{B}\mathbf{U} \\
&= \frac{1}{2}\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^T : \delta\mathbf{B} \\
&= \frac{1}{2}\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{B}^{-1} : \delta\mathbf{B}
\end{aligned}
\tag{A.6}
$$

Thus $\frac{\partial f}{\partial \mathbf{B}} = \frac{1}{2}\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}}\mathbf{B}^{-1}$ which yields $\frac{\partial \hat{f}}{\partial \boldsymbol{\epsilon}} = 2\frac{\partial f}{\partial \mathbf{B}}\mathbf{B}$.

A.2 The Relationship Between Kirchhoff Stress and Hencky Strain

Claim: For any isotropic constitutive model ψ, $\boldsymbol{\tau} = \frac{\partial \psi}{\partial \boldsymbol{\epsilon}}$.

Proof: We have

$$
\begin{aligned}
\mathbf{P} &= \frac{\partial \psi}{\partial \mathbf{F}} \\
\mathbf{P} &= \frac{\partial \tilde{\psi}}{\partial \mathbf{B}}\mathbf{F} + \mathbf{F}^T\frac{\partial \tilde{\psi}}{\partial \mathbf{B}} \\
\mathbf{P} &= 2\frac{\partial \tilde{\psi}}{\partial \mathbf{B}}\mathbf{F} \\
\mathbf{P}\mathbf{F}^T &= 2\frac{\partial \tilde{\psi}}{\partial \mathbf{B}}\mathbf{F}\mathbf{F}^T \\
\boldsymbol{\tau} &= 2\frac{\partial \tilde{\psi}}{\partial \mathbf{B}}\mathbf{B}.
\end{aligned}
\tag{A.7}
$$

By the Hencky strain derivative lemma applied to $\tilde{\psi}(\mathbf{B})$ we have $\boldsymbol{\tau} = \frac{\partial \psi}{\partial \boldsymbol{\epsilon}}$.

A.3 Elastic Component of the Total Work Rate Density

We prove that the elastic component of the total work rate defined as $\dot{w}^E = \boldsymbol{\tau} : \mathbf{l}^E$ satisfies the following:

$$\tau_{ij}(\mathbf{X},t) l^E_{ij}(\mathbf{X},t) = \frac{\partial \psi}{F^E_{ij}}(\mathbf{F}^E(\mathbf{X},t)) \dot{F}^E_{ij}(\mathbf{X},t) \tag{A.8}$$

Recall that $\boldsymbol{\tau} = \mathbf{P}\mathbf{F}^T$, $\mathbf{P} = \frac{\partial \psi}{\partial \mathbf{F}^E}\mathbf{F}^{P-T}$ and $\mathbf{l}^E = \dot{\mathbf{F}}^E \mathbf{F}^{E-1}$. With this we have

$$\begin{aligned}
\boldsymbol{\tau} : \mathbf{l}^E &= \frac{\partial \psi}{\partial \mathbf{F}^E}\mathbf{F}^{P-T}\mathbf{F}^T : \dot{\mathbf{F}}^E \mathbf{F}^{E-1} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E}(\mathbf{F}\mathbf{F}^{P-1})^T : \dot{\mathbf{F}}^E \mathbf{F}^{E-1} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E}(\mathbf{F}^E\mathbf{F}^P\mathbf{F}^{P-1})^T : \dot{\mathbf{F}}^E \mathbf{F}^{E-1} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E}\mathbf{F}^{E^T} : \dot{\mathbf{F}}^E \mathbf{F}^{E-1}
\end{aligned} \tag{A.9}$$

Writing the same in index notation gives the proof to our claim:

$$\begin{aligned}
\boldsymbol{\tau} : \mathbf{l}^E = \tau_{ij}(\mathbf{X},t) l^E_{ij}(\mathbf{X},t) &= \frac{\partial \psi}{\partial \mathbf{F}^E_{ik}}\mathbf{F}^E_{jk}\dot{\mathbf{F}}^E_{il}\mathbf{F}^{E-1}_{lj} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E_{ik}}\dot{\mathbf{F}}^E_{il}\mathbf{F}^{E-1}_{lj}\mathbf{F}^E_{jk} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E_{ik}}\dot{\mathbf{F}}^E_{il}\delta_{lk} \\
&= \frac{\partial \psi}{\partial \mathbf{F}^E_{ij}}\dot{\mathbf{F}}^E_{ij}
\end{aligned} \tag{A.10}$$

where δ_{lk} is the Kronecker delta:

$$\delta_{lk} = \begin{cases} 1 & \text{if } l = k \\ 0 & \text{if } l \neq k \end{cases} \tag{A.11}$$

A.4 Plastic Rate of Deformation

We prove that the choice for $\mathbf{l}^P$ as

$$\mathbf{l}^P = -\frac{1}{2}\mathcal{L}_{\mathbf{v}}\mathbf{b}^E\mathbf{b}^{E^{-1}} \tag{A.12}$$

satisfies

$$\boldsymbol{\tau} : \mathbf{l} = \boldsymbol{\tau} : \mathbf{l}^E + \boldsymbol{\tau} : \mathbf{l}^P. \tag{A.13}$$

First we demonstrate that the total work rate can be written in terms of the rate of the elastic left Cauchy-Green tensor ($\dot{\mathbf{b}}^E$), if we assume no plastic deformation takes place ($\frac{D\mathbf{C}^{P^{-1}}}{Dt} = 0$):

$$\boldsymbol{\tau} : \mathbf{l} = \boldsymbol{\tau} : \frac{1}{2}\left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T\right)\mathbf{b}^{E^{-1}}, \tag{A.14}$$

because, recalling $\mathbf{l} = \dot{\mathbf{F}}\mathbf{F}^{-1}$,

$$\begin{aligned}
\boldsymbol{\tau} : \dot{\mathbf{F}}\mathbf{F}^{-1} &= \boldsymbol{\tau} : \frac{1}{2}\left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T\right)\mathbf{b}^{E^{-1}} \\
&= \frac{1}{2}\boldsymbol{\tau} : \nabla\mathbf{v} + \frac{1}{2}\boldsymbol{\tau} : \mathbf{b}^E(\nabla\mathbf{v})^T\mathbf{b}^{E^{-1}} \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\mathbf{b}^E(\nabla\mathbf{v})^T\mathbf{b}^{E^{-1}}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) + \frac{1}{2}\mathrm{tr}\left(\mathbf{b}^E\boldsymbol{\tau}(\nabla\mathbf{v})^T\mathbf{b}^{E^{-1}}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) + \frac{1}{2}\mathrm{tr}\left(\mathbf{b}^{E^{-1}}\mathbf{b}^E\boldsymbol{\tau}(\nabla\mathbf{v})^T\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}(\nabla\mathbf{v})^T\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\nabla\mathbf{v}\right) \\
&= \boldsymbol{\tau} : \nabla\mathbf{v} \\
&= \boldsymbol{\tau} : \dot{\mathbf{F}}\mathbf{F}^{-1}
\end{aligned} \tag{A.15}$$

