Numerical and Data Analysis of a Portable Free Fall Penetrometer

Jonathan J. Moore

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Civil Engineering

Alba Yerro-Colom, Co-chair

Nina Stark, Co-chair

Kyle Strom

Januray 27th, 2025

Blacksburg, Virginia

Keywords: Portable Free Fall Penetrometer, Numerical Analysis, Material Point Method

Copyright 2025, Jonathan J. Moore

Numerical and Data Analysis of a Portable Free Fall Penetrometer

Jonathan J. Moore

ABSTRACT

Coastal environments are among the most economically and environmentally significant regions on Earth. However, rising sea levels and increasingly frequent storms driven by climate change pose growing risks to these critical zones. Understanding and predicting the evolution of coastal environments requires advanced models and high-resolution geotechnical data to characterize the mechanical behavior of coastal soils. Portable free-fall penetrometers (PFFPs) are widely used for this purpose, offering a rapid and efficient means of collecting in situ soil data. Despite their utility, the interpretation of PFFP data remains uncertain due to the empirical nature of existing methods used to infer soil properties from impact acceleration measurements. This research aims to improve the reliability of PFFP-based soil characterization by advancing both numerical modeling and data processing techniques. To this end, a two-pronged approach was taken. First, efforts were made to refine and streamline numerical modeling techniques to work towards the creation of a digital twin of BlueDrop PFFP impacting into the soil. This included identifying the current limitations of the MPM framework to simulate PFFP impact and addressing some of these limitations. In particular, this thesis focuses on the mitigation of volumetric locking by means of the implementation of the B-Bar algorithm in quadrilateral elements and the availability of strain-rate advanced consecutive models by testing the accuracy and efficiency of different stress integration algorithms. In addition, a Python library was also developed to streamline the testing of soil constitutive models using the IncrementalDriver software. Second, the processing of BlueDrop field data was centralized, standardized, and automated through the development of a Python library integrated with an SQLite database. This ensures consistency and accessibility of PFFP datasets for broader scientific and engineering applications.

By advancing data processing methodologies and improving numerical modeling capabilities, this research contributes to a more rigorous framework for interpreting PFFP measurements and understanding soil behavior during impact. These developments support broader efforts to enhance geotechnical modeling of coastal systems, ultimately aiding in the prediction and management of environmental changes affecting these vulnerable regions.

Numerical and Data Analysis of a Portable Free Fall Penetrometer

Jonathan J. Moore

GENERAL AUDIENCE ABSTRACT

Climate change is intensifying the impacts of storms and flooding on coastal zones, posing a growing threat to the substantial portion of the global population and economically critical assets located in these areas. To predict how coastal environments will evolve and to engineer solutions that mitigate disaster impacts, more accurate models, advanced modeling techniques, and high-quality data are essential.

Portable free-fall penetrometers (PFFPs) offer a promising tool for assessing soil strength in coastal environments. These handheld devices, tethered to ropes, are dropped until they are stopped by the beach or seabed. However, a robust framework is required to more accurately process data from PFFPs and to better understand the interactions between the PFFP and the soil during impact. This thesis introduces physics-based computational models and Python code designed to streamline data processing and improve our understanding of these interactions, contributing to the development of more reliable tools for coastal engineering and environmental analysis.

Dedication

To my mother, who continually encouraged and talked with me; to my sisters, for their constant support; and to my grandparents, who financially and emotionally made my journey possible. This work is dedicated to you. Thank you.

Acknowledgments

During the course of my Master's I've had the great opportunity to work with the research group of all three of my advisors, so I have a lot of thanking and acknowledging to do. First, I'd like to express my deepest gratitude to all members of my committee, Dr. Yerro-Colom, Dr. Nina Stark, and Dr. Kyle Strom. Dr. Yerro, you were the first to take the risk of working with me. You have been invaluable in inspiring me to push on and continue doing research. Dr. Stark, thank you for facilitating and informing me of so many opportunities. My, somewhat, well roundness as researcher has heavily been supported by you. Dr. Strom, thank you for letting me talk your ear off and the opportunity to sit in your lab. Now to the groups underlying each professor. To the VT GeoMPM lab, thank you for your encouragement and insightful questions. A special thank you to Abdel Alsardi and Varsha Swamy, for your discussions and help. To the Coastal & Marine Geotechnics group, it was great doing fieldwork with you. A special thank you to Nick Brilli and Matthew Florence for being my introduction to field work and instrumentation. Finally, to the Baker Environmental Hydraulics Lab, thank you for letting me be your noisy upstairs neighbor and for your help. A special thank you to Ted Balabanski, Sam Kraus, and Brandon Dillon. Ted, thank you for showing me how to understand and use the tools available. Sam, thank you for your questions and for always being there for me (and paying rent each month). Brandon, you showed me that with imagination and pure will power, truly anything is possible. To Dr. Luis Zambrano-Cruzzatty, my thesis is directly built on your work, without your work my work would not have been possible. To anyone I have forgotten, I apologize. All of you have provided invaluable gifts to me. Thank you.

Contents

List of Figures

List of Tables

List of Symbols and Abbreviations

σ_v	Von-Mises Stress
J	Square root of J_2. Sometimes referred to as Deviatoric stress. It is not the same as J_2. See the appendix for more information.
J_2	Offically the second invariant of the deviatoric stress tensor but sometimes (confusingly) referred to as the deviatoric stress. It is not the same as J see the appendix for more info
p	Scalar multiple of the first invariant of the stress tensor, called the mean stress
q	Deviator or Deviatoric stress invariant. The square root of J_2 scaled by $\sqrt{3}$.
$\delta\epsilon_q^p$	Increment of plastic deviatoric strain
$\delta\epsilon_v^p$	Increment of plastic volumetric strain
ε_q	Equivalent Strain, also sometimes called the deviatoric strain
$d\boldsymbol{\varepsilon}^e$	Elastic increment of strain
$d\boldsymbol{\varepsilon}^p$	Plastic increment of strain
ε_v	Volmetric strain invariant
g	Unit of Standard Earth gravitation acceleration ($9.81 m/s^2$)
$d\boldsymbol{\varepsilon}$	Total increment of strain
BlueDrop	Product name for a Portable Free Fall Penetrometer

BlueDrop Analysis Library Python library developed as part of this Master's Thesis to process BlueDrop PFFP data

FEM Finite Element Method

Impact End Index Array index that corresponds to when the PFFP comes to rest after impacting the seabed

Impact Start Index Array index that corresponds to when the PFFP starts to impact into the seabed

Jerk Time derivative of acceleration

MC Mohr-Coulomb

MCSS Mohr-Coulomb Strain Softening

MPM Material Point Method - Particle based numerical technique that is based on a Galerkin formulation to simulate large deformation continuum based problems

OOP Object Orientated Programming

PFFP Portable Free Fall Penetrometer

q Deviatoric stress

Release Index Array index that corresponds to the release of the PFFP

SPH Smooth Particle Hydrodynamics - Particle based (completely mesh-free) numerical technique to simulate large deformation continuum based problems

Chapter 1

Introduction

1.1 Motivation

Coastal environments are the interface for many Earth processes. Climate change is expected to increase the dynamics of coastal zones, increase sea level, and make storms more powerful and frequent [Perry et al., 2018, Lakshmi, 2021, Harley et al., 2006]. Furthermore, climate change is expected to make current systems more chaotic, meaning that traditional data and decision-making procedures may not be as valuable [Harley et al., 2006]. Efforts to reduce the further worsening of climate change are promoting a transition to green energy systems, including offshore wind and wave farms [Pérez-Collazo et al., 2015]. Therefore, to maintain the stability of coastal environments and protect engineered structures, there is a need to integrate high temporal and spatial resolution data with more advanced multiphysics and multiscale models.

Soil provides habitat and nutrients for coastal life, supports foundations and anchoring for coastal structures, and forms flood-reducing dunes [Stark, 2011, Snelgrove, 1997, Zambrano-Cruzatty, 2021]. Determining the mechanical properties of soil in coastal environments can be challenging; as there can be high spatial and temporal variability [Stark, 2011, Jaber and Stark, 2023]. Salt water damages instrumentation, waves prevent or increase the cost of field deployments, and storms destroy or displace instruments [Stark, 2011]. Remote sensing circumvents some of these issues, but still requires ground truthing [Paprocki et al., 2021].

One of the main methods to ground truth the mechanical properties of soil is through soil sample collection and laboratory testing. Sample collection faces the same challenges as listed above, but has the additional constraint that disturbances should be minimized during collection and transport to the laboratory [Stark et al., 2014]. The mechanical response of soil is highly dependent on its underlying particle network, and if that network is disturbed, the laboratory results may not be representative of field conditions [Lambe and Whitman,

1969]. Another method of ground truthing is simply testing the soil as is in the field (aka. in situ testing). Free fall penetrometers (FFPs) are a method to do in-situ testing in coastal and marine environments [Stark et al., 2017, Albatal, 2018]. However, the impact of FFP on multiphase soil at a high strain rate (HSR) is not fully understood. Therefore, this work builds on the framework and subsequent work done by Zambrano-Cruzatty and Yerro [2020], Zambrano-Cruzatty [2021], Yalcin [2021], using numerical modeling and data analysis to better understand impact physics.

1.2 In-Situ Testing and Portable Free Fall Penetrometers (PFFPs)

Free Fall Penetrometers (FFPs) fall under their own weight until they impact into the soil layer. This can result in the FFPs impacting the seabed or beach at a high strain rate (HSR). FFPs can vary widely in size and weight (Stark [2011], Rong et al. [2024]). A subset of FFPs, that push towards optimizing ease and rapidness of deployment, are Portable Free Fall Penetrometers (PFFPs) [Stark, 2011]. One of the most used PFFPs is the BlueDrop, Figure 1.1 [Stark, 2011, Brilli, 2023]. Due to the availability of data, the focus of this thesis will be on the BlueDrop.

1.2.1 BlueDrop Data Products

The BlueDrop, produced by BlueCDesigns, using the steel cone tip, has a mass of 7.71 kg (m_p), is 0.632 meter long and has a volume of 0.002 473 m^3 (V_p).

The deployment consists of typing a rope to the tail of the instrument, allowing the instrument to fall until it impacts the seabed and comes to rest; then, the instrument is retrieved using the attached rope, and the process is repeated. Impact velocities generally

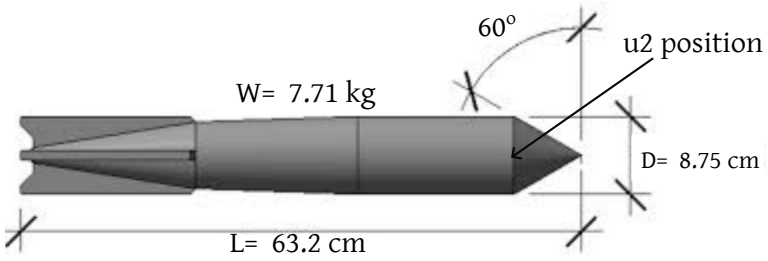


Figure 1.1: Representation of the BlueDrop PFFP. Taken from Zambrano-Cruzatty and Yerro [2020], who modified it from BlueCDesigns

range from $1\,\mathrm{m\,s^{-1}}$ to $6\,\mathrm{m\,s^{-1}}$. The BlueDrop output is a continuous series of binary files that each contain one minute of data from a set of accelerometers and a pressure sensor. All sensors record at 2000 Hz. The set of accelerometers can be broken into two groups. The first group measures acceleration in the vertical (z) direction, sensor ranges are $\pm 1.7g$, $\pm 18g$, $\pm 50g$, $\pm 200g$, and $\pm 250g$. The unit of g stands for one unit of standard Earth gravitational acceleration ($9.81 m/s^2$). The vertical acceleration measurements are integrated in time (t) to obtain the PFFP's velocity (v_z) and displacement (x_z). The second group of accelerometer sensors are used as tilt sensors, determining if the BlueDrop tilts during the fall and impacts at an angle that is beyond a set tolerance. The tilt sensors are mounted in the horizontal ($x-y$) plane, one along the $x-axis$ and the other along the $y-axis$. The range of both tilt sensors is $\pm 55g$. The onboard accelerometers measure in an inertial reference frame. Therefore, when the PFFP is at a constant velocity and is orientated vertically, the vertical accelerometers measure $1g$. The pressure sensor is mounted at the $u2$ position (the base of the tip of the BlueDrop) [Stark, 2011]. While the BlueDrop falls through the fluid (air or water) column, the measured pressure is impacted by Bernoulli's principle. For information on correcting the pressure measurement, see Mumtaz [2018]. In this thesis, drops through the water column are called water drops, and drops through air are called air drops. Figure 1.2 shows the sensor output for an example water drop taken in the York River, Virginia.

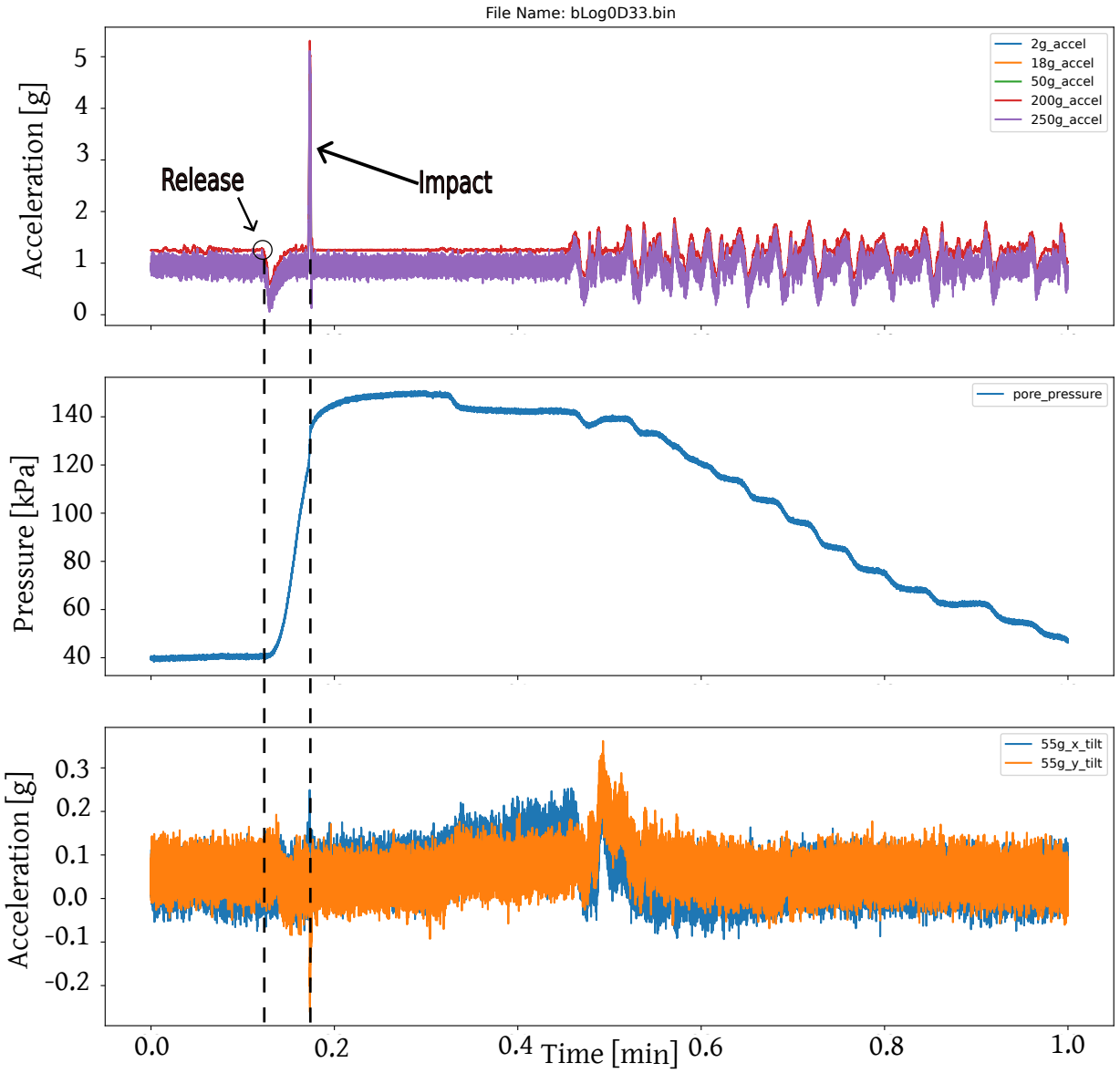


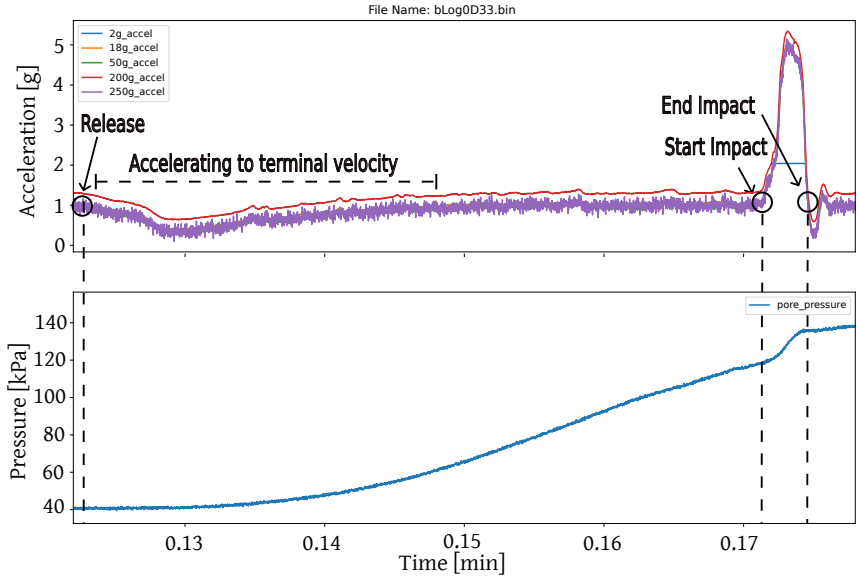
Figure 1.2: Plot of the sensor output from a water drop taken from the York River, Virginia.

Top Panel: Plot of the vertical acceleration versus time. The sensors take the direction opposite of gravity to be positive. *Release* corresponds to when the rope attached to the BlueDrop is released, and the instrument starts to fall. *Impact* corresponds to when the BlueDrop impacts into the seabed. The offset of the red line (200g sensor) and the purple line (250g sensor) is because of the calibration drift of the 200g sensor. The rest of the accelerometer measurements are hidden by the purple line. **Middle Panel:** Plot of the measured pressure versus time. **Bottom Panel:** Plot of the two tilt sensors.

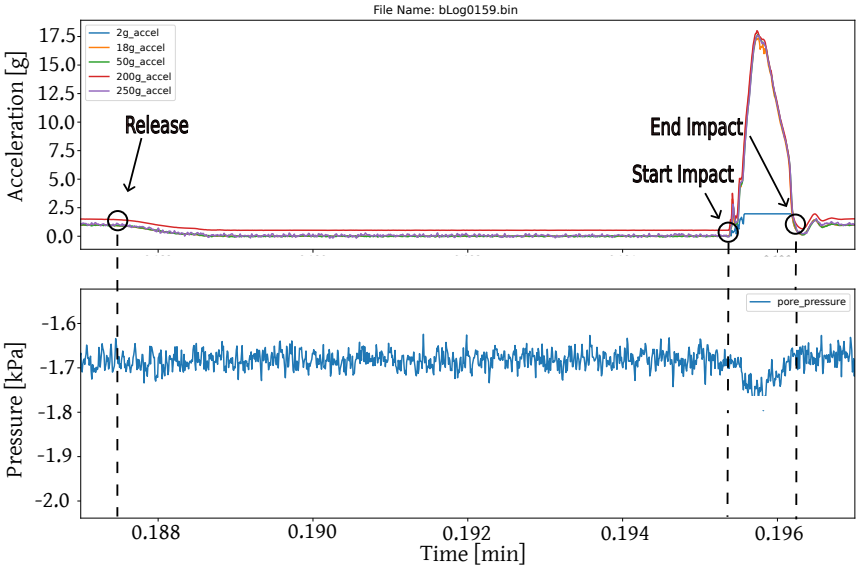
The primary distinction between air and water drops lies in the magnitude of the drag force acting on the system. Drag forces, which influence both the PFFP and the attached rope, are significantly higher in water than in air for a given velocity. Consequently, as long as the length of the rope submerged in the fluid continues to increase, the PFFP system does not reach terminal velocity. However, for many water depths, the vertical accelerometer records an approximate pre-impact acceleration of $1g$.

In contrast, many air drops are performed from a height of only a few feet above the ground, causing the accelerometer to record approximately $0g$ just before impact. Since a sensor measurement of $0g$ corresponds to an acceleration of $9.81\,\mathrm{m\,s^{-2}}$ for an observer fixed to the Earth, this implies that the BlueDrop continues to accelerate in the direction of gravity during the initial phase of penetration into the seabed. These characteristic pre-impact accelerations serve as practical guidelines that simplify data processing. However, their applicability depends on the release height and the medium in which the PFFP is deployed, and they do not always hold under all conditions.

Figure 1.3 provides a detailed comparison of these two scenarios. Figure 1.3a illustrates a water drop where the depth is sufficient for the velocity to approximately peak before impact, while Figure 1.3b depicts an air drop from a height of only a few feet above the ground. The presented drops are ideal PFFP drops. They impacted at a negligibly small angle from the vertical. If the PFFP falls at more of an angle, the vertical accelerometer will only measure a part of the gravitational acceleration. The tilt sensors, an example of which is in the bottom panel of Figure 1.2, would measure the other part of the gravitational acceleration. If the tilt is too great, the drop needs to be discarded.



(a) Example of the vertical acceleration and pressure response for a water drop. Start with the release of the BlueDrop until the end of impact. This is a zoomed in perspective of the drop in Figure 1.2



(b) Example of the vertical acceleration and pressure response for an air drop from the release of the BlueDrop until the end of impact.

Figure 1.3: Comparison of an example water drop (top) and an air drop (bottom).

1.2.2 Measuring Soil Strength with a BlueDrop

The deceleration that the BlueDrop experiences as it impacts into the soil is a function of strain rate effects. Therefore, an empirical model is used to convert the dynamic force from the soil transmitted to the PFFP during impact ($F_{dyn,Br}$) to an equivalent quasi-static force ($F_{qs,Br}$). $F_{dyn,Br}$ is calculated using equation 1.1, where a_s is the acceleration measured during impact. See Appendix D for a derivation.

$$F_{dyn,Br} = m_p(a_s - g) + m_p g \tag{1.1}$$

Two of the most common classes of methods for accounting for strain rate effects will be described. The first class of methods assumes a relationship between $F_{qs,Br}$ and $F_{dyn,Br}$ through a strain rate function, denoted as $f_s = f_s(v_z, v_{ref})$, where v_z represents the vertical velocity of the BlueDrop, and v_{ref} is a reference velocity, typically chosen as $2\,\mathrm{cm\,s^{-1}}$ [Brilli, 2023]. The assumed formulation of this approach is presented in Equation 1.2.

$$F_{qs,Br} = \frac{F_{dyn,Br}}{f_s} \tag{1.2}$$

$$\tag{1.3}$$

The contact area between the BlueDrop and the soil (A) can be used to convert the forces (F) to stresses (q), where A is a function of the depth of penetration, and $qsbc$ is the quasi-static bearing capacity.

$$\begin{aligned} \frac{1}{A}(F_{qs,Br}) &= \frac{1}{A}\left(\frac{F_{dyn,Br}}{f_s}\right) \\ qsbc &= \frac{q_{dyn}}{f_s} \end{aligned} \tag{1.4}$$

The advantage of converting to $qsbc$ is that it represents the strength of the soil as a cone penetrometer (CPT) would measure it, and the large set of well-tested correlations between CPT measurements and geotechnical properties can be used [Lunne, 2010]. Different forms and values have been proposed for f_s and corresponding empirical factors depending on the soil type and saturation [Stark, 2011, Jaber and Stark, 2023, Brilli, 2023]. One of the most common methods to calculate f_s is

$$f_s = 1 + K log_{10}\left(\frac{v_z}{v_{ref}}\right) \tag{1.5}$$

where, K is a soil dependent parameter that can range from 0.3 to 1.5 [Albatal, 2018, Zambrano-Cruzatty and Yerro, 2020]. Equation 1.5 has been used for both cohesive and non-cohesive soils [Albatal, 2018]. Another commonly used formulation is

$$f_s = \left(\frac{v_z}{v_{ref}}\right)^{\beta} \tag{1.6}$$

where, β is a soil dependent strain rate parameter [Jaber and Stark, 2023].

The second class of methods was presented by White et al. [2018]. It is designed for penetration into non-cohesive saturated sand. This method uses the accelerometer and pressure sensor data. A general overview of the process is given herein, but for further details, refer to White et al. [2018]. Following the impact, the PFFP is left in the soil and the pressure is allowed to dissipate to at least 50% of its initial value. This data is used to calculate a consolidation coefficient (c_h). The consolidation coefficient is used along with the PFFP's diameter (D) to define a non-dimensional velocity (V)

$$V = \frac{v_z D}{c_h} \tag{1.7}$$

Then, a set of "backbone curves" are defined,

$$q_{dyn} = q_{ud} + (q_d - q_{ud}) \left(\frac{1}{1 + \frac{V}{V_{50}}} \right) \tag{1.8}$$

where, q_{ud} is the soil's resistance under undrained loading, q_d is the soil's resistance under drained loading, and V_{50} is taken to be 1. Effectively, the backbone curves define a way to interpolate between q_{ud} and q_d as a function of the penetration rate V.

White et al. [2018] uses common empirical geotechnical relations to pose both q_{ud} and q_d as functions of relative density (I_d). Combining all the relationships White et al. [2018] used q_d is

$$q_d = C_0 p_0^{'C_1} \exp(C_2 I_D) \tag{1.9}$$

where, $C_0 = 300$, $C_1 = 0.46$, and $C_2 = 2.96$ are empirical coefficients; p'_0 is the in-situ mean effective stress. p'_0 must be in kPa and can be calculated using an assumed or measured soil unit weight and at-rest earth pressure coefficient (K_0). Similarly, q_{ud} is calculated as

$$q_{ud} = \frac{N_{kt}}{2} \exp \left(Q - \frac{1}{I_D} \right) \left(\frac{6 \sin \phi'_{cv}}{3 - sin\phi'_{cv}} \right) \tag{1.10}$$

where, N_{kt} is a bearing factor and assumed to be 12, Q is a crushing strength parameter and is assumed to be 10, ϕ'_{cv} is the friction angle at constant volume and is assumed to be 32°.

Equations 1.9 and 1.10 can be plugged into equation 1.8 and as q_{dyn} is a measured quantity, an iterative procedure can be used to calculate I_D at each depth. Once I_D is known, the strength a CPT would measure ($qsbc$) can be predicted. This is done by plugging the calculated I_D and the V calculated with the standard $20\,\mathrm{mm\,s^{-1}}$ loading rate for a CPT, into equation 1.8.

1.3 Numerical modeling of Free Fall Penetrometers

The use of empirical equations to convert free fall penetrometers (FFP) measurements to quasi-static values has prompted multiple authors to use numerical modeling techniques to investigate, constrain, and theoretically validate the empirical relations. However, numerically modeling FFPs is challenging. The numerical method must be able to handle rapid impact, incorporate an advanced soil constitutive model, track stress and state variables, handle large deformations, include soil-PFFP contact with interface friction, solve multiphase interactions (water-soil or water-soil-air), and be computationally tractable.

Historically, this problem was approached with well-known mesh-based Lagrangian methods, such as Lagrangian Finite Difference method (FDM) and Lagrangian Finite Element method (FEM) [Abelev et al., 2009a,b]. However, it is well known that standard Lagrangian methods suffer from severe mesh distortion and likely model collapse when modeling large deformation problems. Mesh distortion occurs because the deformation of the material is accounted for by the mesh deforming. This has the advantage that state variables can be tracked, but for large deformation problems, elements can become too distorted causing erroneous results or full model failure.

To remove the need for remeshing, Abelev et al. [2009a] modeled an FFP impact using a *FLAC3D*'s quasi-static implementation of the FDM. The soil was modeled using undrained Mohr-Coulomb. However, using a quasi-static approach for a problem as dynamic as FFP impact is highly limited. The stress and strain fields are not realistic. Additionally, the FFP was modeled as a loading condition so interface friction could not be considered.

Abelev et al. [2009b] used *ABAQUS*'s implementation of the arbitrary Lagrangian-Eulerian Finite Element method (ALE). ALE allows for automatic determination when the mesh is failing and remeshing to prevent numerical collapse from occurring. Impact in a cohesive soil was considered. The soil was modeled as undrained using the Von-Mises failure

criteria. The FFP was modeled as an applied boundary condition, and interface friction was not considered.

Nazem et al. [2012] was a substantial advancement in representing a more physically realistic FFP impact. An in-house ALE model was used. Impact was considered onto a cohesive soil using an undrained strain rate-dependent model. Interface friction was accounted for between the FFP and the soil.

Although ALE mitigates model collapse due to mesh distortion, it introduces new challenges. Remeshing necessitates stress and strain variables be remapped to each new mesh. This can damage the accuracy of the solution, especially when using advanced constitutive behaviors which are highly reliant on accurate tracking of state variables [Zambrano-Cruzatty, 2021]. Additionally, remeshing is computationally expensive, which can severely limit numerical models where non-symmetric 3D effects have to be accounted for.

To remove the need for remeshing and make modeling of large deformation problems easier, while still using a Lagrangian perspective to track state variables, a multitude of particle-based methods have been proposed. Two of the most popular particle methods are the material point method (MPM) and the smooth particle hydrodynamics (SPH).

MPM was introduced by Sulsky et al. [1994], to model history-dependent materials undergoing large deformations[1]. It is a descendant of the Fluid Implicit Particle (FLIP) and the Particle In Cell method (PIC) [Sulsky et al., 1994]. Standard MPM uses an updated Lagrangian-Eulerian framework. This decouples the discretization of the material and the spatial region. Allowing for history dependent materials to be modeled. To discretize the material and spatial geometry, a Galerkin formulation is used. Resulting in a set of particles or integration points, with Dirac delta support, and a mesh composed of elements. Challenges faced by MPM are detailed in section 1.3.2.

SPH was introduced by Lucy [1977] and Gingold and Monaghan [1977]. SPH is truly

[1]Following the convention in Bonus [2023], the original variation of MPM is called standard MPM

meshless and models the continuum as particles with a volume of influence. During each time step, a nearest neighbor search must be done to determine which particles are influencing each other. This process is computationally expensive. Additionally, the lack of a mesh or fixed locations can make prescribing boundary conditions complicated [Bui and Nguyen, 2021].

Previous numerical analyses that used SPH or MPM to model free fall penetrometers include Zambrano-Cruzatty and Yerro [2020], Yalcin [2021], Li [2019], Mohapatra et al. [2024a] and Mohapatra et al. [2024b]. Zambrano-Cruzatty and Yerro [2020], Yalcin [2021], and Li [2019] modeled the BlueDrop impacting into dry, saturated, and submerged saturated sand, respectively. They compared their numerical results to the BlueDrop calibration chamber experiments collected by Albatal [2018]. Albatal [2018] ran two sets of calibration chamber experiments: (1) impact of the BlueDrop onto dry sand with a depth of 20 cm and (2) impact of the BlueDrop into 40 cm of water on top of 20 cm of saturated sand. Zambrano-Cruzatty and Yerro [2020] extended and used Anura3D's implementation of the material point method (MPM) and a strain-softening Mohr-Coulomb model (*SSMC*) for the soil and compared the results to experiment (1). Yalcin [2021] used Anura3D and *SSMC* to model the impact of the BlueDrop into saturated sand, with no water layer, and made an initial comparison between the numerical model and experiment (2). Yalcin [2021] used a fully coupled scheme for the interaction of the soil and pore water. Yalcin [2021] and Zambrano-Cruzatty and Yerro [2020] took advantage of symmetry of the problem and used a 3D-axisymmetric model with tetrahedral elements. Li [2019] used *DualSPHysics'* implementation of the SPH method and a visco-elastoplastic model and compared against experiment (2). Li [2019] used elastic parameters much higher than those used by Zambrano-Cruzatty and Yerro [2020] and Yalcin et al. [2023]. Li [2019] used a full 3D-cylindrical model with a fully coupled scheme for the soil and water.

Mohapatra et al. [2024a] and Mohapatra et al. [2024b] modeled the impact of free

fall cone penetrometers (FF-CPTs) into soft and sensitive marine clay. Both papers used *Uintah's* Generalized Interpolation MPM (GiMP), assumed the clay was undrained, and used a Tresca model with strain softening and rate dependence for the soil. Mohapatra et al. [2024a] modeled the impact of an in-situ scale free fall cone penetrometer (FF-CPT). Mohapatra et al. [2024a] modeled the impact of a laboratory scale FF-CPT.

Currently, no numerical analyses have been done for the impact of Free-Fall Penetrometers (FFPs) on sand, modeled with a strain rate dependent constitutive model. This is important because current PFFP-sand models, Zambrano-Cruzatty and Yerro [2020] and Yalcin [2021], had to backfit the soil strength properties to match the experimental results. A strain rate dependent constitutive model will allow for data from standard geotechnical tests to better constrain the constitutive model without the need for as much backfitting. Zambrano-Cruzatty et al. [2022] developed a visco-elastoplastic extension to the Mohr-Coulomb soil constitutive relation, referred to as strain-rate Mohr-Coulomb (*SRMC*) in this thesis, to capture strain rate effects in sand. More information about *SRMC* will be provided in chapter 4. Given the availability of *SRMC*, the numerical modeling side of this work aims to take a first step in testing and including *SRMC* in Anura3D's implementation of MPM. Anura3D was chosen due to its tested capabilities for multi-phase soil and impact problems.

