Modeling of soil-water-structure interaction

A Finite Volume Method (FVM) approach to fully coupled soil analysis and interactions between wave, seabed and offshore structure

Tang, Tian; Johannesson, Björn ; Hededal, Ole; Rønby, Johan

Modeling of soil-water-structure interaction

- A Finite Volume Method (FVM) approach to fully coupled soil analysis and interactions between wave, seabed and offshore structure

Tian Tang

Ph.D. Thesis

Department of Civil Engineering
Technical University of Denmark

2014

Supervisors:

(Current) Associate Professor Björn Johannesson, Technical University of Denmark - DTU, Denmark
(Previous) Technical Director Ole Hededal, COWI Marine and Foundation Engineering, Denmark
Research Scientist Johan Rønby, DHI Port And Offshore Technology, Denmark

Assessment Committee:

Associate Professor Lars Vabbersgaard Andersen, Aalborg University, Denmark
Professor Hrvoje Jasak, University of Zagreb, Croatia
Professor Ida Lykke Fabricius, Technical University of Denmark - DTU, Denmark

Modeling of soil-water-structure interaction
-A Finite Volume Method (FVM) approach to fully coupled soil analysis and interactions between wave, seabed and offshore structure

Printed by DTU-Tryk
Department of Civil Engineering
Technical University of Denmark
ISBN: 9788778773579
ISSN: -

Preface

This thesis is submitted as a partial fulfillment of the requirements for obtaining the Danish Ph.D. degree from the Technical University of Denmark (DTU). The Ph.D. project was part of the 3-year GTS-university collaboration ”Fremtidens Marine Konstruktioner” (FMK) between DHI and DTU supported by the Ministry of Science, Technology and Innovation. The work title of the project has been ”Modeling of soil-water-structure interaction”.

The Ph.D. was carried out at the Department of Civil Engineering, Section for Geotechnics and Geology in the period of November 15th 2011 - November 14th 2014. The main supervision was undertaken by Associate Professor Ole Hededal from November 2011 to January 2014 and Associate Professor Björn Johannesson from February 2014 until the end of the study. The project also received co-supervision from Research Scientist Dr. Johan Rønby at DHI Port and Offshore Technology.

The thesis is divided into three parts. The first part introduces the research field, discusses the methodology, highlights the major findings and provides an overview of the work carried out within this project. The second part is a collection of papers which constitute the basis of the work and describe the work in greater detail and serves as scientific documentation. The last part is an appended implementation guide that provides practical explanations on the developed numerical codes for knowledge sharing.

Lyngby, the 14$^{\text{th}}$ of November 2014

Tian Tang

Acknowledgements

First and most importantly, I would like to express my sincere gratitude to my supervisors: Ole, Björn and Johan, for providing me this precious learning opportunity at DTU, and for their invaluable suggestions, inspirations and encouragements during the whole project.

As a part of the Ph.D. studies, I went to University College Dublin (UCD) working with the solid mechanics OpenFOAM group. My special gratefulness is given to Dr. Philip Cardiff and Professor Alojz Ivankovic for sharing their experiences and knowledge in the computational solid mechanics.

In addition, I also visited DHI during my study. I greatly appreciate the warm and friendly hosting I received from the Port & Offshore Technology department.

I would also like to thank the entire staff at the Section for Geotechnics and Geology, DTU BYG. Working with them in the past three years has been a joyful memory.

Moreover, I wish to give a big thanks to Kennedy Taveras for helping me with proofreading the thesis, and to Dennis Arreborg Hansen and Stine Skov Madsen for helping with the abstract translation in Danish. And finally, I would like to acknowledge my parents in China for their great love beyond the distance.

Abstract

The trend towards the installation of more offshore constructions for the production and transmission of marine oil, gas and wind power is expected to continue over the coming years. An important process in the offshore construction design is the assessment of seabed soil stability exposed to dynamic ocean waves. The goal of this research project is to develop numerical soil models for computing realistic seabed response in the interacting offshore environment, where ocean waves, seabed and offshore structure highly interact with each other.

The seabed soil models developed are based on the 'modified' Biot's consolidation equations, in which the soil-pore fluid coupling is extended to account for the various nonlinear soil stress-strain relations included. The Finite volume method (FVM) together with a segregated solution strategy has been used to numerically solve the governing equations. In the FVM segregated scheme, the conventional linear and uncoupled terms are discretized implicitly, whereas the nonlinear and coupled terms are discretized explicitly by using available values from previous time level or iteration step. The implicit-explicit discretisation approach leads to linearized and decoupled algebraic systems, which are solved using the fixed-point iteration method. Upon the convergence of the iterative method, fully nonlinear coupled solutions are obtained. The developed nonlinear coupled soil models are capable of predicting the transient and gradual pore pressure variations as well as the developed nonlinear soil displacements and stresses under monotonic and cyclic loading.

With the FVM nonlinear coupled soil models as a basis, multiphysics modeling of wave-seabed-structure interaction is carried out. The computations are done in an open source code environment, OpenFOAM, where FVM models of Computational Fluid Dynamics (CFD) and structural mechanics are available. The interaction in the system is modeled in a 1-way manner: First detailed free surface CFD calculations are executed to obtain a realistic wave field around a given structure. Then the dynamic structural response, due to the motions in the surrounding water, are calculated using a linear elastic solver. Finally, the direct wave loads on the seabed and the indirect wave loads on the seabed-structure interface through the structure are provided as input for a dynamic soil response calculation. Simulation results in general demonstrate that, the interaction modeling provides improved wave loading environments for geotechnical assessment of the seabed soil.

Resumé

Over de kommende år forventes den nuværende trend med installation af flere offshore konstruktioner til transport og produktion af olie, gas og vindenergi at fortsætte. En vigtig del af designprocessen af offshore konstruktioner er vurderingen af havbundens geotekniske stabilitet under dynamisk påvirkning fra bølger. Målet med dette forskningsprojekt er at udvikle numeriske modeller til at beregne et realistisk respons af havbunden omkring offshore konstrucktioner, hvor interaktionen mellem bølger, havbund og konstruktion er høj.

De udviklede modeller af havbunden bygger på modificerede Biot konsolideringsligninger, hvor koblingen mellem kornskelet og porevæske er udvidet til at inkludere forskellige ikke-lineære spændings-tøjnings-relationer i havbunden. Finite Volume metoden (FVM) er benyttet sammen med en opdelt løsningsstrategi til at løse de styrende ligninger. I FVM løsningen blev de konventionelle lineære og ukoblede led diskretiseret implicit, hvorimod de ikke-lineære og koblede led blev diskretiseret eksplicit ved at benytte tilgængelige værdier fra forrige tids- eller iterationsskridt. Denne implicitte og eksplicitte tilgang til diskretiseringen fører til lineariserede og afkoblede ligningssystemer, der løses med fikspunkt iteration. Når der er opnået konvergens i den iterative metode, er resultatet en fuldt ikke-lineær koblet løsning. De udviklede ikke-lineære koblede jordmodeller kan både forudsige de transiente og jævne variationer i poretrykket samt de ikke-linære flytninger og spændinger i jorden under både monoton og cyklisk last.

Med den ikke-lineære finite volume koblede jordmodel som udgangspunkt, blev multifysisk modellering af bølge-havbund-konstruktion interaktion udført. Beregningerne er udført i Open Source miljøet OpenFOAM, hvor finite volume modeller for Computational Fluiddynamik (CFD) og strukturel mekanik er tilgængelige. Interaktionen i de beregnede systemer er envejs: Først udføres detaljerede fri-overflade CFD beregninger for at bestemme et realistisk bølgefelt omkring en given konstruktion. Derefter beregnes det dynamiske strukturelle respons som følge af bevægelserne i det omkringliggende vand, ved hjælp af en lineær-elastisk løser. Til sidst benyttes de direkte påvirkninger fra bølgerne på havbunden og de indirekte påvirkninger på havbund-struktur interfacet gennem konstruktionen som input til beregning af det dynamiske respons af havbunden. Resultaterne af simuleringerne demonstrerer generelt, at modelleringen af bølge-havbund-konstruktion interaktion giver forbedrede bølgelaster til geoteknisk vurdering af havbunden.

Contents

Part I

Extended summary

Chapter 1

Introduction

The awareness of the vast energy resources in the world's ocean has driven an increasing number of offshore constructions globally. Nowadays more than thirty percent of oil and gas energy is extracted from reservoirs beneath the seabed, Ferentinos (2013). Various kinds of platforms are built to support exploratory drilling equipment and the production plants required to process the hydrocarbons. Marine pipelines or tankers are used to transport the hydrocarbons to shore. Meanwhile, offshore wind power as an important renewable energy is also harvested by wind turbines constructed in the ocean. Favorably, offshore wind turbines are acknowledged with lower visual impact and larger production rates compared to onshore wind turbines.

1.1 Safety of offshore structure and foundation

Offshore foundations are used to transmit the structural design loadings to the subsoil, thus supporting the upper offshore structures. The type of foundation element to be employed will depend mainly on the water depth, the nature of loadings, and the seafloor characteristics. The traditional gravity-based foundations are large diameter concrete and/or steel structures. They are often ballasted with stones or sand to attain the necessary stability. Scour protection is placed around the toe of the structure. Another widely applied foundation type are the monopile foundations, which are long steel cylinders rammed down in the seabed. The monopile structures are relatively simple and insensitive to scour. The jacket (space frame) foundations consist of three or four legged steel lattice structures, secured to the seabed with piles at each leg position. This foundation type is very adaptable to increased water depth. Figure 1.1 presents an overview of the three above mentioned common foundation types. A new attractive form of offshore foundation concept is the suction caissons (also referred to as suction buckets), Figure 1.2. They are upturned buckets which are embedded in the soil by creating under pressures inside the caisson skirts. After installation, they are held in place by a combination of weight, friction and suction.

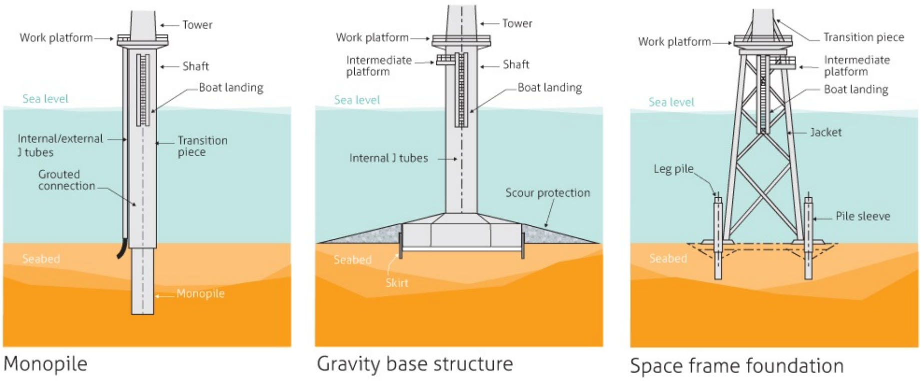


Figure 1.1: Classic foundation concepts for offshore structures, Figure from: http://www.navitusbaywindpark.co.uk/offshore-project-details.

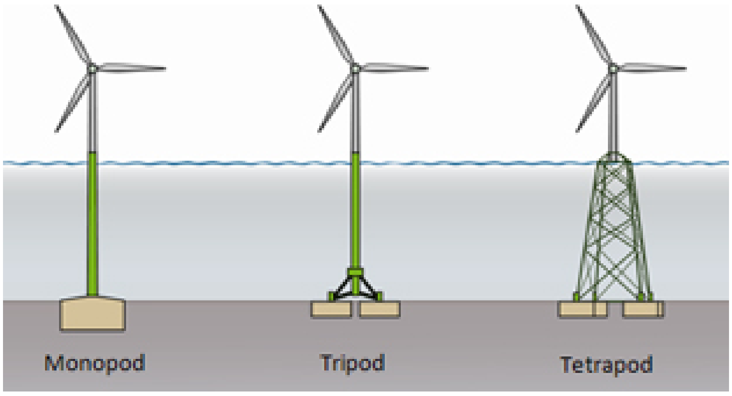


Figure 1.2: The new skirted caisson foundation concepts for offshore wind turbines, Figure from: http://www.ngi.no/en/Contentboxes-and-structures/Reference-Projects/Reference-projects/Skirted-Caisson-Foundations-for-Offshore-Structures.

The offshore structures together with their support foundations must function safely for the designed lifetimes (e.g., 20 years for offshore wind turbines and 25 years or more for offshore oil platforms, Sadeghi (2008)), against harsh marine environments where hydrodynamic interaction effects and dynamic response are present. Some important safety considerations are: peak loads created by storm winds and waves, fatigue loads generated by waves over time, the motions of the offshore structures induced by winds and/or waves, and the underlying soil conditions. Figure 1.3, as an example, demonstrates the various mishaps events on jack-up platforms observed over 20 years worldwide, Kvitrud et al. (2001). In fact, the "Soil" factor, as illustrated in Figure 1.3, has been attracting great attentions from offshore geotechnical engineers, and is also the focus of this PhD project. Seabed soil instability, including soil strength reductions, large permanent settlements and soil liquefaction failures, often involves the foundation of the structure and consequently impacts on the overall safety of the offshore structures.

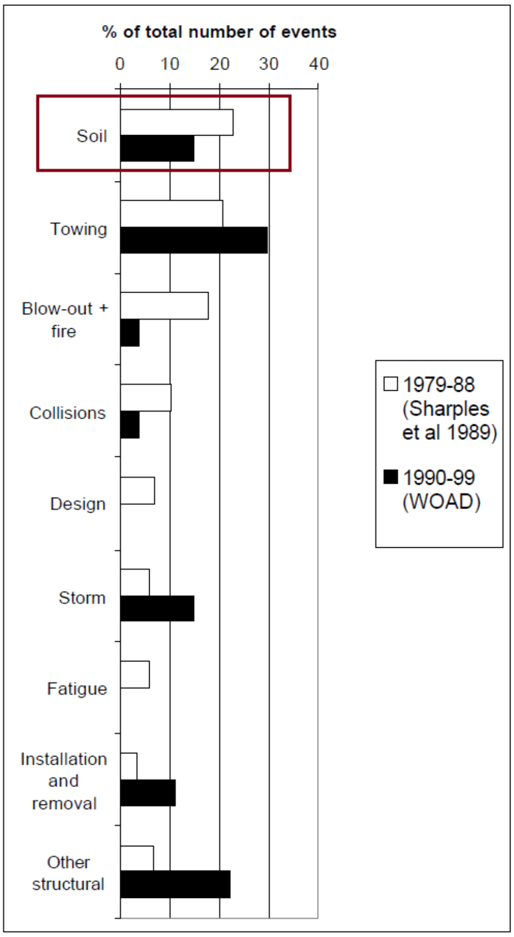


Figure 1.3: Incidents statistics of jack-up platform 'mishaps', from Kvitrud et al. (2001).