Here we used the properties that $\mathbf{b}^E$ and $\boldsymbol{\tau}$ commutes, that is $\boldsymbol{\tau}\mathbf{b}^E = \mathbf{b}^E\boldsymbol{\tau}$; that $\boldsymbol{\tau}$ is symmetric; that $\mathbf{A} : \mathbf{B} = \mathrm{tr}(\mathbf{A}^T\mathbf{B})$, for any $\mathbf{A}$ and $\mathbf{B}$; that $\mathrm{tr}(\mathbf{ABC}) = \mathrm{tr}(\mathbf{CAB})$; and (2.39), that provides $\nabla\mathbf{v} = \dot{\mathbf{F}}\mathbf{F}^{-1}$.

Next, we prove that the elastic component of the total work rate can be written in terms of $\mathbf{b}^E$ as

$$\boldsymbol{\tau} : \mathbf{l}^E = \boldsymbol{\tau} : \frac{1}{2}\dot{\mathbf{b}}^E \mathbf{b}^{E^{-1}} \tag{A.16}$$

because, recalling $\mathbf{l}^E = \dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}$,

$$\begin{aligned}
\boldsymbol{\tau} : \left(\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) &= \boldsymbol{\tau} : \frac{1}{2}\dot{\mathbf{b}}^E \mathbf{b}^{E^{-1}} \\
&= \boldsymbol{\tau} : \frac{1}{2}(\dot{\mathbf{F}}^E \mathbf{F}^{E^T} + \mathbf{F}^E \dot{\mathbf{F}}^{E^T})\mathbf{b}^{E^{-1}} \\
&= \boldsymbol{\tau} : \frac{1}{2}(\dot{\mathbf{F}}^E \mathbf{F}^{E^T} \mathbf{F}^{E^{-T}} \mathbf{F}^{E^{-1}} + \mathbf{F}^E \dot{\mathbf{F}}^{E^T} \mathbf{b}^{E^{-1}}) \\
&= \boldsymbol{\tau} : \frac{1}{2}(\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}} + \mathbf{F}^E \dot{\mathbf{F}}^{E^T} \mathbf{b}^{E^{-1}}) \\
&= \frac{1}{2}\boldsymbol{\tau} : (\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}} + \mathbf{F}^E \dot{\mathbf{F}}^{E^T} \mathbf{b}^{E^{-1}}) \\
&= \frac{1}{2}\boldsymbol{\tau} : (\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}) + \boldsymbol{\tau} : (\mathbf{F}^E \dot{\mathbf{F}}^{E^T} \mathbf{b}^{E^{-1}}) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\left(\mathbf{F}^E \dot{\mathbf{F}}^{E^T} \mathbf{b}^{E^{-1}}\right)^T\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\mathbf{b}^{E^{-T}} \dot{\mathbf{F}}^E \mathbf{F}^{E^T}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\mathbf{b}^{E^{-T}} \dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}} \mathbf{F}^E \mathbf{F}^{E^T}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\mathbf{b}^{E^{-T}} \dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}} \mathbf{b}^E\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\mathbf{b}^E \boldsymbol{\tau}\mathbf{b}^{E^{-T}} \dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\mathbf{b}^E \mathbf{b}^{E^{-T}} \dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) \\
&= \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\tau}\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right) \\
&= \boldsymbol{\tau} : \left(\dot{\mathbf{F}}^E \mathbf{F}^{E^{-1}}\right)
\end{aligned} \tag{A.17}$$

Finally, from the difference of these follows the definition of the plastic rate of deforma-

tion:

$$\begin{aligned}
\boldsymbol{\tau} : \mathbf{l}^P &= \boldsymbol{\tau} : \mathbf{l} - \boldsymbol{\tau} : \mathbf{l}^E \\
&= \boldsymbol{\tau} : \frac{1}{2}\left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T\right)\mathbf{b}^{E-1} - \boldsymbol{\tau} : \frac{1}{2}\dot{\mathbf{b}}^E\mathbf{b}^{E-1} \\
&= \boldsymbol{\tau} : \frac{1}{2}\left[\left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T\right) - \dot{\mathbf{b}}^E\right]\mathbf{b}^{E-1} \\
&= \boldsymbol{\tau} : \frac{1}{2}\left[\left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T\right) - \left(\nabla\mathbf{v}\mathbf{b}^E + \mathbf{b}^E(\nabla\mathbf{v})^T + \mathcal{L}_{\mathbf{v}}\mathbf{b}^E\right)\right]\mathbf{b}^{E-1} \\
&= \boldsymbol{\tau} : -\frac{1}{2}\mathcal{L}_{\mathbf{v}}\mathbf{b}^E\mathbf{b}^{E-1}.
\end{aligned} \tag{A.18}$$

A.5 Derivatives of Elasticity and Plasticity

As part of an implicit formulation, we encounter the combination

$$\mathbf{Y}(\mathbf{F}) = \frac{\partial\psi}{\partial\mathbf{F}}(\mathbf{Z}(\mathbf{F},\alpha)) = \mathbf{W}(\mathbf{Z}(\mathbf{F},\alpha)), \tag{A.19}$$

where $\mathbf{W}(\mathbf{F}) = \frac{\partial\psi}{\partial\mathbf{F}}(\mathbf{F})$. This corresponds to projecting a deformation gradient for plasticity and then using the result as part of a force computation. This function $\mathbf{Y}$ must be differentiated, resulting in the rank-four tensor

$$\mathbf{M} = \frac{\partial\mathbf{Y}}{\partial\mathbf{F}}(\mathbf{F}). \tag{A.20}$$

The tensor $\mathbf{M}$ has $3^4 = 81$ entries in the thee dimensional case and no symmetries. Both the construction and application of $\mathbf{M}$ are somewhat expensive, and both can be avoided.