1.3.1 MPM and Continuum Mechanics Background

In MPM, the material points (MPs) store and track all the kinematic and material properties for the entire computation cycle. In this way, the material history is preserved. For the spatial region, a temporary mesh composed of elements and locally defined basis functions, is constructed each time step. The basis functions serve two purposes. The first is that they provide a way to interpolate the values at the MPs to the nodes of the mesh, and the reverse.

The second, using the basis functions allow the conservations equations to be solved at fixed locations in space. Different element shapes and basis functions can be used. In 2D the most common elements are triangles and quadrilaterals. In 3D tetrahedral and hexahedral elements are the most common [Hughes, 2000]. In standard MPM piece-wise linear basis functions are used. Piece-wise continuous elements are called low-order elements.

In the case of soil modeling the conservation equations are the mass

$$\frac{D\rho}{Dt} + \rho\nabla \cdot \vec{v} = 0 \tag{1.11}$$

and momentum equation

$$\rho\frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \rho\vec{b} \tag{1.12}$$

where, $\frac{D}{Dt}$ is the material derivative, ρ is density, $\nabla\cdot$ is the divergence operator, $\vec{b}$ are the body forces, $\boldsymbol{\sigma}$ is Cauchy's stress tensor and $\vec{v}$ is velocity [Fern et al., 2019, Nguyen et al., 2023].

To model multiphase material continuum mixture theory is required [Fern et al., 2019, Yerro et al., 2015]. MPM has single and multi-point formulations, both can be used to solve multiphase materials. In single-point MPM, the MPs represent the solid and the motion of the fluid phases are calculated in reference to the solid MPs. In double or triple point MPM, each material phase is represented by its own set of MPs. Single-point MPM is computationally cheaper but does not conserve the liquid's mass and should only be used when small gradients in the porosity are expected [Yerro et al., 2015, Chandra et al., 2024]. This means that it cannot be used to solve free surface flow problems. Double or triple MPM is computationally expensive and requires treatment of porosity discontinuities when transitioning from one material phase to another [Chandra et al., 2024]. Both methods have

been successful in modeling a wide range of geotechnical problems [Chandra et al., 2024, Yerro et al., 2015, Zambrano-Cruzatty and Yerro, 2020]. For this thesis single-point MPM is used. Time stepping can be done using either explicitly or implicitly (Fern et al. [2019], [Zhang et al., 2016, sec. 3.2.1], Chandra et al. [2024]).

To close equations 1.12 and 1.11 and represent the material specific behavior a constitutive model is required. The most general form of soil constitutive equations are based on the concept of effective stress, equation 1.13. For saturated and dry soils, effective stress is defined as

$$\boldsymbol{\sigma'} = \boldsymbol{\sigma} - \boldsymbol{p_w} \tag{1.13}$$

where, $\boldsymbol{\sigma'}$ is the effective stress, $\boldsymbol{\sigma}$ is the total stress, and $\boldsymbol{p_w}$ the pressure of the pore fluid [Lambe and Whitman, 1969]. $\boldsymbol{\sigma'}$ is an approximate model for the inter-granular stresses between soil particles. For a physical interpretation of effective stress and a detailed explanation of the surface areas that are used to calculate $\boldsymbol{\sigma}$ and $\boldsymbol{p_w}$, see [sec. 16.3] [Lambe and Whitman, 1969]. In a dry soil, it is assumed that the gauge pressure of the pore gas is zero, $u = 0$, which means $\boldsymbol{\sigma'} = \boldsymbol{\sigma}$. For partially saturated conditions, extensions are made to equation 1.13, see [Lambe and Whitman, 1969, Yerro et al., 2015].

In infinitesimal strain theory, strain ($\boldsymbol{\varepsilon}$) is defined as

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla\vec{u} + (\nabla\vec{u})^T) \tag{1.14}$$

where, $\vec{u}$ is the displacement, ∇ is the gradient operator, and $(\cdot)^T$ is the transpose operator. Classical elastoplastic theory assumes that $\boldsymbol{\varepsilon}$ can be additively decomposed into recoverable (elastic, $\boldsymbol{\varepsilon}^e$) and unrecoverable (plastic, $\boldsymbol{\varepsilon}^p$) portions,

$$\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^e + \boldsymbol{\varepsilon}^p \tag{1.15}$$

Only the elastic portions contribute to changes in stress changes therefore, in incremental form the relation between $d\boldsymbol{\sigma}$ and $d\boldsymbol{\varepsilon}$ is

$$d\boldsymbol{\sigma}' = \boldsymbol{D}^{ep} : [d\boldsymbol{\varepsilon}] = \boldsymbol{D}^{e} : [d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}^{p}] = \boldsymbol{D}^{e} : d\boldsymbol{\varepsilon}^{e} \tag{1.16}$$

where, $\boldsymbol{D}^{ep}$ is elastoplastic tangent stiffness matrix or the constitutive stiffness matrix and $\boldsymbol{D}^{e}$ is the elastic stiffness matrix [Fern et al., 2019][2].

The general explicit single-point updated-Lagrangian MPM computational cycle can be summarized as follows: (1) apply boundary conditions to nodes and/or MPs and map them to the nodes, (2) solve the momentum equation at the nodes, (3) map the nodal acceleration to the MPs, (4) map the momentum of the MPs to the nodes, (5) calculate the nodal momentum, (6) calculate the strain increments for each MP, (7) evaluate the constitutive equations and update the stress, (8) update the nodal displacements, (9) map the nodal displacements to the MPs, (10) update their location and reset the mesh. See Figures 1.4 and 1.5 for a simplified graphical representation of this process and the initial discretization [Martinelli and Galavi, 2022].

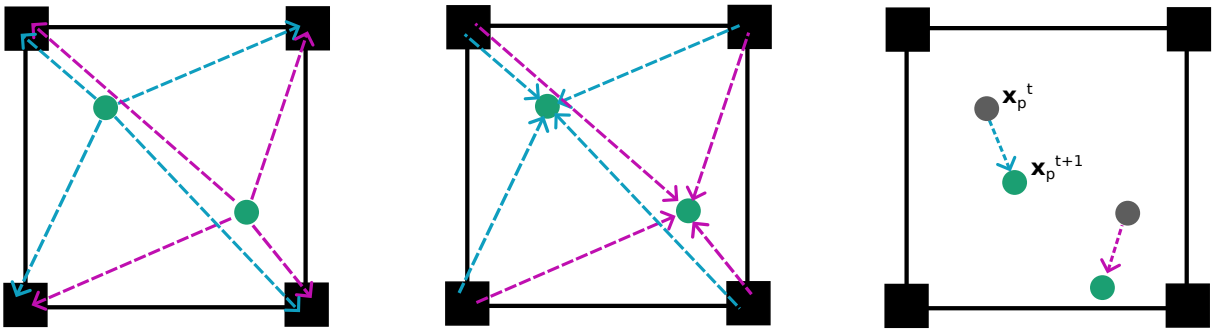


Figure 1.4: Interpolation of information from the mesh to the MPs

[2] $D \equiv \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\epsilon}}$, making it a 4th order, symmetric tensor.

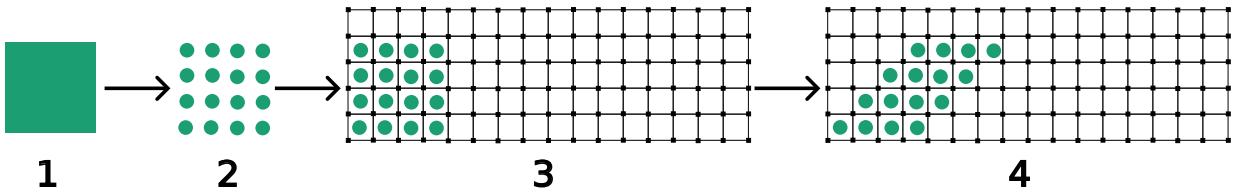


Figure 1.5: Transformation of a continuum material into a MPM simulation

1.3.2 Challenges faced by MPM

Standard MPM is known to suffer from some fundamental issues. The main two are the cell crossing issue and volumetric locking. This subsection gives some information about those issues, what causes them, and possible remedies.

Cell Crossing

The cell crossing error in MPM originates from un-optimal location of MPs during integration [Steffen et al., 2008]. This is especially highlighted in standard MPM, as the derivative of the basis functions are discontinuous between elements (The basis is $C0$ continuous). When a MP crosses over that discontinuity, an oscillation in the derivative driven quantities occurs. Several authors have discussed and proposed methods to mitigate the error [Baumgarten and Kamrin, 2023, Steffen et al., 2008]. There are two main ways to help mitigate the error, (1) increase the support and smoothness of the MP functions (ie. move from the Dirac delta function to a smoother function) and (2) increase the smoothness of the mesh's basis functions. In the class of increasing the support of the MPs, the most common methods are GiMP [Bardenhagen and Kober, 2004] and CPDI [Sadeghirad et al., 2011]. The most common method to increase the smoothness of the mesh is to use quadratic B-Splines [Steffen et al., 2008]. Both classes of methods present their own challenges and are more computationally expensive than standard MPM.

A simpler method, which comes with some theoretical drawbacks, is the Mixed Gauss

integration method. In Mixed Gauss integration, Gaussian integration is used in elements that contain MP volumes that total more than 90% of the volume of the element. For elements that do not meet these criteria, regular MPM MP integration is used. For elements that meet the criteria, a mass weighted average state, stress, and strain variables are found by mapping the values of each MP to the element nodes. Then the averaged nodal values is mapped to a Gauss point at the element's center. The constitutive equation is evaluated for the Gauss point, updating the state, stress and strain variables. The Gauss point values are mapped back to the nodes and then back to the original MPs.

This technique was first proposed by Beuth [2012] for quasi-static simulations and was used by Kafaji [2013] for the same purpose. Both authors used the technique for 3-node triangle and 4-node tetrahedral elements. Those elements, explicity have constant strain updates each time step. There are clear disadvantages to the technique, as maintaining the stress and state variables is one of the primary uses for using MPM in the first place [Kafaji, 2013, sec. 4.6.2]. Martinelli and Galavi [2022] presented an extension of the procedure for 4-node quadrilateral elements. Despite the disadvantages, Mixed Gauss Integration is able to recreate some experimental and theoretical results [Yerro et al., 2015, Li, 2019, Martinelli and Galavi, 2022, Zambrano-Cruzatty and Yerro, 2020]. Other authors have noted limitations of the method [Fern et al., 2019, sec. 9.8].

Volumetric Locking

When a continuum material is discretized, the previously infinite deformation modes are significantly reduced. The reduction makes the problem computationally feasible, but introduces limitations, particularly for nearly and fully incompressible materials. Those materials restrict element deformation to only modes in which the element volume is nearly or fully maintained. If this causes the element deformation to be over-constrained, the element volumetrically locks [Hughes, 2000]. This is particularly apparent in low-order elements as they

already have tightly constrained deformation modes. To mitigate volumetric locking, different techniques have been proposed. Commonly used techniques split a tensor related to the strain increment calculation into a volumetric and a deviatoric portion and average the volumetric component. This technique keeps the volumetric strain of the element constant and reduces the volumetric restrictions on the element [Hughes, 2000].

For box elements (quadrilateral and hexahedral), *B-Bar* is a common solution to mitigate volumetric locking [Hughes, 1980]. For triangle-based elements, Nodal Mixed Discretization (NMD) can be used. NMD was first used in dynamic MPM by Kafaji [2013]. Additional details about *B-Bar* and NMD are in 2.

Addressing volumetric locking is particularly important in Material Point Method (MPM) simulations of portable free-fall penetrometers (PFFPs). When soils are strained to high values, they reach a critical state, where deformation occurs without any change in volume, effectively behaving as an incompressible material. In coupled models involving pore water, the water is typically treated as nearly incompressible. Hence, capturing the material response for both the soil and the pore water relies on effectively mitigating volumetric locking.

1.3.3 Information on Anura3D software

Several MPM codes that have been implemented in the field of geomechanics. Anura3D is one of these open-source codes. Anura3D is an open-source serial MPM software with a scientific and tutorial manual and mesh generation available through GiD software. The standard branch of Anura3D uses triangle elements in 2D and tetrahedral elements in 3D. Stress updates occur using the Modified Update Stress Last (MUSL) algorithm. Information about MUSL can be found in Zhang et al. [2016]. Volumetric locking is mitigated using NMD strain smoothing (section 1.3.2), and the impact of cell crossing is reduced using Mixed

Gauss integration. Anura3D has a plethora of implemented boundary conditions and can model problems in 2D plane strain, 2D axisymmetric, 3D axisymmetric, and 3D Cartesian. Multiphase materials can be modeled using single-point or double-point. Anura3D assumes fluids are weakly compressible.

Multi-body contact can be handled in two ways in Anura3D. The first is to use the in-built no-slip contact inherent to MPM [Sulsky et al., 1994]. To handle frictional contact, Anura3D has an implementation of the method proposed in Bardenhagen et al. [2000] with some recommended modifications from Bardenhagen et al. [2001]. In Anura3D, contact nodes needed for Bardenhagen et al. [2000] are set during geometry initialization.

1.4 Objectives and Methodology

The overarching objective of this research is to enhance the reliability of soil characterization using portable free-fall penetrometers (PFFPs). Achieving this objective necessitates both the centralization and automation of field data processing and the development of high-fidelity numerical models of PFFP-soil impact using the material point method (MPM). A key milestone in this effort is the creation of a highly accurate MPM model, which would enable the development of a digital twin of PFFP impact.

A digital twin offers numerous advantages. First, it provides a framework for rigorously evaluating and refining the empirical relationships currently used to estimate the quasi-static bearing capacity ($qsbc$) from PFFP data. Second, it allows for interpolation and extrapolation of field measurements, thereby expanding the available dataset beyond the constraints of physical testing. Additionally, if the digital twin is computationally efficient, it could facilitate the formulation of an inverse problem wherein soil properties are back-calculated directly from measured PFFP acceleration data.

Beyond its immediate applications in numerical modeling and field data interpretation, a

unified database integrating both numerical simulations and field measurements would serve as an extensive training set for machine learning (ML) applications. Such ML models could be leveraged to: (1) accelerate the solution of inverse problems, reducing computational costs and time; (2) serve as a data-driven approach for converting the dynamic bearing capacity (q_{dyn}) to $qsbc$, improving the accuracy of soil strength estimation; and (3) directly infer geotechnical soil properties from raw PFFP data, potentially bypassing intermediate empirical relationships altogether.

By integrating advanced numerical modeling, automated data processing, and machine learning, this research aims to establish a more robust, efficient, and data-driven approach to PFFP-based soil characterization, ultimately improving the reliability of geotechnical assessments.

Figure 1.6 shows the general process required to complete these objectives. The chart flows from top to bottom. The layering of the three colored boxes, for each flag on the flow chart, represents the steps required before that step is complete. The first step is curation and/or implementation (blue box); the second is validation (orange box); and the third is computational analysis and streamlining (purple box). In practice, all three steps are intercoupled, but these are the general regimes. Curation and/or implementation includes finding, organizing, or implementing software and data. During validation, the numerical software is tested against known solutions, the implemented data processing software is tested using known inputs, and documentation starts to be written. During computational analysis and streamlining, the software now tested is cleaned up and organized. Steps are taken to reduce the computational cost, and documentation is completed.

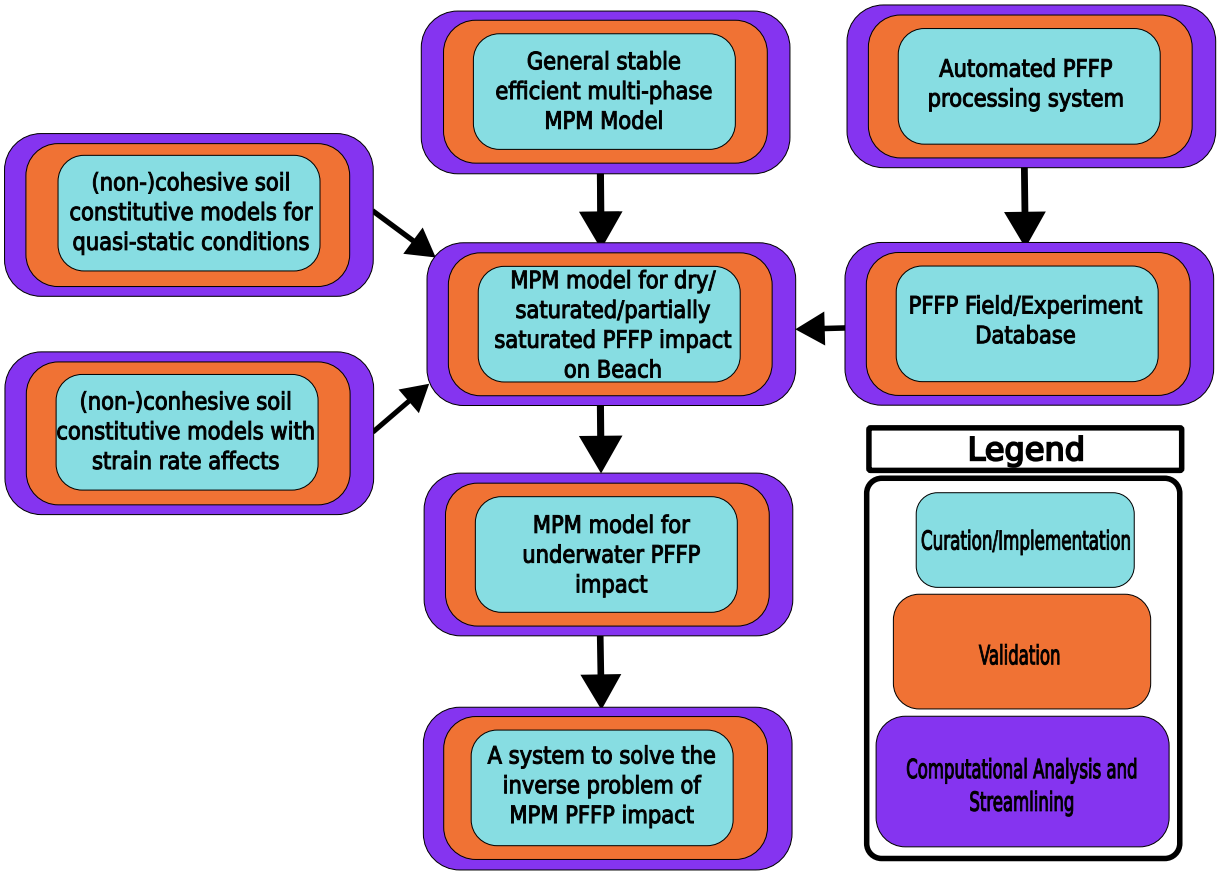


Figure 1.6: Flow chart of work that would have to be done to generate a system to solve the PFFP to soil properties inverse problem.

This thesis made significant progress through a two-pronged approach. The primary aim of the first prong is to advance the creation of a digital twin for the PFFP impacting into sand. This effort is subdivided into three distinct subgoals, each corresponding to an individual chapter in the thesis [Chapter 2-4]. The second prong is the centralization, automation, and standardization of BlueDrop field data [Chapter 5]. The description of each chapter is as follows.

Chapter 2 addresses the first subgoal of doing a comprehensive review and testing of the current limitations of the Anura3D MPM implementation is presented. The focus is on PFFP impact problems when using an advanced constitutive model, namely the strain rate Mohr-Coulomb (*SRMC*) model. On the formulism side, the investigation reveals that the techniques employed by Anura3D to discretize the mesh space, to mitigate volumetric locking, and to handle the cell crossing issue are significant limiting factors. Additionally, from a software perspective, the computational expense associated with the *SRMC* model within Anura3D is found to be excessively high.

Chapter 3 addresses the second subgoal of working towards adopting a more advanced formulism to discretize space and address volumetric locking in Anura3D. This chapter details the implementation of quadrilateral elements enhanced with the *B-Bar* method, followed by a qualitative assessment of the implementation. This effort represents a critical step toward refining the numerical performance of the simulation framework.

Chapter 4 is dedicated to the third subgoal, which focuses on reducing the computational expense of the constitutive models. In this chapter, a detailed analysis is conducted to decrease the computational cost of the *SRMC* model. Simultaneously, the computational burden of a strain softening Mohr-Coulomb (*SSMC*) model is analyzed, and attempts to accelerate it are described. Accelerating the *SSMC* model is of particular importance as it enables faster and more direct comparisons with previous work by Zambrano-Cruzatty and Yerro [2020] and Yalcin [2021]. Testing in this context is performed using *IncrementalDriver*,

which simplifies the analysis by removing the complexities inherent in integrating a soil model within a full FEM or MPM framework. This approach allows the constitutive model to be examined in isolation, bypassing many challenges associated with more advanced numerical methods.

Chapter 5 is the second prong of this work, which presents the development of a Python library known as the *BlueDrop Analysis Library*. This library is designed to automate and centralize the processing of BlueDrop field data. The library standardizes data storage by integrating both inputs and computed outputs with a *SQLite* database, thereby facilitating more systematic data management and analysis.

Finally, **Chapter 6** synthesizes the conclusions and outlines potential directions for future research. A pictorial representation of the workflow followed throughout this thesis is provided in Figure 1.7. In this flow chart, the stages of the project are visually denoted: a colored box behind a flag indicates a completed stage, a box with a missing colored background suggests that the stage is either in progress or has not been started, and a greyed-out box confirms that the stage was not initiated. The details of the work associated with each flag are elaborated upon in the corresponding chapters.

Through this structured approach, the thesis makes meaningful strides toward the overarching goal of developing a digital twin for PFFP impact scenarios, even as certain objectives remain to be fully realized.

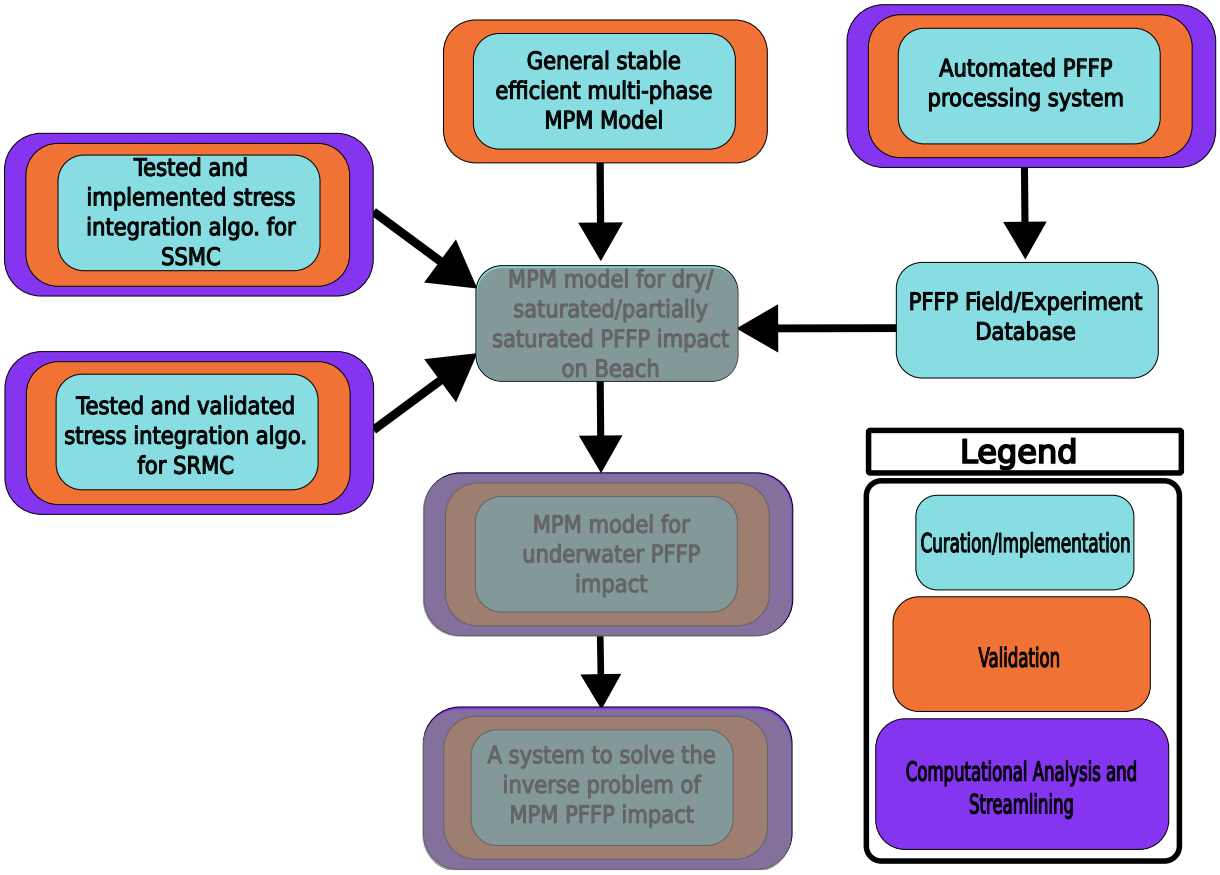


Figure 1.7: Flow chart of work actually done during this thesis. Each leaf shows the actions that were done during this thesis. Boxes that are greyed out were not worked on during this thesis.

Chapter 2

PFFP Analysis using Anura3D

2.1 Introduction

In this section, a qualitative analysis of two Material Point Method (MPM) models simulating a Portable Free-Fall Penetrometer (PFFP) impacting sand is presented. The modeling was performed using the main branch of Anura3D. These models are inspired by the experiment conducted by Albatal [2018], where the BlueDrop PFFP was dropped into 40 cm of dry sand within a calibration chamber (referred to as Experiment (1) in Section 1.3). By backfitting this experiment was numerically reproduced by Zambrano-Cruzatty and Yerro [2020]. The models presented here were developed during the initial part of this thesis to replicate these experiments and assess how incorporating a more advanced constitutive model influences the numerical results.

The modeled PFFP has a density of 2930 $\mathrm{kg\,m^{-3}}$, a mass of 5.35 kg, and a length of 40 cm. In comparison, the BlueDrop PFFP, equipped with a cone tip, has a density of 3120 $\mathrm{kg\,m^{-3}}$, a mass of 7.71 kg, and a length of 63.2 cm. The density of the modeled PFFP was estimated using measurements of BlueDrop 3 (refer to Figure 2.1). Differences in density are likely due to errors in volume calculations. The modeled PFFP was shortened to 40 cm to reduce computational time during initial simulations. Although the intention was to rerun the models using the correct geometry and material properties, time constraints prevented this. Nevertheless, the general impact behavior should remain comparable.

The modeling, meshing, and material specifications closely follow the approaches outlined in Zambrano-Cruzatty [2021] and Yalcin [2021]. To reduce computational costs, the PFFP impact was assumed to be 3D-axisymmetric, and only a 20° slice of the geometry was modeled. The simulation utilized a total of 268,973 material points (MP). Figure 2.2 illustrates the geometry, boundary conditions, and the distribution of material points per element. To address numerical challenges, Mixed Gauss integration was applied to mitigate the cell-crossing issue, and NMD strain smoothing was applied to reduce volumetric locking.

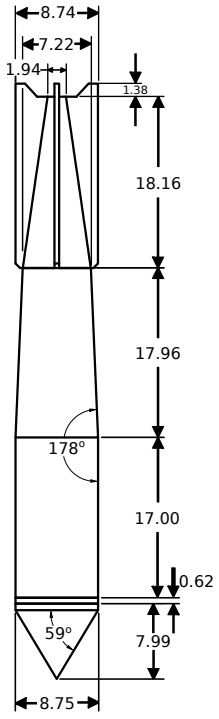


Figure 2.1: Dimensions of BlueDrop 3 with a cone tip in centimeters and degrees

The mesh was refined to a resolution of 1 cm in the region surrounding the PFFP to improve accuracy.

The PFFP was initialized with a velocity of $-5.6\,\mathrm{m\,s^{-1}}$ in the y-direction. The PFFP was modeled as a rigid body and constrained to only move in the y-direction [Zambrano-Cruzatty and Yerro, 2020]. To keep the PFFP aligned with the mesh's nodes, a moving mesh technique was used. The moving mesh technique works by fixing the upper mesh, initially from $y = 0.6\,\mathrm{m}$ to $y = 1.2\,\mathrm{m}$, to move with the PFFP. The mesh initially from $y = 0\,\mathrm{m}$ to $y = 0.6\,\mathrm{m}$ either compresses or extends to accomodate the movement of the upper mesh.

To model the behavior of the sand, the first MPM model adopts a Mohr-Coulomb (MC) constitutive model, while the second utilizes a strain-rate dependent Mohr-Coulomb ($SRMC$) model [Zambrano-Cruzatty et al., 2022]. The material properties of the PFFP, along with the parameters shared by both constitutive models, are summarized in Table 2.1.

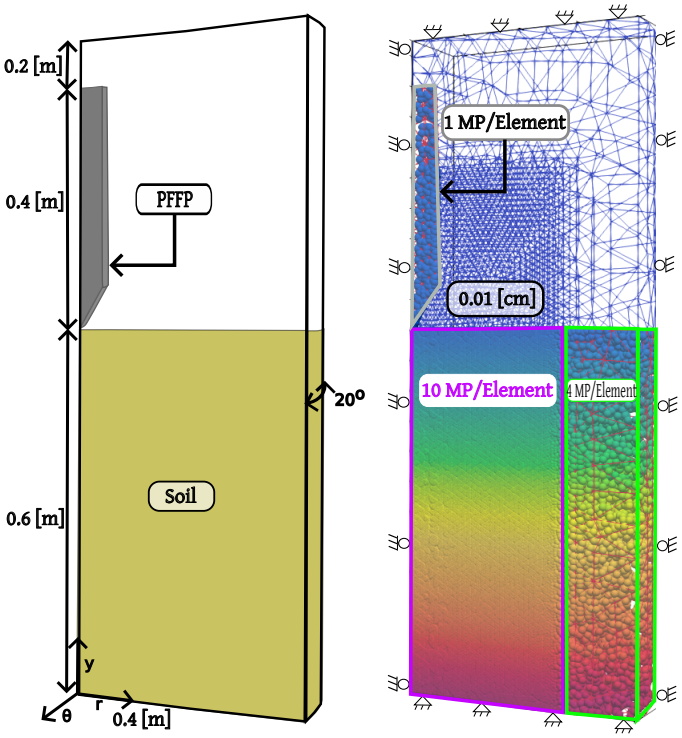


Figure 2.2: Geometry of the PFFP models. Roller boundary conditions on the sides and fully fixed on the top and bottom

The soil parameters were derived from triaxial experiments conducted by Albatal [2018] in conjunction with the BlueDrop experiments. These triaxial data was analyzed and utilized by Zambrano-Cruzatty and Yerro [2020]. The contact between the PFFP and the soil is modeled using the contact algorithm developed by Bardenhagen et al. [2001]. Consistent with Zambrano-Cruzatty and Yerro [2020], a contact friction angle of 20° was applied. A Courant number of 0.7 and 0.5 was used for the *MC* and *SRMC* model, respectively.

The goal numerical time (t) for the PFFP impact was 0.125 s. Vtk files were output for analysis in *Paraview* every 0.005 s. The numerical time was selected following Zambrano-Cruzatty and Yerro [2020], to give the PFFP enough time to impact into the soil and come to rest. The *MC* model completed its run-time but suffered from severe rebound after the impact was done. The *SRMC* model failed due to a numerical instability at 0.096 09 s. The next two sections present some figures and a qualitative discussion. Finally, the chapter

Table 2.1: Modeled PFFP material parameters

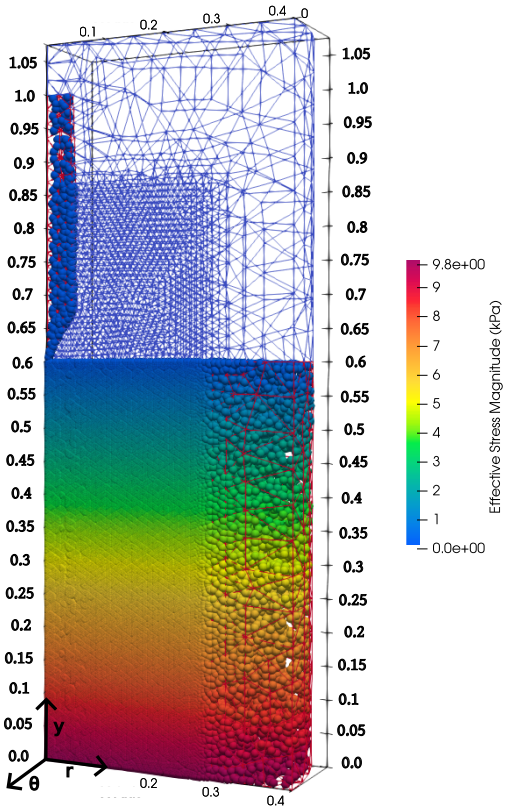
Properties	**Sand**	**Modeled PFFP**
Material Type	**Mohr-Coulomb/*SRMC***	**Rigid Body**
Mass (m_{pffp}) [kg]	-	5.35
Volume (V_{pffp}) [m^3]	-	0.001825
Solid Density (ρ_s) [m^3]	2650	2930
Effective Young's Modulus (E') [kPa]	2500	-
Effective Poisson's Ratio (ν') [-]	0.33	-
Effective cohesion (c') [-]	0	-
Friction Angle (ϕ') [°]	39	-
Coefficient of Earth pressure at rest (K_0) [-]	0.39	0.5

concludes with a discussion on the next steps.