The primary goal of this research work is therefore to improve the understanding, analyzing and modeling of realistic seabed soil responses in an offshore environment, where ocean waves, soil and structures(foundations) highly interact with each other. In this way, we can help to prevent soil-related instabilities and improve the safety of our future offshore constructions.

1.2 Wave-induced seabed soil response

When designing offshore structures, assessing seabed soil behaviors under waves in the vicinity of the structure is critical to geotechnical engineers. Accurate assessment of the wave-induced pore pressure, soil displacements and stresses, however, has been a challenging task, mainly due to not only the complexity of the external dynamic wave loadings but also the complexity of the seabed material itself. The following subsections therefore attempt to provide background information about the two aspects.

1.2.1 The nature of wave loadings

When water waves propagate over the ocean, they generate dynamic wave pressures on the sea floor. The resulting dynamic wave pressures, in general, possess two distinct loading features: the progressive nature and the oscillatory nature.

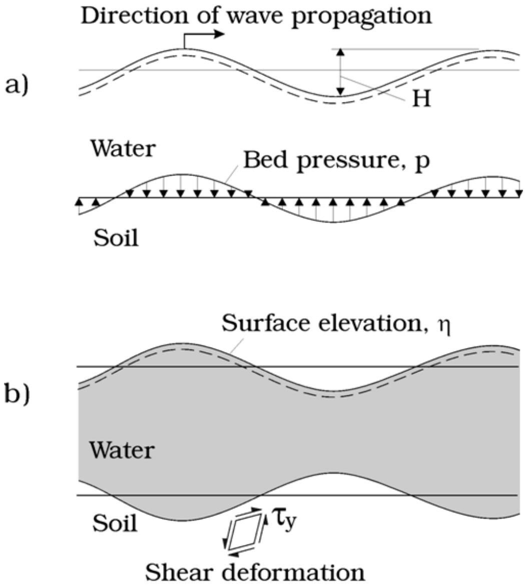


Figure 1.4: Illustration of the progressive wave loading nature, adopted from Sumer and Fredsøe (2002).

The progressive loading nature can be explained by reference to Figure 1.4, where a progressive wave over a horizontal seabed is illustrated. Along with the propagation of waves, the seabed undergoes periodic pressure variations, as sketched in Figure 1.4 a. The soil is compressed under the wave crest due to the increased bed pressure, and expanded under the wave trough owing to the opposite effect. Subsequently, the water-soil interface is nearly 180° out of phase with the water surface elevation, as shown in Figure 1.4 b. This results in the generation of shear stresses in the soil, and these shear stresses will vary periodically as the wave propagates.

The oscillatory loading nature, depicted in Figure 1.5, is mainly related to the generation of transient pore pressure gradient in the soil. The bed pressure fluctuations have a direct influence on changes in pore pressure in the seabed, with significant amplitude damping and phase lag effect. At each moment of wave oscillations, positive excess pore pressure generated in the soil due to a passage of a wave crest creates seepage forces acting downwards. Contrary to that, negative excess pore pressure induced by a passage of a wave trough is responsible for seepage forces directed upwards.

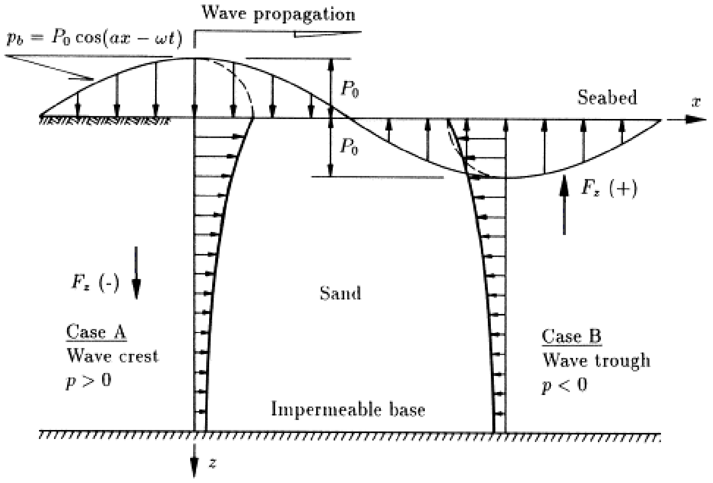


Figure 1.5: Illustration of the oscillatory wave loading nature. Adopted from Magda (2000).

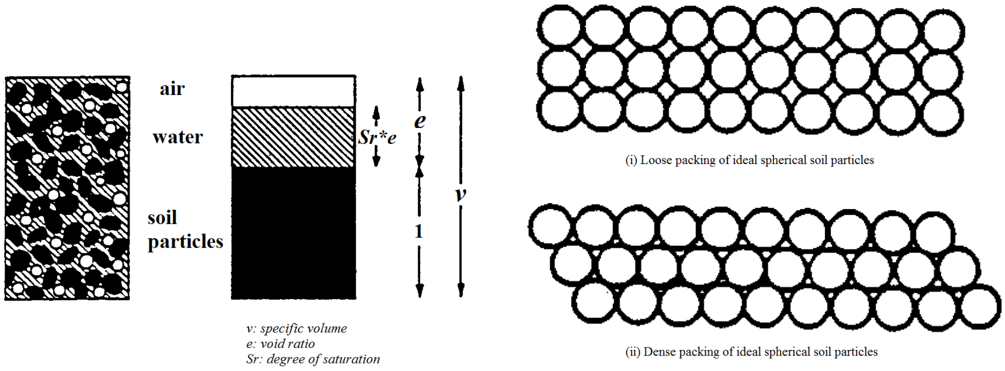
The oscillatory loading can be of great importance under the case of single extreme waves with large amplitudes, as a significant amount of upward seepage forces are generated under the wave trough. The upward seepage forces may completely uplift the soil and hence damage the structure. Whereas, the progressive loading scenario is more relevant to the gradual degradations of the stiffness and strength of soil due to cyclic shearing, for small to medium wave amplitudes.

1.2.2 The seabed soil properties

Seabed sediments cover a wide range of soil types, ranging from soft, impermeable silts and clays to stiff, permeable sands and gravels. Here, the focus will be on the non-cohesive soils, like sand and gravel. The limitation made is owing to that these "sandy" soils are generally the most sensitive to liquefaction - a soil failure phenomenon which is considered as a critical cause of severe offshore structure instabilities, Jeng (2003) and de Groot et al. (2006a,b).

In the case of saturated non-cohesive seabed, one deals with a two-phase medium. The two phases are the soil skeleton and the pore water. The pore water in a sandy seabed is usually not fully saturated and contains small amounts of air bubbles. Hence, a third phase, air, is also present in many cases, as illustrated in Figure 1.6a. Due to the presence of air bubbles in the seabed, the pore fluid is more compressible than compared to a fully saturated seabed condition.

The different volumetric packing of the soil particles due to the history of the soil deposit is another important feature of non-cohesive soils, as it will control the future responses to stress changes. As illustrated in Figure 1.6b, the loosely packed spheres (top) are clearly unstable and will collapse when sheared; the dense packed spheres (bottom) can deform only if spheres in each layer rise up over the spheres in the layer below upon shearing, Wood (1990). Real sandy materials have much more irregular packing than those depicted in Figure 1.6b, but the deformation modes are essentially the same.


air
water
soil particles
Sr*e
e
1
v
v: specific volume
e: void ratio
Sr: degree of saturation
(i) Loose packing of ideal spherical soil particles
(ii) Dense packing of ideal spherical soil particles

(a) Assembles of soil particles with voids filled with mixture of air and water.

(b) Loose and dense packing of idealized soil particles.

Figure 1.6: Illustrations of porous soil structure, from Wood (1990).

The soil properties of air content and the packing density will highly influence how the seabed reacts to the aforementioned wave actions. Figure 1.7 gives an example of the different pore pressure profiles in seabeds with different air content, during the wave trough loading situation. Under the wave trough, the excess pore pressure has a negative sign, and in the case of fully saturated seabed the upward pressure gradient is not tremendously large (Figure 1.7a). However, in the case of partially saturated seabed (Figure 1.7b), the pore pressure changes drastically due to the air content and consequently the pressure gradient is very sharp. The sharp upward pressure gradient creates a significant amount of uplift. If the lift force exceeds the submerged weight of the soil, the soil will be liquefied and flow as a fluid. This type of wave-induced soil failure, due to a combined effect of the oscillatory wave trough loading and the unsaturated seabed state, is often termed the *momentary liquefaction*.

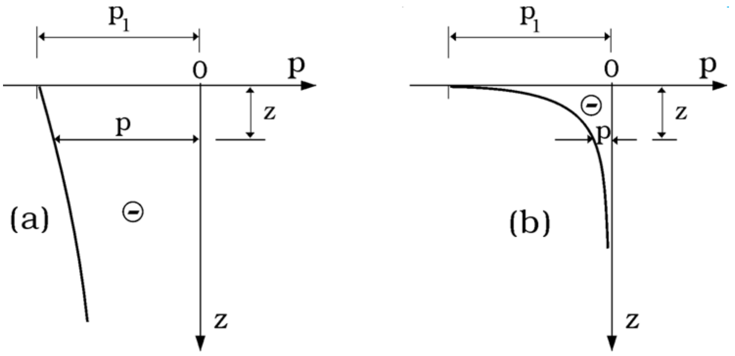


Figure 1.7: Typical distributions of excess pore pressure under wave trough. a) Fully saturated soil. b) Partially saturated soil. From Sumer and Fredsøe (2002).

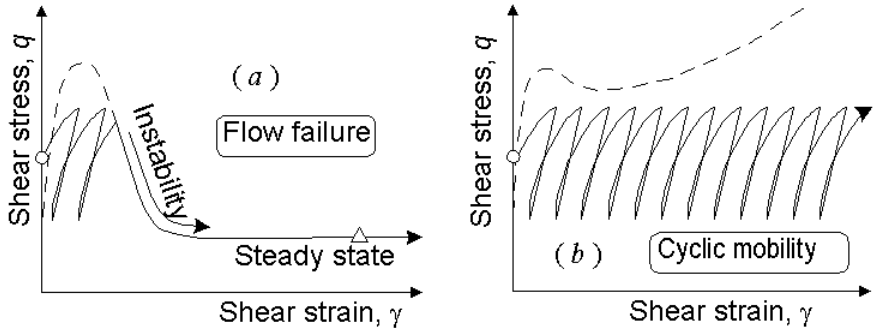


Figure 1.8: Different types of soil instabilities under cyclic shear loading. a) Residual liquefaction in loose sand. b) Cyclic mobility in dense sand. From Ishihara and Yoshimine (1998).

Figure 1.8 gives another example of the distinct soil responses in seabeds with different soil packing density, upon progressive wave shearing. In the case of a loosely packed sand, the periodic wave shear stresses gradually rearrange the soil grains at the expense of the pore volume. The latter effect "presses" the water in the pores and leads to a buildup of pore pressure. After a certain wave cycles, the pore pressure may accumulate to such levels that it exceeds the initial effective stress of the soil. In that case, the soil grains will become unbound and completely free, and the soil will behave almost like a liquid with large deformations (Figure 1.8a). This type of progressive flow failure, often observed in loose sandy seabed, is termed *residual liquefaction*.

While, in the case of a dense sand, the progressive contraction of soil skeleton under moderate shear stress level is compensated with soil dilation when approaching to plastic failure level. Therefore the buildup of pore pressure is often stabilized to a minor level compared to that in the loose sand. However, as continuous shear deformations occur, the soil may still fail due to accumulated shear deformations after large amount of wave cycles (Figure 1.8b). This type of cyclic shear failure exclusively occurred in dense seabed is named *cyclic mobility*.

Reliable models designed to predict the above mentioned seabed failures or any pre-failure states of pore pressure, soil deformations and stresses, shall require realistic descriptions of the wave loading mode as well as the seabed soil material.

1.3 Existing work

Depending on the investigation approach, existing research works studying the wave-induced seabed soil response may be categorized into three groups:

a) The analytical solution approach

b) The physical modeling approach

c) The numerical modeling approach

The analytical solution group represents those methods that directly solve the poro-elasticity governing equations and obtain exact solutions of the wave-induced pore pressure, soil displacements, and effective stresses by using analytical methods. The poro-elasticity theory was proposed initially by Biot (1941), and has been commonly used to describe porous seabed as it permits the strong coupling of pore fluid motion and soil motion. Pioneering efforts on obtaining analytical solutions of Biot's poro-elasticity equations were taken by Madsen (1978) and Yamamoto et al. (1978). They dealt with partially saturated homogenous seabed of infinite thickness under isotropic and hydraulically anisotropic conditions, respectively. Later, the analytical investigations were moved forward to cover nonhomogeneous layered seabed by Yamamoto (1981), finite soil thickness by Jeng and Seymour (1997), and mechanically anisotropic soil condition by Hsu and Jeng (1994). However, the recent development of exact analytical solutions of wave-induced seabed response seems to remain stagnant, with limited applications in offshore engineering practices. This may be partly due to the complicated mathematical forms of the solutions themselves, especially for seabeds of finite thickness; and partly due to the unrealistic simplifications of linear wave and elastic soil material adopted in the analysis. It is very difficult to tackle nonlinear wave effects and elasto-plastic seabed material, which are features commonly present in real offshore environment, by analytical methods.

Physical modeling including wave tank experiments and centrifuge wave facilities, is another approach to explore the wave-induced soil behaviors. In fact, wave tank experiments have been broadly used by coastal engineers, see Sumer et al. (1999); Teh et al. (2003); Chowdhury et al. (2006); Kudella et al. (2006); Sumer et al. (2008); Zhou et al. (2011), among many others. Teh et al. (2003) revealed the progressive instability of marine pipelines on mobile and liquefied seabed. In the momentary liquefaction tests conducted by Chowdhury et al. (2006), they confirmed the air content (also referred as the degree of saturation) as a crucial factor for the pore pressure attenuation and phase lag. Kudella et al. (2006) used large-scale experiments to study pore pressure generation underneath caisson breakwaters. In this investigation it was discovered that significant residual pore pressure can be generated by the caisson motions due to breaking wave loads. It is also worthy to note that Sumer et al. (2012) set up an experimental study directed towards the validation of an empirical mathematical model for the buildup of pore water pressure and the resulting liquefaction of marine soils under progressive waves. In general, these wave tank experiments provide direct and realistic pictures on the spatial and temporal distribution of wave-induced pore pressure and deformations within the soil. However, a potential limitation of applying those experimental findings for geotechnical design lies in the lack of accuracy in determining the various soil parameters. Alternatively, centrifuge wave facilities are a new advanced physical modeling approach, in which the experiments are carried out under N times gravitation acceleration, Sekiguchi and Phillips (1991). A promising work on applying the centrifuge technology to study the wave-induced seabed liquefaction is Sassa and Sekiguchi (1999, 2001). Nevertheless, sample preparation and a well-controlled wave generation system are still key challenges to be solved using this technology.