If $\mathbf{F} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, then it turns out that $\mathbf{Z}(\mathbf{F},\alpha) = \mathbf{U}\hat{\mathbf{Z}}(\boldsymbol{\Sigma},\alpha)\mathbf{V}^T$ and $\mathbf{W}(\mathbf{F}) = \mathbf{U}\hat{\mathbf{W}}(\boldsymbol{\Sigma})\mathbf{V}^T$, where $\hat{\mathbf{Z}}(\boldsymbol{\Sigma},\alpha)$ and $\hat{\mathbf{W}}(\boldsymbol{\Sigma})$ are diagonal matrices. It follows then that $\mathbf{Y}(\mathbf{F}) = \mathbf{U}\hat{\mathbf{Y}}(\boldsymbol{\Sigma})\mathbf{V}^T$, with $\hat{\mathbf{Y}}(\boldsymbol{\Sigma}) = \hat{\mathbf{W}}(\hat{\mathbf{Z}}(\boldsymbol{\Sigma},\alpha))$, where $\hat{\mathbf{Y}}(\boldsymbol{\Sigma})$ is also a diagonal matrix. To be able to carry out these steps, it is required of the energy density function ψ, that it depends only on the singular values of $\mathbf{F}$. In essence, we need to be able to define $\hat{\psi}$ such that, $\hat{\psi}(\boldsymbol{\Sigma}) = \psi(\mathbf{F})$. This allows us to write the definition of $\hat{\mathbf{W}}$ as $\hat{\mathbf{W}}(\boldsymbol{\Sigma}) = \frac{\partial\hat{\psi}}{\partial\boldsymbol{\Sigma}}(\boldsymbol{\Sigma})$.

Note that we have taken advantage of these relationships to avoid computing the sin-

gular value decomposition more often than necessary. Indeed, $\hat{\mathbf{W}}$ is implemented by ENERGY_DERIVATIVE, and $\hat{\mathbf{Z}}$ is implemented by PROJECT.

Since these functions are rather simple in *diagonal space*, it might not be too surprising that the derivatives are also simpler there. Let $\hat{\mathbf{M}}$ be the diagonal space version of $\mathbf{M}$, defined by

$$M_{ijkl} = \hat{M}_{rsuv} U_{ir} V_{js} U_{ku} V_{lv}, \tag{A.21}$$

where index notation is used and summation is implied. In the pseudocode, the operation

$$\mathbf{A} = \hat{\mathbf{M}} : \mathbf{T} \tag{A.22}$$

is requested. This is equivalent to $A_{ij} = \hat{M}_{ijkl} T_{kl}$. What remains is to determine the structure of $\hat{\mathbf{M}}$. The way that this is done follows from the approach taken in (Stomakhin et al., 2012), but we summarize the result here.

Introducing the auxiliary variables $\overline{Y}_{ij}, D_{ij}, S_{ij}$, the nonzero entries of $\hat{\mathbf{M}}$ are (with no summation implied)

$$\hat{M}_{iijj} = \overline{Y}_{ij} = \frac{\partial \hat{Y}_{ii}}{\partial \Sigma_{jj}} \tag{A.23}$$

$$\hat{M}_{ijij} = \frac{D_{ij} + S_{ij}}{2} \qquad i \neq j \tag{A.24}$$

$$\hat{M}_{ijji} = \frac{D_{ij} - S_{ij}}{2} \qquad i \neq j \tag{A.25}$$

$$D_{ij} = \frac{\hat{Y}_{ii} - \hat{Y}_{jj}}{\Sigma_{ii} - \Sigma_{jj}} \tag{A.26}$$

$$S_{ij} = \frac{\hat{Y}_{ii} + \hat{Y}_{jj}}{\Sigma_{ii} + \Sigma_{jj}} \tag{A.27}$$

Note that $D_{ij} = D_{ji}$ and $S_{ij} = S_{ji}$, so that $\hat{M}_{ijij} = \hat{M}_{jiji}$ and $\hat{M}_{ijji} = \hat{M}_{jiij}$. Thus, there are only $9 + 3 + 3 = 15$ distinct nonzero entries to compute, and $9 + 6 + 6 = 21$ multiplications (plus 12 additions) are required to apply the tensor. Of the computations required, $\overline{Y}_{ij}$ and D_{ij} merit further attention. S_{ij} can be computed directly, since division by zero is not a

concern there.

First we describe the computation of $\overline{\mathbf{Y}}$. Since $\hat{\mathbf{Y}}$ is the composition of two functions, $\overline{\mathbf{Y}}$ is computed using the chain rule. Note that both $\hat{\mathbf{W}}$ and $\overline{\mathbf{W}}$ are evaluated at $\hat{\mathbf{Z}} = \hat{\mathbf{Z}}(\boldsymbol{\Sigma}, \alpha)$.

$$\overline{Y}_{ij} = \frac{\partial \hat{Y}_{ii}}{\partial \Sigma_{jj}} = \sum_k \frac{\partial \hat{W}_{ii}}{\partial \Sigma_{kk}} \frac{\partial \hat{Z}_{kk}}{\partial \Sigma_{jj}} = \sum_k \overline{W}_{ik} \overline{Z}_{kj}, \tag{A.28}$$

where the matrices $\overline{\mathbf{Y}}$, $\overline{\mathbf{W}}$, and $\overline{\mathbf{Z}}$ represent the derivatives of the functions $\hat{\mathbf{Y}}$, $\hat{\mathbf{W}}$, and $\hat{\mathbf{Z}}$ when diagonal matrices are treated as functions taking a vector and returning a vector. Differentiating $\hat{\mathbf{W}}$ gives

$$\overline{\mathbf{W}} = \hat{\mathbf{Z}}^{-1}(2\mu\mathbf{I} - 2\mu \ln(\hat{\mathbf{Z}}) + \lambda \mathbf{o}\mathbf{o}^T - \lambda \mathrm{tr}(\ln(\hat{\mathbf{Z}}))\mathbf{I})\hat{\mathbf{Z}}^{-1}, \tag{A.29}$$

where $\mathbf{o}$ is the all-ones vector.

Next, we need to differentiate the projection to get $\overline{\mathbf{Z}}$. There are three cases to consider. In Case I, $\overline{\mathbf{Z}} = \mathbf{I}$. In Case II, $\overline{\mathbf{Z}} = \mathbf{0}$. This leaves only Case III, in which case

$$\boldsymbol{\epsilon} = \mathrm{diag}(\ln \boldsymbol{\Sigma}) \quad \mathbf{w} = \mathrm{diag}(\boldsymbol{\Sigma}^{-1}) \quad k = \mathrm{tr}(\ln \boldsymbol{\Sigma}) \quad \mathbf{s} = \boldsymbol{\epsilon} - \frac{k}{d}\mathbf{o} \quad \hat{\mathbf{s}} = \frac{\mathbf{s}}{\|\mathbf{s}\|} \quad p = \frac{\alpha k(d\lambda + 2\mu)}{2\mu\|\mathbf{s}\|} \tag{A.30}$$

$$\overline{\mathbf{Z}} = \hat{\mathbf{Z}}\Big(\Big(\frac{1+2p}{d}\mathbf{o} - \frac{p}{k}\boldsymbol{\epsilon}\Big)\mathbf{w}^T - p\big(\mathbf{I} - \hat{\mathbf{s}}\hat{\mathbf{s}}^T\big)\boldsymbol{\Sigma}^{-1}\Big). \tag{A.31}$$

With this, $\overline{\mathbf{Y}}$ can be readily computed as $\overline{\mathbf{Y}} = \overline{\mathbf{W}}\,\overline{\mathbf{Z}}$ by (A.28).