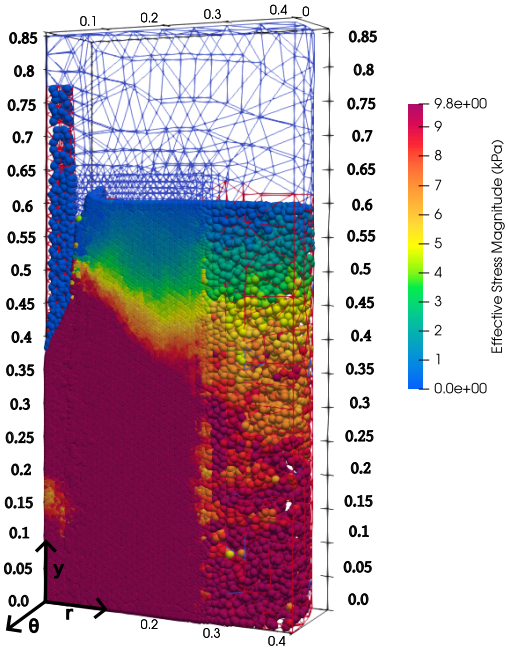
2.2 PFFP impact into dry soil using Mohr-Coulomb

This section discusses the *MC* PFFP model. Figures 2.3a, 2.3b, and 2.3c correspond to $t = 0.0\,\text{s}, 0.06\,\text{s}$, and $0.125\,\text{s}$ during the PFFP impact. The soil is colored by the magnitude of the effective stress tensor ($\boldsymbol{\sigma}$). Figure 2.3d is a plot of how the PFFP's acceleration and velocity changes as it penetrates into the soil. The sign convention of displacement, velocity and acceleration in Figure 2.3d follows the convention in PFFP literature [Albatal, 2018]. Acceleration is positive in the positive y-direction. Velocity is positive in the negative y-direction. This sign convention allows the velocity plotted in $\text{m}\,\text{s}^{-1}$ and acceleration in g's to have similar ranges and be plotted on the same plot. Compared to results presented by [Zambrano-Cruzatty et al., 2022] the acceleration, velocity, and penetration depth are reasonable. However, at the end of impact the acceleration and velocity does not stay at zero. This causes the PFFP to rebound. This is caused by a numerical instability. The instability happens around $t = 0.06\,\text{s}$ at y $= 0.15$. The instability is likely caused by a combination of MPM Gauss Integration not fully mitigating the impact of the cell crossing issue, NMD smoothing not fully resolving volumetric locking and/or an unfavorable

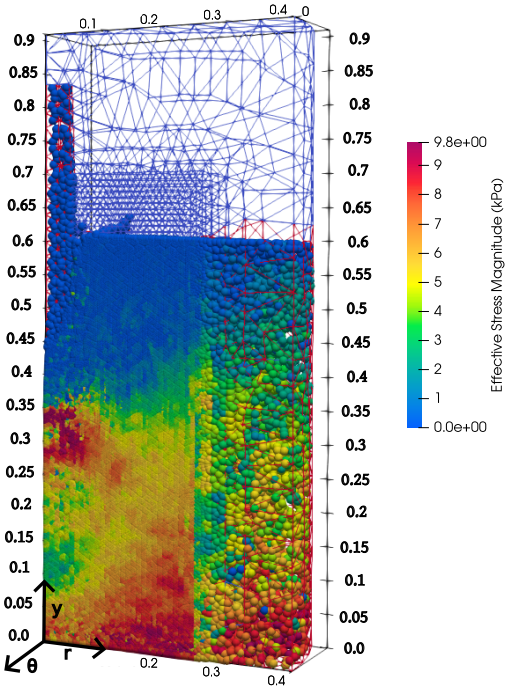
distribution of material points (MPs) in some elements. The instability causes non-physical energy to be produced, pushing the PFFP in the vertical y-direction.



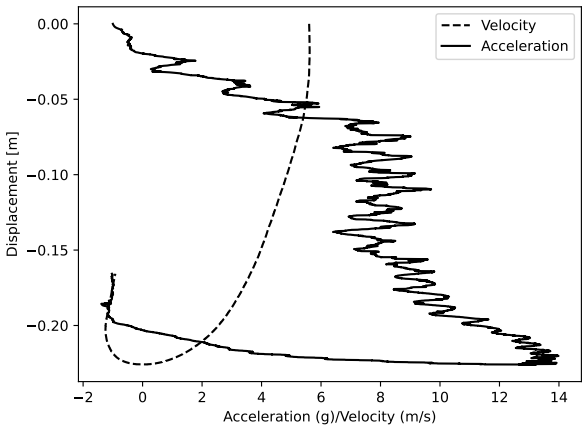
(a) *MC* PFFP model at $t = 0.0\,\mathrm{s}$



(b) *MC* PFFP model at $t = 0.06\,\mathrm{s}$



(c) *MC* PFFP model at $t = 0.125\,\mathrm{s}$



(d) PFFP Acceleration/Velocity vs. Penetration depth during impact using the *MC* soil constitutive relation

Table 2.2: *SRMC* State parameters Initial Values and Other Required Inputs

Property	**Value**
Critical Stress ratio in triaxial compression (M_{tc}) [-]	1.59
Nova's volumetric coupling coefficient	0.0
Initial Minimum Dilatancy (D^0_{min}) [-]	-0.1
Hardening Parameter (h) [-]	10
Shear Viscosity Coefficient (k_G) [-]	0.04
Bulk Modulus Viscosity Coefficient (k_K) [-]	0.1
Dilatancy Viscosity Coefficient (k_D) [-]	0.04
Specific Gravity (G_s) [-]	2.65
Reference deviatoric strain rate $\dot{\varepsilon}_{q_{ref}}$ [-]	2

2.3 PFFP impact into dry soil with a Strain Rate Mohr-Coulomb

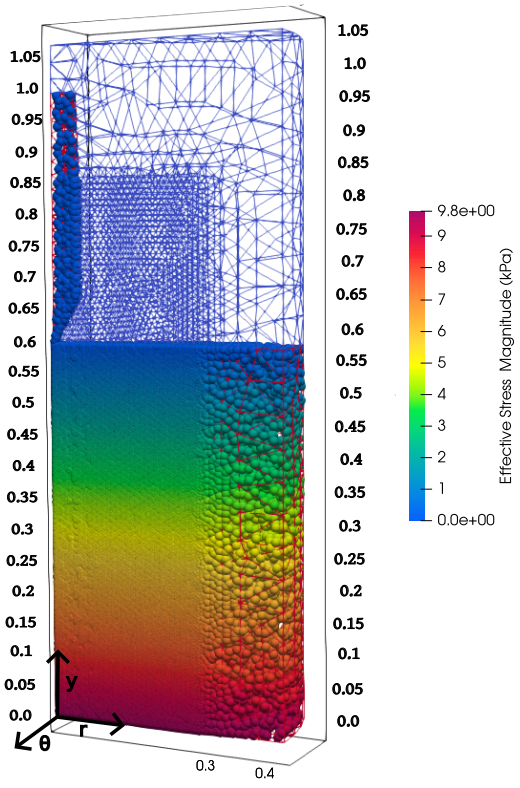
This section discusses the *SRMC* PFFP model. The *SRMC* constitutive model requires more material and fitting parameters than the *MC* model, Table 2.2. The definition of the parameters are provided in 4. The triaxial compression critical stress ratio at constant volume (M_{tc}) was calculated using $\phi' = 39°$ angle through equation 2.1,

$$M_{tc} = \frac{6\sin(\phi')}{3 - \sin(\phi')} \tag{2.1}$$

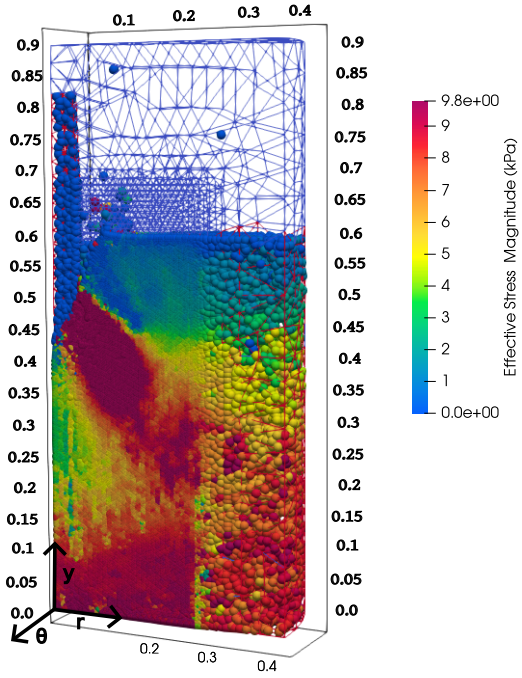
Most values in Table 2.2, are close to the values used by Zambrano-Cruzatty et al. [2022] during calibration with the strain rate sand experiments of Yamamuro et al. [2011]. The exception is the reference strain deviatoric strain rate $\dot{\varepsilon}_q$. $\dot{\varepsilon}_{q_{ref}}$ was raised to $\dot{\varepsilon}_{q_{ref}}$. This is several orders of magnitude higher than the value of $\dot{\varepsilon}_{q_{ref}} = 2.5 \cdot 10^{-5}$ used in Zambrano-Cruzatty et al. [2022]. The higher $\dot{\varepsilon}_{q_{ref}} = 2.5 \cdot 10^{-5}$ was used, so the amount of strain rate effects would be lower for this initial model. This was a modelling decision with the hope of creating a more stable model.

Figures 2.4a, 2.4b, and 2.4c correspond to $t = 0.0\,\text{s}, 0.05\,\text{s}$, and $0.0961\,\text{s}$ during the PFFP

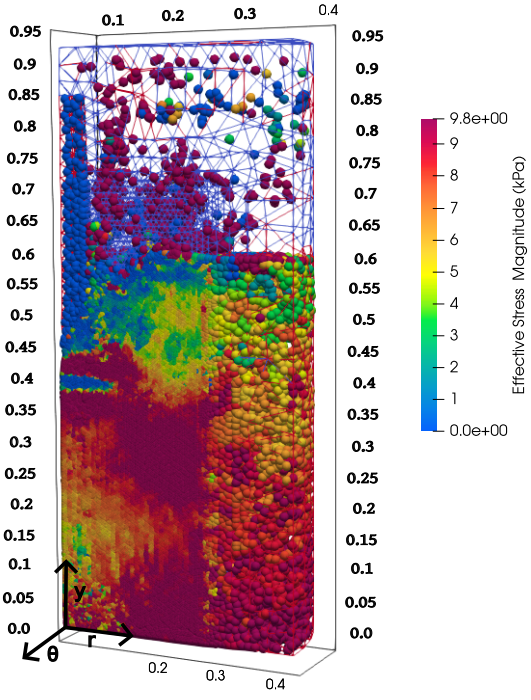
impact. The soil is colored by the magnitude of the effective stress tensor ($\boldsymbol{\sigma}$). Figure 2.4d is a plot of how the PFFP's acceleration and velocity changes as it penetrates into the soil. As in the last section, the acceleration is positive in the positive y-direction and the velocity is positive in the negative y-direction. This model suffers from more instabilities that the *MC* model. *SRMC* has the effect of pushing the soil towards being more incompressible. Incompressible material pushes the capabilities of Mixed Gauss integration and any strain smoothing algorithm. An instability in the model causes the oscillations in the acceleration profile (Figure 2.4d) and launching of the soil's MPs into the air. In particular Mixed Gauss integration mixing the material state parameters is expected to have negative impacts on the *SRMC* model in particular. The *SRMC* requires the state parameters be preserved with greater accuracy. The other challenge of the *SRMC* model was that it was more computationally expensive than the *MC* model. An investigation into a way of reducing the computational expense is done in chapter 4. During that process bugs were found in the implementation of the *SRMC* model.



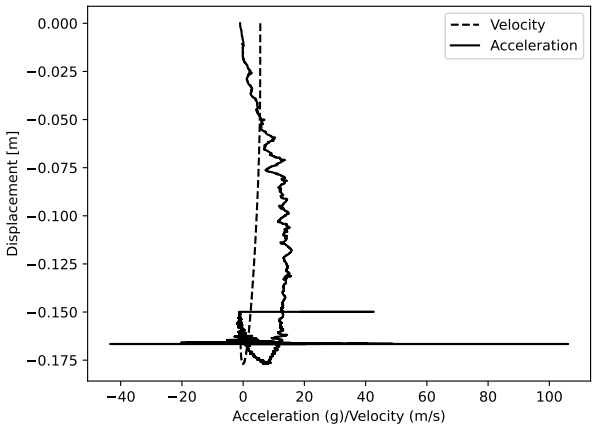
(a) *SRMC* PFFP model at $t = 0\,\mathrm{s}$



(b) *SRMC* PFFP model at $t = 0.05\,\mathrm{s}$



(c) *SRMC* PFFP model at $t = 0.1\,\mathrm{s}$



(d) PFFP Acceleration/Velocity vs. Penetration depth during impact using the *SRMC* soil constitutive relation

2.4 Conclusion

In summary, the models presented here highlight the need for advancements, further investigations, and additional developments in the constitutive models and Anura3D. Compared to Zambrano-Cruzatty [2021] and Yalcin [2021], where a strain softening Mohr-Coulomb constitutive model was used, using a more advanced constitutive model, like strain-rate Mohr-Coulomb, requires additional numerical development of Anura3D. Specifically, improvements are required to address volumetric locking, the cell-crossing instability, and the computational expense associated with the *SRMC* model. These objectives motivated the subsequent chapters, except for Chapter 5, which focuses on processing actual BlueDrop data.

The initial plan was to take a step back and implement 2D quadrilateral elements in Anura3D. Once quadrilateral elements were in place, the goal was to extend the implementation to 3D hexahedral elements. These elements would enable the use of advanced techniques for strain smoothing and addressing the cell-crossing issue, as described by Hughes [1980] and Bardenhagen and Kober [2004]. However, these goals proved too ambitious. Only a preliminary implementation of quadrilateral elements with the *B-Bar* strain smoothing technique presented by Hughes [1980] was completed, as discussed in Chapter 3.

On the side of constitutive models, errors in the *SRMC* model were corrected, and efforts were made to reduce its computational expense, as described in Chapter 4.

Chapter 3

Qualitative Testing of B-Bar and Quadrilateral Elements in Anura3D

3.1 Introduction

As mentioned in Section 1.3.3, Anura3D currently employs NMD to relax volumetric constraints in 3-node triangular and 4-node tetrahedral elements. As part of this thesis, in collaboration with the VT GeoMPM research group[1], 4-node quadrilateral elements incorporating *B-Bar* and the MPM Gauss integration scheme presented in Martinelli and Galavi [2022] were implemented. A key advantage of quadrilateral elements is that their strain field is naturally bi-linear. Since *B-Bar* is localized to each element, the volumetric strain does not spread as it does with NMD. Additionally, MPM Gauss integration is essential for mitigating the stress oscillations associated with cell crossing.

The following sections compare NMD and *B-Bar* and qualitatively assess MPM results for a column collapse, with and without these methods. The column collapse was chosen as a simple first test for implementing quadrilateral elements and *B-Bar*. While more rigorous tests, such as Cooke's Membrane, exist for *B-Bar*, the column collapse model was selected as a more straightforward starting point. A full PFFP impact model was not used, as it is significantly more complex and introduces too many variables (contact algorithm), and does not directly test the implementation of quadrilateral elements with *B-Bar*.

3.2 Comparison of NMD and B-Bar

NMD works by additively splitting the strain increment tensor ($d\boldsymbol{\varepsilon}$) into a volumetric ($d\boldsymbol{\varepsilon}^{vol}$) and deviatoric ($d\boldsymbol{\varepsilon}^{dev}$) part, equation 3.1[2].

$$d\boldsymbol{\varepsilon} = d\boldsymbol{\varepsilon}^{dev} + d\boldsymbol{\varepsilon}^{vol} \tag{3.1}$$

[1]The implementation of this code would not have been possible without the help of Abdel Alsardi.

[2]Note: Since strain is constant in piece-wise linear triangle and tetrahedral element, the values of the strain at each MPM is the same as the values at the nodes.

Then the average volumetric strain ($d\boldsymbol{\varepsilon}^{vol}$) for each node i is calculated as

$$d\boldsymbol{\varepsilon}_i^{vol} = \frac{\sum_{e=1}^{N_e} d\boldsymbol{\varepsilon}_e^{vol} V_e}{\sum_{e=1}^{N_e} V_e} \tag{3.2}$$

where, V_e is the volume of MPs in each element and N_e is the number elements surrounding the node i. Then, an average strain for the element ($d\bar{\boldsymbol{\varepsilon}}_e^{vol}$) is calculated.

$$d\bar{\boldsymbol{\varepsilon}}_e^{vol} = \frac{1}{N_{en}} \sum_{i=1}^{N_{en}} d\boldsymbol{\varepsilon}_i^{vol} \tag{3.3}$$

where, N_{en} is the number of nodes in the element. Finally, the updated total strain ($d\bar{\boldsymbol{\varepsilon}}$) is calculated as

$$d\bar{\boldsymbol{\varepsilon}} = d\boldsymbol{\varepsilon}^{dev} + \frac{1}{d_v}(d\boldsymbol{\varepsilon}^{vol} - d\bar{\boldsymbol{\varepsilon}}_e^{vol}) \tag{3.4}$$

where, d_v is the dimension of the volumetric strain terms (ie. number of strains with $_{ii}$ terms, i.e., 2 for plane strain, 3 axi-symmetric, etc.). This technique is not local to each element. Therefore, it has the effect of homogenizing and spreading the volumetric strain across the mesh, which can affect the accuracy of the numerical results [Martinelli and Galavi, 2022].

In comparison, *B-Bar* modifies the matrix of shape function derivatives ($\boldsymbol{B}$). All modifications are local to the element. In both FEM and MPM, $\boldsymbol{B}$ is evaluated at the integration points (MPs or Gauss points), and multiplied against the nodal incremental displacement ($d\vec{u}_a$) to calculate each MP's incremental strain ($d\boldsymbol{\varepsilon}$) as

$$d\boldsymbol{\varepsilon} = \sum_{a=1}^{nn} \boldsymbol{B}_a d\vec{u}_a \tag{3.5}$$

where nn is the number of element nodes. $\boldsymbol{B}$ can be additively split as

$$\boldsymbol{B} = \boldsymbol{B}^{vol} + \boldsymbol{B}^{dev} \tag{3.6}$$

where, $\boldsymbol{B}^{vol}$ is the volumetric component and $\boldsymbol{B}^{dev}$ is the deviatoric component. To form the *B-Bar* matrix ($\bar{\boldsymbol{B}}$), $\boldsymbol{B}^{vol}$ is selectively evaluated at a fictitious MP at the element center, forming $\bar{\boldsymbol{B}}^{vol}$. $\bar{\boldsymbol{B}}^{vol}$ is then combined back as

$$\bar{\boldsymbol{B}} = \bar{\boldsymbol{B}}^{vol} + \boldsymbol{B}^{dev} \tag{3.7}$$

This makes the volumetric strain in the element constant, reducing volumetric constraints. This has the downside of increasing the likelihood of spurious non-physical oscillation modes (hour-glassing) [Hughes, 2000]. Despite that, 4-node and 8-node hexahedral elements continue to be one of the most used element types in the FEM literature [Hughes, 2000]. Note that *B-Bar* cannot be used for 3-node triangle elements and 4-node tetrahedral elements as the total strain, and the modification of $\boldsymbol{B}$ for these elements would get you right back to where you started.

3.3 Column Collapse Comparison

The aim of this section is to present a qualitative comparison between the results of 3-node triangle and 4-node quadrilateral models with and without their respective strain smoothing technique. All models use Mixed Gauss integration to reduce cell-crossing noise. For simplification purposes, a column collapse problem in plane strain conditions was modeled[3] (Figure 3.1). Given the implementation of plane strain in Anura3D, a modification was made to the B-Bar algorithm, following Bisht [2021]. Details of the implementation are summarized in Appendix B. An analysis similar to the one done here, only using triangle elements but using

[3]As both elements are two-dimensional elements, an assumption had to be made about the out of plane stress or strain. As common in the geotechnical community a plane strain model, the out of plane stress is takes what ever value is required to fix the out of plane strain to zero. A plane strain assumption holds when the material being modeled is long enough, in the out of plane direction, that for the part that is being modeled the adjacent out of plane sections hold each other in place. An example of when this assumption has been successfully applied is for long foundations.

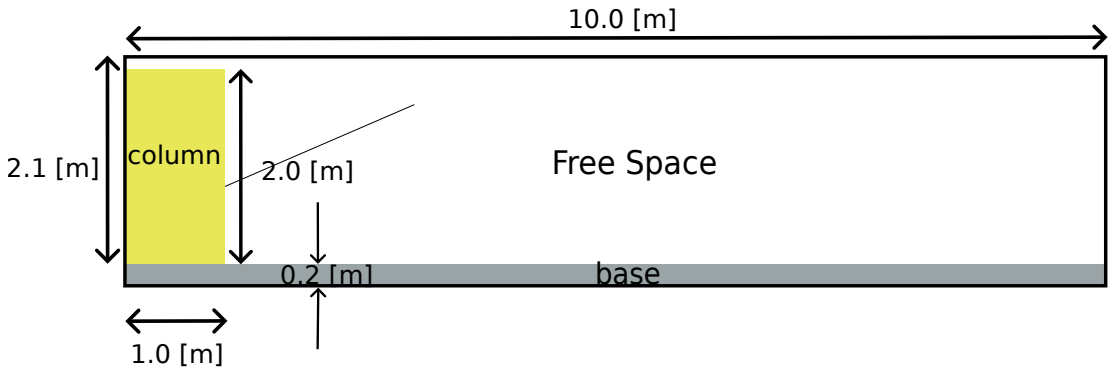


Figure 3.1: Column Collapse Geometry

different constitutive relations, was done in [Fern et al., 2019, chap.9].

The column consists of a dry, drained perfectly-plastic Mohr-Coulomb material and the base is linear elastic. The material properties of the column and base are provided in Table 3.1.

Table 3.1: Column and Base Material Parameters

Properties	**Column**	**Base**
Material Type	**Dry Material**	**Dry Material**
Initial Porosity [-]	0.4	0.4
Particle Density [$\mathrm{kg\,m^{-3}}$]	2650	2650
Material Model Solid	**Mohr-Coulomb**	**Linear-Elastic**
Young Modulus [kPa]	10,000	10,000
Poisson ratio [-]	0.3	0.3
Cohesion [kPa]	0.0	-
Quasi-Static Cohesion Value [kPa]	10,000	-
Friction Angle [°]	20	-
Dilatancy Angle [°]	0	-
Tensile Strength [kPa]	0	-

A structured mesh with 0.1 m elements was used. Using quadrilateral elements with 4 material points (MPs) per element resulted in 2300 elements and 1596 MPs. Using triangle elements with 3 MPs per element resulted in 4600 elements and 2397 MPs[4].

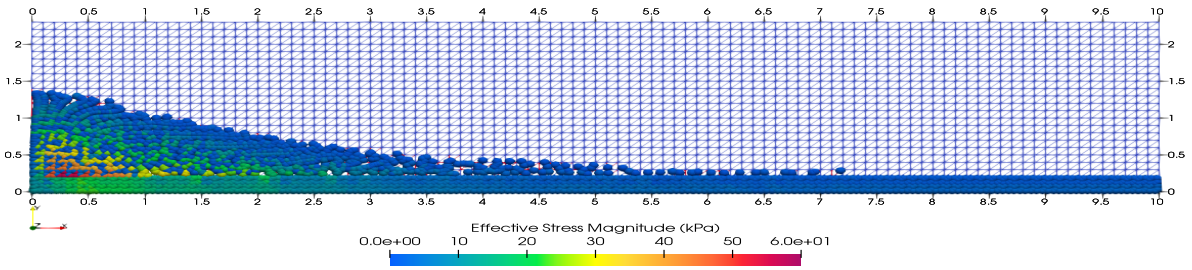
[4]MPs are only generated where the material initially is. The higher number of elements comes from having to discretize enough space for the material to deform into.

Running the model consisted of two stages. During both stages the left and right grid boundaries were fixed in the horizontal direction, the bottom boundary was fully fixed, and the top boundary was fixed in the vertical direction. The first stage consisted of generating the initial gravity stress in the column and base. This was done using the quasi-static convergence algorithm presented in Kafaji [2013]. 70 % local damping was used to help convergence. To prevent horizontal deformation of the column during this stage, deformation on the right side of the column was fixed horizontally. Additionally, a cohesion of 10 000 kPa was used to prevent plastification of the soil column during this stage. Convergence was determined using a kinetic energy and force error tolerance of 0.01. During the second stage, the horizontal fixity on the right side of the column was removed, the cohesion was reduced to 0.0 kPa, local damping was decreased to 5 % and the model was allowed to propogate in time. Contact was handled using MPM's built in no-slip contact algorithm.

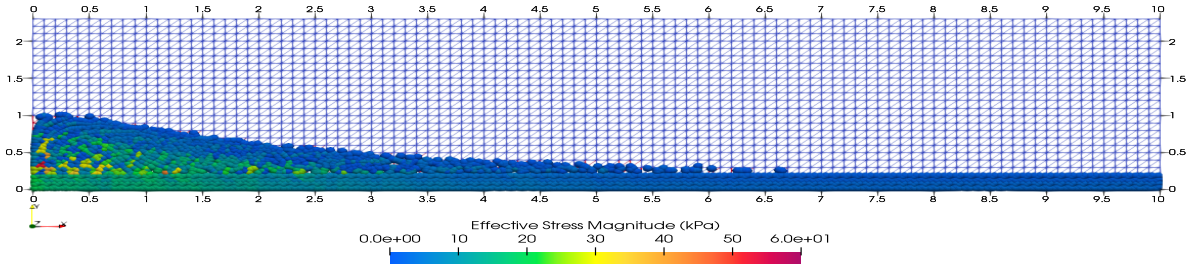
3.3.1 Column Collapse Comparison

Four models were run to do the comparison. These consisted of a triangle model with and without strain smoothing and a quadrilateral model with and without strain smoothing. Figure 3.2 is the final run-out of the model. The animations for the models are available for download on GitHub: Triangle without strain smoothing, Triangle with NMD strain smoothing, Quadrilateral without strain smoothing, and Quadrilateral with B-Bar strain smoothing. The input files to run the models are at the same link, expect up a directory on GitHub.

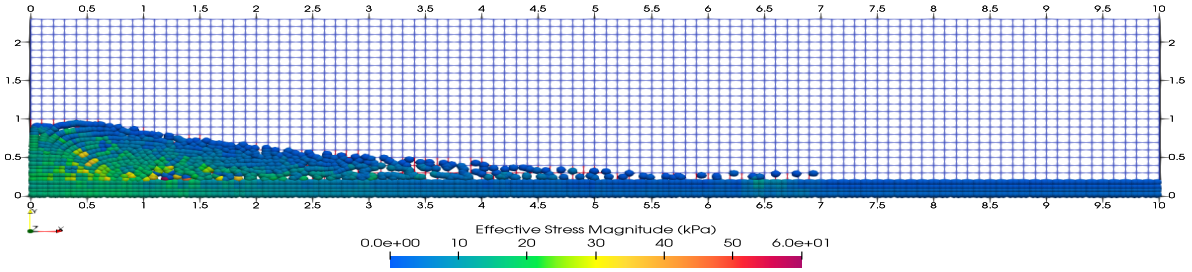
At first glance, the shapes of the final run-outs are relatively similar. However, there are some notable differences. The model with the most difference is the triangle element without strain smoothing, Figure 3.2a. The final peak height is about 1.4 m compared to about 1.0 m for the other models. A likely reason for the discrepancy is volumetric locking.



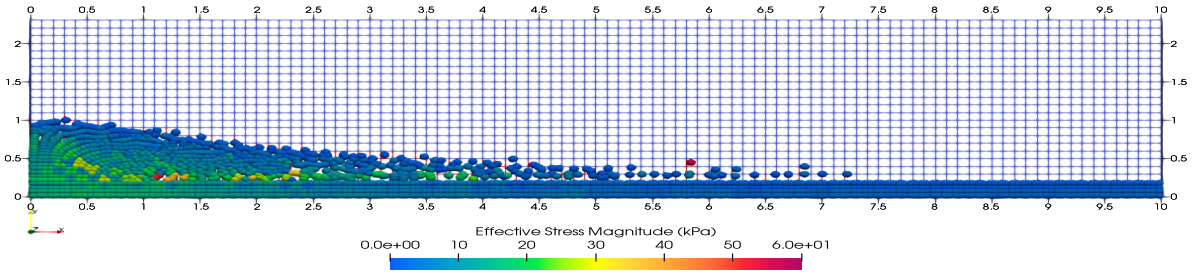
(a) Triangle element without NMD strain smoothing



(b) Triangle element with NMD strain smoothing



(c) Quadrilateral element without B-Bar strain smoothing



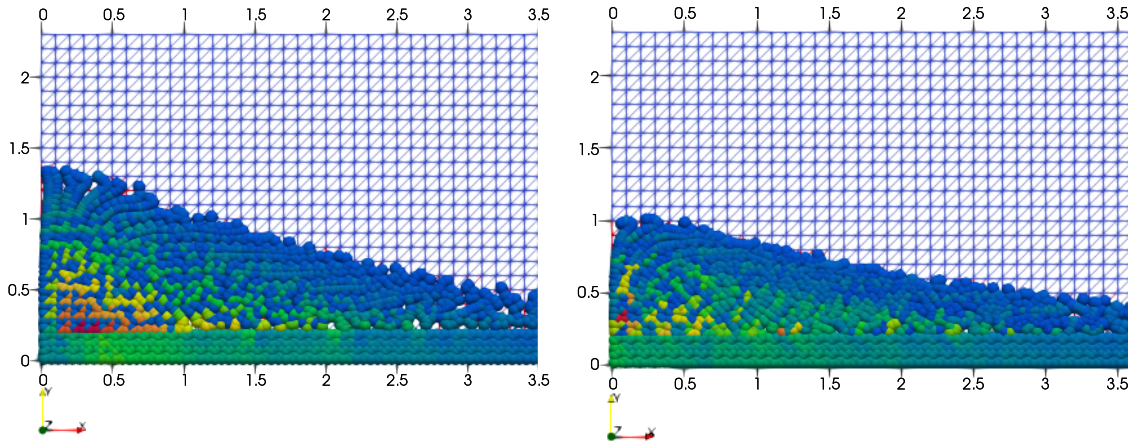
(d) Quadrilateral element with B-Bar strain smoothing

Figure 3.2: Final run-out of the column collapse models.

The triangle model with strain smoothing, Figure 3.2b, seems to have a higher MP density in the run-out. A possible explanation is the NMD smoothing caused the MPs to clump. However, the quadrilateral model with B-Bar smoothing has the MPs that are the most dispersed.

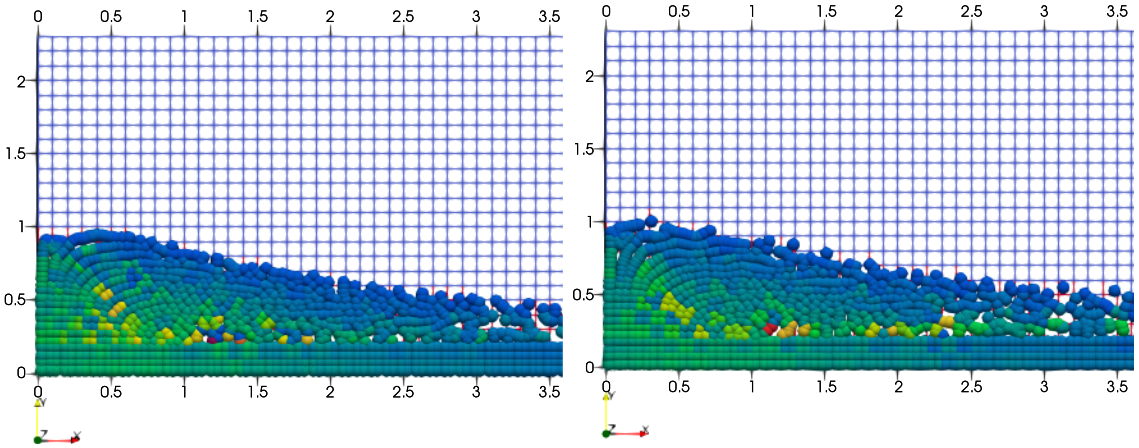
Figure 3.3 is a zoomed in perspective of the left of the column collapse models. The stress field for the triangle model without and with NMD strain smoothing are oscillatory, Figure 3.2a, 3.3b. The stress field of the quadrilateral elements with no strain smoothing is about the same smoothness as the triangle elements with strain smoothing. Figure 3.2d seems to be the smoothest, as expected. It's likely some remaining oscillations are because of the limitations of Mixed Gauss integration. When trying to model more complicated soil behaivor, beyond a simple Mohr-Coulomb constitutive relation, the stress oscillations are particularly damaging to the accuracy of the model.