Numerical modeling may be considered the most flexible approach among the three approaches discussed here. Different numerical methods, like finite element method (FEM), finite difference method (FDM) and boundary element method (BEM), render approximated solutions at discrete points for the wave-induced soil response and seabed instability. A big advantage of those numerical models over the analytical approach is their capability to cover more realistic wave and seabed characteristics. For instance, in the branch of numerous FEM works, various wave conditions have been considered (e.g. non-linear waves Mostafa et al. (1999); Gao et al. (2003); Gao and Wu (2006), 3-dimensional waves Zhang et al. (2011b), and breaking waves Ulker et al. (2012)). More complex seabed features have also been studied, including plasticity by Yang and Poorooshasb (1997) and Li et al. (2002), varying shear modulus through depth by Jeng and Lin (1999), and sloping topography by Zhang and Jeng (2005). Moreover, the dynamic motion of the pore fluid has been investigated by Jeng (2003) and Ulker et al. (2010).

The FDM and BEM approaches, though less popular than FEM in the field, have also been exploited to uncover interesting wave-induced soil behaviors, such as quasi-dynamic soil motion by Wang and Oh (2013) and poroplastic failure of seabed slopes by Raman-Nair and Sabin (1991). Furthermore, a micro-mechanical modeling technique - the discrete element method (DEM) has recently been applied to study granular seabed liquefaction by Scholtes et al. (2014).

The frontier of numerical modeling on wave-induced seabed response can be drawn to the multi-physics technique. In this technique, a complete Navior-Stokes fluid model is solved so as to give a realistic picture of the dynamic wave pressure loading on the structure and the subsoil. A structure solid mechanical model may be used to estimate the resulting structural motions and stresses. A porous soil model is applied to compute the subsequent seabed responses, given the direct wave pressure loading at the wave-soil interface and the indirect wave-induced structural loadings at the soil-structure interface. Such a multi-physics model can adopt different numerical methods for the different governing equations, such as FDM or finite volume method (FVM) for the fluid (wave) and FEM for the solids (i.e. seabed and structure), Zhang et al. (2011a,b, 2012); Ye et al. (2013a) and Ye et al. (2013b); but can also involve a single numerical method like FEM for all the physical sub-domains and thus favored in terms of efficiency by avoiding data transfer between different solvers, Jeng et al. (2010).

There are many appealing features to the numerical modeling approach in the study of wave-induced seabed response, however, careful validations and verifications of the numerical results are always necessary and important. The overall efficiency of a complex multi-physics model for the wave-seabed-structure interacted system is also a challenging task to meet.

1.4 Research scope

The current research adopts numerical modeling as the main tool. First, the FVM is applied to establish an alternative numerical method for predicting the nonlinear soil material behaviors with pore fluid coupling. Thereafter, a multi-physics model of the soil-water-structure interactions is developed. This multi-physics model integrates a wave model, a solid structure model[1] and a porous soil model together using a unifying framework of FVM.

In the following subsections, the initiatives to carry out such research will be explained and an overview of the PhD project will also be given.

1.4.1 Initiatives

A key feature of the wave-induced seabed response is the coupled nonlinear soil-pore fluid behavior. As already described, repeated (cyclic) shearing of a granular sand material results in a change of volume due to particle rearrangement. When a pore fluid is present, this volume change will (via the coupling) tend to increase the pore fluid pressures and hence reduce the soil strength, especially for the case of loosely packed sand. The reproduction of such soil behavior using numerical simulations is not easy. One challenge lies in the fact that it is difficult to find constitutive relations which exactly mimics the real soil response. Another challenge is to develop a robust and efficient numerical tool and solution scheme to handle the nonlinearly coupled governing equations. Efforts devoted to developing cyclic plasticity models for realistic soil constitutive behaviors have been made by e.g. Prevost (1978); Bardet (1986) and Manzari and Dafalias (1997). Their works have achieved reasonable confidences in predicting soil stress-strain behaviors by comparison with experiments. The numerical models used to solve the overall nonlinearly coupled soil equations is however far from well-explored, and hence was the main reason for initiating this Phd study.

FVM is usually second-order accurate, based on the integral form of the governing equation, and uses a segregated solution procedure in which the coupling and the non-linearity is treated in an iterative way, Jasak and Weller (2000). FVM produces diagonally dominant matrices well suited for iterative solvers. Due to this fact it is not surprising - that FVM has been extensively

[1] The structural model will be set as optional, as a complete numerical model may not be necessary in the case of very rigid structures with small magnitude of deformations.

applied to treat complicated, coupled and nonlinear fluid flow problems in the computational fluid dynamics field. Two research questions were proposed:

a) Could the FVM be applied to solve the coupled nonlinear soil-pore fluid problems of interest?

b) Could the FVM and its segregated solution scheme be used to obtain more efficient and faster calculations of the nonlinearly coupled soil behaviors than compared to the traditional FEM strategies?

This study was initialized to address the above questions.

It is also noted that by adopting a unifying FVM multi-physics framework to the interacting wave-seabed-structure system, the usage of additional software for data transfer between different numerical methods (and software) and their maintenance is avoided; thus saving computational time and producing more efficient solvers.

1.4.2 Project overview

This PhD study has been carried out in three main steps: preliminary study, implementation and multi-physical simulation stage. The preliminary study includes the understandings of the physical problems, the numerical tool (FVM), as well as the code structure and re-use of an open source numerical library (OpenFOAM). This first step is summed up in Chapters 1 and 2, and Paper I. The implementation step involves implementing different coupled soil solvers, convergence improvements, and the corresponding solver verifications and applications. This second step is described in Chapter 3 and in Paper II and III. The multi-physical simulation step mainly contains the work on integrating the available CFD wave model, newly developed FVM soil model and existing FVM structure model together for modeling the wave-seabed-structure interaction phenomena. The achievement gained through this work stage are summarized in Chapter 4 and detailed in Paper IV and V. Figure 1.9 provides a schematic work flow of the overall contents of this thesis.

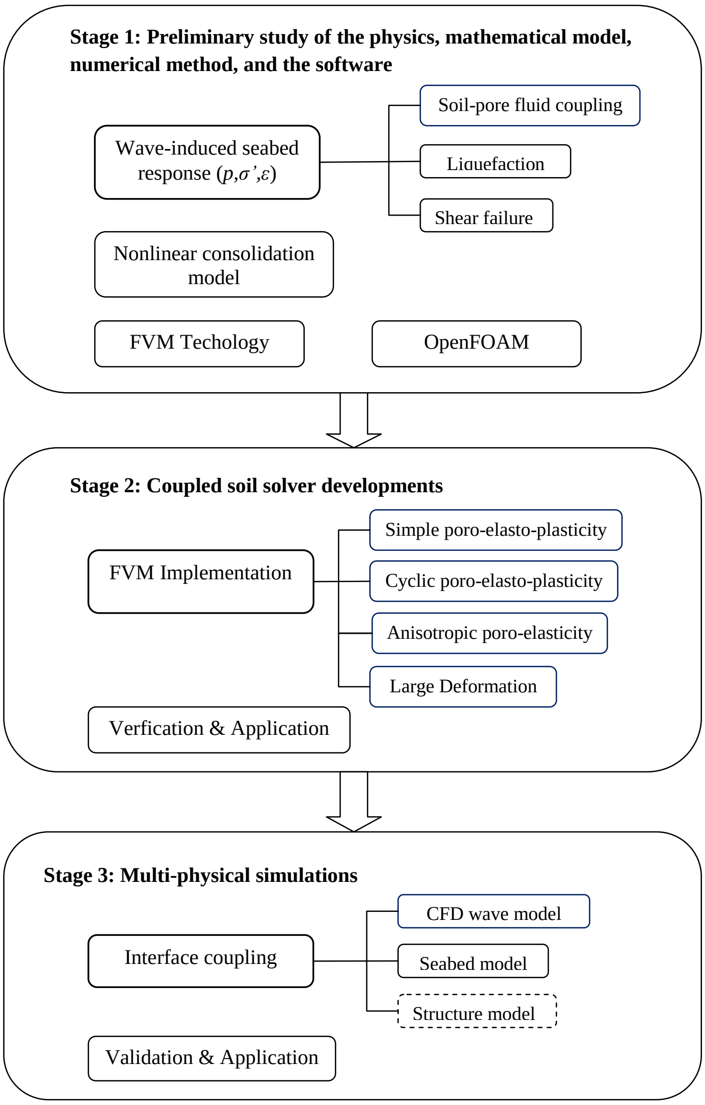

Stage 1: Preliminary study of the physics, mathematical model, numerical method, and the software
Wave-induced seabed response (p,σ',ε)
Soil-pore fluid coupling
Liquefaction
Shear failure
Nonlinear consolidation model
FVM Techology
OpenFOAM
Stage 2: Coupled soil solver developments
FVM Implementation
Simple poro-elasto-plasticity
Cyclic poro-elasto-plasticity
Anisotropic poro-elasticity
Large Deformation
Verfication & Application
Stage 3: Multi-physical simulations
Interface coupling
CFD wave model
Seabed model
Structure model
Validation & Application

Figure 1.9: The work flow of this PhD study.

Chapter 2

Methodology

This chapter provides a summary of the methodologies used in the numerical study. The descriptions are organized into three topics, which in fact correspond to the three key components of any computational model. They are: 1) the mathematical model in Section 2.1, 2) the numerical method in Section 2.2, and 3) the implementation environment (tool) in Section 2.3.

2.1 Coupled soil mathematical model

A proper mathematical model, which is simple but yet effective enough to represent the complex physical problem, can be considered as the keystone of a good computational model. For the wave-induced seabed response problem with strong soil-pore fluid coupling, it is obvious that one need to consider a 'coupled' soil mathematical model which is capable of capturing the coupling effects. The classic Biot's consolidation theory for two-phase porous media is used in this work. It involves several types of formulations, depending on the numbers of unknown variables of interests, such as the pore fluid pressure and the displacement of the soil skeleton phase and the pore fluid phase, respectively. In our study, we have adopted the so-called $\mathbf{u} - p$ formulation where the pore fluid pressure p and the soil (skeleton) displacement $\mathbf{u}$ are the primary solution variables. The pore fluid displacement (often termed as $\mathbf{U}$), is typically included in earthquake engineering problems with strong acceleration of the pore fluid, and hence not considered here.

In the following, the equations of the derived coupled soil mathematical model will be presented. For more details on the derivation process than presented here, the reader is referred to Paper I.

2.1.1 Modified Biot's consolidation equations

The original Biot's consolidation equations only capture the soil-pore fluid coupling in its linear form. Hence, the extra term "Modified" is used here to indicate the modifications including the substitution of the original elastic constitutive model with the nonlinear plastic constitutive models.

The "Modified" Biot's model contains two types of governing partial differential equations:

(i) One scalar equation representing the mass conservation.

(ii) One vector equation describing the momentum equilibrium.

The governing equation based on the mass balance, often referred to as the storage equation or seepage equation Verruijt (1969), is formulated as follows:

$$\frac{n}{K'}\frac{\partial p}{\partial t} = \nabla \cdot (\frac{\mathbf{k}}{\gamma_w} \cdot \nabla p) - \frac{\partial}{\partial t}(\nabla \cdot \mathbf{u}) \tag{2.1}$$

where, the unknown variables are:

p = the pore fluid pressure
$\mathbf{u}$ = the displacement vector of the soil skeleton

The material properties are:

$\mathbf{k}$ = generalized soil permeability tensor[1]
γ_w = density of water
n = soil porosity
K' = effective bulk modulus of the pore fluid

The mathematical operators are:

$\nabla \cdot$ = divergence operator
∇ = gradient operator

The left hand side term of Eq.(2.1) is related to the rate of volume change of the pore fluid. The first and second terms on the right hand side of Eq.(2.1) represent the rate of pore fluid seepage (via the Darcy's law) and rate of volume change of the soil skeleton, respectively. In the Biot's equations, the third phase - air bubbles - is considered to be part of the pore fluid. The influence of the presence of the air bubbles in the pore fluid is taken into account by estimating the effective bulk modulus of the pore fluid K' using the following approximation:

$$\frac{1}{K'} = \frac{Sr}{K_w} + \frac{(1 - Sr)}{p_a} \tag{2.2}$$

[1] In the case of isotropy, it will reduce to a single permeability parameter k; whereas in the case of anisotropy, it is a diagonal tensor with the directional permeability parameters k_x, k_y, and k_z as the diagonal values.

with Sr being the degree of saturation, K_w the pure water bulk modulus ($\approx 2 \times 10^9 Pa$), and p_a the absolute pore water pressure.

The governing equation for the solid matrix, often termed as the displacement equation, is based on the total momentum equilibrium of the soil mixture. By incorporating the incremental nonlinear stress-strain relation, small-strain split and strain-displacement relation below, the momentum balance is conveniently formulated as:

$$\begin{cases} \nabla \cdot (\delta\boldsymbol{\sigma}) = \nabla \cdot (\delta\boldsymbol{\sigma'} - \delta p\mathbf{I}) = \mathbf{0} \\ \delta\boldsymbol{\sigma'} = \mathbf{C} : \delta\boldsymbol{\varepsilon}^e \\ \delta\boldsymbol{\varepsilon}^e = \delta\boldsymbol{\varepsilon} - \delta\boldsymbol{\varepsilon}^p \\ \delta\boldsymbol{\varepsilon} = \frac{1}{2}\left[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}\right] \end{cases}$$

$$\Downarrow$$

$$\nabla \cdot \{\frac{1}{2}\mathbf{C} : [\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}]\} - \nabla \cdot (\mathbf{C} : \delta\boldsymbol{\varepsilon}^p) - \nabla(\delta p) = \mathbf{0} \tag{2.3}$$

in which, the primary unknown variables are:
$\delta\mathbf{u}$ = the incremental displacement vector of the soil skeleton
δp = the incremental pore fluid pressure
The stress variables are:
$\delta\boldsymbol{\sigma}, \delta\boldsymbol{\sigma'}$ = incremental total and effective stress tensor, respectively
The strain variables are:
$\delta\boldsymbol{\varepsilon}, \delta\boldsymbol{\varepsilon}^e, \delta\boldsymbol{\varepsilon}^p$ = incremental total, elastic, plastic strain tensor, respectively
The mechanical property:
$\mathbf{C}$ = generalized forth-order soil elastic stiffness tensor [2]
The mathematical operators:
: is the double dot operator
$^{\mathrm{T}}$ denotes tensor transformation operator
and $\mathbf{I}$: the identity tensor

It should be noted that in this study tension is defined as positive and compression as negative, so as to comply with the tradition in computational continuum mechanics.

There are two important features of the Biot's mathematical model adopted: They are strongly coupled by having a soil volumetric strain rate term present in the pressure equation (2.1) and a pore pressure gradient term present in the displacement equation (2.3). Also the displacement equation itself can be highly nonlinear, depending on the plastic soil material behaviors.