Finally, for D_{ij}, we can avoid potential numerical problems in the case where $\Sigma_{ii} \approx \Sigma_{jj}$ by writing

$$D_{ij} = \frac{\hat{Y}_{ii} - \hat{Y}_{jj}}{\Sigma_{ii} - \Sigma_{jj}} = \left(\frac{\hat{Y}_{ii} - \hat{Y}_{jj}}{\hat{Z}_{ii} - \hat{Z}_{jj}}\right)\left(\frac{\hat{Z}_{ii} - \hat{Z}_{jj}}{\Sigma_{ii} - \Sigma_{jj}}\right). \tag{A.32}$$

This works as long as both factors can be robustly computed. Consider the first term.

$$\hat{Y}_{ii} = \frac{2\mu \ln \hat{Z}_{ii}}{\hat{Z}_{ii}} + \frac{\lambda \mathrm{tr}(\ln \hat{\mathbf{Z}})}{\hat{Z}_{ii}} \tag{A.33}$$

$$\hat{Y}_{ii} - \hat{Y}_{jj} = \frac{2\mu(\ln(\hat{Z}_{ii}) - \ln(\hat{Z}_{jj}))}{\hat{Z}_{ii}} - \frac{\lambda \mathrm{tr}(\ln \hat{\mathbf{Z}}) + 2\mu \ln \hat{Z}_{jj}}{\hat{Z}_{ii}\hat{Z}_{jj}}(\hat{Z}_{ii} - \hat{Z}_{jj}) \tag{A.34}$$

$$\frac{\hat{Y}_{ii} - \hat{Y}_{jj}}{\hat{Z}_{ii} - \hat{Z}_{jj}} = \frac{2\mu}{\hat{Z}_{ii}} \frac{\ln(\hat{Z}_{ii}) - \ln(\hat{Z}_{jj})}{\hat{Z}_{ii} - \hat{Z}_{jj}} - \frac{\lambda \mathrm{tr}(\ln \hat{\mathbf{Z}}) + 2\mu \ln \hat{Z}_{jj}}{\hat{Z}_{ii}\hat{Z}_{jj}} \tag{A.35}$$

The only term that presents further difficulties is the divided difference on the natural log. This can be computed by noting

$$\frac{\ln(x) - \ln(y)}{x - y} = \frac{1}{y} \frac{\ln(w+1)}{w} \qquad x = (w+1)y \tag{A.36}$$

$$= \frac{1}{y} \begin{cases} 1 & |w| < \epsilon \\ \mathrm{log1p}(w)/w & |w| \geq \epsilon \end{cases} \tag{A.37}$$

Here we have made use of the log1p library routine, which is designed to be robust in this case.

The next term that must be considered is the divided difference on $\hat{\mathbf{Z}}$. Following the same general procedure yields

$$\frac{\hat{Z}_{ii} - \hat{Z}_{jj}}{\Sigma_{ii} - \Sigma_{jj}} = \left(1 - \frac{\delta\gamma}{\|\hat{\boldsymbol{\epsilon}}\|_F}\right) \left(\frac{\exp(H_{ii}) - \exp(H_{jj})}{H_{ii} - H_{jj}}\right) \left(\frac{\ln(\Sigma_{ii}) - \ln(\Sigma_{jj})}{\Sigma_{ii} - \Sigma_{jj}}\right). \tag{A.38}$$

The first term is not a problem, and we have already seen how to handle the last term. For the middle term,

$$\frac{e^x - e^y}{x - y} = e^y \frac{e^w - 1}{w} \qquad x = y + w \tag{A.39}$$

$$= e^y \begin{cases} 1 & |w| < \epsilon \\ \mathrm{expm1}(w)/w & |w| \geq \epsilon \end{cases} \tag{A.40}$$

where in this case we have made use of the expm1 library function.

APPENDIX B

Pseudocode

Algorithm 1 Main algorithm

procedure TIME_STEP
    TRANSFER_TO_GRID
    if explicit **then**
        EXPLICIT_GRID_STEP
    else
        IMPLICIT_GRID_STEP
    TRANSFER_TO_PARTICLES
    UPDATE_PARTICLE_STATE
    PLASTICITY_HARDENING

procedure TRANSFER_TO_GRID
    for all grid nodes i **do**
        $m_i^n \leftarrow \sum_p w_{ip}^n m_p$
        $\mathbf{v}_i^n \leftarrow \frac{1}{m_i^n} \sum_p w_{ip}^n m_p \Big(\mathbf{v}_p^n + \frac{3}{h^2} \mathbf{B}_p^n (\mathbf{x}_i - \mathbf{x}_p^n) \Big)$ ▷ assuming cubic spline

procedure TRANSFER_TO_PARTICLES
    for all particles p **do**
        $\mathbf{v}_p^{n+1} \leftarrow \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}$
        $\mathbf{B}_p^{n+1} \leftarrow \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^T$

procedure UPDATE_PARTICLE_STATE
    for all particles p **do**
        $\mathbf{x}_p^{n+1} \leftarrow \sum_i w_{ip}^n (\mathbf{x}_i^n + \Delta t w_{ip}^n \overline{\mathbf{v}}_i^{n+1})$
        $\mathbf{T} \leftarrow \sum_i \overline{\mathbf{v}}_i^{n+1} (\nabla w_{ip}^n)^T$
        $\hat{\mathbf{F}}_p^{E,n+1} \leftarrow (\mathbf{I} + \Delta t \mathbf{T}) \mathbf{F}_p^{E,n}$
        $\hat{\mathbf{F}}_p^{P,n+1} \leftarrow \mathbf{F}_p^{P,n}$