The current qualitative analysis is extremely limited. However, it provides an initial confirmation on the implementation of quadrilateral elements and *B-Bar* in Anura3D. Quadrilateral elements, with and without *B-Bar*, give a satisfactory solution compared to the triangle elements with NMD smoothing. In the future, a mesh and MP convergence study needs to be done. The implementation of the quadrilateral elements and *B-Bar* needs to be tested with different model geometries, boundary conditions, and extended to 2D-axisymmetry. Ideally, an analysis similar to [Fern et al., 2019, chap. 9] would be repeated for the column collapse model, and the numerical results would be compared to actual experimental data. Furthermore, a complete implementation of quadrilateral elements opens the possibility of using GiMP which might be a more effective way of mitigating the cell crossing issue.



(a) Triangle elements without NMD strain smoothing

(b) Triangle elements with NMD strain smoothing



(c) Quadrilateral elements without B-Bar strain smoothing

(d) Quadrialteral elements with B-Bar strain smoothing

Figure 3.3: Zoomed in perspective of the column collapse models.

Chapter 4

Constitutive Modeling

4.1 Introduction

The accurate and efficient evaluation of constitutive equations are required to accurately predict local and global material response in numerical models [Brannon et al., 2009]. Soil constitutive models tend to be highly non-linear. Therefore, calculating the stress increment ($d\boldsymbol{\sigma}$) and state parameters ($\boldsymbol{\chi}$) updates is generally an iterative procedure. The process of doing this update is called stress integration. Stress integration is required for all MPs and/or Gauss points and can be a significant portion of the computational expense of a numerical model [Li, 2023]. Therefore, this chapter compares the accuracy and computational efficiency of the two stress integration methods, an *implicit return mapping* algorithm [Ortiz and Simo, 1986] and an *explicit substepping* algorithm [Sloan et al., 2001] for the Strain Softening Mohr Coulomb (*SSMC* , Yerro Colom [2015]) and the Strain Rate Mohr-Coulomb (*SRMC*, Zambrano-Cruzatty et al. [2022]) constitutive models. Details of both constitutive models are presented in this chapter. The final goal of this analysis is to determine which stress integration algorithm should be used when running the PFFP impact problem.

The *implicit return mapping* algorithm was implemented for the *SSMC* and *SRMC* models as part of this thesis. The original implementation of the *SSMC* and *SRMC* models with the *explicit substepping* algorithm was provided to the author by Dr. Alba Yerro-Colom and Dr. Luis Zambrano-Cruzzatty. Next before presenting the comparison, background on general elastic-plastic/elasto-plastic theory and its application to soil constitutive relations is provided.

4.2 Consistency equation and the yield function

To determine the portions of $d\boldsymbol{\varepsilon}$ that is elastic ($d\boldsymbol{\varepsilon}^e$) and plastic ($d\boldsymbol{\varepsilon}^p$) a yield function (4.1) is required.

$$F = F(\boldsymbol{\sigma}, \boldsymbol{\chi}) = 0 \tag{4.1}$$

Inside the surface ($F < 0$) the material deforms elastically. On the yield surface it deforms plastically ($F = 0$). Being outside of the yield surface is not admissible $F \not> 0$ and, therefore, the differential of the yield surface is zero $dF = 0$ during plastic deformation. dF is defined in equation 4.2. The combination of equation 4.1 and 4.2 is known as the consistency condition [Brannon et al., 2009].

$$dF = \frac{\partial F}{\partial \boldsymbol{\sigma}} d\boldsymbol{\sigma} + \frac{\partial F}{\partial \boldsymbol{\chi}} d\boldsymbol{\chi} = 0 \tag{4.2}$$

where, $\frac{\partial F}{\partial \boldsymbol{\sigma}}$ is the gradient of the yield function with respect to stress, $\frac{\partial F}{\partial \boldsymbol{\chi}}$ the gradient of the yield function with respect to the state variables, $d\boldsymbol{\sigma}$ the stress tensor increment, and $d\boldsymbol{\chi}$, the vector of state variable increments. F is a level-set, therefore the gradients are surface normals. $\frac{\partial F}{\partial \boldsymbol{\sigma}}$ is usually denoted as $\vec{n}$. Equation 4.2 only constrains the material state to lie on the yield surface, during plastic loading, the state is free to shift along the yield surface and move with the yield surface, if the surface changes size. The exact form of F and what parameters are chosen for $\boldsymbol{\chi}$ are constitutive model specific. To couple the changes in stress to increments of plastic strain $d\boldsymbol{\varepsilon}^p$ a plastic multiplier λ and plastic potential surface ($\mathcal{P}$) are assumed, this is called a flow rule, equation 4.3 [Wood, 1991].

$$d\boldsymbol{\varepsilon}^p = \lambda \frac{\partial \mathcal{P}}{\partial \boldsymbol{\sigma}} \tag{4.3}$$

where, $\frac{\partial \mathcal{P}}{\partial \boldsymbol{\sigma}}$ is the gradient of the plastic potential with respect the stress, and is usually denoted as $\vec{m}$. If $\mathcal{P}$ is assumed to be the same function as F then the constitutive model is called associative. If the surfaces are different, then the model is non-associative [Muir Wood, 2017, Wood, 1991, Simo and Hughes, 1998].

4.3 Stress integration algorithms

Computer implementations of stress integration algorithms are generally not able to completely satisfy the consistency condition due to a combination of machine floating point error and finite computational time. Therefore, in stress integration algorithms a tolerance (*FTOL*) around the yield surface is defined. Additionally, a limited number of iterations (*MAXITERS*) are set for this condition to be reached. This relaxes the consistency condition, equation 4.4 [Brannon et al., 2009].

$$|F| \leq \text{FTOL and } i < MAXITERS \tag{4.4}$$

4.3.1 Implicit Return Mapping and Explicit substepping algorithms

The *implicit return mapping* algorithm, is an implicit algorithm that uses $d\boldsymbol{\varepsilon}$ to elastically predict the updated stress state and from there, iterates back until the consistency condition is satisfied. Ortiz and Simo [1986] provides a detailed derivation of the method and pseudocode for its rate-independent and viscoplastic implementation. Theoretically, the algorithm gives quadratic convergence. In Ortiz and Simo [1986], the algorithm was implemented for a Cam-Clay model and was found to give good convergence and error results.

The *explicit substepping* algorithm explicitly updates the stress and state variables in the plastic region and then does a substepping routine to ensure the consistency condition is satisfied [Sloan et al., 2001]. The algorithm gives direct control on the allowed error and provides error estimates. The *explicit substepping* algorithm is commonly used in the soil mechanincs field due to its accuracy and computational efficiency. In addition to the parameters mentioned in Section 4.3 (*FTOL, MAXITERS*), this algorithm requires a minimum pseudo-timestep (*DTmin*) and relative error tolerance (STOL). The tolerances are used to control the increment of substepping and error between two measures of the correct stress,

respectively.

4.4 Error Analysis

The constitutive models are tested using a single element software called *IncrementalDriver*. *IncrementalDriver* is a Fortran 90 code, designed to test and calibrate soil constitutive models under arbitrary stress and strain paths. Single element tests assume a homogeneous block of material with a homogeneous stress and strain state. To use *IncrementalDriver*, the user links their Fortran or C/C++ constitutive model with the *IncrementalDriver* object file, generating the *IncrementalDriver* executable. The user supplies the executable with input files, boundary conditions, initial state, material parameters, yield surface parameters, and the wanted stress/strain path. For a strain controlled K_0-triaxial test the required input parameters are defined in list 4.4. More detailed information can be found in [Niemunis, 2019, Niemunis and Grandas-Tavera, 2017].

- *ddstran* - Final axial strain the test will be run until
- *ninc* - The number of axial strain increments ($d\varepsilon_a$). This determines the size of the strain increments passed to the constitutive model during each step
- *maxiter* - The maximum number of iterations that should be done to ensure the lateral stress boundary conditions are maintained. Currently, *IncrementalDriver* does not have the ability to set a convergence tolerance for the iterations that tries to maintain the lateral stresses. This means this value has to be set to a high value to ensure the stresses have converged.
- *dtime* - The time increment that corresponds to each of the calculated $d\varepsilon_a$ steps. This is used to calculate the strain rate for models that are explicitly strain rate dependent.

To simplify the process of generating input files, running the executable, and processing results, a Python library was developed during this thesis. The library, called *pumat* is available on GitHub[1]. Currently the python library can only generate the input files for what *IncrementalDriver* calls "predefined popular paths". However, *pumat* was built in an object-oriented format that makes the extension to the other test types easy and is planned for the future. The main branch of *IncrementalDriver* is freely available under a gnu license at SoilModel.com. As part of this thesis a fork of the code was made and modified to compile using the Fortran package manager (fpm). Making the compilation, theoretically, operating system agnostic. The fork is available here.

4.4.1 Error Test Setup

The constitutive models were tested using a set of strain controlled K_0-triaxial tests. In a strain controlled test, the soil sample is intialized with an initial stress state, then the vertical axial strain (ε_a) is increased incrementally. During each strain increment the lateral deformation is determined through an iterative procedure so that the initial lateral stresses is maintained. For the models run here, the initial vertical and lateral boundary compressive stresses were 14.3 kPa and 8.2 kPa, respectively. The stresses were calculated using the material parameters provided in Table 4.1 assuming dry conditions. The triaxial tests were run to a final axial strain of 40%, to push the model into the large deformation domain.

The material parameters for the model are provided in Table 4.1. In the case of *SRMC* , the value listed for G and ν are the initial values before strain rate affects are accounted for. The friction angle was assumed to correspond to a stress ratio of one, $M = q/p = 1$. The elastic properties were assumed, but are representative of a possible soil. The assumed density, porosity, and K_0 values, were used to calculate the initial boundary stresses. A depth of 1 m was assumed.

[1]PUMAT was heavily inspired by a library of the same name developed by Luis Zambrano-Cruzatty.

Table 4.1: Material Parameters For Stress Integration Testing

Property	**Value**
Shear Modulus (G) [kPa]	1042
Poisson's Ratio (ν) [-]	0.2
Friction Angle (ϕ') [°]	25.4
Cohesion (c) [kPa]	0.0
Particle Density (ρ_s) [$\mathrm{kg\,m^{-3}}$]	2650
Porosity (n) [-]	0.45
K_0 [-]	0.57

For each constitutive model, the triaxial models were varied in two ways. First, as *SSMC* and *SRMC* are complex, and it is common not to use the models full complexity, the models were tested starting with their simplest form and the level of complexity was increased incrementally to their most complex form. In the case of the *SSMC* model, this meant going from an elastic-perfectly plastic behavior to an elasto-plastic with strain softening behavior with dilatancy. Similarly, for the *SRMC* model, this meant going from elastic-perfectly plastic to visco-elasto-plastic with stress-dilatancy coupling ("visco" is incorporated to account for strain rate effects).

Second, the number of axial strain increments (*ninc*) was varied to account for different strain increments sizes. This was done to test the effect of the strain increment size on the convergence of the algorithm. The maximum *ninc* selected for testing corresponded to the strain increment for which further subdividing did not show significant change in the calculated results. The reason for testing the effect of *ninc* , is in full-fledged MPM codes, the size of the time step can change to satisfy the stability CFL condition in explicit integration schemes [Courant et al., 1967]. Therefore, having an understanding of how the stress integration algorithm performs under small and large strain increments is important.

An assumption had to be made on which algorithm was more "correct" to evaluate the relative error. In this paper the *explicit substepping* algorithm was assumed to be "correct". This choice was made because Sloan et al. [2001] and Potts and Ganendra [1994] found the

explicit substepping algorithm to be accurate for the constitutive models they tested. The set of constitutive models tested did not include *SRMC* and *SSMC* . Therefore, for each of the model runs, the relative error is calculated using *explicit substepping* with the highest number of *ninc* as the correct solution. The tolerances for the stress integration algorithms used in the analyses are presented in Table 4.2.

Table 4.2: Yield Surface Parameters

Property	**Value**
Yield Surface Tolerance ($FTOL$)	$1e-8$
implicit return mapping Maximum integration iterations	1000
explicit substepping Maximum number of substepping integrations	20
explicit substepping Minimum Psuedo-Time Step ($DTmin$)	$1e-8$
explicit substepping User defined relative error tolerance ($STOL$)	$1e-3$

The plots generated for each test set are:

1. Maximum absolute value error vs. average computational time. Each data point on the plot corresponds to increasing number of *ninc* .

2. Absolute error in p and ε_q vs. ε_a

3. A geotechnical quad plot

Average computational time refers to the mean of 10 model run-times for each model. The realization of these plots are presented in their respective sections.

4.5 Strain Softening Mohr-Coulomb

In this section, the error analysis for the *SSMC* model is presented. First, the state parameters for the model are defined. Second, the types of tests performed are listed and defined. Third, the results of the error analysis are presented and discussed. Finally, at the end of the section, the pseudocode for the *implicit return mapping* algorithm for the *SSMC* model is provided.

4.5.1 *SSMC* State Parameters

In a *SSMC* model, the state variables decrease, or "soften", as a function of accumulated plastic strain. Typically, the scalar magnitude of the equivalent plastic strain ε_q^p is selected as a good representation of the plastic strain tensor, equation C.48. The state variables ($\boldsymbol{\chi}$) for the *SSMC* model are the effective friction angle (ϕ', [°]), dilatancy angle (ψ, [°]), and effective cohesion (c, [kPa]). All three parameters are bound between a peak $(\cdot)_p$, and a residual value $(\cdot_r)$, and are controlled with equation 4.5

$$(\cdot) = (\cdot)_r + [(\cdot)_p - (\cdot)_r] \exp(-\eta \varepsilon_q^p) \tag{4.5}$$

where, η is the user calibrated softening shape factor and $(\cdot)$ represents any of the state parameters [Yerro et al., 2015]. η is mesh size dependent.

4.5.2 Error Analysis Details and Figures

SSMC was tested in elastic-perfectly plastic, elastic-perfectly plastic with dilatancy, elasto-plastic with ϕ' softening, and elasto-plastic with ϕ' and ψ softening conditions. If a parameter is not mentioned in the test name, this means it was set to zero for that test. The state parameter bounds and other required input parameters are given in Table 4.3.

Table 4.3: *SSMC* State Parameters Bounds and other Required Input Parameters

Property	**Value**
Peak Friction Angle (ϕ'_p) [°]	25.4
Residual Friction Angle (ϕ'_r) [°]	15
Peak Dilatancy Angle (ψ_p) [°]	10.0
Residual Dilatancy Angle (ψ_r) [°]	5.0
Peak and Residual Cohesion (c_p, c_r) [kPa]	0.0
Strain Softening Factor (η) [-]	2.5

For *SSMC* , the results were found to converge for *ninc* = 5000. Therefore, the *explicit substepping* algorithm with *ninc* = 5000 is used as the basis for calculating the relative error and thus always has zero relative error. To compare the effect of the strain increment size, two other *ninc* values were used, i.e., 500 and 1000. For the *SSMC* tests, the *IncrementalDriver* parameters are summarized in Table 4.4 [2]. *dtime* is not applicable to the model because *SSMC* is not strain rate dependent.

Table 4.4: *SSMC IncrementalDriver* Parameters For Stress Integration Testing

Property	**Value**
ddstran	-0.4
ninc	500, 1000, 5000
maxiter	100
dtime	N/A

Figure 4.1 compares the stress and strain tensor error vs. computational time for all the models. For the *SSMC* model, the *explicit substepping* algorithm is more for the stress and for all cases for the strain. The *implicit return mapping* algorithm is faster for the tested cases.

To get a picture of what is contributing to the absoloute error in Figure 4.1, the absoloute error in p and ε_q vs. ε_a was plotted for all models, Figure 4.2. Figure 4.2a is the error for the elastic-perfectly plastic with and without dilatancy. From that figure, it's clear that the

[2]Parameters were defined in list 4.4

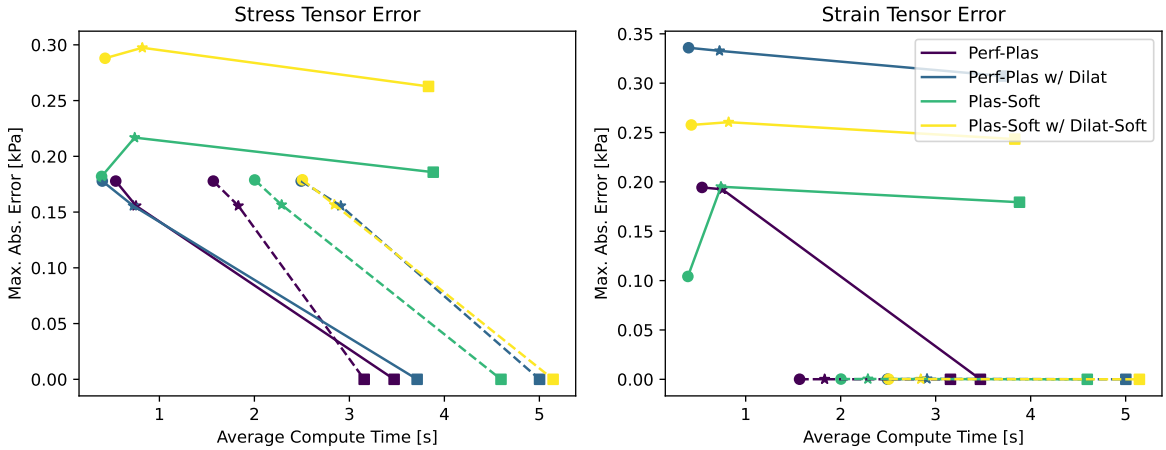


Figure 4.1: Maximum absolute value $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ tensor error for the *SSMC* model. The *explicit substepping* and *implicit return mapping* models are denoted with dashed and solid lines, respectively. The circle, star, and square marks the model with the least, middle, and greatest *ninc*.

main source of error in the stress calculation for the *explicit substepping* algorithm for a small *ninc* and for the *implicit return mapping* algorthim across the range of *ninc* the error comes from determining when the material transitions from being elastic to plastic. However, for strain the *implicit return mapping* algorithm increases error with each axial strain increment. Figure 4.2b is the error for the elasto-plastic with $effFricAng$ smoothing and elasto-plastic with friction and dilatancy angle softening. Unlike the elastic perfectly-plastic cases the error in the *implicit return mapping* algortithm stress calculation does not decrease after the transition from elastic to plastic is complete. Instead, the error continues to grow until a peak. The error in the strain follows a similar trend as before.

As the errors being presented are in terms of absoloute error, a geotechnical quad plot for each model has been included to put the test and the error into context. Only the geotechnical quad plot for the *SSMC* model with ϕ' and ψ softening is presented in this section, Figure 4.3. The quad plots for the elastic perfectly-plastic (Figure A.1), elastic perfectly-plastic with dilatancy (Figure A.2), and elasto-plastic with ϕ' softening (Figure A.3) are provided in the appendix. The error in the elastic-plastic cases are not visually

noticeable for the *implicit return mapping* or *explicit substepping* algorithm. The error in the elasto-plastic cases using *implicit return mapping* are. With softening the *implicit return mapping* and *explicit substepping* algorithms diverge. The code used to generate these results are available on Github.

4.5.3 *SSMC* Pseudo-code for the *implicit return mapping* algorithm

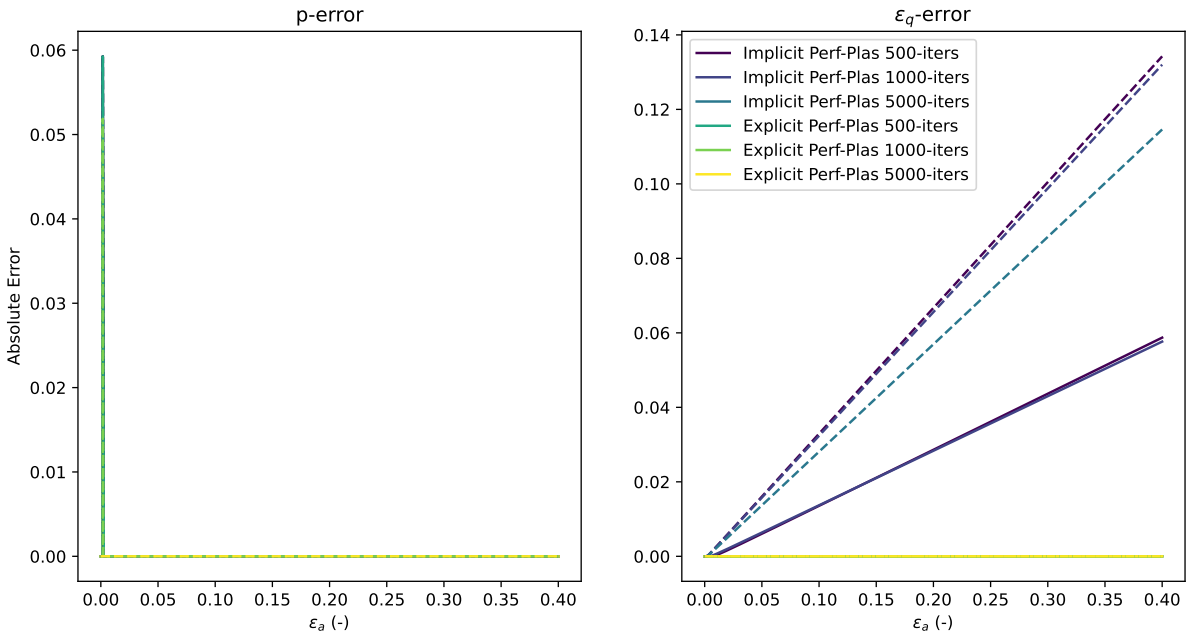
The pseudocode for the *SSMC implicit return mapping* algorithm is provided in 1. To ease notation the following definitions are noted in equation 4.6. Variables with a subscript u are the updated variables for that iteration.

$$\boldsymbol{\chi} = [\phi', \psi, c']; \boldsymbol{m} = \frac{\partial \mathcal{P}}{\partial \boldsymbol{\sigma}}; \boldsymbol{n} = \frac{\partial F}{\partial \boldsymbol{\sigma}} \tag{4.6}$$

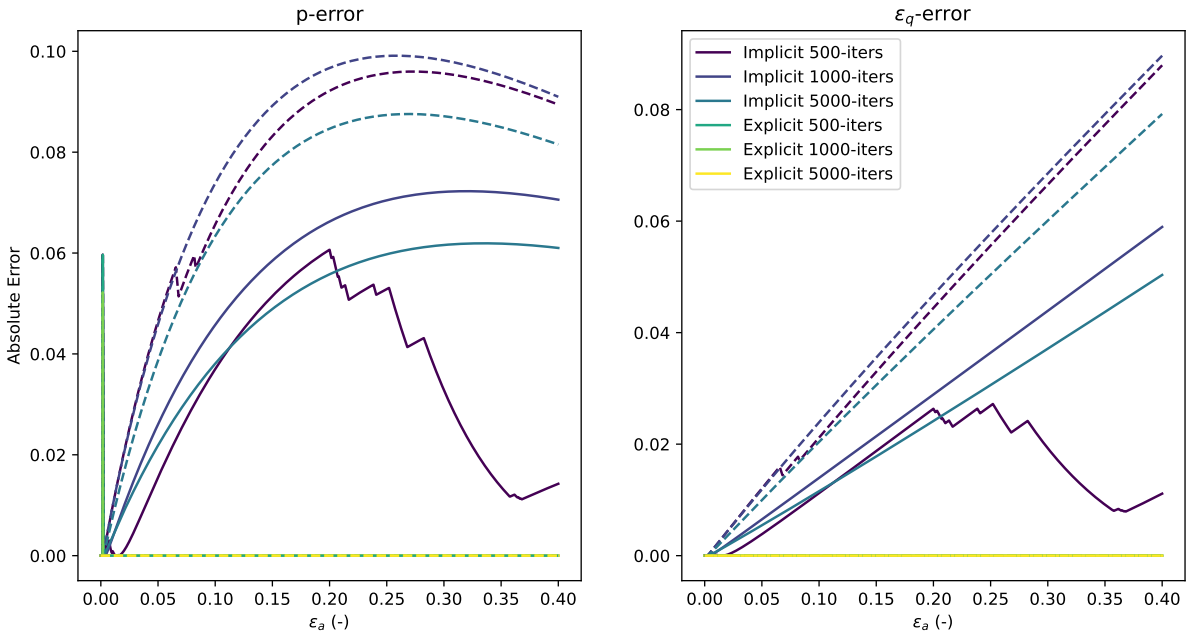
Algorithm 1 Pseudocode for *SSMC implicit return mapping* algorithm

Data: $\boldsymbol{D}$, $\boldsymbol{\sigma}_0$, $\boldsymbol{\chi}_0$, $\boldsymbol{\varepsilon}^p$, $d\boldsymbol{\varepsilon}$
Result: Updated $\boldsymbol{\sigma}_u$, $\boldsymbol{\chi}_u$, $\boldsymbol{\varepsilon}^p_u$

```
begin
Predict σ_u ← σ_0 + D : dε
χ_u ← χ_0
F_u ← F(σ, χ_0)
if F_u ≤ 0 then
    RETURN σ_u, χ_u, ε^p_u
end if
i ← 0
while |F_u| ≤ FTOL and i < MAXITERS do
    n_u ← n(σ_u, χ_u)
    m_u ← m(σ_u, χ_u)
    H ← ∂F_u/∂ε^p_u · m_u
    dλ ← F_u / (n_u : D : m_u − H)
    σ_u ← σ_u − dλ · D : m_u
    ε^p_u ← ε^p_u + dλ · m_u
    χ_u ← Update χ using equation 4.5
    F_u ← F(σ_u, χ_u)
    i ← i + 1
end while
RETURN σ_u, χ_u, ε^p_u
```



(a) The error for the **elastic-perfectly plastic model** is marked with solid lines. The error for the **elastic-perfectly plastic with dilatancy model** is marked with dashed lines.



(b) The error for the **elasto-plastic softening model** is marked with solid lines. The error for the **elasto-plastic softening and dilatancy softening model** is marked with dashed lines.

Figure 4.2: Absolute error for each strain increment for mean stress (p) and deviatoric strain (ε_q) invariant

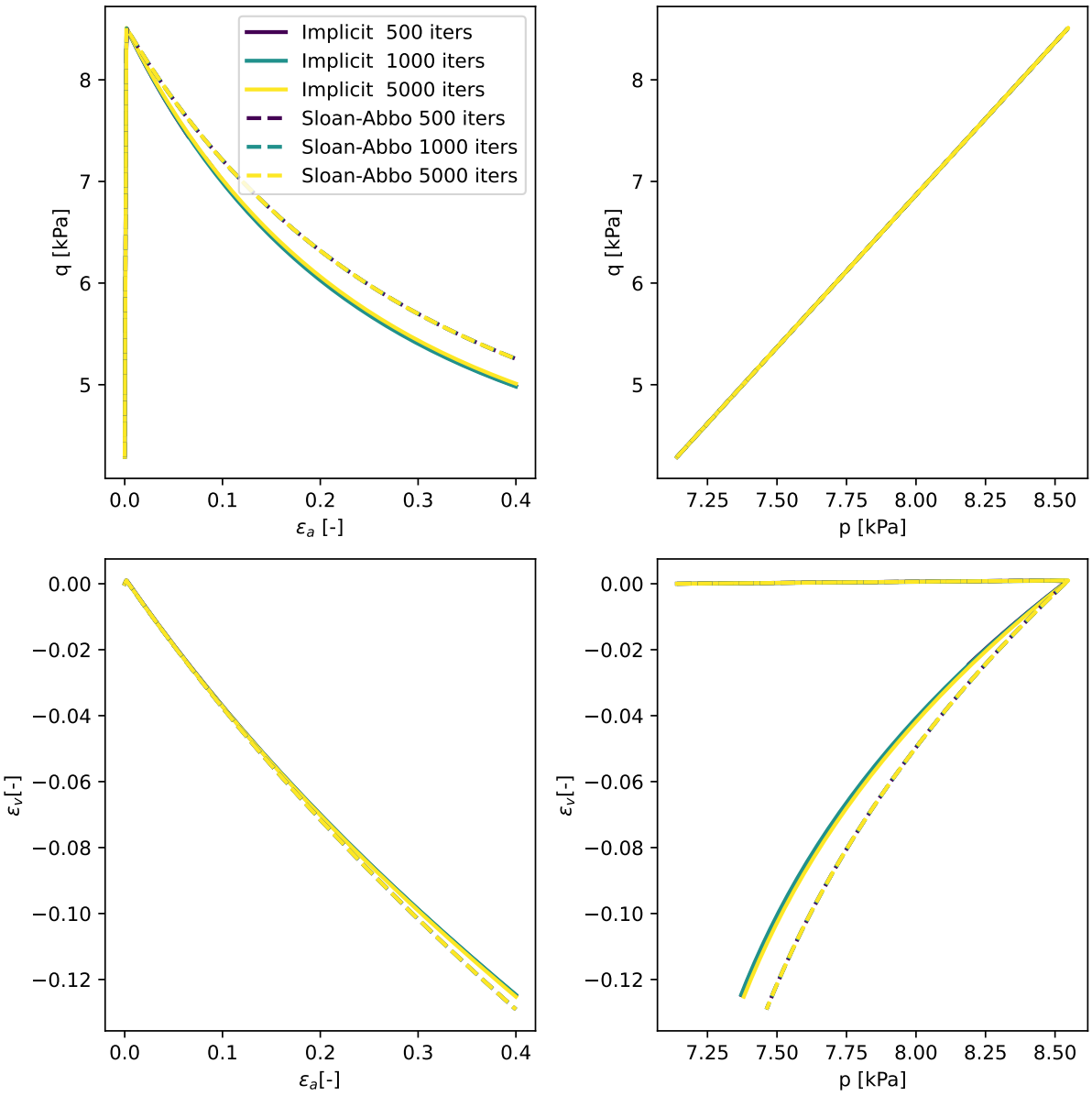


Figure 4.3: Elasto-Plastic with dilatancy and ϕ' softening quad plot

4.6 Strain Rate Dependent Mohr-Coulomb

The *SRMC* constitutive relation [Zambrano-Cruzatty et al., 2022] is a more complex constitutive relation than the *SSMC* model. The additional complexity of the *SRMC* model comes in two forms. The simpler one, is a coupling between the friction and dilation of the material through a stress-dilatancy equation. The coupling happens through a mobilized stress ratio (η_y), equation 4.7,

$$\eta_y = M - D^p(1 - N) \tag{4.7}$$

where, M is the critical stress ratio for shearing at constant volume, $D^p = d\varepsilon_v^p / d\varepsilon_q^p$ is the plastic dilatancy, N is Nova's volumetric coupling coefficient, and $d\varepsilon_v^p$, $d\varepsilon_q^p$ are the increments of plastic volumetric and deviatoric strain, respectively. M is calculated from the triaxial compression critical stress ratio, M_{tc}, and Lode's angle to account for the M's dependence on the angle of loading, equation 4.8 [Muir Wood, 2017]. D^p is calculated using equation 4.9.

$$M = M_{tc}\left(1 + \frac{1}{4}\cos\left[\frac{3\theta}{2} + \frac{\pi}{4}\right]^{1.2}\right) \tag{4.8}$$

$$D^p = D^p_{min} h \varepsilon_q^p \exp(1 - h\varepsilon_q^p) \tag{4.9}$$

where, D^p_{min} is the minimum plastic dilatancy, ε_q^p is the plastic deviatoric strain invariant, and h is the hardening parameter.

The second main difference is that *SRMC* is strain rate ($\dot{\boldsymbol{\varepsilon}}$) dependent. This means that when the material is loaded at a faster rate, the constitutive relation provides a stiffer response. The effect is incorporated by making the shear modulus (G), bulk modulus (K), and minimum dilatancy (D^p_{min}) dependent on the rate of deviatoric strain invariant ($\dot{\varepsilon_q}$). $\dot{\varepsilon_q}$

is included in a non-dimensional number that coins the inertial coefficient (I) by equation A.2 [Zambrano-Cruzatty et al., 2022],

$$I = D\varepsilon_q \sqrt{\frac{G_s}{|p'|}} \tag{4.10}$$

where, D is the particle diameter [m], ε_q is the deviatoric strain rate [s^{-1}], G_s is the soil particle's specific gravity [-], and $|p'|$ is the absolute value of the mean effective stress [kPa]. I accounts for the "effect of mobilizing the weight of the particle" at the enforced strain rate $\dot{\varepsilon_q}$ normalized by the confinement pressure p'[3] [Zambrano-Cruzatty et al., 2022]. Then, using I, a power law is used to update the strain rate dependent parameters with equation 4.11.

$$(\cdot) = (\cdot)_0 (\frac{I}{I_0})^{k_{(\cdot)}} \tag{4.11}$$

where, $(\cdot)$ is a strain rate dependent parameter, $(\cdot)_0$ is the parameter's value at the reference rate, I_0 is the reference strain rate, and $k.$ is the viscosity or power law index for the state parameter.

4.6.1 Error Analysis Details and Figures

To test the *SRMC* constitutive model, different numerical tests were run in *IncrementalDriver* while activating different levels of complexity: elastic-perfectly plastic, elasto-plastic with dilatancy, elastic-perfectly plastic with rate effects, and elasto-plastic with rate and dilatancy. Similar to section 4.5.2, if a parameter is not mentioned in the test name, this means that it was set to zero for that test. The state parameter bounds and the other parameters for the *SRMC* test are provided in Table 4.5. The code used to run the *IncrementalDriver* models and generate the error plots for the *SRMC* model is available on

[3] In Zambrano-Cruzatty et al. [2022], the equation for I uses the particle's density (ρ_s) instead of G_s. Equation A.2 is used in the Fortran implementation to account for *IncrementalDriver* and *Anura3D* using kPa as the unit of stress.