[2] In the case of isotropy, it reduces to two scalar constants, usually as a pair of μ and λ (the Lame's coefficients), or K and G (the bulk modulus and shear modulus), or some other similar combinations.

So far the important nonlinear variable - the incremental plastic strain $\delta\boldsymbol{\varepsilon}^p$ - which closely depends on the effective stress state $(\boldsymbol{\sigma}', \delta\boldsymbol{\sigma}')$ through a given constitutive relation, has not been defined. The soil permeability property $\mathbf{k}$ and the stiffness property $\mathbf{C}$ have also been kept as generalized higher-order tensors. The intention to leave them open in this chapter is to promote the coupled soil mathematical model as a general framework, any user-desired constitutive model to predict certain soil feature(s) can later be incorporated into this framework in a straightforward manner.

In Chapter 3, three specific soil constitutive models are selected and implemented into the coupling framework:

(a) A simple plasticity model, namely the non-associated Mohr-Coulomb model (Section 3.1), capable of capturing transient nonlinear seabed soil behaviors.

(b) A cyclic plasticity model, i.e. the critical state two-surface plasticity model (Section 3.2), effective for evaluating cyclic nonlinear seabed soil behaviors.

(c) An anisotropic elasticity model (Section 3.3) capable of predicting anisotropic effects on the seabed response.

However, the choices are not limited to the three models mentioned above. Other appropriate soil constitutive models, like those defined within the critic state soil mechanics concept describing important soil features, can also be considered.

2.2 FVM soil stress analysis

Since 1990s, FVM has been gradually established as a noteworthy alternative to FEM in the field of solid mechanics through the work of many researchers' efforts. Some important references are: Demirdzic and Martinovic (1993); Ivankovic et al. (1994); Demirdzic and Muzaferija (1994, 1995); Wheel (1996); Demirdzic et al. (1997, 2000); Jasak and Weller (2000); Maneeratana (2000); Slone et al. (2003); Taylor et al. (2003); Cardiff et al. (2012, 2014a,b); Tukovic et al. (2013). In particular, Demirdzic and Martinovic (1993) dealt with thermo-elasto-plastic solid materials and Cardiff et al. (2014b) studied orthotropic solid bodies with large strain. Their works demonstrate the capability of FVM in handling nonlinear partial differential equations of various solid mechanical stress problems.

Applying FVM to solve nonlinearly coupled soil materials in the field of geotechnics, is still rare. Of this reason the FVM approach will be described

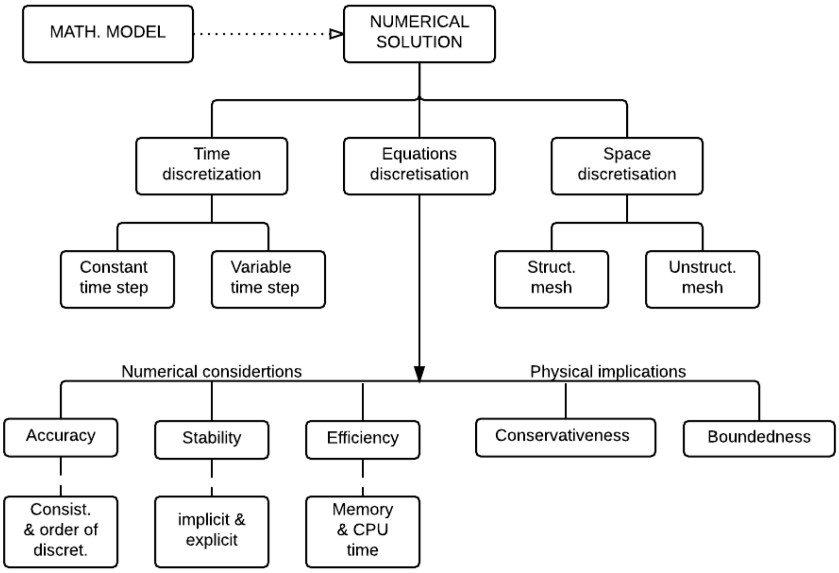


Figure 2.1: Main concepts of numerical stress analysis. Modified from Demirdzic and Ivankovic (1999).

in quite details: starting by the discretization to the solution procedure and the special FVM traction boundary condition treatment.

2.2.1 Discretization

Firstly, FVM needs to transform the governing equations (Eqs. 2.1 and 2.3 in this case) into a system of algebraic equations. In order to achieve that, the time, the space and the equations themselves have to be discretised as illustrated in Figure 2.1.

The time discretisation subdivides the given total time interval of interest into a number of smaller subintervals or time steps.

The space discretisation splits the spatial domain considered into a number of polyhedral control volumes (CV) that do not overlap and which completely fill the domain. A typical CV denoted V_P, with the computational point (node) P in its centroid, is shown in Figure 2.2. The face f and the centroid N of the neighboring CV sharing that face are also marked.

The equation discretisation uses the integral form of the governing equations over the computational spatial domain and time. Individual integral terms in the governing equations are then replaced (approximated) by algebraic expressions connecting nodal values in the spatial domain. Usually, a

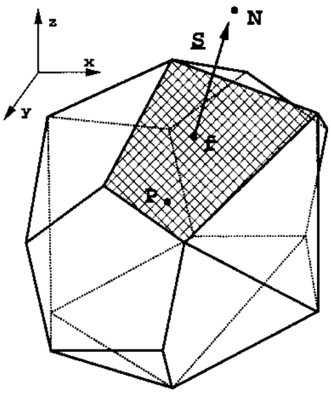


Figure 2.2: FVM polyhedral control volume. From Versteeg and Malalasekera (2007).

Table 2.1: A category of derivative terms in Eqs.(2.1) and (2.3).

Generic type	Example	ϕ	Γ
Temporal derivative $\Gamma\frac{\partial\phi}{\partial t}$	$\frac{n}{K'}\frac{\partial p}{\partial t}$	p	$\frac{n}{K'}$
Laplacian derivative $\nabla\cdot(\Gamma\nabla\phi)$	$\nabla\cdot(\frac{\mathbf{k}}{\gamma_w}\cdot\nabla p)$	p	$\frac{\mathbf{k}}{\gamma_w}$
	$\nabla\cdot[\frac{1}{2}\mathbf{C}:\nabla(\delta\mathbf{u})]$	$\delta\mathbf{u}$	$\frac{1}{2}\mathbf{C}$
	$\nabla\cdot[\frac{1}{2}\mathbf{C}:\nabla(\delta\mathbf{u})^{\mathrm{T}}]$	$\delta\mathbf{u}$	$\frac{1}{2}\mathbf{C}$
Divergence/gradient $\nabla\cdot(\Gamma\phi)$ / $\nabla(\Gamma\phi)$	$\frac{\partial}{\partial t}(\nabla\cdot\mathbf{u})$	$\mathbf{u}$	1
	$\nabla\cdot(\mathbf{C}:\delta\boldsymbol{\varepsilon}^p)$	$\delta\boldsymbol{\varepsilon}^p$	$\mathbf{C}$
	$\nabla(\delta p)$	δp	1

second-order accuracy is obtained in the approximation by assuming a linear variation of the variables over the CV. Consider the introduction of a symbol Γ representing a generic material coefficient and ϕ being an unknown variable (p,δp or $\mathbf{u}$,$\delta\mathbf{u}$,$\delta\boldsymbol{\varepsilon}^p$ in this case), in Table 2.1. The discretization of the different types of derivatives terms present in Eqs. (2.1) and (2.3) may be explained one by one:

(i) Temporal derivative:

$$\int_{V_P}\Gamma\frac{\partial\phi}{\partial t}dV \simeq \int_{V_P}\Gamma\frac{\phi^n-\phi^o}{\delta t}dV \simeq \Gamma_P\frac{\phi_P^n-\phi_P^o}{\delta t} \tag{2.4}$$

where $\phi^{n,o}$ corresponds to the 'new' and 'old' time value of the variable, respectively. The mid-point rule has been used to evaluate the volume integrals.

(ii) Laplacian derivative:

$$\int_{V_P} \nabla \cdot (\Gamma \nabla \phi) dV = \oint_{\partial V_P} d\mathbf{s} \cdot (\Gamma \nabla \phi) \simeq \sum_f \Gamma_f \mathbf{s}_f \cdot (\nabla \phi)_f \tag{2.5}$$

where ∂V_P is the closed surface bounding the volume V_P, $d\mathbf{s}$ represents an infinitesimal surface element with an associated outward pointing normal on ∂V_P, the subscript f implies the value of the variable in the center of the face, and $\mathbf{s}_f$ is the outward-pointing face area vector.

We shall recognize two options for evaluating the gradient variable at the face center:

a. The *implicit* approach. The term will be approximated assuming that the face area vector $\mathbf{s}_f$ and the distance vector $\mathbf{d}_N = \overline{PN}$ are parallel. It follows:

$$\mathbf{s}_f \cdot (\nabla \phi)_f = |\mathbf{s}_f| \frac{\phi_N - \phi_P}{|\mathbf{d}_N|} \tag{2.6}$$

The above approximation allows us to create an algebraic equation in which the Laplacian $\nabla \cdot \nabla \phi_P$ depends on the values in P and its nearest neighbors:

$$\int_{V_P} \nabla \cdot (\Gamma \nabla \phi) dV = a_P \phi_P + \sum_N a_N \phi_N \tag{2.7}$$

where

$$a_N = \Gamma_f \frac{|\mathbf{s}_f|}{|\mathbf{d}_N|}, \quad a_P = \sum_N -a_N \tag{2.8}$$

If the vectors $\mathbf{s}_f$ and $\mathbf{d}_N$ are not parallel, a “non-orthogonal correction” is necessary. However, since this study only employ regular orthogonal meshes, the details of non-orthogonality treatment are not covered here, see instead Jasak (1996) for this matter.

b. The *explicit* approach. The term will be evaluated using the linearly interpolated gradients:

$$\mathbf{s}_f \cdot (\nabla \phi)_f = \mathbf{s}_f \cdot [f_x (\nabla \phi)_P + (1 - f_x)(\nabla \phi)_N] \tag{2.9}$$

in which f_x is the interpolation factor, and it is defined as the ratio of distances $\overline{fN}$ and $\overline{PN}$, that is:

$$f_x = \frac{\overline{fN}}{\overline{PN}} \tag{2.10}$$

Different from the implicit approach, the face center gradient term is now evaluated from the current cell center values of $\nabla\phi$ (i.e. from the available distribution of ϕ). The gradients at the cell center can be calculated using the least-square method.

(iii) Divergence or gradient derivative.

$$\int_{V_P} \nabla \cdot (\Gamma\phi) dV = \oint_{\partial V_P} d\mathbf{s} \cdot (\Gamma\phi) = \sum_f \Gamma_f \mathbf{s}_f \cdot \phi_f, \quad \text{for } \phi = \mathbf{u} \tag{2.11}$$

$$\int_{V_P} \nabla(\Gamma\phi) dV = \oint_{\partial V_P} d\mathbf{s}(\Gamma\phi) = \sum_f \Gamma_f \mathbf{s}_f \phi_f, \quad \text{for } \phi = p \tag{2.12}$$

The face value is calculated assuming linear variation of ϕ between P and N, as:

$$\phi_f = f_x \phi_P + (1 - f_x)\phi_N \tag{2.13}$$

Similarly, an *implicit* or *explicit* approach can be chosen for the above approximation. The implicit approach again allows us to form an algebraic equation in which the divergence $\nabla \cdot \phi_P$ or the gradient $\nabla\phi_P$ depends on the values in P and its nearest neighbors, e.g.:

$$\int_{V_P} \nabla \cdot (\Gamma\phi) dV = a_P \phi_P + \sum_N a_N \phi_N \tag{2.14}$$

with

$$a_N = \Gamma_f \mathbf{s}_f (1 - f_x), \quad a_P = \sum_N (\Gamma_f \mathbf{s}_f - a_N) \tag{2.15}$$

whereas, the explicit approach directly evaluates the terms using the known available values of ϕ.

The actual choice of implicit or explicit discretization approach for each spatial derivative term will be described in the following subsection.

2.2.2 Segregated solution procedure and iterative method

In most CFD codes, a *segregated* solution approach is used to deal with the nonlinearity and pressure-velocity coupling in the flow equations, Versteeg and Malalasekera (2007). Similarly, the following implicit-explicit discretization split is used to facilitate the segregated solution of the soil governing equations:

$$\underbrace{\frac{n}{K'} \frac{\partial p}{\partial t} - \nabla \cdot (\frac{\mathbf{k}}{\gamma_w} \cdot \nabla p)}_{implicit} = \underbrace{- \frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{explicit} \tag{2.16}$$

$$\begin{aligned}\underbrace{\nabla\cdot[\mathbf{K}\cdot\nabla(\delta\mathbf{u})]}_{\textit{implicit}} &= \underbrace{-\left(\nabla\cdot\{\frac{1}{2}\mathbf{C}:[\nabla(\delta\mathbf{u})+\nabla(\delta\mathbf{u})^{\mathrm{T}}]\}-\nabla\cdot[\mathbf{K}\cdot\nabla(\delta\mathbf{u})]\right)}_{\textit{explicit}} \\ &\quad +\underbrace{\nabla\cdot(\mathbf{C}:\delta\boldsymbol{\varepsilon}^p)}_{\textit{explicit}} \\ &\quad +\underbrace{\nabla(\delta p)}_{\textit{explicit}} \qquad (2.17)\end{aligned}$$

The terms $\nabla\cdot(\mathbf{C}:\delta\boldsymbol{\varepsilon}^p)$, $\frac{\partial}{\partial t}(\nabla\cdot\mathbf{u})$ and $\nabla(\delta p)$, are treated explicitly, as they contain nonlinearity and coupling depending on the unknown solutions. The original Laplacian term $\nabla\cdot\{\frac{1}{2}\mathbf{C}:[\nabla(\delta\mathbf{u})+\nabla(\delta\mathbf{u})^{\mathrm{T}}]\}$ in the displacement equation is rearranged into an implicit part $\nabla\cdot[\mathbf{K}\cdot\nabla(\delta\mathbf{u})]$ and an explicit part $\left(\nabla\cdot\{\frac{1}{2}\mathbf{C}:[\nabla(\delta\mathbf{u})+\nabla(\delta\mathbf{u})^{\mathrm{T}}]\}-\nabla\cdot[\mathbf{K}\cdot\nabla(\delta\mathbf{u})]\right)$. The $\mathbf{K}$ is a second-order diagonal stiffness tensor with diagonal elements K_{ii} which fulfill the stress-strain relation $\sigma_{ii}=K_{ii}\varepsilon_{ii}$. In this way, the implicit term $\nabla\cdot[\mathbf{K}\nabla(\delta\mathbf{u})]$ contains the maximum implicit contribution from the component-wise discretization, while the remaining terms in the explicit part include all the off-diagonal coupling. Such a rearrangement has proved to have considerably enhanced convergence in the work of Jasak and Weller (2000) (linearly elasticity) as well as Cardiff et al. (2014b) (large strain with orthotropic material properties).