Algorithm 2 Plasticity routines

procedure PLASTICITY_HARDENING
  for all particles p **do**
    $(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \text{SVD}(\hat{\mathbf{F}}_p^{E,n+1})$
    $(\mathbf{T}, \delta q) \leftarrow \text{PROJECT}(\mathbf{\Sigma}, \alpha_p^n)$
    $\mathbf{F}_p^{E,n+1} \leftarrow \mathbf{UTV}^T$
    $\mathbf{F}_p^{P,n+1} \leftarrow \mathbf{VT}^{-1}\mathbf{\Sigma}\mathbf{V}^T\hat{\mathbf{F}}_p^{P,n+1}$
    $q_p^{n+1} \leftarrow q_p^n + \delta q$
    $\phi_F \leftarrow h_0 + (h_1 q_p^{n+1} - h_3)e^{-h_2 q_p^{n+1}}$
    $\alpha_p^{n+1} \leftarrow \sqrt{\frac{2}{3}}\frac{2\sin\phi_F}{3-\sin\phi_F}$

function PROJECT($\mathbf{\Sigma}, \alpha$)
  $\boldsymbol{\epsilon} \leftarrow \ln \mathbf{\Sigma}$
  $\hat{\boldsymbol{\epsilon}} \leftarrow \boldsymbol{\epsilon} - \frac{\text{tr}(\boldsymbol{\epsilon})}{d}\mathbf{I}$
  if $\hat{\boldsymbol{\epsilon}} = \mathbf{0}$ **or** $\text{tr}(\boldsymbol{\epsilon}) > 0$ **then** ▷ Case II
    return $(\mathbf{I}, \|\boldsymbol{\epsilon}\|_F)$
  $\delta\gamma \leftarrow \|\hat{\boldsymbol{\epsilon}}\|_F + \frac{d\lambda+2\mu}{2\mu}\text{tr}(\boldsymbol{\epsilon})\alpha$
  if $\delta\gamma \leq 0$ **then** ▷ Case I
    return $(\mathbf{\Sigma}, 0)$
  $\mathbf{H} \leftarrow \boldsymbol{\epsilon} - \delta\gamma\frac{\hat{\boldsymbol{\epsilon}}}{\|\hat{\boldsymbol{\epsilon}}\|_F}$ ▷ Case III
  return $(\exp(\mathbf{H}), \delta\gamma)$

Algorithm 3 Explicit integration

procedure EXPLICIT_GRID_STEP
  $\langle \mathbf{v}_i^\star \rangle \leftarrow \langle \mathbf{v}_i^n \rangle + \text{FORCE\_INCREMENT}(\langle \mathbf{F}_p^{E,n} \rangle, 0)$
  $\langle \overline{\mathbf{v}}_i^{n+1} \rangle \leftarrow \text{GRID\_COLLISIONS}(\langle \mathbf{v}_i^\star \rangle)$
  $\langle \tilde{\mathbf{v}}_i^{n+1} \rangle \leftarrow \text{FRICTION}(\langle \overline{\mathbf{v}}_i^{n+1} \rangle, \langle \overline{\mathbf{v}}_i^{n+1} - \mathbf{v}_i^\star \rangle)$

Algorithm 4 Force grid computation

function FORCE_INCREMENT($\langle \mathbf{F}_p \rangle, b$)
  for all particles p **do**
    $(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \text{SVD}(\mathbf{F}_p)$
    if $b \neq 0$ **then**
      $(\mathbf{\Sigma}, \cdot) \leftarrow \text{PROJECT}(\mathbf{\Sigma}, \alpha_p^n)$ ▷ ignore hardening
    $\mathbf{T} \leftarrow \text{ENERGY\_DERIVATIVE}(\mathbf{\Sigma})$
    $\mathbf{A}_p \leftarrow V_p^0\mathbf{UTV}^T(\mathbf{F}_p^n)^T$
  for all grid nodes i **do**
    $\mathbf{f}_i \leftarrow -\frac{\Delta t}{m_i^n}\sum_p \mathbf{A}_p \nabla w_{ip}^n$
  return $\langle \mathbf{f}_i \rangle$

function ENERGY_DERIVATIVE($\mathbf{F}$)
  return $2\mu\mathbf{\Sigma}^{-1}\ln\mathbf{\Sigma} + \lambda\text{tr}(\ln\mathbf{\Sigma})\mathbf{\Sigma}^{-1}$

Algorithm 5 Implicit integration

procedure IMPLICIT_GRID_STEP
  COUPLED_SOLVE
  $\langle \mathbf{v}_i^\star \rangle \leftarrow \langle \mathbf{v}_i^n \rangle + \text{FORCE\_INCREMENT\_VEL}(\langle \overline{\mathbf{v}}_i^{n+1} \rangle)$
  $\langle \Delta \mathbf{v}_i \rangle \leftarrow \text{GRID\_COLLISIONS}(\langle \mathbf{v}_i^\star \rangle) - \langle \mathbf{v}_i^\star \rangle$
  $\langle \hat{\mathbf{v}}_i^{n+1} \rangle \leftarrow \text{FRICTION}(\langle \overline{\mathbf{v}}_i^{n+1} \rangle, \langle \Delta \mathbf{v}_i \rangle)$

procedure COUPLED_SOLVE
  $\langle \hat{\mathbf{v}}_i \rangle \leftarrow \langle \mathbf{v}_i^n \rangle + \text{FORCE\_INCREMENT\_VEL}(\langle \mathbf{v}_i^n \rangle)$
  return $\text{CONSTRAINED\_NEWTON}(\langle \hat{\mathbf{v}}_i \rangle)$

function FORCE_INCREMENT_VEL($\langle \mathbf{v}_i \rangle$)
  for all particles p **do**
    $\mathbf{T} \leftarrow \sum_i \mathbf{v}_i (\nabla w_{ip}^n)^T$
    $\mathbf{A}_p \leftarrow (\mathbf{I} + \Delta t \mathbf{T}) \mathbf{F}_p^{E,n}$
  return $\text{FORCE\_INCREMENT}(\langle \mathbf{A}_p \rangle, 1)$

function CONSTRAINED_NEWTON($\langle \mathbf{z}_i \rangle$)
  $\langle \mathbf{z}_i \rangle \leftarrow \text{GRID\_COLLISIONS}(\langle \mathbf{z}_i \rangle)$
  while not too many iterations **do**
    $\langle \mathbf{y}_i \rangle \leftarrow \text{NONLINEAR\_FUNCTION}(\langle \mathbf{z}_i \rangle)$
    $a \leftarrow \sqrt{\sum_i m_i \mathbf{y}_i \cdot \mathbf{y}_i}$
    if $a < \tau \Delta t$ **and** not first iteration **then**
      return $\langle \mathbf{z}_i \rangle$
    Find $\langle \Delta \mathbf{z}_i \rangle$ so $\text{TIMES\_DIFF}(\langle \Delta \mathbf{z}_i \rangle) = -\langle \mathbf{y}_i \rangle$ ▷ Solve using GMRES
    $\langle \mathbf{z}_i \rangle \leftarrow \langle \mathbf{z}_i \rangle + \langle \Delta \mathbf{z}_i \rangle$
    $\langle \mathbf{z}_i \rangle \leftarrow \text{GRID\_COLLISIONS}(\langle \mathbf{z}_i \rangle)$
  return $\langle \mathbf{z}_i \rangle$

function NONLINEAR_FUNCTION($\langle \mathbf{v}_i \rangle$)
  $\langle \hat{\mathbf{v}}_i \rangle \leftarrow \text{GRID\_COLLISIONS}(\langle \mathbf{v}_i \rangle)$
  $\langle \overline{\mathbf{v}}_i \rangle \leftarrow \langle \hat{\mathbf{v}}_i \rangle - \langle \mathbf{v}_i^n \rangle - \text{FORCE\_INCREMENT\_VEL}(\langle \hat{\mathbf{v}}_i \rangle)$
  return $\text{PRUNE\_COLLISIONS}(\langle \overline{\mathbf{v}}_i \rangle)$