Table 4.5: *SRMC* State parameters Initial Values and Other Required Inputs

Property	**Value**
Initial Shear Modulus (G_0) [kPa]	1042
Initial Poisson's Ratio (ν) [-]	0.2
Critical Stress ratio in triaxial compression (M_{tc}) [-]	1.0
Nova's volumetric coupling coefficient	0.0
Initial Minimum Dilatancy (D^0_{min}) [-]	-0.2
Hardening Parameter (h) [-]	20
Shear Viscosity Coefficient (k_G) [-]	0.04
Bulk Modulus Viscosity Coefficient (k_K) [-]	0.1
Dilatancy Viscosity Coefficient (k_D) [-]	0.04
Specific Gravity (G_s) [-]	2.68
Reference deviatoric strain rate $\dot{\varepsilon_q}$ [-]	$2.5e-5$

Table 4.6: *SRMC IncrementalDriver* Parameters For Stress Integration Testing

Property	**Value**
ddstran	-0.4
ninc	500, 1000, 5000, 20000
maxiter	100
dtime	0.04

GitHub.

For the *SRMC* model, the results converged for $ninc = 20000$. The *IncrementalDriver* parameters are summarized in Table 4.6. The selected *dtime* and *ddstran* results in a strain rate of $1000\,\%\,\mathrm{s}^{-1}$. Figure 4.4 compares the stress and strain error vs. computation time for all models. From this figure it seems both the *implicit return mapping* and *explicit substepping* algorithms converge towards a relative of zero with increasing *ninc*. For the strain, the error is particularly small as the plot is scaled by a factor of 10^{-5}.

However, when looking at the error in p and ε_q with each strain increment for the elastic-perfectly plastic and elasto-plastic with dilatancy (Figure 4.5a) and for elasto-plastic with a strain rate dependency and elasto-plastic with strain rate dependency and dilatancy (Figure 4.5b) the behaivor of the error for the *implicit return mapping* and *explicit substepping* algorithms are different. The *implicit return mapping* models have a tendency to have more

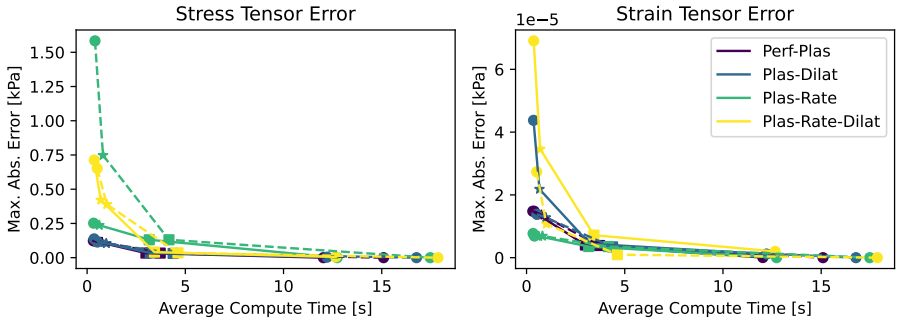


Figure 4.4: Maximum absolute value $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ tensor error for the *SRMC* model. The *explicit substepping* models are the dashed lines. The *implicit return mapping* models are the solid lines.

bounce. Bounce refers to the error decreasing close to zero but then rebounding back up. Particularly, the bounce seems to happen more often for models with diltancy. A possible explanation is the dilatancy (D^p), as a function of the plastic strain, is only updated at the end of the *implicit return mapping* iterations. A little more detail about that choice is given in section 4.6.2.

At lower *ninc* the *explicit substepping* algorithm has a tendency to maintain a constant error. This happens for the $ninc = 500$ and $ninc = 1000$ cases in the elasto-plastic rate dependent model, Figure 4.5b.

To put the error in context, a geotech quad plot for each model was created. The one for the strain rate with diltancy model is presented in Figure 4.6. The other *SRMC* model quad plots are presented in Appendix A.2. For the elastic perfectly-plastic (Figure A.4), elasto-plastic with diltancy (Figure A.5), and elasto-plastic with dilatancy (Figure 4.6) models the visual difference between the *implicit return mapping* and *explicit substepping* results are negligible. For the elastic-perfectly with strain rate affects (Figure A.6) the *explicit substepping* algorithm diverges for $ninc = 500$ and $ninc = 1000$. This agrees with the error seen in Figure 4.5b plot.

4.6.2 *SRMC* Pseudo-code for the *implicit return mapping* algorithm

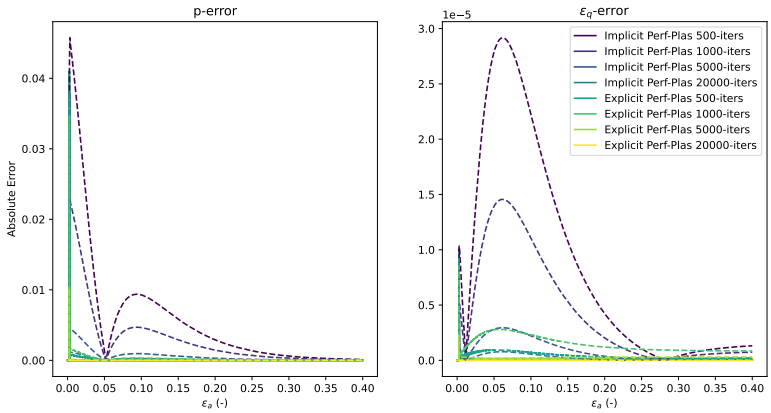
Equation 4.12 is the vector of state variables for the *SRMC* constitutive model. As described in Ortiz and Simo [1986], all the state parameters dependent on the plastic strain ($\boldsymbol{\varepsilon}^p$) should be updated each iteration. For the *SRMC* model, updating D^p during each iteration caused spurious oscillations. Therefore, D^p, as a function of $\boldsymbol{\varepsilon}^p$, is updated at the end once the consistency equation is satisfied. This is reflected in the psuedo-code 2.

$$\boldsymbol{\chi} = [G, K, M, D^p, I, N] \tag{4.12}$$

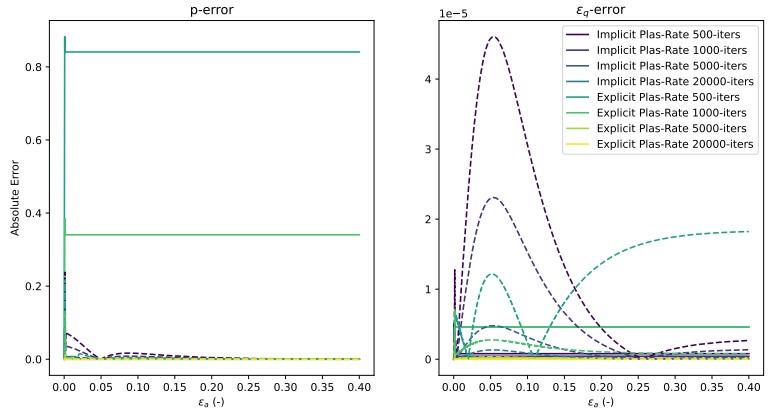
Algorithm 2 Pseudocode for *SRMC implicit return mapping* algorithm

Data: $G_0, K_0, M, \boldsymbol{\varepsilon}^p, d\boldsymbol{\varepsilon}, I_0, k_k, k_G, k_{D^p}, N$
Result: Updated $\boldsymbol{\sigma}_u$, $\boldsymbol{\chi}_u$, $\boldsymbol{\varepsilon}^p_u$

begin
$\boldsymbol{\chi}_u \leftarrow \boldsymbol{\chi}_0; \boldsymbol{\varepsilon}^p_u \leftarrow \boldsymbol{\varepsilon}^p_0; \boldsymbol{\sigma}_u \leftarrow \boldsymbol{\sigma}_0$
Update inertial coefficient (I_u) using equation A.2
Check if I_u mandates internal updates
if Apply Intertial Updates **then**
$\boldsymbol{\chi}_u \leftarrow \boldsymbol{\chi}(I, \boldsymbol{\varepsilon}^p)$ using equation 4.11
end if
$M_u \leftarrow M(\boldsymbol{\sigma}_u)$ using equation 4.8
$\eta_{y,u} \leftarrow \eta_y(\boldsymbol{\chi}_u, \boldsymbol{\sigma})$
Construct $\boldsymbol{D} \leftarrow \boldsymbol{D}(G, K)$
Predict $\boldsymbol{\sigma}_u \leftarrow \boldsymbol{\sigma}_0 + \boldsymbol{D} : d\boldsymbol{\varepsilon}$
$\eta_{y,u} \leftarrow \eta_y(\boldsymbol{\chi}_u, \boldsymbol{\sigma}_u)$
$F_u \leftarrow F(\boldsymbol{\sigma}_u, \boldsymbol{\chi}_u)$
if $F_u \leq 0$ **then**
RETURN $\boldsymbol{\sigma}_u$, $\boldsymbol{\chi}_u$, $\boldsymbol{\varepsilon}^p_u$
end if
$i \leftarrow 0$
while $|F_u| \leq FTOL$ and i $< MAXITERS$ **do**
$\mathbf{n}_u \leftarrow \mathbf{n}(\boldsymbol{\sigma}_u, \boldsymbol{\chi}_u); \mathbf{m}_u \leftarrow \mathbf{m}(\boldsymbol{\sigma}_u, \boldsymbol{\chi}_u)$
$H \leftarrow \partial F_u / \partial \boldsymbol{\varepsilon}^p_u \cdot \boldsymbol{m}_u$
$d\lambda \leftarrow F_u / (\boldsymbol{n}_u : \boldsymbol{D} : \boldsymbol{m}_u - H)$
$\boldsymbol{\sigma}_u \leftarrow \boldsymbol{\sigma}_u - d\lambda \cdot \boldsymbol{D} : \boldsymbol{m}_u$
$M_u \leftarrow M(\boldsymbol{\sigma}_u)$ using equation 4.8
$\boldsymbol{\varepsilon}^p_u \leftarrow \boldsymbol{\varepsilon}^p_u + d\lambda \cdot \boldsymbol{m}_u$
$\eta_{y,u} \leftarrow \eta_y(\boldsymbol{\chi}_u, \boldsymbol{\sigma}_u)$
$F_u \leftarrow F(\boldsymbol{\sigma}_u, \boldsymbol{\chi}_u)$
$i \leftarrow i + 1$
end while
$D^p_u \leftarrow D^p(\boldsymbol{\varepsilon}^p)$ using equation 4.9
RETURN $\boldsymbol{\sigma}_u$, $\boldsymbol{\chi}_u$, $\boldsymbol{\varepsilon}^p_u$



(a) The error for the **elastic perfectly-plastic model** is marked with **solid lines**. The error for the **elasto-plastic with dilatancy model** is marked with **dashed lines**.



(b) The error for the **elasto-plastic rate dependent model** is marked with **solid lines**. The error for the **elasto-plastic rate dependent model with dilatancy** is marked with **dashed lines**.

Figure 4.5: Absolute relative error in mean stress (p) and deviatoric strain (ε_q) invariant for each axial strain increment (ε_a)

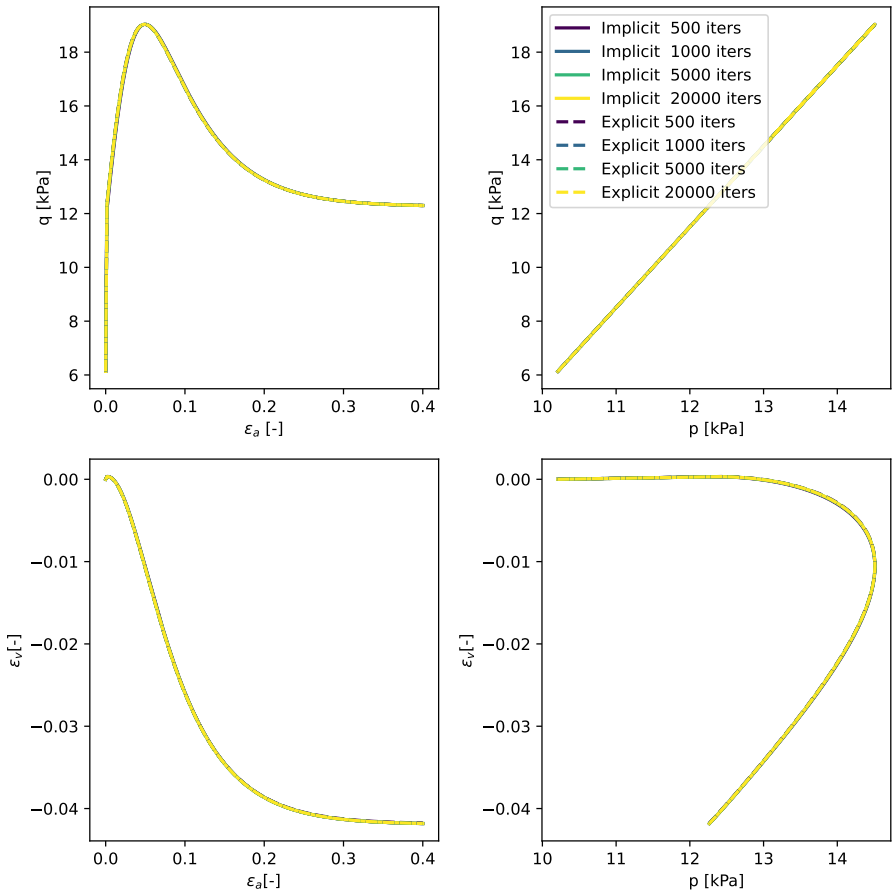


Figure 4.6: Elasto-Plastic with rate and dilatancy quad plot. *implicit return mapping* results in solid lines and *explicit substepping* results in dashed lines

4.7 *SSMC* and *SRMC* Error Analysis Discussion

For the tests presented here, the *implicit return mapping* algorithm converges in less time but is less accurate than the *explicit substepping* algorithm. This agrees with the findings of other authors [Sloan et al., 2001, Potts and Ganendra, 1994]. One notable exception is the elastic-perfectly plastic test for the *SRMC* model (Figure 4.5b). For that test the *explicit substepping* algorithm has significantly more error for the lower *ninc* tests. The peak in the stress and strain error at low axial strain is caused by the stress integration algorithm not correctly determining the axial strain the material transitions from elastic to plastic. This problem occurs for both algorithms. This could be problematic in a full numerical model. The stress integration algorithms should be tested for other loading cases before final conclusions are made.

Additionally, the current timing procedure has a flaw. In *IncrementalDriver*, maintaining the boundary stresses requires determining which strain results in that stress. To accommodate this, *IncrementalDriver* contains an iterative procedure that tests different lateral strains. Ideally, this would be done with a combination of error tolerance on the lateral stress and a limit on the number of iterations. However, currently, *IncrementalDriver* only limits the number of iterations, *maxiter*. This has two downsides. The first is lateral stress convergence needs to be checked manually. The second is that no matter what the error in the lateral stress is, *IncrementalDriver* will calculate all iterations until *maxiter*. Therefore, if one of the stress integration algorithms can converge to the lateral stress faster, this effect is not noticed. This could have resulted in higher computational times for one or both stress integration schemes. Therefore, if future tests use *IncrementalDriver*, an error tolerance for satisfying the stress boundary conditions must be implemented.

Finally, additional tests in an MPM or FEM software are still needed. FEM tests would be able to highlight the algorithm's efficiency and accuracy for small strain tests. MPM tests

would be able to highlight how the stress integration algorithms handle large deformation problems and the challenges unique to MPM.

Chapter 5

Field Data Analysis

5.1 BlueDrop Analysis Automation

The second prong of this thesis focuses on developing and testing a consistent, centralized code to automatically process, collect, and store data from actual BlueDrop drops. Currently, standard practice requires each user processing BlueDrop data to develop their own code and storage method, often involving manual processing of each drop. This fragmented approach introduces errors and inconsistent biases, complicating both the analysis of historic data and the development of new empirical relationships. Moreover, the diversity in data storage methods further complicates reanalysis by new users.

An automated, centralized code that integrates with a database, such as SQLite, would eliminate many of these challenges. First, automatic processing reduces the processing bias introduced by individual users. Second, a single code base enables the adoption of advanced software development practices, such as unit testing and system testing. Third, centralizing outputs into a uniformly formatted database facilitates systematic comparisons between historic data and numerical model results. Additionally, a well-structured database opens new possibilities for employing machine learning techniques to derive correlations between BlueDrop data and geotechnical properties. These objectives have been achieved using a Python library called *BlueDrop Analysis Library*.

The development of this library required achieving three subgoals. First, an algorithm—termed *drop detection*—was created to determine whether a file contains a drop and, if so, to count the number of drops present. Second, a method for accurately identifying the timestamps corresponding to the release of the BlueDrop, the onset of impact with the seabed, and the point at which the BlueDrop comes to rest after impact was established; this procedure is referred to as *drop selection*. At the same time, the construction of data structures to facilitate efficient and organized data processing was required.

5.1.1 Drop detection

The major hurdle to drop detection is that bumps and jostles, that are not drops, cause spikes in the vertical acceleration data that are hard for a computer to differentiate from a impact into the seabed. To address the problem, approximately 300 files were collected, containing a mix of cases with and without impact events. A tolerance parameter for a peak-finding algorithm was then calibrated to detect drop impacts within the vertical acceleration signal. Additionally, for water drops, pressure sensor data was utilized to provide an independent verification of the detected acceleration peaks.

When the BlueDrop device impacts the seabed, a change in the slope of the pressure signal occurs. This transition, marked by the dashed lines in Figure 1.3a, delineates the start and end of impact. However, a direct correlation between peaks in the acceleration and pressure measurements was not feasible. The magnitude of the pressure readings can vary by orders of magnitude depending on water depth, making it impractical to establish a fixed tolerance for peak detection.

To address this challenge, peak identification was performed on the time derivative of the pressure signal. Prior to computing the derivative, a window-smoothing technique was applied to mitigate spurious peaks caused by noise. The algorithm was implemented within the *analyze_file* function of the *pffpFile* class. The following pseudocode outlines the procedure for air drops:

1. Merge the vertical accelerometer readings into a single array with optimal accuracy for the current measurement's magnitude.

2. Set a tolerance for what acceleration magnitude is considered to be a peak.

3. Find the peaks in the acceleration file.

4. The found peaks are considered to be the deceleration peaks during impact.

The pseudocode for water drops starts similarly, but an additional check using the pressure sensor is used:

1. Do steps 1-3 of the air drop pseudocode, 5.1.1
2. Smooth the pressure reading using window averaging
3. Calculate the time derivative of the smoothed pressure reading
4. Set a tolerance for what is considered a peak in the smoothed pressure reading
5. Find the peaks in the smoothed pressure data and the time that they occurred
6. IF the number of pressure peaks does not equal the number of acceleration peaks, give a warning
7. ELSE IF the number of acceleration and pressure peaks match
   (a) Check that the time that the acceleration and pressure peaks occur are within a tolerance.
   (b) IF the times align
      i. The selected peaks are considered to be peaks corresponding to drops
      ii. RETURN the peak information and the times that they occur

The tolerances were set using various BlueDrop data collected from multiple sites and in different conditions. However, there are occasions when the set tolerance does not find all the drops. This does not happen frequently, but a function was added to the library to allow the manual selection of drops.

5.1.2 Drop Selection

Drop selection is done by finding the release, impact start, and impact endpoints. An algorithm corresponding to each point was required. The release point is found first. The algorithm for finding the release poin was implemented in the *find_release* method in the *Drop* class. The pseudocode of the algorithm is as follows

1. Find the index of the troughs in the PFFPacceleration data, below a set threshold
2. Get the closest trough index to the drop's peak acceleration index
3. FOR indices less than the closest trough index
    (a) Get the acceleration that corresponds to that index
    (b) IF the acceleration, is close to $1g$ the PFFP is at rest
        i. Store the acceleration
4. Of the stored accelerations, pick the acceleration that is closest in time to the drop's peak acceleration
5. Set the release index as the index of the selected acceleration

Next, the start of the impact is found. The algorithm was implemented in the *get_impulse_start* method in the *Drop* object. The pseudocode for finding the start of the impact is

1. Calculate the time derivative of the smoothed acceleration data (The Jerk).
2. Get the index of the first Jerk value above a tolerance. This is equivalent to finding the first time that the Jerk is greater than a tolerance.
3. Map the found Jerk index back to the index that corresponds to the original full acceleration data

4. This mapped index is defined as the impact start index

Finally, the end of the impact is found. The algorithm was implemented in the *get_impulse_end* method in the *Drop* object. The pseudocode for the algorithm is as follows

1. Get the index of where the peak of the drop is

2. Find the first time that the vertical accelerometer returns to a measurement of $1g$

3. The index that corresponds to the found time is taken as the impact end index

5.1.3 Code Structure and Organization

The second broad goal of the *BlueDrop Analysis Library* was to centralize and organize processing so that users can manipulate the data more naturally and efficiently. To accomplish this, an object-orientated programming (OOP) framework was implemented. Objects were created to abstract operations on the folder that contains the BlueDrop binary files (*pffpFolder* class), the BlueDrop binary files (*pffpFile* class), and the drops (*Drop* class). Figure E.1 shows the series of inheritances used to construct the class and the functions contained in each class. The final class that was created, and is still in development, is a class to represent an SQLite database (*pffpDatabase*). That class is used to store the outputs from the library. Some values included in the database are: metadata about each drop, the three points generated during vertical accelerometer analysis, and the configuration properties of the BlueDrop during deployment.

A literature review was done for correlations to the BlueDrop [Stark, 2011, Brilli, 2023, Jaber and Stark, 2023, Albatal, 2018, White et al., 2018]. All functions found were implemented in the library's modules. For documentation on all functions and classes, see BlueDrop Analysis Library Code Documentation.

5.2 Results and Discussion

The library is able to process and generate most of the figures required by day-to-day Blue-Drop analysis. One of the figures the library can generate is the standard acceleration and velocity vs. penetration depth into the seabed plot, 5.1. Example Jupyter Notebooks that show the process of analyzing the data and generation of the figures can be found at BlueDrop Library Examples.

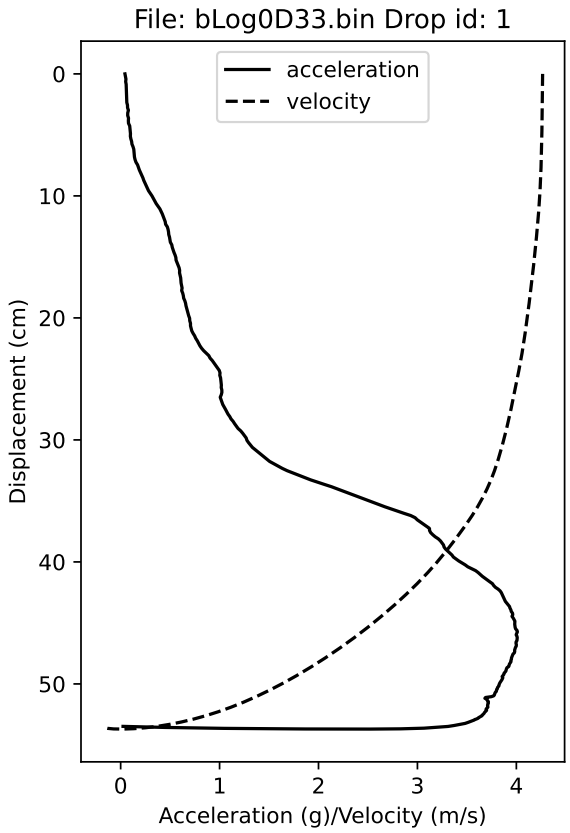


Figure 5.1: Example Acceleration (g)/Velocity (m/s) vs. Penetration Depth (m) generated using the *BlueDrop Analysis Library* for the drop in Figure 1.3a.

A qualitative comparison between the automatic drop start and end selection was done. Data from the York River in Virginia and the Artic were used to do the comparison. The

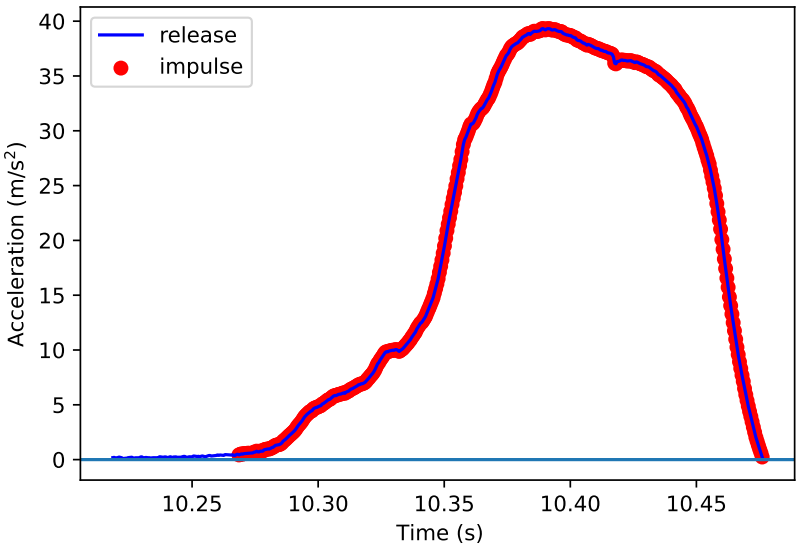


Figure 5.2: Overlay of the selected impulse points (red) with a section of the release data (blue) for the drop in Figure 1.3a

test data included air and water drops in cohesive and non-cohesive soil. Figure 5.2, shows an example, from the *Mouth* of the York River, of where along the acceleration profile the code selected the start and end of the drop. Qualitatively, the results look good. This is a water drop, so, theoretically, the impact begins when the acceleration is greater than 0, and the code selects a point within a close tolerance of that value.

The average time to read in a file, do drop detection, and convert the data to engineering units is 3.49 s per file. It is likely reading the file into memory is a majority of this time. The raw binary file has a column of for the row count, an unknown value, and each of the sensors (7 acceleration sensors and 1 pressure sensor). Each column contains 120,000 rows. Meaning the raw file has 1.2 million data points. If reading in the file is taking up most of the time, the current dataframe library (*pandas*) might be switched out for another dataframe implementation. The average time to process a drop, which includes determining the start and end time of the release and the impact and integrating to get the velocity and displacement, is 0.0074 s. All initial processes and calculations could be parallized, further

decreasing compute time. Unit tests have been implemented for most of the mechanics and empirical equations. Unit tests have not been implemented for the class bound methods.

The current major limitation is that, on rare occasions, files that contain drops are incorrectly selected as having no drops. To circumvent this, the user should plot all drop files and move the files that are incorrectly assigned as having no drops into the funky data folder, and manually add the files to the analysis list. In the future, this issue will be addressed by trying to better calibrate the *drop detection* or implement another algorithm if that does not work.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis worked towards (1) taking initial steps to implement and validate a reliable numerical framework for the simulation of FFP impact and (b) streamlining the data processing of PFFPs. The most important conclusions and takeouts are summarized below.

1. Numerical models of a PFFP using Anura3D software were generated using a Mohr-Coulomb (*MC*) and Strain rate Mohr-Coulomb (*SRMC*) model. This is the first time a strain rate-dependent constitutive model for sands was tested and included in MPM for modeling PFFP-soil impact. The results presented several numerical instabilities, which were attributed to volumetric locking and limited mitigation of the cell crossing issue. The mitigation of such numerical issues served as the inspiration for the other numerical investigations done as part of this thesis. Proper inclusion of a strain rate-dependent model would enable forward estimation instead of requiring backfitting to experimental data.

2. To improve the underlying numerical MPM framework and mitigate volumetric locking, quadrilateral elements and *B-Bar* were implemented into a fork of Anura3D. A theoretical comparison between *B-Bar* and other smoothening techniques was presented. Then, a simple benchmark (i.e, a column collapse model) was run to compare triangle and quadrilateral elements with and without strain smoothing. *B-Bar* and quadrilateral elements seemed to not suffer from the same locking as the no-smoothing triangle element model.

3. In order to contribute to optimizing, standardizing, and validating the implementation of soil constitutive models in MPM, the accuracy and performance of two stress integration methods has been investigated using the *IncrementalDriver* single element test software. In particular, the *explicit substepping* algorithm [Sloan et al., 2001] has been

implemented and compared with *implicit return mapping* [Ortiz and Simo, 1986] for a strain softening Mohr-Coulomb (*SSMC*) and a strain-rate Mohr-Coulomb (*SRMC*) constitutive relation. For tested loading conditions, the *explicit substepping* algorithm was more accurate, especially for the *SSMC* model. The *implicit return mapping* algorithm was faster. Furthermore, a Python library was developed to expedite testing of constitutive models with *IncrementalDriver*.

4. To centralize and enable the reproducibility of BlueDrop PFFP data outputs, a Python library was developed. Using the new library, BlueDrop data files can be automatically loaded and manipulated to generate standard outputs. Initial work was done to integrate the library with a SQLite database. An extensive literature review of equations related to the PFFPs was done. Those equations were included as functions in the library. This library will allow more efficient data processing and comparison with numerical models. This work is the first centralized, generalized, and automatic library for processing and storing BlueDrop data. Additionally, to the author's knowledge, no other library has collected as many empirical functions for transforming PFFP to data geotechnical soil properties. This allows users to more easily validate and calculate their desired parameters.

6.2 Future Work

The work done in this thesis continued to lay the groundwork for gaining a better understanding of PFFP impact into sand. There are several directions for future work on the side of numerical modeling and data analysis. Examples of possible future steps and projects that were began during this thesis but not completed enough to be included in the main text are listed here:

1. The *SSMC* and *SRMC* constitutive models need to be included in Anura3D and PFFP models using those constitutive models need to be created and run. The second is that the presented PFFP models do not consider underwater impacts. A good first pass at approximating underwater impacts would be to apply stress boundary conditions on the PFFP and soil to approximate hydrostatic pressure and fluid drag conditions. The drag force could be approximated using a drag coefficient and the rigid body velocity. A more accurate method would be using a double point formulation. Likely, the extension to a double point formulation would require refactoring of Anura3D.

2. Modeling PFFP impact on slopes, such as the beach, needs to be considered. To accomplish this, a 3D-Cartesian model will likely be required, as axial symmetry won't be present. To solve this in a tractable amount of time Anura3D needs to be parallelized. This will likely require refactoring the code. Towards that goal, work was done to compile a fork of Anura3D for Linux. This fork is not the same fork as the one with quadrilateral elements. The Linux fork must be tested and merged into the main code.

3. The current implementation of the quadrilateral elements and *B-Bar* only works for plane strain conditions and has presented array access errors for models other than the one presented. More testing is required. Once the errors are remedied, the implementation needs to be extended to 2D-axisymmetric conditions. At that point, initial models of PFFP impact could be run. Once that extension is tested, implementation of hexahedral elements is required.

4. The *SSMC* and *SRMC* constitutive models should be tested, using *IncrementalDriver* , under other stress/strain paths. Analytical solutions exist for some elastic-plastic and elastoplastic paths, and those should be tested against, and others should be derived [Brannon et al., 2009, Potts and Ganendra, 1994]. Other paths may reveal bugs or errors in the implementation. The *SRMC* model needs to be re-calibrated

with the experimental data from Yamamuro et al. [2011]. Bugs were found in the implementation during testing for this thesis. The bugs have been listed on GitHub under the issues window and more information is available in Appendix A.3.

5. In the processes of debugging the *SRMC* model, unit tests were created for all functions. Running these tests needs to be automated with GitHub actions. The stress/strain invariant functions and their derivatives should be turned into a library so that all implemented constitutive models in Anura3D can use the same functions. Moving these functions into a library will promote finding bugs and accuracy of results. This can be done using the Fortran package manager (fpm). Before this happens, the functions should be compared to those implemented in the *SSMC* model. The plan is to implement these changes into the CriticalSoilModels GitHub repository. The goal is to build an extendable library of constitutive models and stress integration schemes. Similar work was done by Sandia National Lab in the form of *Kayenta* [Brannon et al., 2009]. *Kayenta* is behind a Department of Energy (DOE) request for an access wall. It is currently unknown to the author if *Kayenta* is available to all members of the public, so *CriticalSoilModels* may be able to supplement or build on its framework.

Bibliography

General mathematical functions. URL https://fortran-lang.org/en/learn/intrinsics/math/#norm2.

Andrei Abelev, Kevin Tubbs, and Philip Valent. Numerical investigation of dynamic free-fall penetrometers in soft cohesive marine sediments using a finite difference approach. In *OCEANS 2009*, pages 1–10, Biloxi, MS, October 2009a. IEEE. ISBN 978-1-4244-4960-6. doi: 10.23919/OCEANS.2009.5422350. URL https://ieeexplore.ieee.org/document/5422350/.

Andrei Abelev, Kevin Tubbs, and Philip Valent. Numerical investigation of dynamic free-fall penetrometers in soft cohesive marine sediments using a finite difference approach. In *OCEANS 2009*, pages 1–10, Biloxi, MS, October 2009b. IEEE. ISBN 978-1-4244-4960-6. doi: 10.23919/OCEANS.2009.5422350. URL https://ieeexplore.ieee.org/document/5422350/.

Ali Albatal. *Advancement of Using Portable Free Fall Penetrometers for Geotechnical Site Characterization of Energenic Sandy Nearshore Areas.* PhD thesis, Virginia Tech, March 2018. URL http://hdl.handle.net/10919/94608.