As a consequence of the choice of the above discretization split, four linear algebraic equations for each CV are produced:

$$a_P^p p_P + \sum_N a_N^p p_N^p = r_P^p \qquad (2.18)$$

$$a_P^{\mathbf{u}}(\delta\mathbf{u})_P + \sum_N a_N^{\mathbf{u}}(\delta\mathbf{u})_N^p = \mathbf{r}_P^{\mathbf{u}} \qquad (2.19)$$

where the source term r_P^p in the pressure equation contains the explicit displacement coupling, and the source term $\mathbf{r}_P^{\mathbf{u}}$ in the displacement equations contains all the explicit nonlinearity and the explicit pore pressure coupling. Those source terms also include the contributions from the temporal term and/or the boundary conditions as usual.

Assembling Eqs. (2.18) and (2.19) for all the CVs results in four systems of algebraic equations:

$$[A^p][p] = [r^p] \qquad (2.20)$$

$$[A^{\mathbf{u}}]\,[\delta\mathbf{u}] = [\mathbf{r}^{\mathbf{u}}] \qquad (2.21)$$

where the sparse matrices $[A^p]$ and $[A^{\mathbf{u}}]$ have coefficients a_P^p and $a_P^{\mathbf{u}}$ in the diagonal and a_N^p and $a_N^{\mathbf{u}}$ off the diagonal. $[p]$ and $[\delta\mathbf{u}]$ are the vector of $p's$ and $\delta\mathbf{u}'s$ for all CVs, respectively. $[r^p]$ and $[\mathbf{r}^{\mathbf{u}}]$ are the right-hand side vectors.

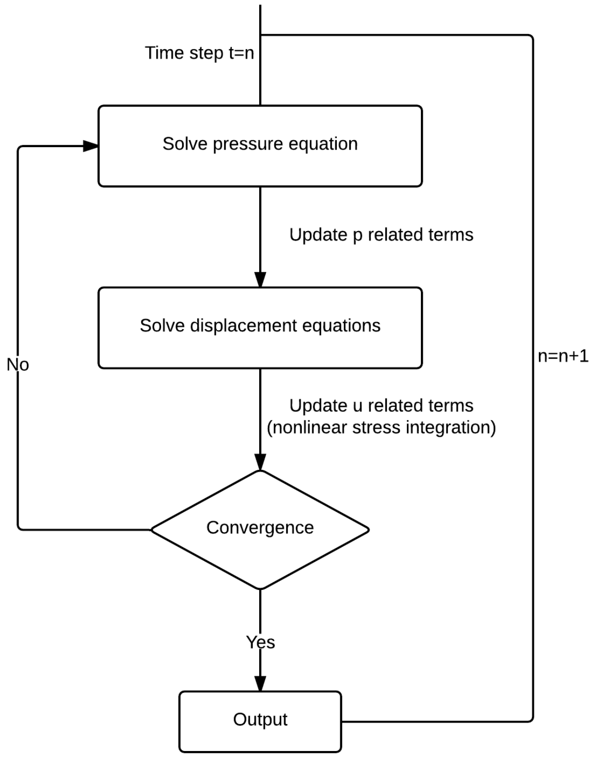


Figure 2.3: The segregated solution procedure with fixed point iteration

The above systems will be solved sequentially for the pore pressure and the three components of $\delta\mathbf{u}$. Due to the fact that the matrices $[A]$ are symmetric and diagonally dominant, the system of equations can be solved using efficient iterative solvers, such as the Generalized geometric-algebraic multigrid (GAMG) solver.

In order to achieve full convergence of the nonlinearly coupled equations system, the above segregated procedure must be combined with a certain iterative method, i.e., the fixed point iteration method. In other words, the algebraic equation systems (Eqs. 2.20 and 2.21) are solved iteratively, with explicit nonlinear and coupling contributions in the source terms being updated according to the latest iterative values. Only when the solution changes less than some pre-defined tolerance, which implies that the explicit terms essentially becomes implicit, the iteration will be stopped. As we are dealing with transient calculations, the iteration will be performed for each time step using the available solution from the previous time step as the initial guess. Figure 2.3 illustrates such iterative process schematically.

It is interesting to note that the above fixed point iterative scheme has some nice properties. One is that it is usually not necessary to converge the solutions of Eqs. (2.20) and (2.21) to a very tight tolerance, as the new solutions will only be used to update the explicit terms. Another point is that within the local nonlinear stress integration process, namely the calculation of $\delta\boldsymbol{\varepsilon}^p$ based on the iterative value of $\delta\mathbf{u}$, one can select cheap explicit or return-mapping algorithms with loose tolerance, as again the term $\delta\boldsymbol{\varepsilon}^p$ will only be used as an approximated explicit term in the global solution procedure.

2.2.3 Boundary condition treatment

There are two basic types of boundary conditions relevant to our study: the Dirichlet boundary condition, which prescribes the value of the pore pressure and/or the soil displacement on the boundary; and the Neumann boundary condition, which on the other hand prescribes the pore water flow and/or the external force on the boundary. For description of the treatment of the Dirichlet and Neumann conditions in FVM in general can be found in, e.g., (Jasak, 1996, Chapter 3), and will therefore not be repeated here. However, the traction force boundary which is important to the stress analysis problems in solids and soils deserves some extra clarifications.

The traction boundary condition involves the prescription of the traction force on the boundary. Unlike in the case of FEM, where external traction is directly accounted due to separating the boundary contribution using the weak form together with the Green-Gauss theorem, the traction in FVM is treated in a more indirect manner and requires some extra computational effort to transform the traction into the gradient of displacement normal to the boundary (i.e., the Neumann type).

Based on the Cauchy's stress theorem, one have:

$$\delta\mathbf{t} = \mathbf{n}_b \cdot \delta\boldsymbol{\sigma} \tag{2.22}$$

where $\delta\mathbf{t}$ is the (prescribed) incremental traction, and $\mathbf{n}_b$ is the face area vector normal to the boundary.

Recall the formulations in Eq. (2.3), the stress tensor can be expressed as:

$$\delta\boldsymbol{\sigma} = \frac{1}{2}\mathbf{C} : [\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}] - \mathbf{C} : \delta\boldsymbol{\varepsilon}^p - \delta p\mathbf{I} \tag{2.23}$$

Substituting Eq. (2.23) into (2.22), one obtain:

$$\delta\mathbf{t} = \mathbf{n}_b \cdot \left\{\frac{1}{2}\mathbf{C} : [\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}] - \mathbf{C} : \delta\boldsymbol{\varepsilon}^p - \delta p\mathbf{I}\right\} \tag{2.24}$$

It is shown in Eq. (2.24) that, to compute the displacement gradient $\nabla(\delta\mathbf{u})$ normal to the boundary, we have to deal with the same nonlinearity and coupling issues as those present in the inner solution domain. Again, the segregated solution approach with fixed point iterations can be adopted to yield an effective resolution for the above case. The the following split on the boundary is used:

$$\begin{aligned}\underbrace{\mathbf{n}_b \cdot [\mathbf{K} \cdot \nabla(\delta \mathbf{u})]}_{implicit} = \delta \mathbf{t} \quad &- \quad \underbrace{\mathbf{n}_b \cdot \left\{ \frac{1}{2}\mathbf{C} : [\nabla(\delta \mathbf{u}) + \nabla(\delta \mathbf{u})^{\mathrm{T}}] - \mathbf{K} \cdot \nabla(\delta \mathbf{u}) \right\}}_{explicit} \\ &+ \quad \underbrace{\mathbf{n}_b \cdot (\mathbf{C} : \delta \boldsymbol{\varepsilon}^p) + \mathbf{n}_b \delta p}_{explicit} \end{aligned} \tag{2.25}$$

In the case of isotropy, $\mathbf{K}$ reduces to a single constant and therefore $\mathbf{n}_b \cdot \nabla(\delta \mathbf{u})$ is obtained directly. In the case of anisotropy, some mathematical manipulations are needed to calculate $\mathbf{n}_b \cdot \nabla(\delta \mathbf{u})$ from the term $\mathbf{n}_b \cdot [\mathbf{K} \cdot \nabla(\delta \mathbf{u})]$. A possible way is documented in Chapter 4 of Appendix. The Neumann type boundary discussed here will be iteratively updated until both the convergence of the inner solution domain and the convergence of the boundaries have been obtained.

The treatment of the traction boundary together with the computational cost required using FVM may have some minor drawbacks as compared to the clean and simple traction boundary implemented in FEM. However, a consistent iterative scheme is used in the traction boundary as in the inner solution domain of the FVM scheme, and hence no effort on inventing a new scheme was necessary. Furthermore, the computational cost related to the traction boundary is in fact marginal, as the size of the boundary patches is usually small compared to that of the inner solution domain. The overall efficiency of the segregated FVM procedure, in handling the coupled soil model, may still be better than that of the implicit block matrix FEM approach, which will be demonstrated in Chapter 3.

2.3 OpenFOAM

After defining a mathematical framework and the numerical scheme, a proper and convenient platform for code implementation needs also to be sought. Here the OpenFOAM (Open source Field Operation And Manipulation), which is a C++ class library for the development of customized numerical codes in continuum mechanics, Weller et al. (1998), will be used. There are several attractive features of OpenFOAM:

- It contains various tensor fields and tensorial derivatives required in the FVM technique. The code syntax can be written very similar to the partial differential equations one wish to solve (e.g., Eqs. 2.16 and 2.17).

- It supports automatic computational parallelization.
- It includes free-surface wave solvers and structure solvers ready to use. Multi-physic simulations of soil-wave-structure interaction can be carried out straightforwardly, by integrating the existing OpenFOAM solvers with the soil solver developed in this work.

In the following, the soil solver implementation within OpenFOAM will be briefly described. Readers are suggested to refer to the Part III Appendix for more details. Also the OpenFOAM's potential in dealing with multi-physical problems will be discussed.

2.3.1 Soil implementation

The implementation of the governing soil equations in OpenFOAM is done in a straightforward manner, as the necessary tensor classes, their associated tensor algebra, and the FVM discretization techniques are already there.

The "geometric tensor field" classes, like `volScalarField`, `volVectorField`, and `volSymmTensorField` are used to define the different working variables, such as the pressure, displacement and stress in the domain of interest. A customized forth-order symmetric tensor field `volSymmTensor4thOrderField` and its associated double dot operator `&&`, implemented by Cardiff et al. (2014b), are also adopted for the generalized higher-order stiffness tensor **C** in the case of anisotropic material behavior.

The two types of tensor-derivative classes implemented in OpenFOAM: `fvm(finiteVolumeMethod)` and `fvc(finiteVolumeCalculus)`, directly correspond to the aforementioned *implicit* and *explicit* discretization approaches. All required tensor derivative operations: $\partial/\partial t$, $\nabla\cdot\nabla$, $\nabla\cdot$, and ∇, are present in the two classes as member functions and can simply be used as, e.g., `fvm::ddt`, `fvm::laplacian`, and `fvc::div`.

As an example, the Eq. (2.16) for solving the pore pressure is simply represented by the code:

```
solve
(
    fvm::ddt(p)
   -fvm::laplacian(Dp1, p)
  ==
   -fvc::div(fvc::ddt(Dp2, p))
);
```

The Eq. (2.17) for obtaining the incremental displacement of soil skeleton is written as:

```
volTensorField gradDU = fvc::grad(dU);
solve
(
    fvm::laplacian(K, dU)
  ==
   -(fvc::div(C && dEpsilon) - fvc::div(K & gradDU))
   +fvc::div(C && dEpsilonP)
   +fvc::grad(dP)
);
```

The above syntax closely resembles the form of the previously described differential equations.

The other important part of the code, which relates to the time loop, the fixed point iteration (global loop) and the nonlinear stress integration process (local loop), are reported in details in Paper II and III as well as in the Appendix, and will therefore not be repeated here.

2.3.2 Multi-physics handling in OpenFOAM

OpenFOAM is a general computational continuum mechanics library, in which various fluid and solid solvers have been developed and released. It provides a natural platform for multi-physical simulations of interest in this work. Flexible wave generation tools, e.g., `waves2foam` by Jacobsen et al. (2012) and `IHFOAM` by Higuera et al. (2014a,b), can generate and actively absorb free surface water waves of many types in a 3D domain. Furthermore, the computational solid mechanics group at University College Dublin contributed to a `solidMechanics` solver library that can solve stresses and displacements in various solid bodies. The wave and solid structure modeling capabilities of OpenFOAM facilitate wave-soil-structure interaction simulations: the solvers for the different physical domains (fluid/soil/solid) are used together, and the coupling is achieved through various boundary condition updates.

Doing a simulation in a single software like OpenFOAM simplifies the operation, as there is no need for multi-threaded simulations of software to software coupling. Moreover, various mesh-to-mesh mapping tools that are already implemented in OpenFOAM, further, simplify the boundary coupling between the different domains. Some results regarding the integrated multi-physic simulation are presented in Chapter 4.

Chapter 3

FVM coupled soil solvers

This chapter presents examples of the FVM coupled soil solvers developed through this study applying the aforementioned methodology. Each example is introduced in terms of the specific constitutive relation and thereafter supported by relevant test cases. A discussion of the possibility to extend the methodology to cope with other types of soil constitutive behaviors is also given.

3.1 Simple poro-elasto-plasticity model

A simple poro-elasto-plasticity model, in which the constitutive relation employed is a linear elastic/plastic model based on a classical non-associated Mohr-Coulomb formulation, is first solved using FVM. This type of constitutive model is ideally suitable for granular soil materials with dilative behavior under monotonic loading. It is therefore expected that the simple poro-elasto-plasticity model can effectively capture the plastic deformations of the soil frame coupled with the pore flow response. Furthermore, after validation, the FVM model should be capable to predict wave-induced seabed soil plastic failures as well as pore pressure variations within the transient (oscillatory) loading scenario.

Here it is necessary to note that, due to the use of isotropic linear elasticity in the constitutive relation, the forth-order elastic stiffness tensor **C** and the second-order diagonal "stiffness" tensor **K** can be expressed as:

$$\mathbf{C} = 2\mu\mathbf{l} + \lambda\mathbf{I} \otimes \mathbf{I}, \quad \mathbf{K} = (2\mu + \lambda)\mathbf{I} \tag{3.1}$$

where μ and λ are the Lamé's constants, $\mathbf{l}$ is the symmetric part of the forth-order identity tensor, and $\mathbf{I}$ is the second-rank identity tensor and $\otimes$ is the dyad product.