Algorithm 6 Implicit integration - continued

function TIMES_DIFF($\langle \mathbf{v}_i \rangle$)
  $\langle \hat{\mathbf{v}}_i \rangle \leftarrow \langle \mathbf{v}_i \rangle$
  for all recorded sticky $(\cdot, i)$ **do**
    $\hat{\mathbf{v}}_i \leftarrow \mathbf{0}$
  for all recorded collisions $(\cdot, i, \mathbf{n})$ **do**
    $\hat{\mathbf{v}}_i \leftarrow \hat{\mathbf{v}}_i - (\hat{\mathbf{v}}_i \cdot \mathbf{n})\mathbf{n}$
  $\langle \overline{\mathbf{v}}_i \rangle \leftarrow$ HESSIAN_TIMES($\langle \hat{\mathbf{v}}_i \rangle$)
  for all grid nodes i **do**
    $\overline{\mathbf{v}}_i \leftarrow \hat{\mathbf{v}}_i + \frac{\Delta t^2}{m_i^n} \overline{\mathbf{v}}_i$
  for all recorded sticky $(\cdot, i)$ **do**
    $\overline{\mathbf{v}}_i \leftarrow \mathbf{0}$
  for all recorded collisions $(\cdot, i, \mathbf{n})$ **do**
    $\overline{\mathbf{v}}_i \leftarrow \overline{\mathbf{v}}_i - (\overline{\mathbf{v}}_i \cdot \mathbf{n})\mathbf{n}$
  return $\langle \mathbf{z}_i \rangle$

function HESSIAN_TIMES($\langle \mathbf{z}_i \rangle$)
  for all particles p **do**
    LOAD($\mathbf{U}_p, \mathbf{Q}_p, \hat{\mathbf{M}}$)
    $\mathbf{T}_1 \leftarrow \sum_i \mathbf{z}_i (\nabla w_{ip}^n)^T$
    $\mathbf{T}_2 \leftarrow \mathbf{U}_p^T \mathbf{T}_1 \mathbf{Q}_p$
    $\mathbf{T}_3 \leftarrow \hat{\mathbf{M}} : \mathbf{T}_2$ ▷ See Section A.5
    $\mathbf{A}_p = V_p^0 \mathbf{U}_p \mathbf{T}_3 \mathbf{Q}_p^T$
  for all grid nodes i **do**
    $\mathbf{y}_i \leftarrow \sum_p \mathbf{A}_p \nabla w_{ip}^n$
  return $\langle \mathbf{y}_i \rangle$

Algorithm 7 Collision and friction

function GRID_COLLISIONS($\langle \mathbf{v}_i \rangle$)
  FORGET_RECORDED_ITEMS
  $\langle \hat{\mathbf{v}}_i \rangle \leftarrow \langle \mathbf{v}_i \rangle$;
  for all collision bodies b **do**
    $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i^n + \Delta t \hat{\mathbf{v}}_i$
    if IS_STICKY(b) **and** $\phi_b(\mathbf{x}_i^n) < 0$ **then**
      RECORD_STICKY(b, i)
      $\hat{\mathbf{v}}_i \leftarrow \mathbf{v}_b(\hat{\mathbf{x}}_i)$
    else
      $\hat{\phi} \leftarrow \phi_b(\hat{\mathbf{x}}_i) - \min(\phi_b(\mathbf{x}_i^n), 0)$
      if (IS_SEPARATING(b) **and** $\hat{\phi} < 0$) **or**
          (IS_SLIPPING(b) **and** $\phi_b(\mathbf{x}_i^n) < 0$) **then**
        RECORD_COLLISION($b, i, \nabla\phi_b(\hat{\mathbf{x}}_i)$)
        $\hat{\mathbf{v}}_i \leftarrow \hat{\mathbf{v}}_i - \hat{\phi}\nabla\phi_b(\hat{\mathbf{x}}_i)/\Delta t$
  return $\langle \hat{\mathbf{v}}_i \rangle$

function PRUNE_COLLISIONS($\langle \mathbf{v}_i \rangle$)
  $\langle \hat{\mathbf{v}}_i \rangle \leftarrow \langle \mathbf{v}_i \rangle$
  for all recorded sticky $(\cdot, i)$ **do**
    $\hat{\mathbf{v}}_i \leftarrow \mathbf{0}$
  for all recorded collisions $(b, i, \mathbf{n})$ **do**
    if IS_SEPARATING(b) **and** $\hat{\mathbf{v}}_i \cdot \mathbf{n} < 0$ **then**
      FORGET_COLLISION($\cdot, i$)
    else
      $\hat{\mathbf{v}}_i \leftarrow \hat{\mathbf{v}}_i - (\hat{\mathbf{v}}_i \cdot \mathbf{n})\mathbf{n}$

function FRICTION($\langle \mathbf{v}_i \rangle, \langle \Delta\mathbf{v}_i \rangle$)
  for all recorded collisions $(b, i, \cdot, \mathbf{n}, \cdot)$ **do**
    $\mathbf{v}_t \leftarrow \overline{\mathbf{v}}_i^{n+1} - \mathbf{n}(\mathbf{n} \cdot \mathbf{v}_i)$
    $\mathbf{t} \leftarrow \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|}$
    $\mathbf{v}_i \leftarrow \mathbf{v}_i - \min(\|\mathbf{v}_t\|, \mu_b\|\Delta\mathbf{v}_i\|)\mathbf{t}$
  return $\langle \mathbf{v}_i \rangle$

Algorithm 8 Force and force derivative precompute

procedure PRECOMPUTE($\langle \mathbf{v}_i \rangle$)
  for all particles p **do**
    $\mathbf{T} \leftarrow \sum_i \mathbf{z}_i (\nabla w_{ip}^n)^T$
    $\mathbf{F}_p \leftarrow (\mathbf{I} + \Delta t \mathbf{T}) \mathbf{F}_p^{E,n}$
    $(\mathbf{U}_p, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \mathrm{SVD}(\mathbf{F}_p)$
    $\mathbf{Q}_p \leftarrow \mathbf{F}_p^{E,n} \mathbf{V}$
    Construct $\hat{\mathbf{M}}$ ▷ See Section A.5
    STORE($\mathbf{U}_p, \mathbf{Q}_p, \hat{\mathbf{M}}$)