Biswajit Banerjee. Derivatives of the invariants of a tensor | iMechanica, May 2007. URL https://imechanica.org/node/1403.

S G Bardenhagen and E M Kober. The Generalized Interpolation Material Point Method. *Computer Modeling in Engineering and Sciences*, 2004.

S G Bardenhagen, J E Guilkey, K M Roessig, J U Brackbill, and W M Witzel. An Improved

Contact Algorithm for the Material Point Method and Application to Stress Propagation in Granular Material. page 15, 2001.

S.G. Bardenhagen, J.U. Brackbill, and D. Sulsky. The material-point method for granular materials. *Computer Methods in Applied Mechanics and Engineering*, 187(3-4): 529–541, July 2000. ISSN 00457825. doi: 10.1016/S0045-7825(99)00338-2. URL https://linkinghub.elsevier.com/retrieve/pii/S0045782599003382.

Aaron S. Baumgarten and Ken Kamrin. Analysis and mitigation of spatial integration errors for the material point method. *International Journal for Numerical Methods in Engineering*, 124(11):2449–2497, June 2023. ISSN 0029-5981, 1097-0207. doi: 10.1002/nme.7217. URL https://onlinelibrary.wiley.com/doi/10.1002/nme.7217.

Lars Beuth. *Formulation and application of a quasi-static material point method*. Number 66 in Mitteilung ... des Instituts für Geotechnik. Inst. f. Geotechnik d. Univ. Stuttgart, Stuttgart, 2012. ISBN 978-3-921837-66-5.

Vibhav Bisht. *Cone penetration analysis using the Material Point Method*. Dissertation, Purdue University Graduate School, 2021. URL https://hammer.purdue.edu/articles/thesis/Cone_penetration_analysis_using_the_Material_Point_Method/15055686.

Justin Bonus. *Evaluation of Fluid-Driven Debris Impacts in a High-Performance Multi-GPU Material Point Method*. Ph.D., University of Washington, United States – Washington, 2023. URL https://www.proquest.com/docview/2915819774/abstract/777668C451714C5APQ/1. ISBN: 9798381406665.

R M Brannon, A F Fossum, and O E Strack. KAYENTA: Theory and User's Guide. 2009.

Nicola Brilli. *Influence of Geotechnical Properties on Sediment Dynamics, Erodibility, and*

Geomorphodynamics in Coastal Environments Based on Field Measurements. PhD thesis, Virginia Tech, Blacksburg, April 2023. URL http://hdl.handle.net/10919/115358.

Ha H. Bui and Giang D. Nguyen. Smoothed particle hydrodynamics (SPH) and its applications in geomechanics: From solid fracture to granular behaviour and multiphase flows in porous media. *Computers and Geotechnics*, 138:104315, October 2021. ISSN 0266352X. doi: 10.1016/j.compgeo.2021.104315. URL https://linkinghub.elsevier.com/retrieve/pii/S0266352X2100313X.

Bodhinanda Chandra, Ryota Hashimoto, Ken Kamrin, and Kenichi Soga. Mixed material point method formulation, stabilization, and validation for a unified analysis of free-surface and seepage flow, February 2024. URL http://arxiv.org/abs/2402.11719. arXiv:2402.11719 [physics].

R. Courant, K. Friedrichs, and H. Lewy. On the Partial Difference Equations of Mathematical Physics. *IBM Journal of Research and Development*, 11(2):215–234, March 1967. ISSN 0018-8646. doi: 10.1147/rd.112.0215. Conference Name: IBM Journal of Research and Development.

James Fern, Alexander Rohe, Kenichi Soga, and Eduardo Alonso, editors. *The Material Point Method for Geotechnical Engineering: a Practical Guide*. CRC Press, Boca Raton : CRC Press, Taylor & Francis Group, [2019], 1 edition, January 2019. ISBN 978-0-429-02809-0. doi: 10.1201/9780429028090-1. URL https://www.taylorfrancis.com/books/9780429650666/chapters/10.1201/9780429028090-1.

R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181:375–389, November 1977. ISSN 0035-8711. doi: 10.1093/mnras/181.3.375. URL

https://ui.adsabs.harvard.edu/abs/1977MNRAS.181..375G. Publisher: OUP ADS Bibcode: 1977MNRAS.181..375G.

Christopher D. G. Harley, A. Randall Hughes, Kristin M. Hultgren, Benjamin G. Miner, Cascade J. B. Sorte, Carol S. Thornber, Laura F. Rodriguez, Lars Tomanek, and Susan L. Williams. The impacts of climate change in coastal marine systems. *Ecology Letters*, 9 (2):228–241, 2006. ISSN 1461-0248. doi: 10.1111/j.1461-0248.2005.00871.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1461-0248.2005.00871.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1461-0248.2005.00871.x.

Thomas J. R. Hughes. Generalization of selective integration procedures to anisotropic and nonlinear media. *International Journal for Numerical Methods in Engineering*, 15(9): 1413–1418, September 1980. ISSN 0029-5981, 1097-0207. doi: 10.1002/nme.1620150914. URL https://onlinelibrary.wiley.com/doi/10.1002/nme.1620150914.

Thomas J. R. Hughes, editor. *The finite element method: linear static and dynamic finite element analysis*. Dover Publications, Mineola, NY, 2000. ISBN 978-0-486-41181-1 978-0-486-13502-1 978-1-62198-588-4.

Reem Jaber and Nina Stark. Geotechnical Properties from Portable Free Fall Penetrometer Measurements in Coastal Environments. *Journal of Geotechnical and Geoenvironmental Engineering*, 149(12):04023120, December 2023. ISSN 1090-0241, 1943-5606. doi: 10.1061/JGGEFK.GTENG-11013. URL https://ascelibrary.org/doi/10.1061/JGGEFK.GTENG-11013.

Issam K. J. Al Kafaji. *Formulation of a dynamic material point method (MPM) for geomechanical problems*. Doctoral Thesis, Universität Stuttgart, 2013. URL http://elib.uni-stuttgart.de/handle/11682/513. Publisher: Universität Stuttgart.

Ahana Lakshmi. Coastal ecosystem services & human wellbeing. *The Indian Journal of Medical Research*, 153(3):382–387, March 2021. ISSN 0971-5916. doi: 10.4103/ijmr.IJMR_695_21. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8204820/.

Thomas William Lambe and Robert V. Whitman. *Soil mechanics*. Series in soil engineering. Wiley, New York, nachdr. edition, 1969. ISBN 978-0-471-51192-2.

Yiming Li. Simulation of rapid spudcan penetration with the material point method, 2023. URL https://research-repository.uwa.edu.au/en/publications/43f0396a-a6c9-4e90-90f9-f1bcab85f950.

Zilong Li. Dynamics of coupled flow-object-sediment systems with smoothed particle hydrodynamics (SPH) simulations, November 2019. URL https://etda.libraries.psu.edu/catalog/17277zzl157.

L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013, December 1977. ISSN 00046256. doi: 10.1086/112164. URL http://adsabs.harvard.edu/cgi-bin/bib_query?1977AJ.....82.1013L.

T Lunne. The CPT in offshore soil investigations - a historic perspective. *International Symposium on Cone Penetration Testing*, 2nd:71–113, May 2010.

Mario Martinelli and Vahid Galavi. An explicit coupled MPM formulation to simulate penetration problems in soils using quadrilateral elements. *Computers and Geotechnics*, 145:104697, May 2022. ISSN 0266352X. doi: 10.1016/j.compgeo.2022.104697. URL https://linkinghub.elsevier.com/retrieve/pii/S0266352X22000611.

Debasis Mohapatra, Saeideh Mohammadi, Maarit Saaresma, Joonas Virtasalo, and Wojciech Sołowski. Numerical simulation of in-situ free fall cone penetrometer tests using the

material point method. In *Proceedings of the 2024 UK Association for Computational Mechanics Conference*, pages 157–160. Durham University, April 2024a. doi: 10.62512/conf.ukacm2024.101. URL https://durham-repository.worktribe.com/output/2396359.

Debasis Mohapatra, Saeideh Mohammadi, Maarit Saresma, Joonas J. Virtasalo, and Wojciech T. Sołowski. Laboratory-Scale Free Fall Cone Penetrometer Test on Marine Clay: A Numerical Investigation Using the Generalized Interpolation Material Point Method. *International Journal for Numerical and Analytical Methods in Geomechanics*, n/a(n/a), 2024b. ISSN 1096-9853. doi: 10.1002/nag.3929. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nag.3929. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nag.3929.

David Muir Wood. *Geotechnical Modelling*. CRC Press, 0 edition, December 2017. ISBN 978-1-315-27355-6. doi: 10.1201/9781315273556. URL https://www.taylorfrancis.com/books/9781482288315.

Muhammad Bilal Mumtaz. *Investigation of Pore Pressures During High-Velocity Impact by a Free Fall Penetrometer*. PhD thesis, Virginia Tech, June 2018. URL http://hdl.handle.net/10919/83800.

M. Nazem, J.P. Carter, D.W. Airey, and S.H. Chow. Dynamic analysis of a smooth penetrometer free-falling into uniform clay. *Géotechnique*, 62(10):893–905, October 2012. ISSN 0016-8505, 1751-7656. doi: 10.1680/geot.10.P.055. URL https://www.icevirtuallibrary.com/doi/10.1680/geot.10.P.055.

Vinh Phu Nguyen, Alban De Vaucorbeil, and Stephane Bordas. *The Material Point Method: Theory, Implementations and Applications*. Scientific Computation. Springer International Publishing, Cham, 2023. ISBN 978-3-031-24069-0 978-3-031-24070-6. doi: 10.1007/978-3-031-24070-6. URL https://link.springer.com/10.1007/978-3-031-24070-6.

A Niemunis. Preparing input for incrementalDriver. October 2019.

Andrzej Niemunis and Carlos Eduardo Grandas-Tavera. Computer Aided Calibration, Benchmarking and Check-Up of Constitutive Models for Soils. Some Conclusions for Neohypoplasticity. In Theodoros Triantafyllidis, editor, *Holistic Simulation of Geotechnical Installation Processes: Theoretical Results and Applications*, Lecture Notes in Applied and Computational Mechanics, pages 168–192. Springer International Publishing, Cham, 2017. ISBN 978-3-319-52590-7. doi: 10.1007/978-3-319-52590-7_7. URL https://doi.org/10.1007/978-3-319-52590-7_7.

M. Ortiz and J. C. Simo. An analysis of a new class of integration algorithms for elastoplastic constitutive relations. *International Journal for Numerical Methods in Engineering*, 23(3): 353–366, March 1986. ISSN 0029-5981, 1097-0207. doi: 10.1002/nme.1620230303. URL https://onlinelibrary.wiley.com/doi/10.1002/nme.1620230303.

Julie Paprocki, Nina Stark, Hans C Graber, Heidi Wadman, and Jesse E McNinch. Assessment of Moisture Content in Sandy Beach Environments from Multispectral Satellite Imagery. *Canadian Geotechnical Journal*, pages cgj–2020–0624, April 2021. ISSN 0008-3674, 1208-6010. doi: 10.1139/cgj-2020-0624. URL https://cdnsciencepub.com/doi/10.1139/cgj-2020-0624.

Chris T. Perry, Lorenzo Alvarez-Filip, Nicholas A. J. Graham, Peter J. Mumby, Shaun K. Wilson, Paul S. Kench, Derek P. Manzello, Kyle M. Morgan, Aimee B. A. Slangen, Damian P. Thomson, Fraser Januchowski-Hartley, Scott G. Smithers, Robert S. Steneck, Renee Carlton, Evan N. Edinger, Ian C. Enochs, Nuria Estrada-Saldívar, Michael D. E. Haywood, Graham Kolodziej, Gary N. Murphy, Esmeralda Pérez-Cervantes, Adam Suchley, Lauren Valentino, Robert Boenish, Margaret Wilson, and Chancey Macdonald. Loss of coral reef growth capacity to track future increases in sea level. *Nature*, 558

(7710):396–400, June 2018. ISSN 1476-4687. doi: 10.1038/s41586-018-0194-z. URL https://www.nature.com/articles/s41586-018-0194-z. Number: 7710 Publisher: Nature Publishing Group.

D M Potts and D Ganendra. An evaluation of substepping and implicit stress point algorithms. *Computer methods in applied mechanics and engineering*, 119.3-4 (1994):341–354, 1994.

David M. Potts and Lidija Zdravković. *Finite element analysis in geotechnical engineering. 2: Application*. Telford, London, 2001. ISBN 978-0-7277-2783-1.

C. Pérez-Collazo, D. Greaves, and G. Iglesias. A review of combined wave and offshore wind energy. *Renewable and Sustainable Energy Reviews*, 42:141–153, February 2015. ISSN 13640321. doi: 10.1016/j.rser.2014.09.032. URL https://linkinghub.elsevier.com/retrieve/pii/S1364032114008053.

Junlin Rong, Majidreza Nazem, Shiao Huey Chow, Annan Zhou, and Sara Moridpour. Design and application of dynamic seabed penetrators in offshore geotechnical engineering: A review. *Ocean Engineering*, 293:116599, February 2024. ISSN 00298018. doi: 10.1016/j.oceaneng.2023.116599. URL https://linkinghub.elsevier.com/retrieve/pii/S0029801823029839.

A. Sadeghirad, R. M. Brannon, and J. Burghardt. A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations. *International Journal for Numerical Methods in Engineering*, 86 (12):1435–1456, June 2011. ISSN 0029-5981, 1097-0207. doi: 10.1002/nme.3110. URL https://onlinelibrary.wiley.com/doi/10.1002/nme.3110.

Juan Simo and Thomas J. R. Hughes. *Computational Inelasticity*, volume 7 of *Interdisci-*

plinary Applied Mathematics. Springer-Verlag, New York, 1998. ISBN 978-0-387-97520-7. doi: 10.1007/b98904. URL http://link.springer.com/10.1007/b98904.

Scott W. Sloan, Andrew J. Abbo, and Daichao Sheng. Refined explicit integration of elastoplastic models with automatic error control. *Engineering Computations*, 18(1/2): 121–194, February 2001. ISSN 0264-4401. doi: 10.1108/02644400110365842. URL https://www.emerald.com/insight/content/doi/10.1108/02644400110365842/full/html.

Paul V. R. Snelgrove. The Importance of Marine Sediment Biodiversity in Ecosystem Processes. *Ambio*, 26(8):578–583, 1997. ISSN 0044-7447. URL https://www.jstor.org/stable/4314672. Publisher: [Springer, Royal Swedish Academy of Sciences].

Nina Stark. *Geotechnical investigation of sediment remobilization processes using dynamic penetrometers*. PhD thesis, 2011.

Nina Stark, Alex E. Hay, and Greg Trowse. Cost-effective geotechnical and sedimentological early site assessment for ocean renewable energies. In *2014 Oceans - St. John's*, pages 1–8, St. John's, NL, September 2014. IEEE. ISBN 978-1-4799-4918-2 978-1-4799-4920-5 978-1-4799-4919-9. doi: 10.1109/OCEANS.2014.7003004. URL http://ieeexplore.ieee.org/document/7003004/.

Nina Stark, Boris Radosavljevic, Brandon Quinn, and Hugues Lantuit. Application of portable free-fall penetrometer for geotechnical investigation of Arctic nearshore zone. *Canadian Geotechnical Journal*, 54(1):31–46, January 2017. ISSN 0008-3674. doi: 10.1139/cgj-2016-0087. URL https://cdnsciencepub.com/doi/10.1139/cgj-2016-0087. Publisher: NRC Research Press.

Michael Steffen, Robert M. Kirby, and Martin Berzins. Analysis and reduction of quadrature errors in the material point method (MPM). *International Journal for Numerical Methods*

in Engineering, 76(6):922–948, November 2008. ISSN 0029-5981, 1097-0207. doi: 10.1002/nme.2360. URL https://onlinelibrary.wiley.com/doi/10.1002/nme.2360.

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

Johannes Traa. Matrix Calculus - Notes on the Derivative of a Trace. URL https://paulklein.ca/newsite/teaching/matrix%20calculus.pdf.

C. Truesdell. *The Elements of Continuum Mechanics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1984. ISBN 978-3-540-03683-8 978-3-642-64976-9. doi: 10.1007/978-3-642-64976-9. URL http://link.springer.com/10.1007/978-3-642-64976-9.

Clifford Truesdell and Walter Noll. *The Non-Linear Field Theories of Mechanics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992. ISBN 978-3-662-13185-5 978-3-662-13183-1. doi: 10.1007/978-3-662-13183-1. URL http://link.springer.com/10.1007/978-3-662-13183-1.

David White, Conleth O'Loughlin, and Shiao Huey Chow. *Interpretation of free fall penetrometer tests in sands: An approach to determining the equivalent static resistance*. Cone Penetration Testing, June 2018.

David Muir Wood. *Soil Behaviour and Critical State Soil Mechanics*. Cambridge University Press, 1 edition, April 1991. ISBN 978-0-521-33249-1 978-0-521-33782-3 978-1-139-87827-2. doi: 10.1017/CBO9781139878272. URL https://www.cambridge.org/core/product/identifier/9781139878272/type/book.

Fuat Furkan Yalcin. Numerical Analysis of FFP Impact on Saturated Loose Sand. page 79, September 2021.

Fuat Furkan Yalcin, Luis Zambrano-Cruzatty, and Alba Yerro-Colom. Numerical Analysis of FFP Impact on Saturated Sands. In *Geo-Congress 2023*, pages 180–189, Los Angeles, California, March 2023. American Society of Civil Engineers. ISBN 978-0-7844-8469-2. doi: 10.1061/9780784484692.019. URL https://ascelibrary.org/doi/10.1061/9780784484692.019.

Jerry A. Yamamuro, Antonio E. Abrantes, and Poul V. Lade. Effect of Strain Rate on the Stress-Strain Behavior of Sand. *Journal of Geotechnical and Geoenvironmental Engineering*, 137(12):1169–1178, December 2011. ISSN 1090-0241, 1943-5606. doi: 10.1061/(ASCE)GT.1943-5606.0000542. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29GT.1943-5606.0000542.

A. Yerro, E.E. Alonso, and N.M. Pinyol. The material point method for unsaturated soils. *Géotechnique*, 65(3):201–217, March 2015. ISSN 0016-8505, 1751-7656. doi: 10.1680/geot.14.P.163. URL https://www.icevirtuallibrary.com/doi/10.1680/geot.14.P.163.

Alba Yerro Colom. *MPM modelling of landslides in brittle and unsaturated soils*. PhD thesis, Universitat Politècnica de Catalunya, November 2015. URL http://hdl.handle.net/2117/102412.

Luis Zambrano-Cruzatty and Alba Yerro. Numerical simulation of a free fall penetrometer deployment using the material point method. *Soils and Foundations*, 60(3):668–682, June 2020. ISSN 0038-0806. doi: 10.1016/j.sandf.2020.04.002. URL https://www.sciencedirect.com/science/article/pii/S0038080620336179.

Luis E Zambrano-Cruzatty. *Advancements for the Numerical Simulation of Free Fall Penetrometers and the Analysis of Wind Erosion of Sands*. PhD thesis, Virginia Tech, Blacksburg, VA, July 2021. URL http://hdl.handle.net/10919/104861.

Luis E. Zambrano-Cruzatty, Alba Yerro, and Jorge Macedo. Constitutive modelling of non-cohesive soils under high-strain rates: a consistency approach. *Géotechnique*, pages 1–18, August 2022. ISSN 0016-8505, 1751-7656. doi: 10.1680/jgeot.21.00192. URL https://www.icevirtuallibrary.com/doi/10.1680/jgeot.21.00192.

Xiong Zhang, Zhen Chen, and Yan Liu. *The material point method: a continuum-based particle method for extreme loading cases*. Academic Press, 2016.