As a consequence of the above formulations, the soil model described by Eqs. (2.16) and (2.17) can now be written in a simpler form without involving those double dot tensor contractions:

$$\frac{n}{K'}\frac{\partial p}{\partial t} = \frac{k}{\gamma}\nabla^2 p - \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{\text{explicit}} \tag{3.2}$$

$$\begin{aligned}\nabla \cdot [(2\mu + \lambda)\nabla(\delta\mathbf{u})] = &- \underbrace{\nabla \cdot \left\{\mu\nabla(\delta\mathbf{u})^{\mathrm{T}} + \lambda\mathbf{I}\mathrm{tr}[\nabla(\delta\mathbf{u})] - (\mu+\lambda)\nabla(\delta\mathbf{u})\right\}}_{\text{explicit}} \\ &+ \underbrace{\nabla \cdot [2\mu(\delta\boldsymbol{\varepsilon}^p) + \lambda\mathbf{I}\mathrm{tr}\,(\delta\boldsymbol{\varepsilon}^p)]}_{\text{explicit}} + \underbrace{\nabla(\delta p)}_{\text{explicit}}\end{aligned} \tag{3.3}$$

in this specific poro-elasto-plasticity model, a constant permeability scalar k has also been used assuming hydraulic isotropy.

The definition of plastic strain term $\delta\boldsymbol{\varepsilon}^p$ is prescribed by the non-associated Mohr Coulomb formulation as described below.

3.1.1 Non-associated Mohr Coulomb model

We adopted a non-associated Mohr Coulomb model with perfect plasticity formulated in the principal stress space, which in turn conveniently allowed us to employ an efficient explicit return mapping algorithm originally proposed by Clausen et al. (2007) for the stress integration procedure.

In summary, the ingredients of this constitutive model are the following:

a. The yield surface, which specifies the occurrence of plastic deformation.

$$f = (\sigma'_1 - \sigma'_3) + (\sigma'_1 + \sigma'_3)\sin\varphi - 2c\cos\varphi \tag{3.4}$$

where f is the yield surface, σ'_1 and σ'_3 are the maximum and minimum principal effective soil stresses, respectively. They are equivalent to the largest and smallest eigenvalues of the general effective stress tensor $\boldsymbol{\sigma}'$. The soil properties φ and c are the friction angle and cohesion.

b. The plastic potential, which prescribes the mode of the non-associated plastic deformation (dilative/contractive).

$$g = (\sigma'_1 - \sigma'_3) + (\sigma'_1 + \sigma'_3)\sin\psi \tag{3.5}$$

where g is the plastic potential function, and ψ is the soil dilation angle.

c. The hardening rule, which defines the magnitudes of the plastic deformations.

$$\delta \boldsymbol{\varepsilon}^p = \langle \delta\Lambda \rangle \frac{\partial g}{\partial \boldsymbol{\sigma}'}, \quad \langle \delta\Lambda \rangle = \begin{cases} 0, & \text{when pre-yielding } f < 0. \\ \delta\Lambda, & \text{when yielding } f = 0. \end{cases} \tag{3.6}$$

where return mapping is used for the evaluation of the plastic multiplier $\delta\Lambda$, which can be evaluated by:

$$\delta\Lambda = \frac{f(\boldsymbol{\sigma}'_{tl})}{H + (\partial f/\partial \boldsymbol{\sigma}') : \mathbf{C} : (\partial g/\partial \boldsymbol{\sigma}')} \tag{3.7}$$

In the above equation $f(\boldsymbol{\sigma}'_{tl})$ is the yield function evaluated at the trial elastic stress, and $\mathbf{C}$ is the elastic stiffness tensor completely defined by μ and λ using Eq. (3.1). Moreover, as we are considering a non-hardening model the plastic modulus $H = 0$.

The stress integration procedure, which is schematically summarized in Table 3.1, is a local computation at each CV after the global iterative solution of the poro-elasto-plasticity equations.

3.1.2 Test cases

Three test cases have been studied in order to assess the performance of the implemented simple poro-elasto-plasticity solver, including: (i) an initial verification of the elastic soil/pore fluid coupling with the classic 1-D Terzaghi's consolidation problem, (ii) a further verification of the elaso-plastic soil/pore fluid coupling with a strip footing bearing capacity problem, and (iii) an example of the ability to predict the uplift capacity of a circular suction caisson foundation. These cases are all described and discussed in Paper II. Here the findings from the second test case are highlighted.

The strip footing case setup is as follows: a smooth, perfectly flexible, uniformly loaded, permeable strip footing acts on a layer of soil resting on a smooth rigid base. In order to completely define the problem, it is assumed that there is no horizontal force on any vertical section; and plain strain condition has been considered. This test case was originally examined by Small et al. (1976), Figure 3.1, to investigate elasto-plastic consolidation using FEM. We hence re-study the test using the developed FVM solver in OpenFOAM, Figure 3.2, for comparison. Due to the lack of information on computational costs in Small et al.'s work, a comparable simulation in a commercial FEM software Abaqus was also performed. In this way, a clear picture of the computational efficiency of the two methods in handling the nonlinearly coupled soil problem could be estimated.

Table 3.1: Local return mapping stress integration

```
INPUT: $\delta\mathbf{u}$, displacement increments
       $\boldsymbol{\sigma}'^{A}$, initial/old-time stress

1. Compute the elastic trial stress $\boldsymbol{\sigma}'^{B}$ by:
   $\boldsymbol{\sigma}'^{B} = \boldsymbol{\sigma}'^{A} + \{\mu\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})]^{\mathrm{T}} + \lambda\mathbf{I}\mathrm{tr}[(\delta\mathbf{u})]\}$
2. Transform $\boldsymbol{\sigma}^{B}$ into principal space as $\boldsymbol{\sigma}'^{B}_{prin}$ (eigenvalues).
   Store the principal directions (eigenvectors).
3. Evaluate the yield function $f(\boldsymbol{\sigma}'^{B}_{prin})$:
   if $f < 0$, EXIT, $\boldsymbol{\sigma}'^{C} = \boldsymbol{\sigma}^{B}$, $d\boldsymbol{\varepsilon}^{p} = 0$
   if $f \geq 0$, CONTINUE
4. Determine the right stress return type.
   Obtain the principal plastic corrector stress $\boldsymbol{\sigma}'^{C}_{prin}$.
5. Reuse the preserved principal directions.
   Transform $\boldsymbol{\sigma}'^{C}_{prin}$ back to the general space as $\boldsymbol{\sigma}'^{C}$
6. Compute the correct elastic strain increment $\delta\boldsymbol{\varepsilon}^{e}$ by:
   $\delta\boldsymbol{\varepsilon}^{e} = \dfrac{\mathrm{iso}(\boldsymbol{\sigma}'^{C} - \boldsymbol{\sigma}'^{A})}{3\lambda + 2\mu} + \dfrac{\mathrm{dev}(\boldsymbol{\sigma}'^{C} - \boldsymbol{\sigma}'^{A})}{2\mu}$
7. Calculate the plastic strain increment $\delta\boldsymbol{\varepsilon}^{p}$ by:
   $\delta\boldsymbol{\varepsilon}^{p} = \dfrac{1}{2}[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}] - \delta\boldsymbol{\varepsilon}^{e}$
8. Go back to step 3.

OUTPUT: $\boldsymbol{\sigma}'^{C}$, $\delta\boldsymbol{\varepsilon}^{p}$
```

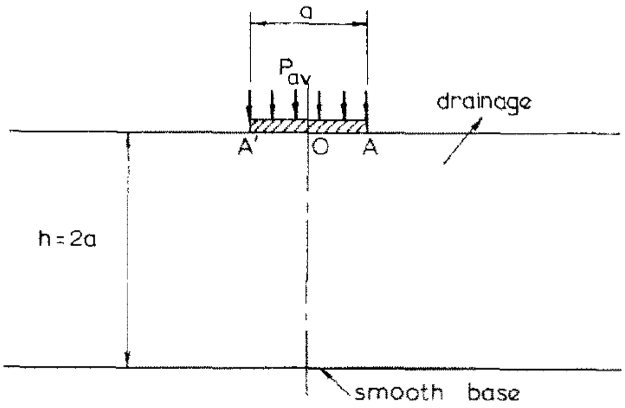


Figure 3.1: Illustration of the strip footing case geometry. Adopted from Small et al. (1976).

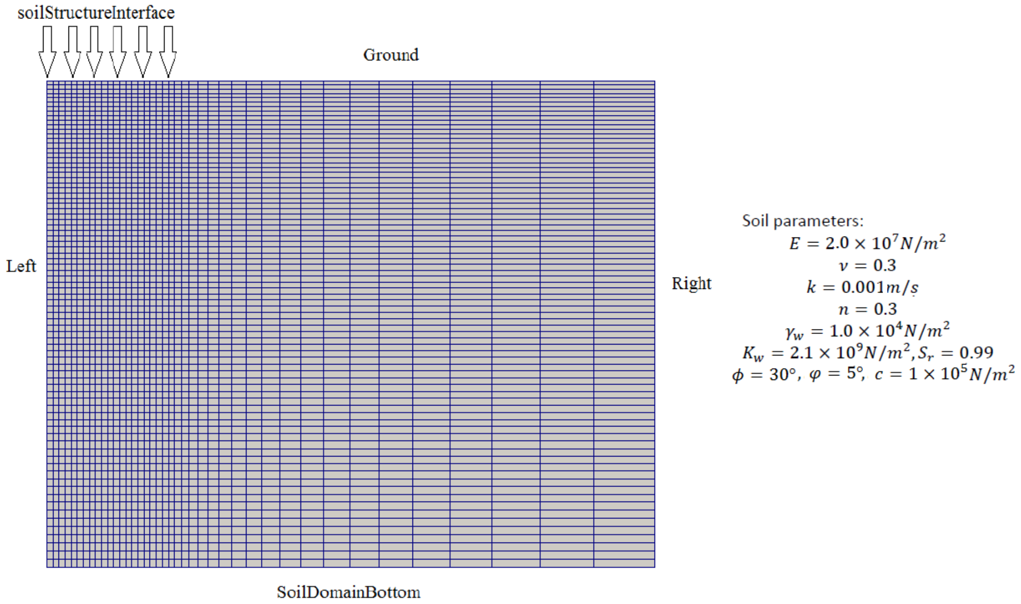

soilStructureInterface
Ground
Left
Right
SoilDomainBottom
Soil parameters:
$E = 2.0 \times 10^7 N/m^2$
$\nu = 0.3$
$k = 0.001 m/s$
$n = 0.3$
$\gamma_w = 1.0 \times 10^4 N/m^2$
$K_w = 2.1 \times 10^9 N/m^2, S_r = 0.99$
$\phi = 30°,\ \varphi = 5°,\ c = 1 \times 10^5 N/m^2$

Figure 3.2: (Half domain) mesh setup in OpenFOAM with soil properties.

In particular, a load rate parameter ω is used. The parameter ω is originally defined by Small et al. (1976) to indicate the different loading conditions with various load rates:

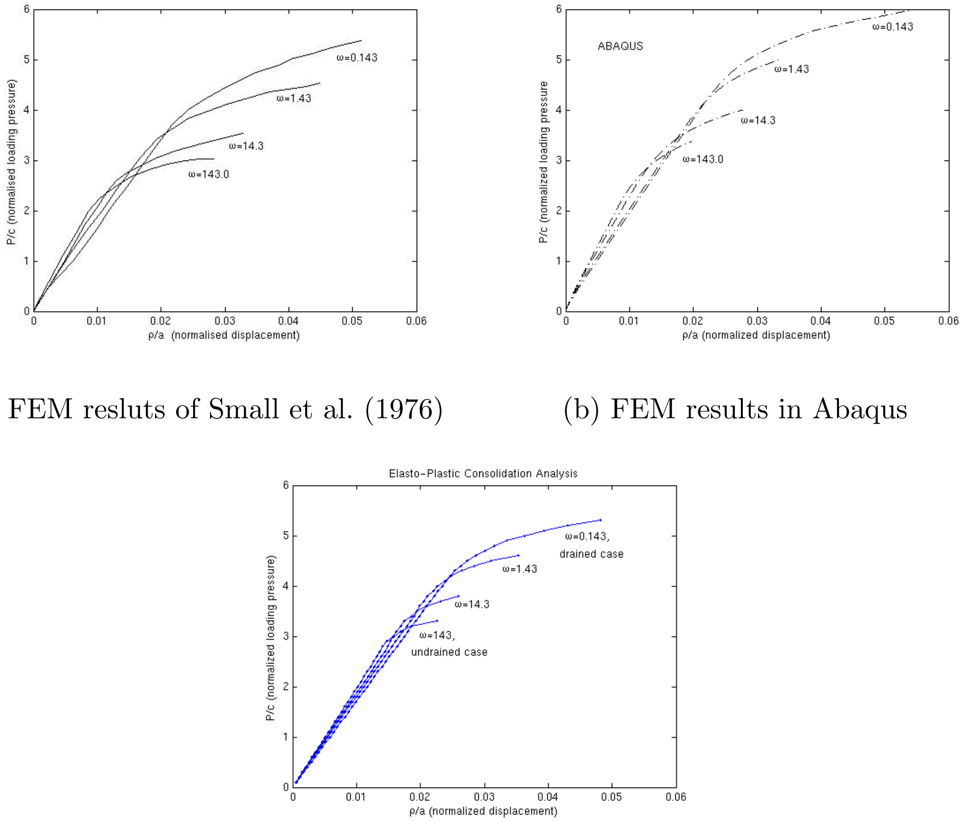
$$\omega = \frac{d(P/c)}{d(T_v)}, \qquad T_v = \frac{c_v t}{a^2} \tag{3.8}$$

where P/c represents the external load pressure (normalized by soil cohesion), and T_v is the dimensionless time, dependent on c_v, the one-dimensional consolidation coefficient, and a, the width of the strip footing.

The validity of the simulation results has been successfully verified in two aspects:

1). A well-reproduced load rate effect on the soil bearing capacity.

A commonly observed feature of a saturated soil material is that: the faster the load, the weaker the soil response. This is because of the influence of the generated excess pore pressure on the soil effective stresses and in turn the soil strength. Figure 3.3 represents cases where a very fast load rate $\omega = 143$ is applied for a fully undrained condition, a very slow load rate $\omega = 0.143$ for a completely drained condition, and the two intermediate rates of $\omega = 14.3$ and $\omega = 1.43$ for a fast load rate and a slow load rate in partially drained conditions, respectively. Our FVM predictions are: the lowest level of footing bearing capacity under undrained condition (full impact from the high amount of excess pore pressure); and thereafter a gradual increase of the capacity along with the decreasing load speed (gradual dissipation of the excess pore pressure). These results agree well with the FEM solutions obtained by Small et al. (1976) and by calculations using Abaqus.



(a) FEM resluts of Small et al. (1976)

(b) FEM results in Abaqus

(c) FVM results in OpenFOAM

Figure 3.3: Bearing capacities of the footing under different loading rates.