Bibliography

Alduán, I. and Otaduy, M. (2011). SPH granular flow with friction and cohesion. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pages 25–32. 2

Alduán, I., Tena, A., and Otaduy, M. (2009). Simulation of high-resolution granular media. In *Proc Cong Español Inf Graf*. 3

Andersen, S. and Andersen, L. (2010). Modelling of landslides with the material-point method. *Computational Geosciences*, 14(1):137–147. 4

Bardenhagen, S., Brackbill, J., and Sulsky, D. (2000). The material-point method for granular materials. *Comp Meth App Mech Eng*, 187(3–4):529–541. 1

Bell, N., Yu, Y., and Mucha, P. (2005). Particle-based simulation of granular materials. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pages 77–86. 3

Bonet, J. and Wood, R. (2008). *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press. 8, 10, 15, 29, 35

Brackbill, J. (1988). The ringing instability in particle-in-cell calculations of low-speed flow. *J Comp Phys*, 75(2):469–492. 59

Brackbill, J., Kothe, D., and Ruppel, H. (1988). FLIP: A low-dissipation, PIC method for fluid flow. *Comp Phys Comm*, 48:25–38. 59

Brackbill, J. and Ruppel, H. (1986). FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *J Comp Phys*, 65:314–343. 38, 59

Chanclou, B., Luciani, A., and Habibi, A. (1996). Physical models of loose soils dynamically marked by a moving object. In *Comp Anim*, pages 27–35. 3

Chang, Y., Bao, K., Zhu, J., and Wu, E. (2012). A particle-based method for granular flow simulation. *Sci China Inf Sci*, 55(5):1062–1072. 2

Chen, P. and Wong, S. (2013). Real-time auto stylized sand art drawing. In *CAD Comp Graph*, pages 439–440. 3

Chen, W.-F. and Saleeb, A. F. (1994a). Constitutive equations for engineering materials. Volume 1: Elasticity and modeling. *Studies in Applied Mechanics*, 37. 8, 19

Chen, W.-F. and Saleeb, A. F. (1994b). Constitutive equations for engineering materials. Volume 2: Plasticity and modeling. *Studies in Applied Mechanics*, 37. 8, 19

Coetzee, C., Vermeer, P., and Basson, A. (2005). The modelling of anchors using the material point method. *International Journal for Numerical and Analytical Methods in Geomechanics*, 29(9):879–895. 4

Gast, T., Schroeder, C., Stomakhin, A., Jiang, C., and Teran, J. (2015). Optimization integrator for large time steps. *IEEE Trans Vis Comp Graph*, 21(10):1103–1115. 63

Gast, T. F. and Schroeder, C. (2014). Optimization integrator for large time steps. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*. 63

Gonzalez, O. and Stuart, A. (2008). *A first course in continuum mechanics*. Cambridge University Press. 8, 12, 28

Gritton, C. E. (2014). *Ringing Instabilities in Particle Methods*. PhD thesis, The University of Utah. 59

Guilkey, J. E., Hoying, J. B., and Weiss, J. A. (2006). Computational modeling of multicellular constructs with the material point method. *Journal of Biomechanics*, 39(11):2074–2086. 4

Harlow, F. (1964). The particle-in-cell method for numerical solution of problems in fluid dynamics. *Meth Comp Phys*, 3:319–343. 38

Harlow, F. and Welch, E. (1965). Numerical calculation of time dependent viscous flow of fluid with a free surface. *Phys Fluid*, 8(12):2182–2189. 38

Hashiguchi, K. (2009). *Elastoplasticity Theory*. Springer-Verlag Berlin Heidelberg. 15

Hegemann, J., Jiang, C., Schroeder, C., and Teran, J. M. (2013). A level set method for ductile fracture. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, pages 193–201. 4

Hill, R. (1948). A variational principle of maximum plastic work in classical plasticity. *The Quarterly Journal of Mechanics and Applied Mathematics*, 1(1):18–28. 29

Hill, R. (1998). *The mathematical theory of plasticity*, volume 11. Oxford university press. 29

Ihmsen, M., Wahl, A., and Teschner, M. (2013). A lagrangian framework for simulating granular material with high detail. *Comp Graph*, 37(7):800–808. 2

Irgens, F. (2008). *Continuum mechanics*. Springer Science & Business Media. 8, 22

Jaeger, H., Nagel, S., and Behringer, R. (1996). Granular solids, liquids, and gases. *Rev Mod Phys*, 68:1259–1273. 1

Jiang, C. (2015). *The material point method for the physics-based simulation of solids and fluids*. PhD thesis, University of California, Los Angeles. 4

Jiang, C., Schroeder, C., Selle, A., Teran, J., and Stomakhin, A. (2015). The affine particle-in-cell method. *ACM Trans Graph*, 34(4):51:1–51:10. 59

Jiang, C., Schroeder, C., and Teran, J. (2016). An angular momentum conserving Affine-Particle-In-Cell method. *ArXiv E-Prints*. 59

Jiang, C., Schroeder, C., Teran, J., Stomakhin, A., and Selle, A. (2016). The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16. ACM. 4

Kézdi, Á. and Rétháti, L. (1974). *Handbook of soil mechanics*, volume 1. Elsevier Amsterdam. 70

Klár, G. (2014). Speculative atomics: Case-study of the GPU optimization of the material point method for graphics. In *Presentation at GPU Technology Conference*. 68, 79

Klár, G., Gast, T., Pradhana, A., Fu, C., Schroeder, C., Jiang, C., and Teran, J. (2016a). Drucker-prager elastoplasticity for sand animation. *ACM Trans Graph*, 35(4). 5

Klár, G., Gast, T., Pradhana, A., Fu, C., Schroeder, C., Jiang, C., and Teran, J. (2016b). Drucker-prager elastoplasticity for sand animation: Supplementary technical document. *ACM Trans Graph*. 5

Kolymbas, D. (1999). *Introduction to Hypoplasticity: Advances in Geotechnical Engineering and Tunnelling 1*, volume 1. CRC Press. 3

Kolymbas, D. (2000). *Constitutive Modelling of Granular Materials*. Springer Science & Business Media. 3

Lajeunesse, E., Mangeney-Castelnau, A., and Vilotte, J. (2004). Spreading of a granular mass on a horizontal plane. *Physics of Fluids (1994-present)*, 16(7):2371–2381. 69