Appendices

Appendix A

Constitutive Model Additional Quad Plots

A.1 Strain Softening Mohr-Coulomb(*SSMC*) Additional Quad Plots

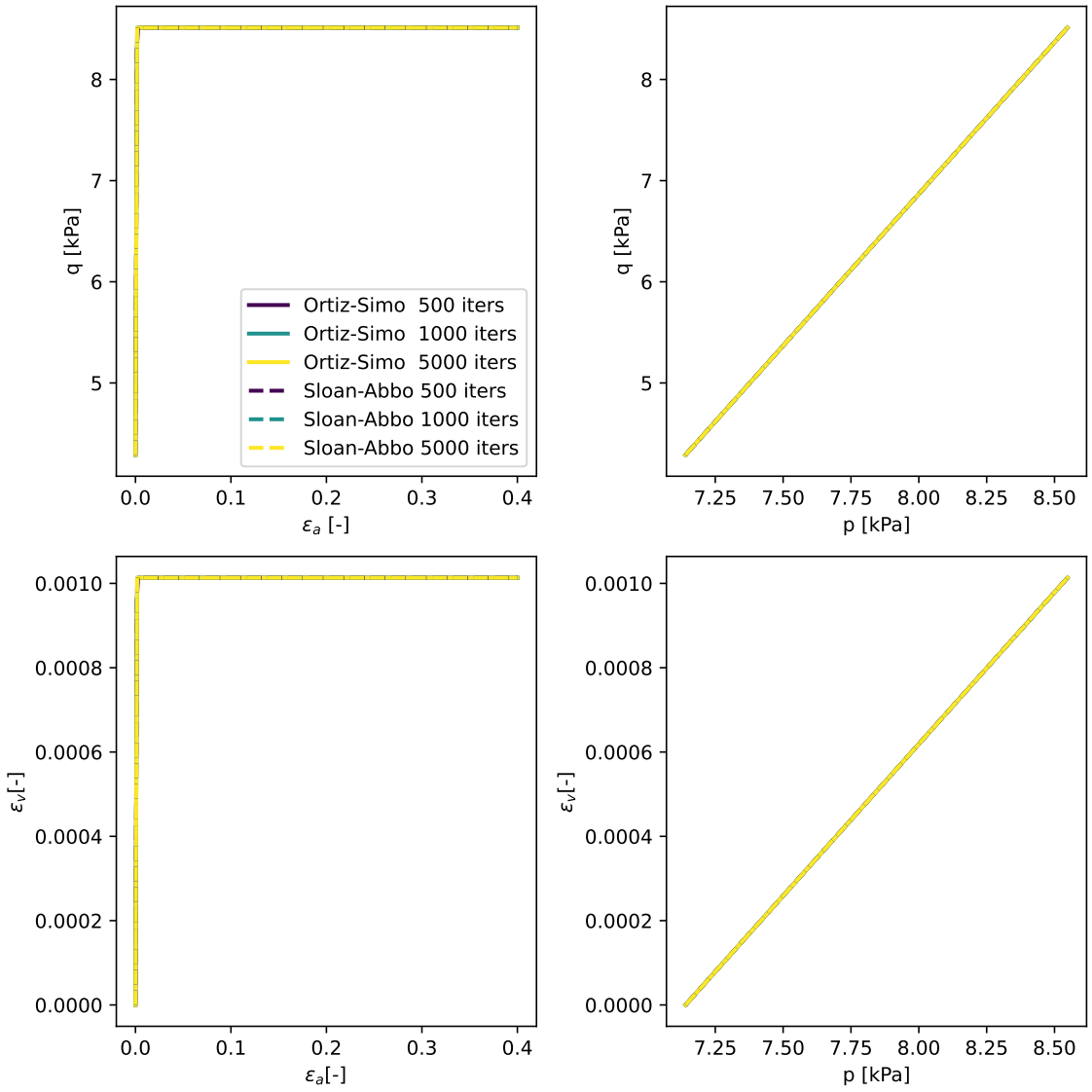


Figure A.1: *SSMC* Elastic-perfectly plastic quad plot

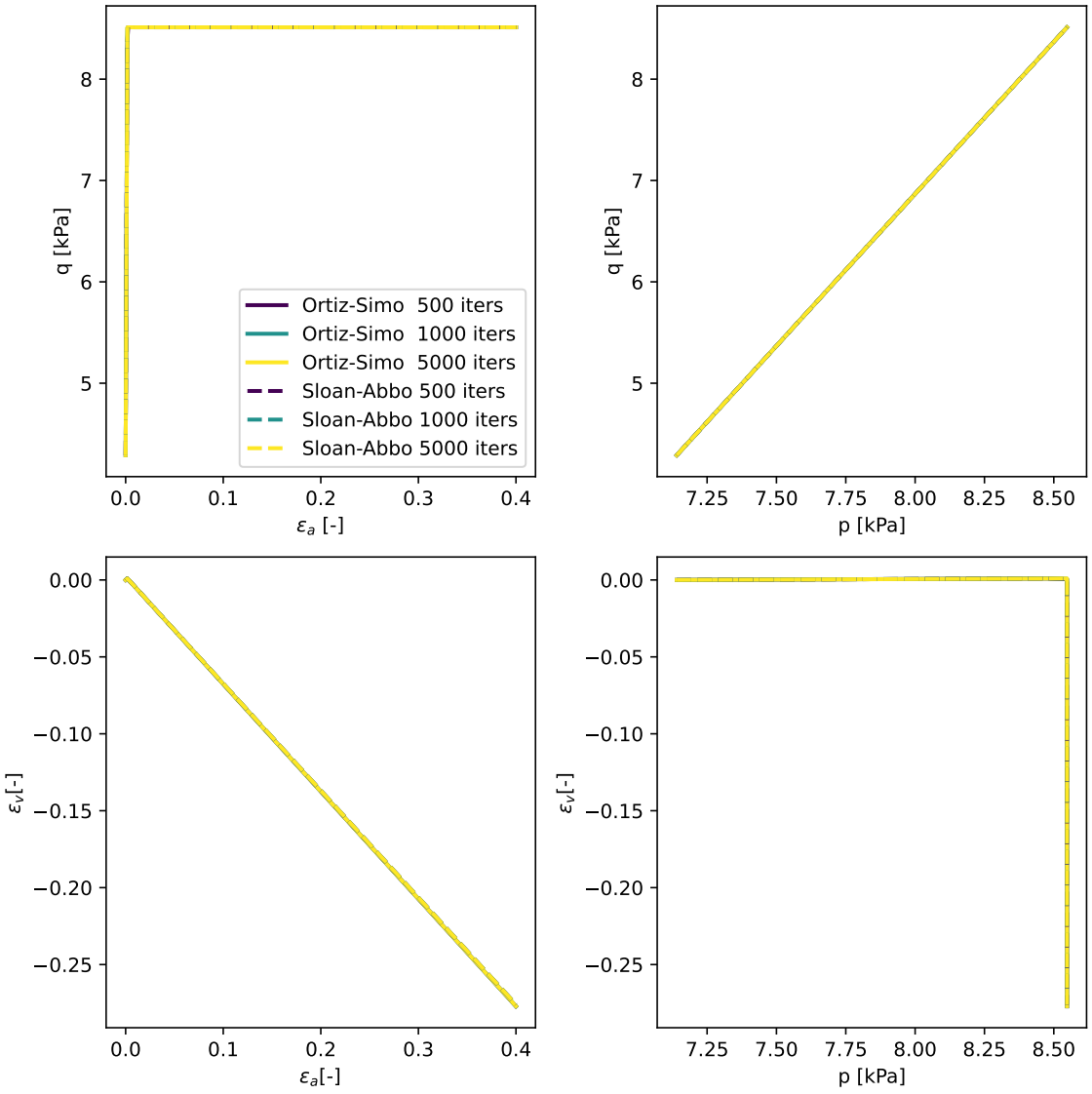


Figure A.2: *SSMC* Elastic perfectly-plastic with dilatancy quad plot

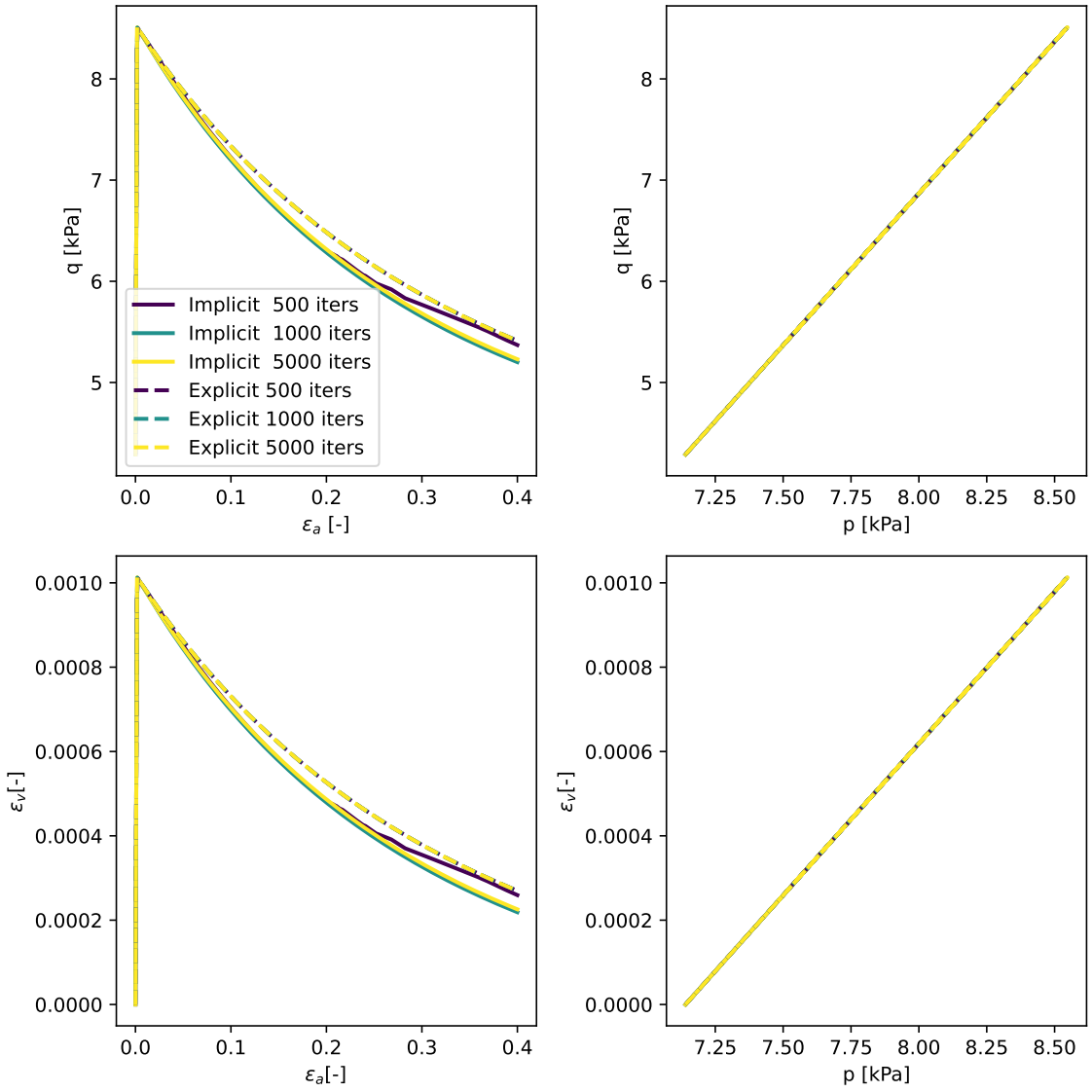


Figure A.3: *SSMC* Elasto-plastic with ϕ' softening quad plot

A.2 Strain Rate Mohr-Coulomb Additional Quad Plots

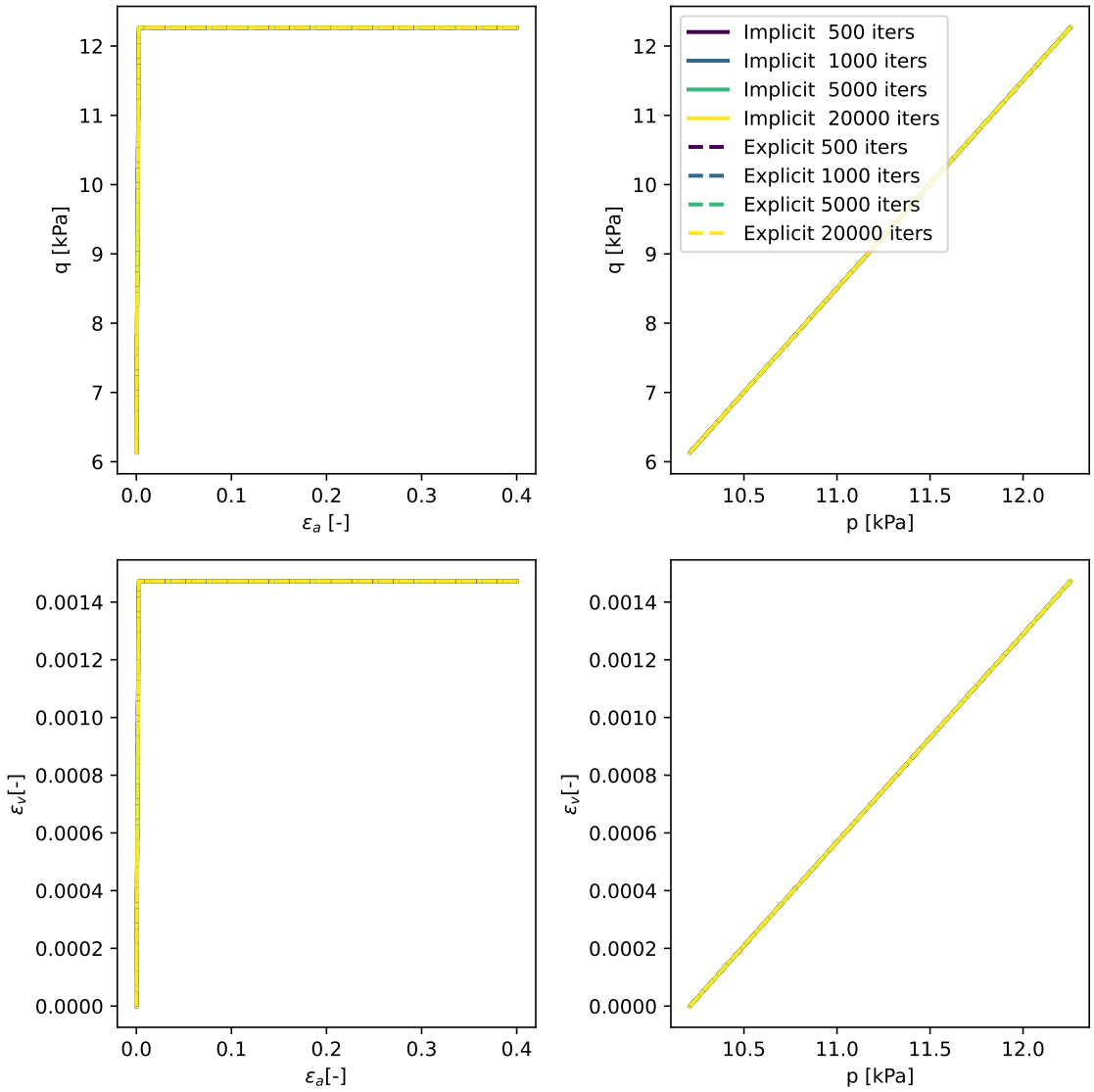


Figure A.4: Elastic-perfectly plastic quad plot

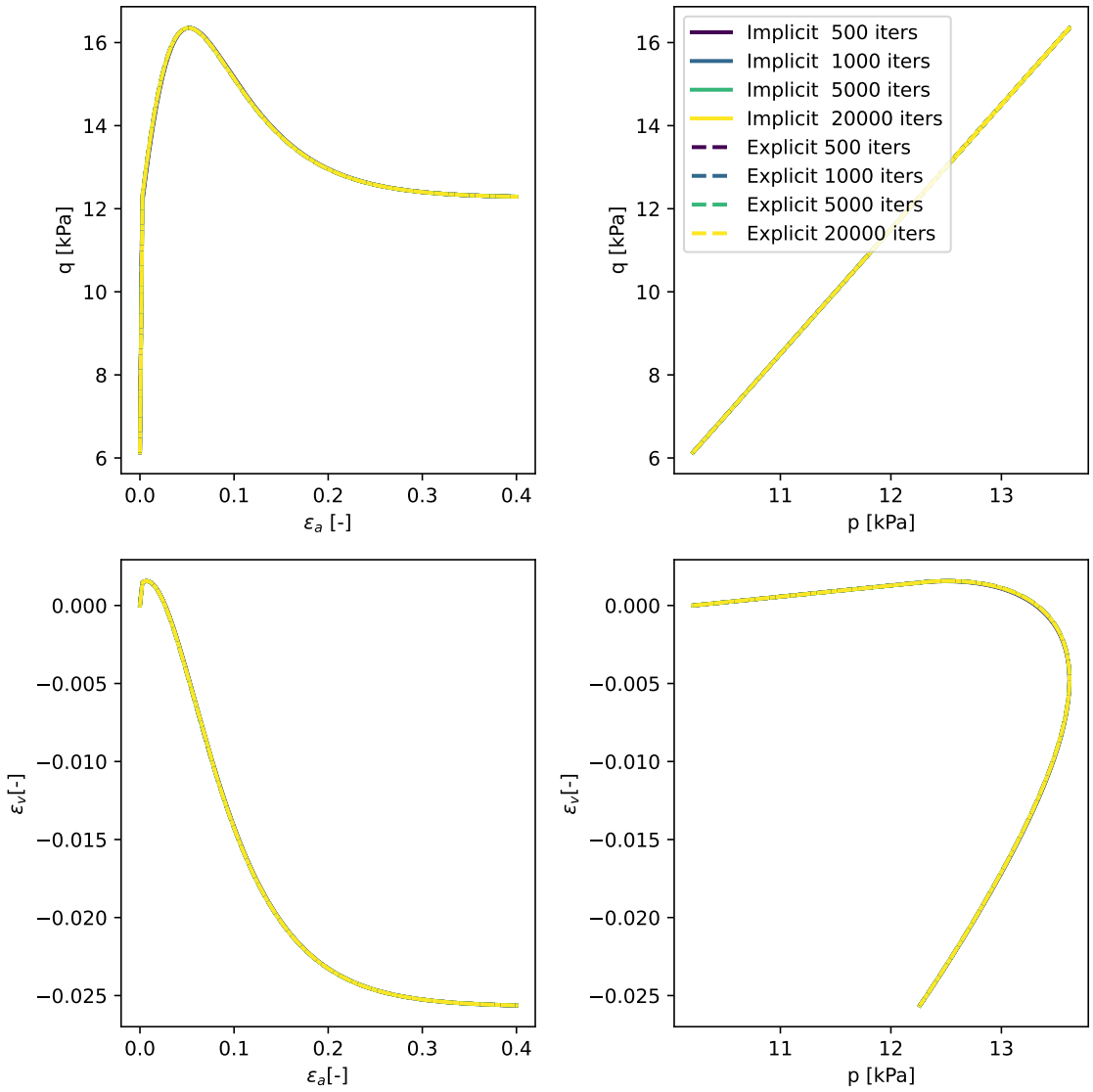


Figure A.5: Elasto-plastic with diltancy quad plot

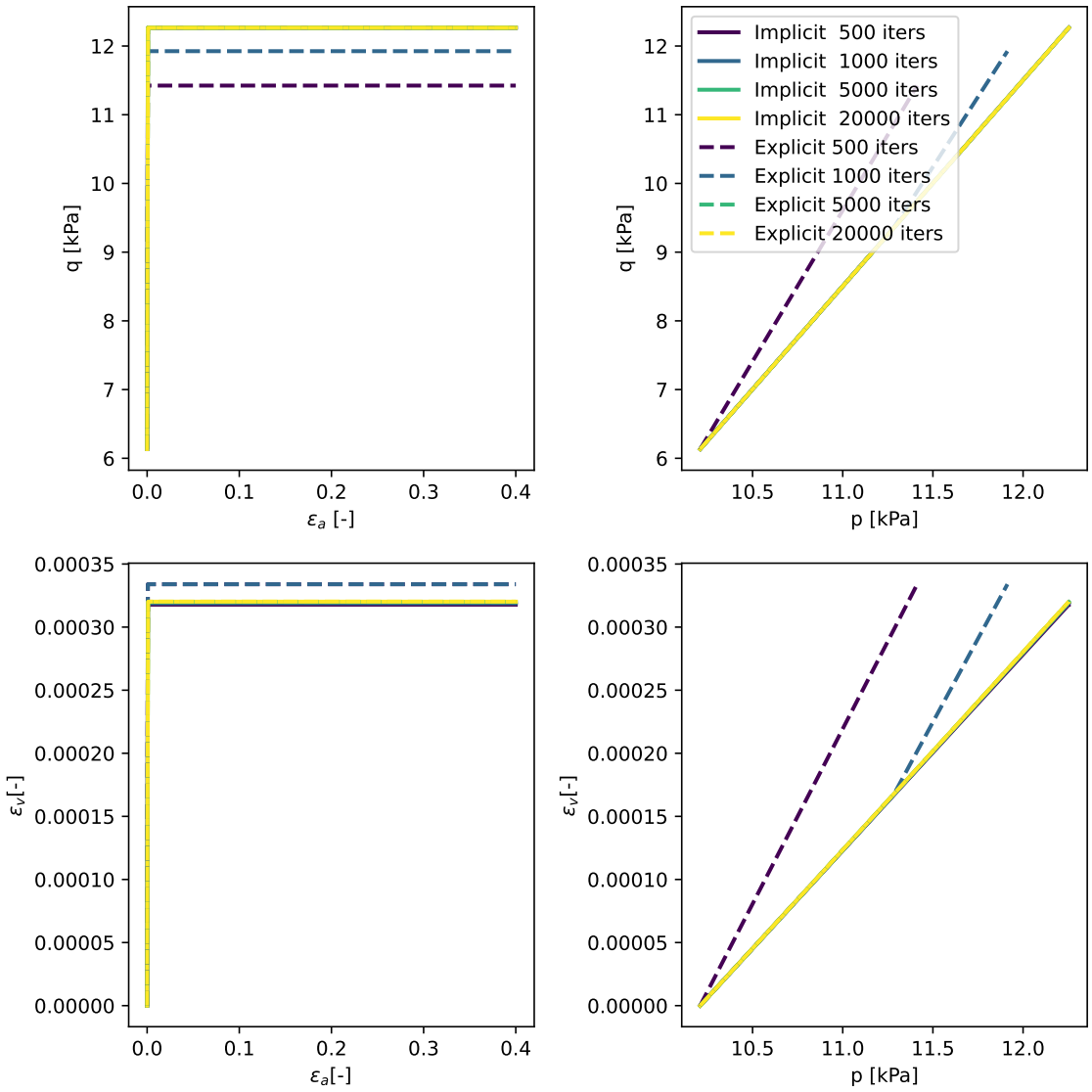


Figure A.6: Elasto-plastic with rate affects quad plot

A.3 Addressing and Discussion of bugs in *SRMC*

The strain rate Mohr-Coulomb (*SRMC*, Zambrano-Cruzatty et al. [2022]) model used in this study was graciously provided by Luis Zambrano-Cruzatty. During this thesis, two main bugs were found in the implementation. The first was an inconsistent particle diameter unit (D) in the equation for the inertial coefficient (I, equation A.2). In Zambrano-Cruzatty et al. [2022], the particle diameter was input with the intention of making it 0.32 mm, however due to the assumed units of the other terms in equation A.2, the value entered was 0.32 m. Therefore, the constitutive model needs to be recalibrated with the Yamamuro et al. [2011] data. This wasn't done during this thesis, so the viscous coefficients (k_G, k_K, k_D) used here are not calibrated against any physical samples. Justification for this decision was that the plots and results presented for the *SRMC* was for an imagined soil. Before applying the constitutive model to real soils, recalibration to the Yamamuro et al. [2011] dataset or another dataset with experiments that studied high strain rate in sand is required.

The second set of bugs was in the calculation of the stress invariants and there derivatives. The invariants and derivatives were corrected and are available in the *SRMC* repo. The main set of stress and strain invariants and their derivatives, not the unit test functions, are voigt vector ordering agnostic as long as the volumetric terms are the first three components. Furthermore, this correction prompted the inclusion of Appendix C in this thesis. If errors are found in the listed invariants, or in the implementation, please let the author know at *jonathanm@vt.edu* or by filing a GitHub issue on any of the *CriticalSoilModels* repos.

A.3.1 Note on implementation of inertial coefficient function in the *SRMC* code

Also as a note to clarify some confusion, the reason the form of the inertial coefficient equation is implemented in the code as

$$I = D\varepsilon_q q\sqrt{\frac{G_s}{|p'|}} \tag{A.1}$$

where

- D: Particle diameter, [m]
- $\varepsilon_q q$: Deviatoric strain rate [s^{-1}]
- G_s: Particle's specific gravity [-]
- $|p'|$: Absolute value of mean effective stress [kPa]

instead of the form presented in Zambrano-Cruzatty et al. [2022],

$$I = D\varepsilon_q q\sqrt{\frac{\rho_s}{|p'|}} \tag{A.2}$$

where

- D: Particle diameter, [m]
- $\varepsilon_q q$: Deviatoric strain rate [s^{-1}]
- G_s: Particle's density [-]
- $|p'|$: Absolute value of mean effective stress [kPa]

is to account for *IncrementalDriver* and Anura3D using units of kilopascal for stress. The factor of 1000 is removed from the density to account for the extra factor of 1000 to convert from pascal to kilopascal. Third, a check on how the elastic moduli are viscously updated is needed. When k_K and k_G are not equal Poisson's ratio (ν) is updated. Therefore, given the right loading rate and initial values for bulk (K) and shear (G) modulus, it's possible that K and G can combine to generate $\nu > 0.5$. This is unphysical. From K and G ν is calculated as, (Muir Wood [2017])

$$\frac{3K - 2G}{2(3K + G)} \tag{A.3}$$

The check should either fix the values of K and G when they reach incompressibilty or the form of the equation that updates the elastic moduli, equation 4.11, should be constrained. The trend to $v \geq 0.5$ is asymtoptic so it takes a large magnitude difference between the reference strain rate and the applied strain rate and a specific coupling of K and G to create this problem.

Appendix B

B-Bar Derivation

B.1 Detailed Derivation of B-Bar Method

The calculations done here are for a single node. For 4-noded quadrilateral elements, the nodal calculations will need to be repeated 4 times and summed. In general this means that

$$\delta\boldsymbol{\varepsilon} = \sum_{a=1}^{nn} \mathbf{B}_a \delta\mathbf{u}_a$$

where nn is the number of nodes, B_a is the matrix of shape function derivatives for and single node and $\mathbf{u}_a$ is the vector of nodal displacements. The same format holds for the B-bar technique except $\mathbf{B}_a$ is replaced with $\bar{\mathbf{B}}_a$.

$$\delta\bar{\boldsymbol{\varepsilon}} = \sum_{a=1}^{nn} \bar{\mathbf{B}}_a \delta\mathbf{u}_a \tag{B.1}$$

This is not the derivation of a new technique, but a detailed explanation and component form of the information provided in Hughes [2000].

1. The $\mathbf{B}_a$ is presented

2. The $\bar{\mathbf{B}}_a$ is presented and some comments are made

3. The node component incremental strains $\delta\varepsilon_{ij}^a$ are presented

B.1.1 3D derivation

3D Normal Strain vector

For a specific node in 3D, the general form of the B-matrix is:

$$\left[\begin{array}{ccc} B_1 & 0 & 0 \\ 0 & B_2 & 0 \\ 0 & 0 & B_3 \\ -- & -- & -- \\ 0 & B_3 & B_2 \\ B_3 & 0 & B_1 \\ B_2 & B_1 & 0 \end{array}\right]$$

in which

$$B_i = \partial N_a / \partial x_i \quad 1 \leq i \leq 3$$

where N_a is the shape function associated with node a, and x_i is the i the Cartesian coordinate (Hughes [2000]).

Defining the nodal incremental strain vector ($\delta\boldsymbol{\varepsilon}^a$) as

$$\delta\boldsymbol{\varepsilon}^a = \left[\begin{array}{c} \delta\varepsilon^a_{xx} \\ \delta\varepsilon^a_{yy} \\ \delta\varepsilon^a_{zz} \\ \delta\gamma^a_{yz} \\ \delta\gamma^a_{xz} \\ \delta\gamma^a_{xy} \end{array}\right]$$

In component form $\delta\boldsymbol{\varepsilon}$, replacing the B_i values with there shape function derivative definitions, is

$$\delta\varepsilon^a_{xx} = \frac{\partial N_a}{\partial x_1}\delta u_1 \tag{B.2}$$

$$\delta\varepsilon^a_{yy} = \frac{\partial N_a}{\partial x_2}\delta u_2 \tag{B.3}$$

$$\delta\varepsilon^a_{zz} = \frac{\partial N_a}{\partial x_3}\delta u_3 \tag{B.4}$$

$$\delta\gamma^a_{yz} = \frac{\partial N_a}{\partial x_3}\delta u_2 + \frac{\partial N_a}{\partial x_2}u_3 \tag{B.5}$$

$$\delta\gamma^a_{xz} = \frac{\partial N_a}{\partial x_1}\delta u_3 + \frac{\partial N_a}{\partial x_3}u_1 \tag{B.6}$$

$$\delta\gamma^a_{xy} = \frac{\partial N_a}{\partial x_2}\delta u_1 + \frac{\partial N_a}{\partial x_1}u_2 \tag{B.7}$$

$$\tag{B.8}$$

Derivation of the B-matrix

Let $\mathbf{B}^{\mathbf{dil}}_{\mathbf{a}}$ denote the dilational part of $\mathbf{B_a}$, i.e.,

$$\mathbf{B}^{\mathbf{dil}}_{\mathbf{a}} = \frac{1}{3}\begin{bmatrix} B_1 & B_2 & B_3 \\ B_1 & B_2 & B_3 \\ B_1 & B_2 & B_3 \\ -- & -- & -- \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{B.9}$$

Multiplying this matrix against the incremental displacement gives the volumetric strain tensor, $(\delta\varepsilon^{\mathbf{a}}_{\mathbf{V}})$, which when symmetry is used, can be written as

$$\delta\boldsymbol{\varepsilon}^{\boldsymbol{a}}_{\boldsymbol{V}} = \frac{1}{3}\begin{bmatrix} B_1\delta u_1 + B_2\delta u_2 + B_3\delta u_3 \\ B_1\delta u_1 + B_2\delta u_2 + B_3\delta u_3 \\ B_1\delta u_1 + B_2\delta u_2 + B_3\delta u_3 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{B.10}$$

which maintains the relationship of the incremental volumetric strain scalar $(\delta\varepsilon_p)$ as the trace of the strain matrix. The symmetry of the strain tensor guarantees that the trace is invariant to a change of basis and is commonly referred to as the first invariant of the tensor.

$$\delta\varepsilon_p = \mathrm{tr}(\boldsymbol{\delta\varepsilon}_V) = \frac{1}{3}(\delta\varepsilon_{xx}+\delta\varepsilon_{yy}+\delta\varepsilon_{zz})+\frac{1}{3}(\delta\varepsilon_{xx}+\delta\varepsilon_{yy}+\delta\varepsilon_{zz})+\frac{1}{3}(\delta\varepsilon_{xx}+\delta\varepsilon_{yy}+\delta\varepsilon_{zz}) = (\delta\varepsilon_{xx}+\delta\varepsilon_{yy}+\delta\varepsilon_{zz}) \tag{B.11}$$

$$\delta\varepsilon_p = \mathrm{tr}(\boldsymbol{\delta\varepsilon}) = \delta\varepsilon_{xx} + \delta\varepsilon_{yy} + \delta\varepsilon_{zz} \tag{B.12}$$

$\boldsymbol{\delta\varepsilon}_{dev}$ is by construction traceless, $\mathrm{tr}(\boldsymbol{\delta\varepsilon}_{dev}) = 0$, and for a specific node can be calculated as

$$\mathbf{B}^{dev}_a = \mathbf{B}_a - \mathbf{B}^{dil}_a \tag{B.13}$$

which in matrix form gives

$$\mathbf{B^{dev}} = \begin{bmatrix} B_1 - \frac{1}{3}B_1 & -\frac{1}{3}B_2 & -\frac{1}{3}B_3 \\ -\frac{1}{3}B_1 & B_2 - \frac{1}{3}B_2 & -\frac{1}{3}B_3 \\ -\frac{1}{3}B_1 & -\frac{1}{3}B_2 & B_3 - \frac{1}{3}B_3 \\ -- & -- & -- \\ 0 & B_3 & B_2 \\ B_3 & 0 & B_1 \\ B_2 & B_1 & 0 \end{bmatrix}$$

Defining the $\mathbf{B_a^{dil}}-$matrix as

$$\mathbf{B_a^{dil}} = \frac{1}{3}\begin{bmatrix} \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ -- & -- & -- \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where (when using the selective integration scheme) $\bar{B}_i = B_i(\mathbf{0})$. The $\bar{B}_i^{dil}$ are the element averaged volumetric strain-displacement relationships. Using *B-Bar* makes the volumetric strain constant in the element. **Link** to notes on FEM techniques for volumetric locking.

Combinining the $\mathbf{B^{dev}}_a$ and $\mathbf{\bar{B}^{dil}}_a$ matrices to form the $\bar{\mathbf{B}}_a$ matrix gives

$$\mathbf{B_a} = \mathbf{B_a^{dev}} + \bar{\mathbf{B}}_a^{dil} = \begin{bmatrix} B_1 - \frac{1}{3}B_1 & -\frac{1}{3}B_2 & -\frac{1}{3}B_3 \\ -\frac{1}{3}B_1 & B_2 - \frac{1}{3}B_2 & -\frac{1}{3}B_3 \\ -\frac{1}{3}B_1 & -\frac{1}{3}B_2 & B_3 - \frac{1}{3}B_3 \\ -- & -- & -- \\ 0 & B_3 & B_2 \\ B_3 & 0 & B_1 \\ B_2 & B_1 & 0 \end{bmatrix} + \frac{1}{3} \begin{bmatrix} \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ \bar{B}_1 & \bar{B}_2 & \bar{B}_3 \\ -- & -- & -- \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\bar{\mathbf{B}}_a = \begin{bmatrix} B_1 + \frac{1}{3}(\bar{B}_1 - B_1) & \frac{1}{3}(\bar{B}_2 - B_2) & \frac{1}{3}(\bar{B}_3 - B_3) \\ \frac{1}{3}(\bar{B}_1 - B_1) & B_2 + \frac{1}{3}(\bar{B}_2 - B_2) & \frac{1}{3}(\bar{B}_3 - B_3) \\ \frac{1}{3}(\bar{B}_1 - B_1) & \frac{1}{3}(\bar{B}_2 - B_2) & B_3 + \frac{1}{3}(\bar{B}_3 - B_3) \\ -- & -- & -- \\ 0 & B_3 & B_2 \\ B_3 & 0 & B_1 \\ B_2 & B_1 & 0 \end{bmatrix}$$

or equivalently

$$\bar{\mathbf{B}}_a = \frac{1}{3} \begin{bmatrix} 2B_1 + \bar{B}_1 & \bar{B}_2 - B_2 & \bar{B}_3 - B_3 \\ \bar{B}_1 - B_1 & 2B_2 + \bar{B}_2 & \bar{B}_3 - B_3 \\ \bar{B}_1 - B_1 & \bar{B}_2 - B_2 & 2B_3 + \bar{B}_3 \\ -- & -- & -- \\ 0 & 3B_3 & 3B_2 \\ 3B_3 & 0 & 3B_1 \\ 3B_2 & 3B_1 & 0 \end{bmatrix}$$

Multiplying $\bar{\mathbf{B}}_a$ and the incremental displacement vector ($\delta\mathbf{u}_a$) together gives the modified

incremental strain vector ($\delta\bar{\boldsymbol{\varepsilon}}^a$)

$$\delta\bar{\boldsymbol{\varepsilon}}^a = \bar{\mathbf{B}}_a \delta\mathbf{u}_a \tag{B.14}$$

In 3D $\delta\bar{\boldsymbol{\varepsilon}}^a$ in component form is

$$\delta\overline{\varepsilon_{xx}}^a = \frac{1}{3}\left(2\frac{\partial N_a}{\partial x_1}(P) + \frac{\partial N_a}{\partial x_1}(C)\right)\delta u_x + \frac{1}{3}\left(\frac{\partial N_a}{\partial x_2}(C) - \frac{\partial N_a}{\partial x_2}(P)\right)\delta u_y + \frac{1}{3}\left(\frac{\partial N_a}{\partial x_3}(C) - \frac{\partial N_a}{\partial x_3}(P)\right)\delta u_z \tag{B.15}$$

$$\delta\overline{\varepsilon_{yy}}^a = \frac{1}{3}\left(\frac{\partial N_a}{\partial x_1}(C) - \frac{\partial N_a}{\partial x_1}(P)\right)\delta u_x + \frac{1}{3}\left(2\frac{\partial N_a}{\partial x_2}(P) + \frac{\partial N_a}{\partial x_2}(C)\right)\delta u_y + \frac{1}{3}\left(\frac{\partial N_a}{\partial x_3}(C) - \frac{\partial N_a}{\partial x_3}(P)\right)\delta u_z \tag{B.16}$$

$$\delta\overline{\varepsilon_{zz}}^a = \frac{1}{3}\left(\frac{\partial N_a}{\partial x_1}(C) - \frac{\partial N_a}{\partial x_1}(P)\right)\delta u_x + \frac{1}{3}\left(\frac{\partial N_a}{\partial x_2}(C) - \frac{\partial N_a}{\partial x_2}(P)\right)\delta u_y + \frac{1}{3}\left(2\frac{\partial N_a}{\partial x_3}(P) + \frac{\partial N_a}{\partial x_3}(C)\right)\delta u_z \tag{B.17}$$

$$\delta\overline{\gamma_{yz}}^a = \frac{\partial N_a}{\partial x_3}(P)\delta u_2 + \frac{\partial N_a}{\partial x_2}(P)\delta u_3 \tag{B.18}$$

$$\delta\overline{\gamma_{xz}}^a = \frac{\partial N_a}{\partial x_1}(P)\delta u_3 + \frac{\partial N_a}{\partial x_3}(P)\delta u_1 \tag{B.19}$$

$$\delta\overline{\gamma_{xy}}^a = \frac{\partial N_a}{\partial x_2}(P)\delta u_1 + \frac{\partial N_a}{\partial x_1}(P)\delta u_2 \tag{B.20}$$

where (P) denotes that the function is evaluated at the location of the particle and C means that the function is evaluated at the center of the element. The shear terms remain the same

the volumetric terms are now a function of the full incremental displacement vector and $\bar{B}_i$. In the next sections the write out will be repeated for the plane strain and 2D axisymmetric cases.

B.1.2 2D axisymmetric

The 2D (torsionless) axisymmetric $\bar{\mathbf{B}}_a^{axi}$-matrix is (Hughes [2000])

$$\bar{\mathbf{B}}_a^{axi} = \begin{bmatrix} B_{12} & B_6 \\ B_{10} & B_7 \\ B_2 & B_1 \\ - - - - - & - - - - - \\ B_{11} & B_6 \end{bmatrix} \tag{B.21}$$

The B_i values in matrix B.21 are defined as

$$B_0 = \frac{N_a}{r_1} \tag{B.22}$$

$$B_4 = \frac{1}{3}(\bar{B}_1 - B_1) \tag{B.23}$$

$$B_5 = B_1 + B_4 = B_1 + \frac{1}{3}(\bar{B}_1 - B_1) \tag{B.24}$$

$$B_6 = \frac{1}{3}(\bar{B}_2 - B_2) \tag{B.25}$$

$$B_7 = B_2 + B_6 = B_2 + \frac{1}{3}(\bar{B}_2 - B_2) \tag{B.26}$$

$$B_8 = \frac{1}{3}(\bar{B}_3 - B_3) \tag{B.27}$$

$$B_9 = B_3 + B_8 = B_3 + \frac{1}{3}(\bar{B}_3 - B_3) \tag{B.28}$$

$$B_{10} = B_4 + \frac{1}{3}(\bar{B}_0 - B_0) = \frac{1}{3}(\bar{B}_1 - B_1) + \frac{1}{3}(\frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1}) \tag{B.29}$$

$$B_{11} = B_0 + B_{10} = \frac{N_a}{r_1} + \frac{1}{3}(\bar{B}_1 - B_1) + \frac{1}{3}(\frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1}) \tag{B.30}$$

$$B_{12} = B_1 + B_{10} = B_1 + \frac{1}{3}(\bar{B}_1 - B_1) + \frac{1}{3}(\frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1}) \tag{B.31}$$

Replacing the values of B.21 with the expanded forms shown above gives

$$\bar{\mathbf{B}}_a^{axi} = \begin{bmatrix} \frac{1}{3}(2B_1 + \bar{B}_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1}) & \frac{1}{3}(\bar{B}_2 - B_2) \\ \frac{1}{3}(\bar{B}_1 - B_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1}) & \frac{1}{3}(2B_2 + \bar{B}_2) \\ B_2 & B_1 \\ - - - - - & - - - - - \\ \frac{1}{3}(\bar{B}_1 - B_1 + 2\frac{N_a}{r_1} + \frac{\bar{N}_a}{r_1}) & \frac{1}{3}(\bar{B}_2 - B_2) \end{bmatrix} \tag{B.32}$$

Factoring out the $\frac{1}{3}$

$$\bar{\mathbf{B}}_a^{axi} = \frac{1}{3}\begin{bmatrix} 2B_1 + \bar{B}_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1} & \bar{B}_2 - B_2 \\ \bar{B}_1 - B_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1} & 2B_2 + \bar{B}_2 \\ 3B_2 & 3B_1 \\ - - - - - & - - - - - \\ \bar{B}_1 - B_1 + 2\frac{N_a}{r_1} + \frac{\bar{N}_a}{r_1} & \bar{B}_2 - B_2 \end{bmatrix} \tag{B.33}$$

Defining the δu as

$$\delta u = \begin{bmatrix} \delta u_r \\ \delta u_z \end{bmatrix} \tag{B.34}$$

The above assumed $\delta u_\theta = 0$ to form torsionless axi-symmmetry. The strains can be calculated as

$$\delta\bar{\varepsilon}^a = \overline{\mathbf{B}}_a^{axi}\delta\mathbf{u}_a$$

component-wise this results in

$$\delta\bar{\varepsilon}^a_{rr} = \frac{1}{3}((2B_1 + \bar{B}_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1})\delta u_r + (\bar{B}_2 - B_2)\delta u_z) \tag{B.35}$$

$$\delta\bar{\varepsilon}^a_{zz} = \frac{1}{3}((\bar{B}_1 - B_1 + \frac{\bar{N}_a}{r_1} - \frac{N_a}{r_1})\delta u_r + (2B_2 + \bar{B}_2)\delta u_z) \tag{B.36}$$

$$\delta\bar{\gamma}^a_{rz} = \frac{1}{3}(3B_2\delta u_r + 3B_1\delta u_z) \tag{B.37}$$

$$\delta\bar{\varepsilon}^a_{\theta\theta} = \frac{1}{3}((\bar{B}_1 - B_1 + 2\frac{N_a}{r_1} + \frac{\bar{N}_a}{r_1})\delta u_r + (\bar{B}_2 - B_2)\delta u_z) \tag{B.38}$$

Replacing the B_i terms with the shape function values

$$\delta\bar{\varepsilon}^a_{rr} = \frac{1}{3}\left[\left(2\frac{\partial N}{\partial r} + \overline{\frac{\partial N}{\partial r}} + \overline{\frac{N_a}{r_1}} - \frac{N_a}{r_1}\right)\delta u_r + \left(\overline{\frac{\partial N}{\partial z}} - \frac{\partial N}{\partial z}\right)\delta u_z\right] \tag{B.39}$$

$$\delta\bar{\varepsilon}^a_{zz} = \frac{1}{3}\left[\left(\overline{\frac{\partial N}{\partial r}} - \frac{\partial N}{\partial r} + \overline{\frac{N_a}{r_1}} - \frac{N_a}{r_1}\right)\delta u_r + (2\frac{\partial N}{\partial z} + \overline{\frac{\partial N}{\partial z}})\delta u_z\right] \tag{B.40}$$

$$\delta\bar{\gamma}^a_{rz} = \frac{\partial N}{\partial z}\delta u_r + \frac{\partial N}{\partial r}\delta u_z \tag{B.41}$$

$$\delta\bar{\varepsilon}^a_{\theta\theta} = \frac{1}{3}\left[\left(\overline{\frac{\partial N}{\partial r}} - \frac{\partial N}{\partial r} + 2\frac{N_a}{r_1} + \overline{\frac{N_a}{r_1}}\right)\delta u_r + (\overline{\frac{\partial N}{\partial z}} - \frac{\partial N}{\partial z})\delta u_z\right] \tag{B.42}$$

The above components should reduce to the regular definitions of the strain components when the bar terms are evaluated at the particle locations instead of the element centers. The results of this process are shown below

$$\delta\varepsilon^a_{rr} = \frac{\partial N}{\partial r}\delta u_r \tag{B.43}$$

$$\delta\varepsilon^a_{zz} = \frac{\partial N}{\partial z}\delta u_z \tag{B.44}$$

$$\delta\gamma^a_{rz} = \frac{\partial N}{\partial z}\delta u_r + \frac{\partial N}{\partial r}\delta u_z \tag{B.45}$$

$$\delta\varepsilon^a_{\theta\theta} = \frac{N_a}{r_1}\delta u_r \tag{B.46}$$

The above components are correct.