2). A good representation of the soil plastic dilation being coupled to the pore pressure variation.

The results from the slow load rate test, $\omega = 1.43$, are used as an example. A qualitative proof of the captured strong coupling between the soil dilation and the pore pressure response is illustrated in Figure 3.4. The subfigures on the left panel depict the excess pore pressure distributions at various load levels; and the subfigures on the right panel present the development of the plastic zone in the soil correspondingly. In the case of the initial low load levels, the whole soil domain behaves elastically (Figures 3.4b and 3.4d), and the excess pore pressure simply accumulates because of the increasing load (Figures 3.4a and 3.4c). Whereas, when load pressure increases to higher levels, the soil reaches plastic failure (Figures 3.4f and 3.4h) and starts to dilate, and in turn the excess pore pressure in the soil starts to dissipate at a fast rate (Figures 3.4e and 3.4g).

An illustration of the above phenomena is also given in Figure 3.5, where the excess pore pressure distributions along the footing center line are sampled. The lines in the Figure 3.5 correspond to the different load levels (steps). It is very clear that the excess pore pressure starts to dissipate as long as plastic deformation occurs and results in dilation.

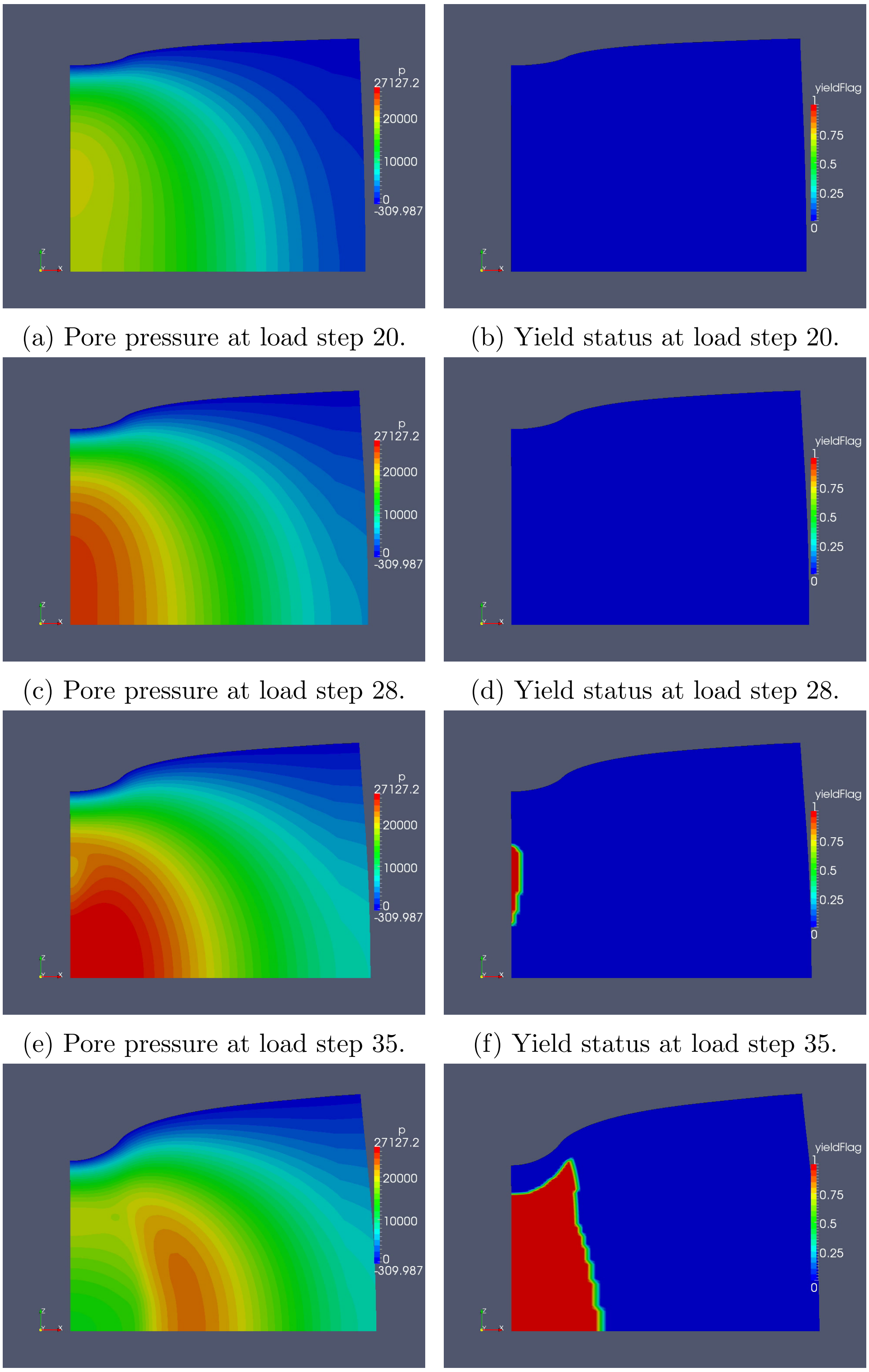
The efficiency of the simulations in terms of computational costs is compared with with those of Abaqus. The OpenFOAM simulation was performed under parallel computation using 8 cores on an IBM HPC cluster (2.66 GHz Intel Xeon E5430 cores). The Abaqus simulations were each run on 6 cores in parallel on an SGI ICE X cluster (2.4 GHz Intel Xeon E5-2695 v2 Ivy Bridge cores). The overall computational costs of FVM and FEM analysis are listed in Table 3.2 and 3.3, respectively.

Table 3.2: FVM OpenFOAM computation cost of the strip footing case

Case	No. CV	No. time steps/ plasticity occurs at	Max. No. outer iter. per plastic time step	Total wall time (hr)
Drained	3200	53/38	3.4×10^4	00:17:23
Slow load rate	3200	46/34	1.0×10^5	00:42:43
Fast load rate	3200	38/27	1.2×10^4	00:18:58
Undrained	3200	33/25	1.0×10^5	00:33:04

Table 3.3: FEM Abaqus computation cost of the strip footing case

Case	No. ele./ No. nod. per ele.	No. time steps/ plasticity occurs at	Max. No. equil iter. per plastic time step	Total wall time (hr)
Drained	4800/8	$1.150 \times 10^4/396$	16	14:50:00
Slow load rate	4800/8	$6.142 \times 10^3/358$	16	08:07:00
Fast load rate	4800/8	$9.558 \times 10^3/263$	16	13:21:00
Undrained	4800/8	$1.752 \times 10^4/257$	16	06:50:00



(a) Pore pressure at load step 20.

(b) Yield status at load step 20.

(c) Pore pressure at load step 28.

(d) Yield status at load step 28.

(e) Pore pressure at load step 35.

(f) Yield status at load step 35.

(g) Pore pressure at load step 43.

(h) Yield status at load step 43.

Figure 3.4: Nonlinear soil-pore fluid coupling, loading rate $\omega = 1.43$.

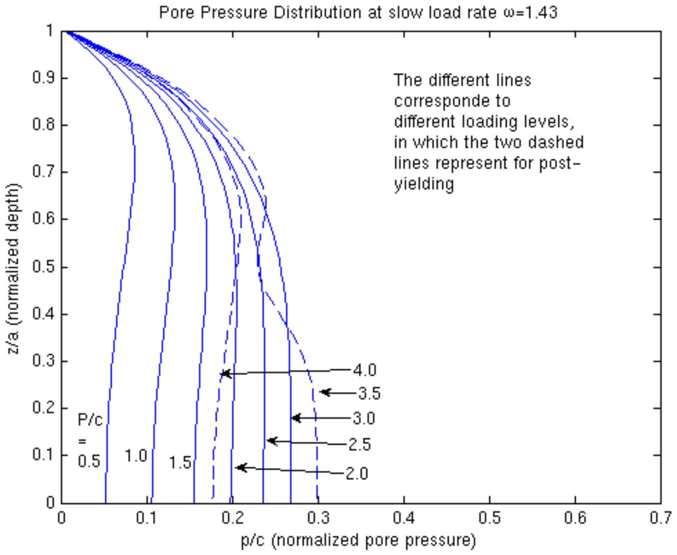


Figure 3.5: Pore pressure distributions along the footing center line using a slow load rate, $\omega = 1.43$.

Before discussing the computational efficiency of the two numerical methods, it is necessary to point out their different solution strategies in handling the nonlinearity and coupling. The proposed segregated FVM simulations use constant time steps with large amounts of fixed-point iterations, while the implicit FEM predictions in Abaqus employ automatic (small) time stepping approaches with Newton-type iterations. The two methods encounter the same challenge - convergence rate decreases during plastic steps and requires special algorithm handling. Therefore under-relaxation methods is applied in OpenFOAM and automatic stabilization with fixed damping factor in Abaqus to help for convergence.

It is not surprising that the current Abaqus FEM coupled analysis turned out to be quite slow, which is due to the limitation that it only assumes a fully saturated condition (i.e. incompressible pore fluid) using the standard FE coupled formulation. The fully-saturated assumption yields singularities in the stiffness matrix which significantly slows down the convergence. The solution of the large implicit matrix system in Abaqus also requires much more RAM and longer computational time. Contrary to that, the FVM poro-elasto-plastic solver is formulated on the basis of an improved approximation of the soil saturation condition. The effective bulk modulus of pore fluid (comparable to that of soil skeleton) enhances the convergence of the coupled algorithm. The staggered solution procedure used in FVM also enables efficient iterative solver to be used for solving the small matrix system.

3.2 Cyclic poro-elasto-plasticity model

A simple poro-elasto-plasticity model was proven to work in a satisfactory manner in the previous section. In this section, the inclusion of a more advanced cyclic plasticity constitutive relation into the coupled framework is studied. The cyclic poro-elasto-plasticity model aims at capturing the gradual buildup of pore pressure and soil strength degradation upon cyclic loading. In this way, a proper tool to investigate the wave-induced seabed soil responses within the progressive(periodic) loading scenario can be obtained.

The specific cyclic constitutive soil model employed is a critical state two-surface plasticity model originally proposed by Manzari and Dafalias (1997). They used isotropic hypoelasticity and distinguished the volumetric and deviatoric strains in the formulation, thus the governing momentum equation (2.17) is now instead written in the form:

$$\begin{aligned}\nabla\cdot[(K+\tfrac{4}{3}G)\nabla(\delta\mathbf{u})] = &-\underbrace{\nabla\cdot[G\nabla(\delta\mathbf{u})^{\mathrm{T}}+(K-\tfrac{2}{3}G)\mathbf{I}\nabla\cdot(\delta\mathbf{u})-(K+\tfrac{1}{3}G)\nabla(\delta\mathbf{u})]}_{explicit}\\ &+\underbrace{\nabla\cdot[K\,(\delta\varepsilon_v^p)+2G\,(\delta\boldsymbol{\varepsilon}_d^p)]}_{explicit}+\underbrace{\nabla(\delta p)}_{explicit} \qquad (3.9)\end{aligned}$$

where the soil properties K and G are the elastic bulk and shear modulus, respectively. They define the full elastic stiffness tensors: $\mathbf{C} = 2G\mathbf{l} + (K - 2/3G)\mathbf{I}\otimes\mathbf{I}$ and $\mathbf{K} = (K + 4/3G)\mathbf{I}$. The plastic strain terms $\delta\varepsilon_v^p$ and $\delta\boldsymbol{\varepsilon}_d^p$ are the volumetric and deviatoric parts, respectively. The determination of $\delta\varepsilon_v^p$ and $\delta\boldsymbol{\varepsilon}_d^p$ are given in the constitutive formulations introduced below.

Note that the governing pore pressure equation is kept the same as before, that is, Eq. (3.2), indicating a hydraulic isotropy condition.

3.2.1 Critical state two-surface cyclic plasticity model

The adopted critical state two-surface cyclic plasticity model can effectively reproduce cyclic stress-strain behaviors considering several important soil features, such as phase transformation, critical state and peak failure, etc. It involves more complicated formulations and a considerable amount of extra model parameters, compared to the previously discussed Mohr Coulomb model. However, that is always the rule: the more soil features to predict, the more complex soil constitutive model one tend to need.

The essential ingredients of this constitutive model is summarized as follows:

a hypo-elastic properties are,

$$\delta\boldsymbol{\varepsilon}_d^e = \frac{\delta\mathbf{s}}{2G} \quad \text{and} \quad \delta\varepsilon_v^e = \frac{\delta p'}{K} \tag{3.10}$$

with

$$G = G_0\left(\frac{p'}{p_{at}}\right)^a \text{ and } K = K_0\left(\frac{p'}{p_{at}}\right)^a \tag{3.11}$$

where $\delta\boldsymbol{\varepsilon}_d^e$ and $\delta\varepsilon_v^e$ are the deviatoric and volumetric part of the elastic strain increment, respectively, $\delta\mathbf{s}$ is the deviatoric effective stress increment tensor, and $p'(\delta p')$ is the mean effective stress (increment). The atmospheric pressure p_{at} is used as a reference pressure, for which $G = G_0$ and $K = K_0$. The property a is a model constant usually set to $a = 0.5$.

b yield surface adopted is,

$$f = \sqrt{(\mathbf{s} - p'\boldsymbol{\alpha}) : (\mathbf{s} - p'\boldsymbol{\alpha})} - \sqrt{2/3}mp' \tag{3.12}$$

where m and $\boldsymbol{\alpha}$ denote the internal-state variables which determine the 'size' and the position of the axis of the yield surface (cone-type), respectively.

c hardening rule is described as,

$$\delta\boldsymbol{\varepsilon}_d^p = \mathbf{n}\delta\Lambda \quad \text{and} \quad \delta\boldsymbol{\varepsilon}_v^p = D\delta\Lambda \tag{3.13}$$

with

$$\delta\Lambda = \frac{2G\mathbf{n} : \delta\boldsymbol{\varepsilon}_d - KN\delta\boldsymbol{\varepsilon}_v}{K_p + 2G - KDN}, \quad K_p = p'(\mathbf{n} : \tilde{\boldsymbol{\alpha}} + \sqrt{2/3}\tilde{m}) \tag{3.14}$$

$$D = A_0(1 + \langle\mathbf{F} : \mathbf{n}\rangle)\mathbf{d} : \mathbf{n} \tag{3.15}$$

$$\delta\boldsymbol{\alpha} = \tilde{\boldsymbol{\alpha}}\delta\Lambda, \quad \tilde{\boldsymbol{\alpha}} = h_0\frac{|\mathbf{b} : \mathbf{n}|}{b_{ref} - |\mathbf{b} : \mathbf{n}|}\mathbf{b} \tag{3.16}$$

$$\delta m = \tilde{m}\delta\Lambda, \quad \tilde{m} = c_m(1 + e_0)D \tag{3.17}$$

where $\mathbf{n}$ is the unit deviatoric stress-ratio tensor and D is the dilatancy coefficient. Detailed explanations on the volumetric part N of the normal to the yield surface, the plastic modulus K_p, the 'distance' vectors $\mathbf{b}$ and $\mathbf{d}$, the fabric tensor $\mathbf{F}$, and the model parameters h_0, b_{ref}, c_m and A_0 can be found in Manzari and Dafalias (1997).