Lenaerts, T. and Dutré, P. (2009). Mixing fluids and granular materials. *Comp Graph Forum*, 28(2):213–218. 2

Li, X. and Moshell, J. (1993). Modeling soil: Realtime dynamic models for soil slippage and manipulation. In *Proc SIGGRAPH*, pages 361–368. 3

Luciani, A., Habibi, A., and Manzotti, E. (1995). A multi-scale physical model of granular materials. In *Proc Graph Int*, pages 136–146. 3

Macklin, M., Müller, M., Chentanez, N., and Kim, T. (2014). Unified particle physics for real-time applications. *ACM Trans Graph*, 33(4):153:1–153:12. 3

Mast, C. (2013). *Modeling landslide-induced flow interactions with structures using the Material Point Method*. PhD thesis, University of Washington, Department of Civil and Environmental Engineering. 1, 22, 29

Mast, C., Arduino, P., Mackenzie-Helnwein, P., and Miller, R. (2014). Simulating granular column collapse using the material point method. *Acta Geotech*, 10(1):101–116. 1, 58

Mazhar, H., Heyn, T., Negrut, D., and Tasora, A. (2015). Using Nesterov's method to accelerate multibody dynamics with friction and contact. *ACM Trans Graph*, 34(3):32:1–32:14. 3

Mehta, A. and Barker, G. (1994). The dynamics of sand. *Reports on Progress in Physics*, 57(4):383. 1

Milenkovic, V. (1996). Position-based physics: Simulating the motion of many highly interacting spheres and polyhedra. In *Proc SIGGRAPH*, pages 129–136. 3

Miller, G. and Pearce, A. (1989). Globular dynamics: A connected particle system for animating viscous fluids. *Comp Graph*, 13(3):305–309. 3

Museth, K. (2013). Vdb: High-resolution sparse volumes with dynamic topology. *ACM Trans Graph*, 32(3):27:1–27:22. 79

Narain, R., Golas, A., and Lin, M. (2010). Free-flowing granular materials with two-way solid coupling. *ACM Trans Graph*, 29(6):173:1–173:10. 2

Nkulikiyimfura, D., Kim, J., and Kim, H. (2012). A real-time sand simulation using a GPU. In *Comp Tech Inf Man*, volume 1, pages 495–498. 2

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer series in operations research and financial engineering. Springer. 62

Onoue, K. and Nishita, T. (2003). Virtual sandbox. In *Proc Pac Conf Comp Graph App*, pages 252–262. 3

Pla-Castells, M., Garcia-Fernandez, I., and Martinez, R. (2006). Interactive terrain simulation and force distribution models in sand piles. In *Cellular Automata*, volume 4173 of *Lecture Notes Comp Sci*, pages 392–401. Springer. 3

Ram, D., Gast, T., Jiang, C., Schroeder, C., Stomakhin, A., Teran, J., and Kavehpour, P. (2015). A material point method for viscoelastic fluids, foams and sponges. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pages 157–163. 4

Sikora, Z. (1992). *Hypoplastic flow of granular materials: A numerical approach.* Institutes fur Bodenmechanik und Felsmechanik der Universitat Fridericiana in Karlsruhe. 3

Simo, J. and Meschke, G. (1993). A new class of algorithms for classical plasticity extended to finite strains. application to geomaterials. *Comput Mech*, 11(4):253–278. 50

Steffen, M., Kirby, R. M., and Berzins, M. (2008). Analysis and reduction of quadrature errors in the material point method (MPM). *Int J Numer Meth Eng*, 76(6):922–948. 44

Stomakhin, A., Howes, R., Schroeder, C., and Teran, J. (2012). Energetically consistent invertible elasticity. In *Proc Symp Comp Anim*, pages 25–32. 88

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., and Selle, A. (2013). A material point method for snow simulation. *ACM Trans Graph*, 32(4):102:1–102:10. 4, 38

Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., and Selle, A. (2014). Augmented MPM for phase-change and varied materials. *ACM Trans Graph*, 33(4):138:1–138:11. 4

Sulsky, D., Chen, Z., and Schreyer, H. L. (1994). A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1):179–196. 3, 38

Sulsky, D., Schreyer, H., Peterson, K., Kwok, R., and Coon, M. (2007). Using the material-point method to model sea ice dynamics. *Journal of Geophysical Research: Oceans*, 112(C2). 4

Sulsky, D., Zhou, S., and Schreyer, H. (1995). Application of a particle-in-cell method to solid mechanics. *Comp Phys Comm*, 87(1):236–252. 3, 38

Sumner, R., O'Brien, J., and Hodgins, J. (1999). Animating sand, mud and snow. *Comp Graph Forum*, 18(1):17–26. 3

Terzopoulos, D. and Fleischer, K. (1988a). Deformable models. *The Vis Comp*, 4(6):306–331. 2

Terzopoulos, D. and Fleischer, K. (1988b). Modeling inelastic deformation: Viscolelasticity, plasticity, fracture. *SIGGRAPH Comp Graph*, 22(4):269–278. 2

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. *SIGGRAPH Comp Graph*, 21:205–214. 2

Więckowski, Z. (2004). The material point method in large strain engineering problems. *Computer Methods in Applied Mechanics and Engineering*, 193(39):4417–4438. 4

Yasuda, R., Harada, T., and Kawaguchi, Y. (2008). Real-time simulation of granular materials using graphics hardware. In *Comp Graph Imag Vis*, pages 28–31. 3

York, A. R., Sulsky, D., and Schreyer, H. L. (1999). The material point method for simulation of thin membranes. *International Journal for Numerical Methods in Engineering*, 44(10):1429–1456. 4

York, A. R., Sulsky, D., and Schreyer, H. L. (2000). Fluid–membrane interaction based on the material point method. *International Journal for Numerical Methods in Engineering*, 48(6):901–924. 4

Yoshioka, N. (2003). A sandpile experiment and its implications for self-organized criticality and characteristic earthquake. *Earth, Planets and Space*, 55(6):283–289. 69

Yu, H.-S. (2007). *Plasticity and geotechnics*, volume 13. Springer Science & Business Media. 8

Yue, Y., Smith, B., Batty, C., Zheng, C., and Grinspun, E. (2015). Continuum foam: A material point method for shear-dependent flows. *ACM Trans Graph*, 34(5):160:1–160:20. 4

Zhou, S., Stormont, J., and Chen, Z. (1999). Simulation of geomembrane response to settlement in landfills by using the material point method. *International Journal for Numerical and Analytical Methods in Geomechanics*, 23(15):1977–1994. 3

Zhu, Y. and Bridson, R. (2005). Animating sand as a fluid. *ACM Trans Graph*, 24(3):965–972. 2