B.1.3 2D plane strain

The 2D plane strain $\bar{\mathbf{B}}_a^{plane}$-matrix (from Hughes [2000]) can be generated from the $\bar{\mathbf{B}}_a^{axi}$-matrix by setting the $B_0 = 0$. Defining the incremental displacement vector as

$$\delta\mathbf{u} = \begin{bmatrix} \delta u_x \\ \delta u_y \end{bmatrix}$$

The $\bar{\mathbf{B}}_a^{plane}$ matrix takes the following form (Hughes [2000])

$$\bar{\mathbf{B}}_a^{plane} = \frac{1}{3}\begin{bmatrix} 2B_1 + \bar{B}_1 & \bar{B}_2 - B_2 \\ \bar{B}_1 - B_1 & 2B_2 + \bar{B}_2 \\ 3B_2 & 3B_1 \\ - - - - - & - - - - - \\ \bar{B}_1 - B_1 & \bar{B}_2 - B_2 \end{bmatrix} \tag{B.47}$$

Altrernatively $\bar{\mathbf{B}}_a^{plane}$ can be derived from B.1.1, by setting $u_z = 0$ and assuming full slip on the z-face ($\frac{\partial \cdot}{\partial z} = 0$). The reason you zero the z-terms is because there is zero displacement out of "plane" however due to the averaging that takes place

$$\delta\overline{\boldsymbol{\varepsilon}_{zz}}^a \neq 0$$

as it's a function of u_x and u_y. Applying these assumptions to the incremental displacement vector and $\mathbf{B}_a$ gives

$$\delta\mathbf{u}^a = \begin{bmatrix} \delta u_x \\ \delta u_y \\ \delta u_z = 0 \end{bmatrix} \tag{B.48}$$

$$\mathbf{B}_{\mathbf{a}}^{\mathbf{plane}} = \frac{1}{3}\begin{bmatrix} 2B_1 + \bar{B}_1 & \bar{B}_2 - B_2 & 0 \\ \bar{B}_1 - B_1 & 2B_2 + \bar{B}_2 & 0 \\ \bar{B}_1 - B_1 & \bar{B}_2 - B_2 & 0 \\ -- & -- & -- \\ 0 & 0 & 3B_2 \\ 0 & 0 & 3B_1 \\ 3B_2 & 3B_1 & 0 \end{bmatrix} \tag{B.49}$$

The 3rd column can be removed because all the terms operate on $u_z = 0$, secondly the 4th and 5th rows can be removed because they are filled with zeros. Resulting in

$$\mathbf{B}_{\mathbf{a}}^{\mathbf{plane}} = \frac{1}{3}\begin{bmatrix} 2B_1 + \bar{B}_1 & \bar{B}_2 - B_2 \\ \bar{B}_1 - B_1 & 2B_2 + \bar{B}_2 \\ \bar{B}_1 - B_1 & \bar{B}_2 - B_2 \\ -- & -- \\ 3B_2 & 3B_1 \end{bmatrix}$$

As it confused the author, note the matrix dimensions from Hughes [2000] are correct. The averaging adds terms to the strains and B-matrix.

Rearranging the above matrix to follow the order Hughes Hughes [2000] follows

$$\mathbf{B}_{\mathbf{a}}^{\mathbf{plane}} = \frac{1}{3}\begin{bmatrix} 2B_1 + \bar{B}_1 & \bar{B}_2 - B_2 \\ \bar{B}_1 - B_1 & 2B_2 + \bar{B}_2 \\ 3B_2 & 3B_1 \\ -- & -- \\ \bar{B}_1 - B_1 & \bar{B}_2 - B_2 \end{bmatrix}$$

This rearrangement reorders the elements of $\delta\overline{\boldsymbol{\varepsilon}}$ but doesn't change any values. In component form the nodal strains are

$$\delta\overline{\varepsilon}^a_{xx} = \frac{1}{3}\left[\left(2B_1 + \overline{B_1}\right)\delta u_x + \left(\overline{B_2} - B_2\right)\delta u_y\right] \tag{B.50}$$

$$\delta\overline{\varepsilon}^a_{yy} = \frac{1}{3}\left[\left(\overline{B_1} - B_1\right)\delta u_x + \left(2B_2 + \overline{B}_2\right)\delta u_y\right] \tag{B.51}$$

$$\delta\overline{\gamma}^a_{xy} = B_2\delta u_x + B_1\delta u_y \tag{B.52}$$

$$\delta\overline{\varepsilon}^a_{zz} = \frac{1}{3}\left[\left(\overline{B_1} - B_1\right)\delta u_x + \left(\overline{B_2} - B_2\right)\delta u_y\right] \tag{B.53}$$

Replacing the B_i values with their shape function derivative definitions

$$\delta\overline{\varepsilon}^a_{xx} = \frac{1}{3}\left[\left(2\frac{\partial N}{\partial x} + \overline{\frac{\partial N}{\partial x}}\right)\delta u_x + \left(\overline{\frac{\partial N}{\partial y}} - \frac{\partial N}{\partial y}\right)\delta u_y\right] \tag{B.54}$$

$$\delta\overline{\varepsilon}^a_{yy} = \frac{1}{3}\left[\left(\overline{\frac{\partial N}{\partial x}} - \frac{\partial N}{\partial x}\right)\delta u_x + \left(2\frac{\partial N}{\partial y} + \overline{\frac{\partial N}{\partial y}}\right)\delta u_y\right] \tag{B.55}$$

$$\delta\overline{\gamma}^a_{xy} = \frac{\partial N}{\partial y}\delta u_x + \frac{\partial N}{\partial x}\delta u_y \tag{B.56}$$

$$\delta\overline{\varepsilon}^a_{zz} = \frac{1}{3}\left[\left(\overline{\frac{\partial N}{\partial x}} - \frac{\partial N}{\partial x}\right)\delta u_x + \left(\overline{\frac{\partial N}{\partial y}} - \frac{\partial N}{\partial y}\right)\delta u_y\right] \tag{B.57}$$

To check the derivations the bar terms are evaluated at the particle locations, the following strain result,

$$\delta\varepsilon^a_{xx} = \frac{\partial N}{\partial x}\delta u_x \tag{B.58}$$

$$\delta\varepsilon^a_{yy} = \frac{\partial N}{\partial y}\delta u_y \tag{B.59}$$

$$\delta\gamma^a_{xy} = \frac{\partial N}{\partial y}\delta u_x + \frac{\partial N}{\partial x}\delta u_y \tag{B.60}$$

$$\delta\varepsilon^a_{zz} = 0 \tag{B.61}$$

The above is correct.

Confusion on 2D-plane strain

The last row of the $\bar{\mathbf{B}}^{plane}_a$ presented in Hughes [2000] results from $\bar{\varepsilon_{zz}}$ being a function of u_x and u_y. The derivation presented by Bisht (2021) Bisht [2021] seems to be a an approximate solution for $\bar{\mathbf{B}}^{plane}_a$. If $\bar{\varepsilon_{zz}} = 0$ is assumed and the average is only done for the u_x and u_y the the Bisht (2021) matrix is formed.

From Bisht [2021]

$$\bar{\mathbf{B}}^{plane}_a = \frac{1}{2}\begin{bmatrix} \frac{\partial N}{\partial x}(P) + \frac{\partial N}{\partial x}(C) & \frac{\partial N}{\partial y}(C) - \frac{\partial N}{\partial y}(P) \\ \frac{\partial N}{\partial x}(C) - \frac{\partial N}{\partial x}(P) & \frac{\partial N}{\partial y}(P) + \frac{\partial N}{\partial y}(C) \\ 2\frac{\partial N}{\partial y}(P) & 2\frac{\partial N}{\partial x}(P) \end{bmatrix}$$

B.1.4 B-bar Implementation

The B-bar implementation detailed here is **Example 1 - A Generalization of Selective Integration** (Hughes (1987)). Compared to Example 2 of Hughes (1987), selective integration seems to simpler to implement and is the technique used by Martinelli and Galvi (2022) Martinelli and Galavi [2022] and is a subcase of the technique used by Bisht (2021) Bisht [2021]. Bisht (2021) used the B-bar method with GIMP, which modifies the shape function

derivatives that are used in forming the **B** and **B** matrix. Specifically, the technique in the above papers, uses a one point reduced Gauss quadrature integration scheme. The general idea of the selective integration, is that a new set of Gauss points (reduced quadrature points) are defined and used to form the $\bar{B}_i$ components. To form $\mathbf{B}^{dil}$ the following equation is used (This is a combination of Hughes Eqn. (4.5.17) and (4.5.18)),

$$\bar{B}_i(\xi) = \sum_{\tilde{a}=1}^{\tilde{n}_{int}} \tilde{N}_{\tilde{a}}(\xi) B_i(\tilde{\xi}_{\mathbf{a}}) \tag{B.62}$$

(Note for 3D problems i =1, 2, 3)

where $\bar{B}_i(\xi)$ is the ith element of the $\bar{B}^{dil}$ matrix, $\tilde{N}_{\tilde{a}}$ are the newly defined shape functions which serve to interpolate the reduced quadrature points, B_i is the shape function derivates that we already use and $\tilde{\xi}_a$ are the locations of the reduced quadrature Gauss points. For the one point reduced Gauss quadrature equation B.62 reduces to

$$\bar{B}_i(\xi) = B_i(\mathbf{0}) \tag{B.63}$$

This is what Martinelli and Galavi [2022] and Bisht [2021] refer to when they say the B values are evaluated at the element centers.

Therefore to form the $\mathbf{B}_i$ matrix, we need to evaluate the shape functions (N_a), these are required for the 2D-Axisymmetric case, and the shape function derivatives $N_{a,i}$ at the $\xi = \mathbf{0}$.

Appendix C

Invariant Notes

This section contains information on the stress/strain invariants and their derivatives. This was done to help clarify future work and act as a centralized resource.

C.1 Information on Stress Invariants

This section holds information about the stress invariants used here and the relationship between them. The mean stress, p is defined as

$$p = \frac{I_1(\boldsymbol{\sigma})}{3} \tag{C.1}$$

$$= \frac{Tr(\boldsymbol{\sigma})}{3} \tag{C.2}$$

$$= \frac{\sigma_{ii}}{3} = \frac{(\sigma_{11} + \sigma_{22} + \sigma_{33})}{3} \tag{C.3}$$

C.1.1 Deviatoric Stress Tensor Invariants

This section is information on the invariants of the deviatoric stress tensor. Good resources for this and their implications are in [Brannon et al., 2009, Truesdell and Noll, 1992, Potts and Zdravković, 2001].

First Invariant of the deviatoric stress tensor, J_1

The first invariant of the stress tensor is:

$$J_1 = Tr(\boldsymbol{s}) = 0 \tag{C.4}$$

as the tensor is traceless by construction.

Second Invariant of the deviatoric stress tensor, J_2

The J_2 invariant is defined as

$$J_2 = \frac{1}{2} s_{ij} s_{ji} = \frac{1}{2} Tr(\boldsymbol{s}^2) \tag{C.5}$$

If $\boldsymbol{s}$ is symmetric then $s_{ij} = s_{ji}$. (Note: if you square a matrix the diagonal stores the vector inner product of the columns and rows, therefore taking the trace of squared matrix gives the summation of the inner product between the $i = 1, 2, 3$ rows and columns.)

Here are a couple of the stress invariants that are direct relations to the J_2 invariant. Generally, q and J are called deviatoric stress in geotechnical engineering. Sometimes q is called the deviator stress (Wood [1991]). The use of q or J depends on the author. In general plasticity mechanics, q is called the Von-Mises stress and is denoted by σ_v.

$$J_2 = J^2 = \frac{\sigma_v^2}{3} = \frac{q^2}{3} \tag{C.6}$$

In component form J_2 for a symmetric matrix can be calculated as

$$J_2 = \frac{1}{2}(s_{11}^2 + s_{22}^2 + s_{33}^2 + 2s_{12}^2 + 2s_{13}^2 + 2s_{23}^2) \tag{C.7}$$

As q is commonly used in geotechnical engineering the component form for a symmetric $\boldsymbol{s}$ is defined below. This definition is taken from Wood [1991] equation (1.35) page 21.

$$q = \sqrt{3J_2} \tag{C.8}$$

$$= \sqrt{\frac{(\sigma_{22} - \sigma_{33})^2 + (\sigma_{zz} - \sigma_{xx})^2 + (\sigma_{xx} - \sigma_{yy})^2}{2} + 3(\tau_{23}^2 + \tau_{31}^2 + \tau_{12}^2)} \tag{C.9}$$

$$= \sqrt{\frac{3}{2} s_{ij} s_{ji}} \tag{C.10}$$

Third invariant of the deviatoric stress tensor, J_3

The J_3 stress invariant is defined as follows

$$J_3 = \det(dev(\boldsymbol{\sigma})) = \det(\boldsymbol{s}) \tag{C.11}$$

Lode's Angle

Lode's angle is useful as a representation of loading angle in stress invariant space (Brannon et al. [2009]). There are a couple definition of Lode's angle that are commonly used. In this work the one that was used is defined as

$$\theta = -\frac{1}{3} arcsin\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right) \tag{C.12}$$

As defined here the range of Lode's Angle is

$$-\frac{\pi}{6} \leq \theta \leq \frac{\pi}{6} \tag{C.13}$$

A scaling of the mean stress ($\sqrt{3}p$), a scaling of the deviatoric stress ($\sqrt{2J_2}$) and Lode's angle forms an orthogonal basis for stress space. Figure C.1 from Potts and Zdravković [2001] shows the invariants in principal stress space.

C.1.2 Derivatives of the stress tensor invariants

Using equation C.1, the derivative is p with respect to stress is

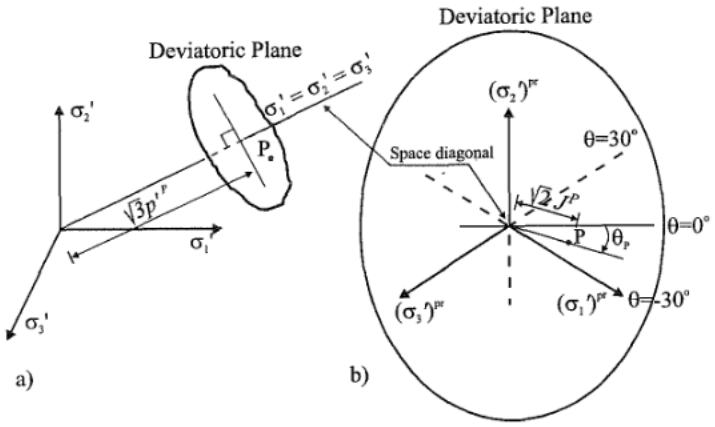


Figure 5.2: Invariants in principal stress space

Figure C.1: Common geotechnical invariants in principal stress space (Taken from Potts and Zdravković [2001])

$$\frac{\partial p}{\partial \boldsymbol{\sigma}} = \begin{bmatrix} 1/3 & 0 & 0 \\ 0 & 1/3 & 0 \\ 0 & 0 & 1/3 \end{bmatrix} \tag{C.14}$$

In Voigt notation

$$\frac{\partial p}{\partial \boldsymbol{\sigma}} = \begin{bmatrix} 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \end{bmatrix} \tag{C.15}$$

C.1.3 Derivatives of the deviatoric stress tensor invariants

This section holds the derivatives of the deviatoric stress invariants that are commonly of interest for soil constitutive models.

Derivative of the J_2 invariant

For a general deviatoric stress tensor, the derivative is

$$\frac{\partial J_2}{\partial \boldsymbol{\sigma}} = \frac{\partial J_2}{\partial \boldsymbol{s}} = \boldsymbol{s} \tag{C.16}$$

However, for a symmetric matrix if the shear terms are combined in J_2 the shear terms in the derivative are doubled. In Voigt notation this gives

$$\frac{\partial J_2}{\partial \boldsymbol{\sigma}} = \frac{\partial J_2}{\partial \boldsymbol{s}} = \begin{bmatrix} s_{11} & s_{22} & s_{33} & 2\sigma_{12} & 2\sigma_{13} & 2\sigma_{23} \end{bmatrix} \tag{C.17}$$

To get the derivative of q and J you can use the product rule and the above definitions. $\frac{\partial J}{\partial \boldsymbol{\sigma}}$ can be found in Potts and Zdravković [2001] equation (VII.6) page 186 and for convience is repeated here.

$$\frac{\partial J}{\partial \boldsymbol{\sigma}} = \frac{1}{2J} \begin{bmatrix} s_{11} & s_{22} & s_{33} & 2\sigma_{12} & 2\sigma_{13} & 2\sigma_{23} \end{bmatrix} \tag{C.18}$$

For completeness $\frac{\partial q}{\partial \boldsymbol{\sigma}}$ is also included.

$$\frac{\partial q}{\partial \boldsymbol{\sigma}} = \frac{3}{2q} \frac{\partial J_2}{\partial \boldsymbol{\sigma}} \tag{C.19}$$

Derivative of the J_3 invariant

This derivative holds for all symmetric deviatoric tensors, whether invertible or not (Banerjee [2007]).

$$\frac{\partial J_3}{\partial \boldsymbol{s}} = \frac{\partial J_3}{\partial \boldsymbol{\sigma}} = \boldsymbol{s}^2 - \frac{2}{3} J_2 \mathbf{1} \tag{C.20}$$

$$= \boldsymbol{s}^2 - \frac{1}{3} (\boldsymbol{s}^2 : \mathbf{1}) \mathbf{1} \tag{C.21}$$

$$= \boldsymbol{s}^2 - \frac{I_1(\boldsymbol{s}^2)}{3} \mathbf{1} \tag{C.22}$$

$$= dev[\boldsymbol{s}^2] \tag{C.23}$$

The equality of $\frac{\partial J_3}{\partial \boldsymbol{s}}$ and $\frac{\partial J_3}{\partial \boldsymbol{\sigma}}$ comes from an identity for

$$\frac{\partial \sigma_{mn}}{\partial s_{ij}} = \frac{1}{2} (\delta_{mi}\delta_{nj} + \delta_{mj}\delta_{ni}) \tag{C.24}$$

This equality is presented in the Option 3: J3 derivative appendix of Imechanica. However, if Voigt notation is used the contributions of the shear terms need to be doubled. The following shows the component form, assuming Voigt notation or simply the shear terms in the matrix are combined. For example, when σ_{12} changes in Voigt notation this means that σ_{21} has also changed and therefore this has to be accounted for in the calculation.

If you start from Voigt notation (where the shear terms are combined) then $\frac{\partial J_3}{\partial \boldsymbol{\sigma}}$ is

$$\frac{\partial J_3}{\partial \sigma_{11}} = \frac{1}{9}(2\sigma_{11}^2 - \sigma_{22}^2 - \sigma_{33}^2 - 2\sigma_{11}\sigma_{22} - 2\sigma_{11}\sigma_{33} + 4\sigma_{22}\sigma_{33}) + \frac{1}{3}(\sigma_{12}^2 + \sigma_{13}^2 - 2\sigma_{23}^2) \quad \text{(C.25)}$$

$$\frac{\partial J_3}{\partial \sigma_{22}} = \frac{1}{9}(-\sigma_{11}^2 + 2\sigma_{22}^2 - \sigma_{33}^2 - 2\sigma_{11}\sigma_{22} + 4\sigma_{11}\sigma_{33} - 2\sigma_{22}\sigma_{33}) + \frac{1}{3}(\sigma_{12}^2 - 2\sigma_{13}^2 + \sigma_{23}^2) \quad \text{(C.26)}$$

$$\frac{\partial J_3}{\partial \sigma_{33}} = \frac{1}{9}(-\sigma_{11}^2 - \sigma_{22}^2 + 2\sigma_{33}^2 + 4\sigma_{11}\sigma_{22} - 2\sigma_{11}\sigma_{33} - 2\sigma_{22}\sigma_{33}) + \frac{1}{3}(-2\sigma_{12}^2 + \sigma_{13}^2 + \sigma_{23}^2) \quad \text{(C.27)}$$

$$\frac{\partial J_3}{\partial \sigma_{12}} = \frac{1}{3}(2t_{11}t_{12} + 2t_{12}t_{22} - 4t_{12}t_{33}) + 2t_{13}t_{23} \quad \text{(C.28)}$$

$$\frac{\partial J_3}{\partial \sigma_{13}} = \frac{1}{3}(2t_{11}t_{13} - 4t_{13}t_{22} + 2t_{13}t_{33}) + 2t_{12}t_{23} \quad \text{(C.29)}$$

$$\frac{\partial J_3}{\partial \sigma_{23}} = \frac{1}{3}(-4t_{11}t_{23} + 2t_{22}t_{23} + 2t_{23}t_{33}) + 2t_{12}t_{13} \quad \text{(C.30)}$$

C.1.4 Derivatives of stress invariants

This section includes the derivatives of the invariants of the stress and deviatoric stress tensors.

Derivative of Lode's Angle

The following is a derivation of $\frac{\partial \theta}{\partial \boldsymbol{\sigma}}$

$$\theta = \frac{-1}{3} arcsin\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right) \tag{C.31}$$

$$sin(-3\theta) = \frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}} \tag{C.32}$$

$$\tag{C.33}$$

Looking at the term on the left of the equal sign

$$\frac{\partial(sin(-3\theta))}{\partial \boldsymbol{\sigma}} = -3cos(-3\theta)\frac{\partial \theta}{\partial \sigma} \tag{C.34}$$

using that $cos(\theta)$ is an even function (ie. $cos(\theta) = cos(-\theta)$) then

$$\frac{\partial(sin(-3\theta))}{\partial \boldsymbol{\sigma}} = -3cos(3\theta)\frac{\partial \theta}{\partial \sigma} \tag{C.35}$$

On the right side

$$\frac{\partial}{\partial \boldsymbol{\sigma}} \left(\frac{3\sqrt{3}}{2} \frac{J_3}{J_2^{3/2}} \right) \tag{C.36}$$

$$= \frac{3\sqrt{3}}{2} \left(J_2^{-3/2} \frac{\partial J_3}{\partial \boldsymbol{\sigma}} + J_3 \frac{\partial (J_2^{-3/2})}{\partial \boldsymbol{\sigma}} \right) \tag{C.37}$$

$$= \frac{3\sqrt{3}}{2} \left(J_2^{-3/2} \frac{\partial J_3}{\partial \boldsymbol{\sigma}} - \frac{3J_3}{2} J_2^{-5/2} \frac{\partial J_2}{\partial \boldsymbol{\sigma}} \right) \tag{C.38}$$

$$= \frac{3\sqrt{3}}{2J_2^{3/2}} \left(\frac{\partial J_3}{\partial \boldsymbol{\sigma}} - \frac{3J_3}{2J_2} \frac{\partial J_2}{\partial \boldsymbol{\sigma}} \right) \tag{C.39}$$

Equating the two sides together

$$-3cos(3\theta) \frac{\partial \theta}{\partial \boldsymbol{\sigma}} = \frac{3\sqrt{3}}{2J_2^{3/2}} \left(\frac{\partial J_3}{\partial \boldsymbol{\sigma}} - \frac{3J_3}{2J_2} \frac{\partial J_2}{\partial \boldsymbol{\sigma}} \right) \tag{C.40}$$

$$\frac{\partial \theta}{\partial \boldsymbol{\sigma}} = \frac{\sqrt{3}}{2cos(3\theta) J_2^{3/2}} \left(\frac{3J_3}{2J_2} \frac{\partial J_2}{\partial \boldsymbol{\sigma}} - \frac{\partial J_3}{\partial \boldsymbol{\sigma}} \right) \tag{C.41}$$

Using the definition of J this can be cast into the same vaiables as used by Potts and Zdravković [2001], equation C.42 has an additional 3 on the first term in the parenthesis compared to equation (VII.8) in Potts and Zdravković [2001]. It's possible I made a mistake in my derivation, but I haven't been able to find it.

$$\frac{\partial \theta}{\partial \boldsymbol{\sigma}} = \frac{\sqrt{3}}{2cos(3\theta) J} \left(\frac{3J_3}{J} \frac{\partial J}{\partial \boldsymbol{\sigma}} - \frac{\partial J_3}{\partial \boldsymbol{\sigma}} \right) \tag{C.42}$$

Equation C.42 is undefined when $|\theta| = \frac{\pi}{6}$ due to the $cos(3\theta)$ in the denominator. Alternatively, the derivative of equation C.31 could be directly taken without taking the *sin* of both sides. This is shown in equation C.43. Equation C.31 and C.43 are equivalent.

$$\frac{\partial\theta}{\partial\boldsymbol{\sigma}} = -\frac{1}{3}\frac{\partial}{\partial\boldsymbol{\sigma}}\left[arcsin\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right)\right] \tag{C.43}$$

$$= -\frac{1}{3}\left[1-\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right)^2\right]^{-1/2}\frac{3\sqrt{3}}{2}\frac{\partial}{\partial\boldsymbol{\sigma}}(J_3 J_2^{-3/2}) \tag{C.44}$$

$$= -\frac{\sqrt{3}}{2}\left[1-\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right)^2\right]^{-1/2}\left(\frac{\partial J_3}{\partial\boldsymbol{\sigma}}J_2^{-3/2}-\frac{3}{2}J_3 J_2^{-5/2}\frac{\partial J_2}{\partial\boldsymbol{\sigma}}\right) \tag{C.45}$$

$$= \frac{\sqrt{3}}{2J_2^{3/2}}\left[1-\left(\frac{3\sqrt{3}}{2}\frac{J_3}{J_2^{3/2}}\right)^2\right]^{-1/2}\left(\frac{3J_3}{2J_2}\frac{\partial J_2}{\partial\boldsymbol{\sigma}}-\frac{\partial J_3}{\partial\boldsymbol{\sigma}}\right) \tag{C.46}$$

C.2 Information on Strain invariants

C.2.1 Definition of strain invariants

The volumetric strain invariant (ε_v) is defined as

$$\varepsilon_v = I_1(\boldsymbol{\varepsilon}) = Tr(\boldsymbol{\varepsilon}) = \varepsilon_{11}+\varepsilon_{22}+\varepsilon_{33} \tag{C.47}$$

The deviatoric strain invariant (ε_q) is defined as

$$\varepsilon_q = \sqrt{\frac{2}{3}I_2(\boldsymbol{\varepsilon^{dev}})} \tag{C.48}$$

$$= \sqrt{\frac{2}{3}\|\boldsymbol{\varepsilon}^{dev}\|_F^2} \tag{C.49}$$

$$= \sqrt{\frac{2}{3}Tr(\boldsymbol{\varepsilon}^{dev}\cdot\boldsymbol{\varepsilon}^{dev})} \tag{C.50}$$

$$= \sqrt{\frac{2}{3}\boldsymbol{\varepsilon}^{dev}:\boldsymbol{\varepsilon}^{dev}} \tag{C.51}$$

$$\tag{C.52}$$

where $\boldsymbol{\varepsilon}^{dev}$ is the deviatoric strain tensor.

C.2.2 Derivatives of the strain invariant

As ε_v is the first invariant of the strain tensor

$$\frac{\partial \varepsilon_v}{\partial \boldsymbol{\varepsilon}} = \mathbf{1} \tag{C.53}$$

where $\mathbf{1}$ is the second order identity tenor (Identity matrix). As ε_q is the second invariant of the deviatoric strain tensor

$$\frac{\partial \varepsilon_q}{\partial \boldsymbol{\varepsilon}} = \frac{\partial \varepsilon_q}{\partial \boldsymbol{\varepsilon}^{dev}} = \frac{\partial}{\partial \boldsymbol{\varepsilon}} \left(\sqrt{\frac{2}{3} Tr(\boldsymbol{\varepsilon}^{dev} \boldsymbol{\varepsilon}^{dev})} \right) \tag{C.54}$$

$$= \frac{1}{3\varepsilon_q} (\boldsymbol{\varepsilon}^{dev,T} + \boldsymbol{\varepsilon}^{dev,T}) = \frac{1}{3\varepsilon_q} (\boldsymbol{\varepsilon}^{dev} + \boldsymbol{\varepsilon}^{dev}) \tag{C.55}$$

$$= \frac{2}{3\varepsilon_q} \boldsymbol{\varepsilon}^{dev} \tag{C.56}$$

For more information on taking the derivative of the trace of a matrix see Traa. Similar to the derivatives of the stress invariants when the calculations are done in Voigt notation the shear terms need to be doubled from there original matrix values. However, unlike the stress invariants the Voigt strain vector already contains shear terms that are doubled. Hence, that vector that's used in the calculation of ε_q is the already created Voigt deviatoric strain vector and the additional factor of 2 is already considered.

C.3 Frobenius Norm and the tensor inner products

The Frobenius norm for a matrix is defined as

$$\|A\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} \tag{C.57}$$

or alternatively where Tr is the trace of matrix and A^H is the conjugate transpose of the matrix [Truesdell, 1984].

$$\|A\|_F \equiv \sqrt{Tr(AA^H)} \tag{C.58}$$

Hence, the contraction of a symmetric matrix is equivalent to the square of the Frobenius norm of the matrix. In Fortran, if the dimension isn't passed, the function $norm2$ calculates the Frobenius norm of an arbitrarily sized tensor [Gen].

Appendix D

BlueDrop Force Derivation

D.1 BlueDrop Force Derivation

Newton's second law is

$$\sum \vec{F} = m\vec{a} \tag{D.1}$$

Assumming a reference frame fixed to the Earth and neglecting rotational affects, Newton's second law applied to the PFFP gives

$$\sum \vec{F} = m_p \vec{a}_p \tag{D.2}$$

where $\sum \vec{F}$ is the net force on the penetrometer, m_p is the mass of the penetrometer and $\vec{a_p}$ is the acceleration of the penetrometer. Taking the direction opposite of gravity (towards the sky) to be positive the following force balance results during impact,

$$m_p \vec{a}_p = \vec{F}_T + \vec{F}_B + \vec{F}_D + \vec{F}_{SB} + \vec{F}_f + \vec{F}_{SD} + \vec{F}_{Br} - \vec{F}_g \tag{D.3}$$

where, F_T is the tension in the rope, F_B is the buoyant force due to the fluid, F_D is the drag force due to the fluid, F_{SB} is the buoyant force due the soil, F_f is the frictional resistance from the soil, F_{SD} is the drag force in the soil, F_{Br} is the soils bearing resistance. Assuming that soil buoyant forces, soil frictional resistance, and the tension in the rope is negligible in equation D.3, gives

$$m_p \vec{a}_p = \vec{F}_B + \vec{F}_D + \vec{F}_{Br} - \vec{F}_g \tag{D.4}$$

then plugging in the associated models

$$m_p \vec{a}_p = \rho_f V_p g + \frac{1}{2}\rho_f v_p^2 C_D A + F_{Br} - m_p g \tag{D.5}$$

where ρ_f is the density of fluid (usually assumed to be $1020\,kg/m^3$ for seawater and $0\,kg/m^3$ for air), V_p is the volume of the penetrometer, g is the magnitude of gravitational acceleration ($9.81\,m/s^2$), v_p is the velocity of the penetrometer, C_D is the PFFP's drag coefficient, and A is the PFFPs frontal area. Rearranging for F_{Br} gives

$$F_{Br} = m_p\vec{a}_p + m_p g - \frac{1}{2}\rho_f v_p^2 C_D A - \rho_f V_p g \tag{D.6}$$

The accelerometer is calibrated to measure acceleration in an inertial reference frame. Therefore, the raw acceleration readings (a_s) are $\vec{a}_s = \vec{a}_p + g$. Rearranging equation D.6 so that the raw sensor acceleration can be inserted gives

$$F_{Br} = m_p(\vec{a}_s - g) + m_p g - \frac{1}{2}\rho_f v_p^2 C_D A - \rho_f V_p g \tag{D.7}$$

This is the general form of the equation that is used to compute the bearing resistance of the soil during impact. For drops of the PFFP that only fall through the air, $\vec{F}_D$ and $\vec{F}_B$ is assumed negligible. Therefore, for air drops the equation reduces to

$$F_{Br} = m_p(\vec{a}_s - g) + m_p g \tag{D.8}$$

At the time of writing this paper, the drag coefficient of the BlueDrop has not been determeined. Laboratory, field, and/or numerical testing needs to be done to determine the value. Therefore, for the time being the drag coeffient for fluid drops is assumed to be zero.

Appendix E

BlueDrop Analysis Library Structure

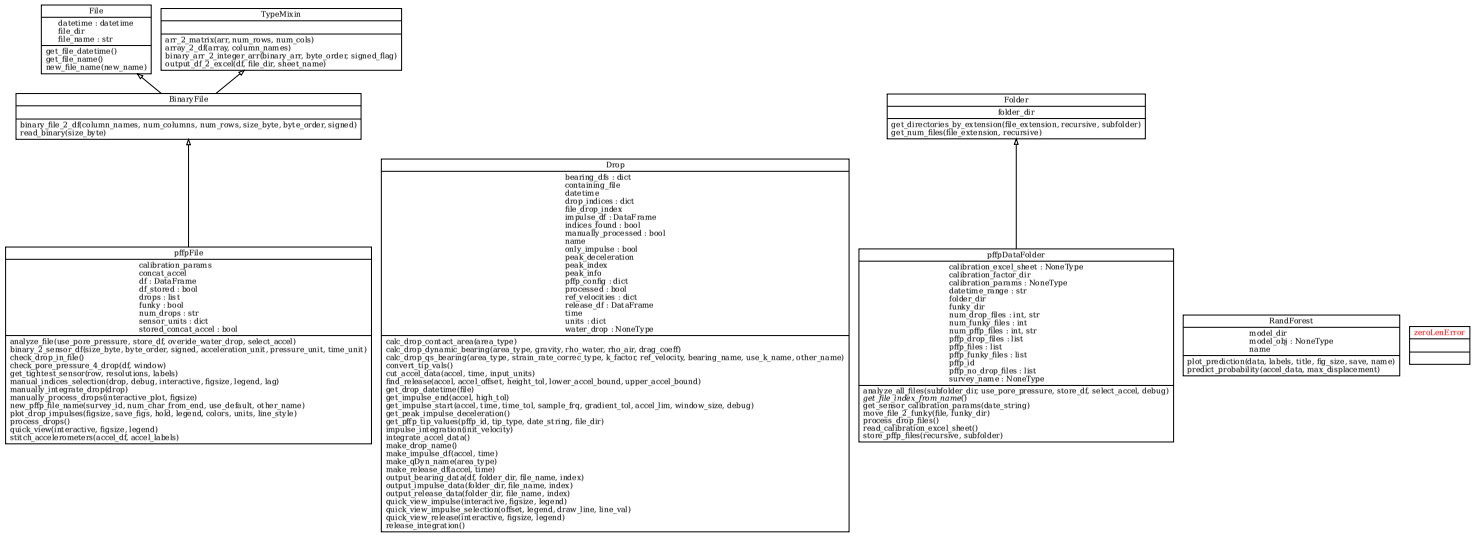


Figure E.1: Flow Chart of the OOP structure in the BlueDrop Analysis Library