Given the above constitutive formulations, an efficient return mapping algorithm, adopted from Manzari and Prachathananukit (2001) and Bakmar et al. (2008), to achieve the local stress integration process at each CV level is used. Figure 3.6 summarizes the essential steps of this algorithm.

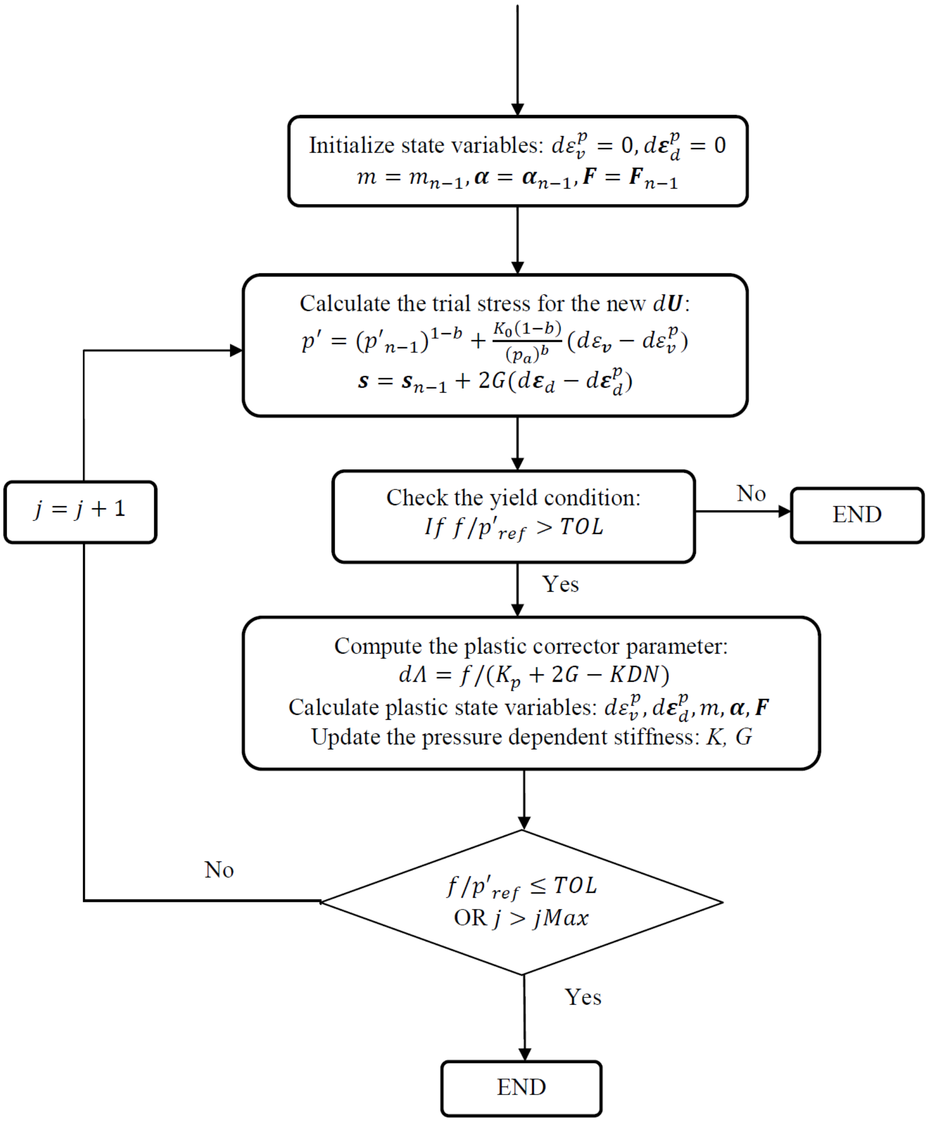

Initialize state variables: $d\varepsilon_v^p = 0, d\boldsymbol{\varepsilon}_d^p = 0$
$m = m_{n-1}, \boldsymbol{\alpha} = \boldsymbol{\alpha}_{n-1}, \boldsymbol{F} = \boldsymbol{F}_{n-1}$
Calculate the trial stress for the new $d\boldsymbol{U}$:
$p' = (p'_{n-1})^{1-b} + \frac{K_0(1-b)}{(p_a)^b}(d\varepsilon_v - d\varepsilon_v^p)$
$\boldsymbol{s} = \boldsymbol{s}_{n-1} + 2G(d\boldsymbol{\varepsilon}_d - d\boldsymbol{\varepsilon}_d^p)$
$j = j + 1$
Check the yield condition:
$If\ f/p'_{ref} > TOL$
No
END
Yes
Compute the plastic corrector parameter:
$d\Lambda = f/(K_p + 2G - KDN)$
Calculate plastic state variables: $d\varepsilon_v^p, d\boldsymbol{\varepsilon}_d^p, m, \boldsymbol{\alpha}, \boldsymbol{F}$
Update the pressure dependent stiffness: K, G
No
$f/p'_{ref} \leq TOL$
OR $j > jMax$
Yes
END

Figure 3.6: Local stress-strain update procedure at each CV level.

3.2.2 Test cases

Previously, the capability of the segregated FVM algorithm to capture the nonlinear coupling effects through the verification of the simple poro-elasto-plasticity model was demonstrated. Here, for the cyclic poro-elasto-plasticity model, the focus is on validating the implemented complex constitutive relation and its ability to facilitate pore pressure buildup when strong coupling is present.

Two types of numerical tests are studied, they are: (i) simulations of small-scale monotonic and cyclic undrained triaxial tests on full 3D saturated soil samples, and (ii) a hypothetical test case with wave pressure loading on a large-scale porous seabed. These tests have also been documented in Paper III. Here we briefly present the results from these cyclic triaxial tests and the wave pressure loading case. It is worth mentioning, that the same soil properties have been used in all the simulations, as summed up in Table 3.4, for simplicity.

Table 3.4: Soil properties used in the simulations of this section, with the plasticity model parameters adopted from Manzari and Dafalias (1997).

Elasticity		Pore fluid			
G_0(MPa)	31.4	k(m/s)	0.0001	p_a(kPa)	100
K_0(MPa)	31.4	K_w(GPa)	2.1	γ_w(kPa)	10
a	0.6	S_r	$0.90 - 1.00$		
Critical state		State parameter			
M_c	1.2	k_c^b	3.975		
M_e	0.857	k_e^b	2.0		
λ	0.025	k_c^d	4.2		
$(e_c)_{\text{ref}}$	1.2	k_e^d	0.07		
Dilatancy		Hardening			
A_0	0.6	h_0	800		
C_f	100	m	0.05		
$F_{\max}$	100	c_m	0.0		

i). Cyclic undrained triaxial tests.

Two kinds of saturated soil samples are considered for the tests, with initial void ratio of 0.65 (dense) and 0.85 (loose). To mimic the laboratory test procedure, they are firstly isotropically consolidated to a mean confining stress of $p_0 = 80$kPa and then loaded by a cyclic shear stress with the amplitude of $30 - 40$kPa and the frequency of 1Hz. As can be seen from Figure 3.7, the well-known butterfly-shape stress path is predicted for the dense soil that undergoes 'cyclic mobility' failure with accumulated large shear strains (left); likewise, the complete loss of mean effective stress leading to liquefaction failure is captured in the loose sample (right). These simulations conform with experimental evidences, e.g., Hyodo et al. (1991) and Flora et al. (2012).

ii). Standing wave pressure loading on the seabed.

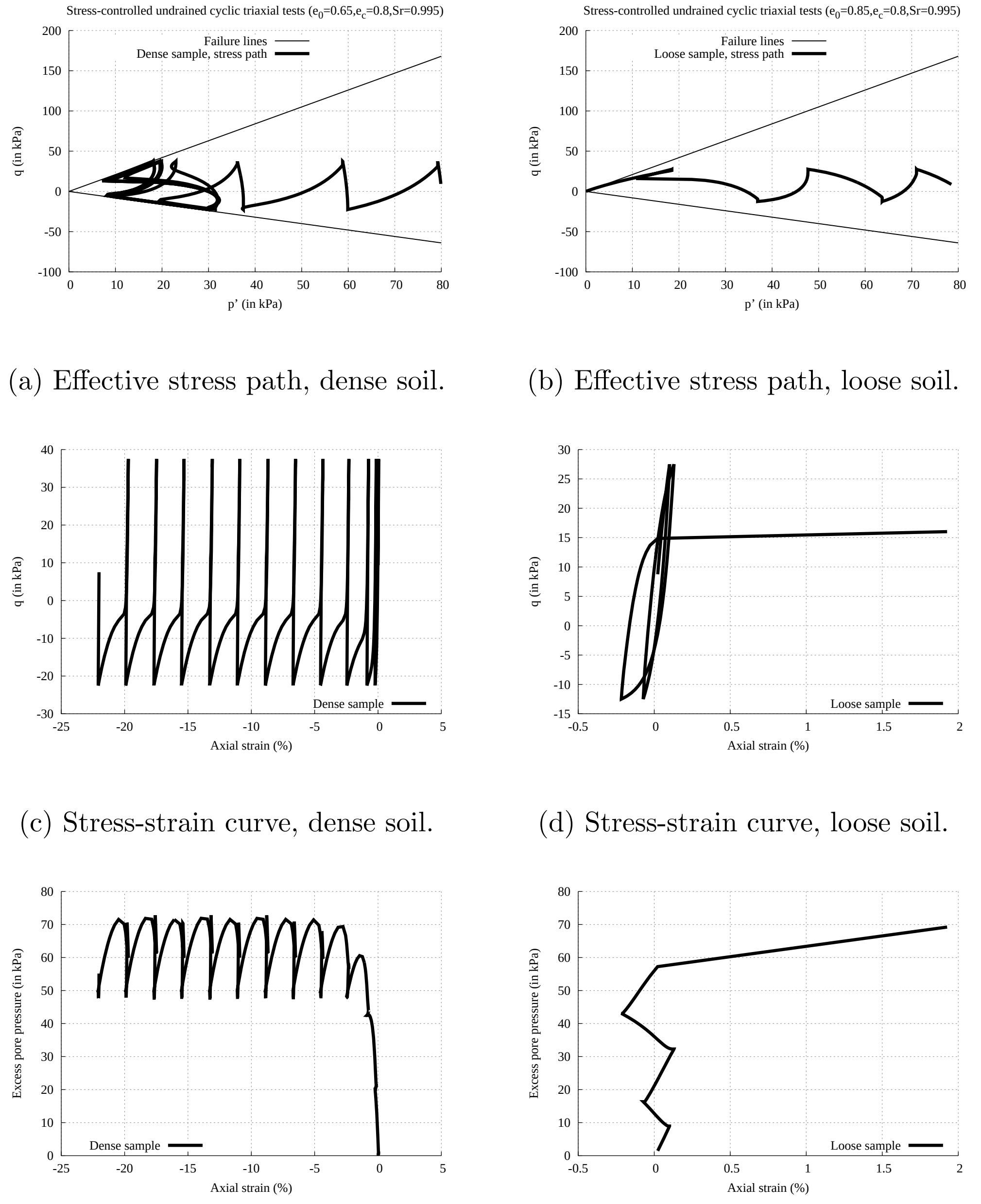
In this hypothetical case, a standing wave pressure (from linear wave theory) is acting on a porous seafloor with finite depth, and an impermeable object is present in the center of the seabed surface resulting in a partially drained condition, as illustrated in Figure 3.8. The test case explores the model's capability of predicting accumulated excess pore pressure inside the seabed given the applied cyclic wave loading condition.

Figure 3.9 plots the simulated excess pore pressure variations, cycle by cycle, at a certain soil depth ($1/10d$ beneath the object center). Results from the classic Biot's theory, i.e. the poro-elastic model, are also presented for comparison. The current cyclic poro-elasto-plastic model predicts a gradual accumulation of excess pore pressure inside the soil corresponding to a gradual decrease of pore volume; whereas, the simple poro-elastic model, which has constant pore volume, only captures the steady state pore pressure variation. The former represents the more realistic seabed response as described before: shear strains induced by the standing wave pressure gradually rearrange the soil grains at the expense of the pore volume of the soil, the latter pressurizes the pore water, and as a result lead to the built-up of pore pressure.

3.3 Anisotropic poro-elasticity model

Most soils display some degree of material anisotropy: they possess different mechanic and hydraulic properties in different directions, due to the deposition mode, particle shape, and loading history. Besides from the inherent anisotropy, soils may also develop plastic stress-induced anisotropy while being loaded. Ideally, soil models that incorporate such anisotropic features could give better approximations of the soil response, compared to conventional isotropic solutions. Hence it is of interest to explore the possibility of including soil anisotropy into the coupled framework.

At this stage, the considerations are limited to a simple anisotropic poro-elastic model, namely considering an orthotropic linear elastic soil skeleton and an anisotropic pore permeability. It is believed that if successful, the



(a) Effective stress path, dense soil.

(b) Effective stress path, loose soil.

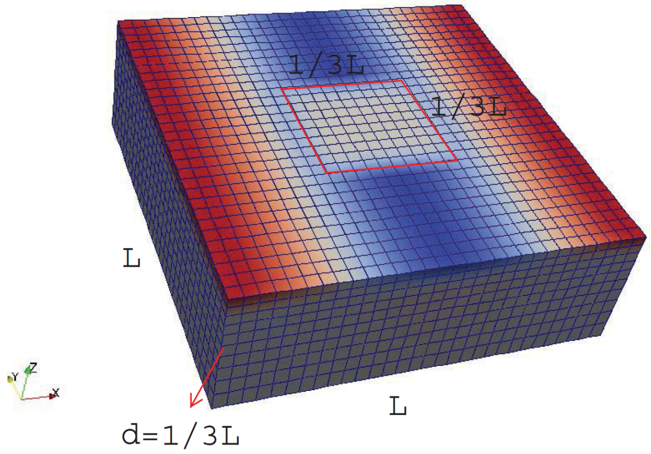
(c) Stress-strain curve, dense soil.

(d) Stress-strain curve, loose soil.

(e) Pore pressure variation, dense soil.

(f) Pore pressure variation, loose soil.

Figure 3.7: Simulation results of stress-controlled cyclic undrained triaxial tests on a dense soil sample (left) and a loose soil sample (right).



Patch	Boundary type
Seabed sides, bottom and the object	impermeable & zero displacement
Seabed top	standing wave pressure: amplitude= 5kPa, period=0.1s & free traction

Figure 3.8: Mesh setup of the porous seabed case with dimensions and boundaries.

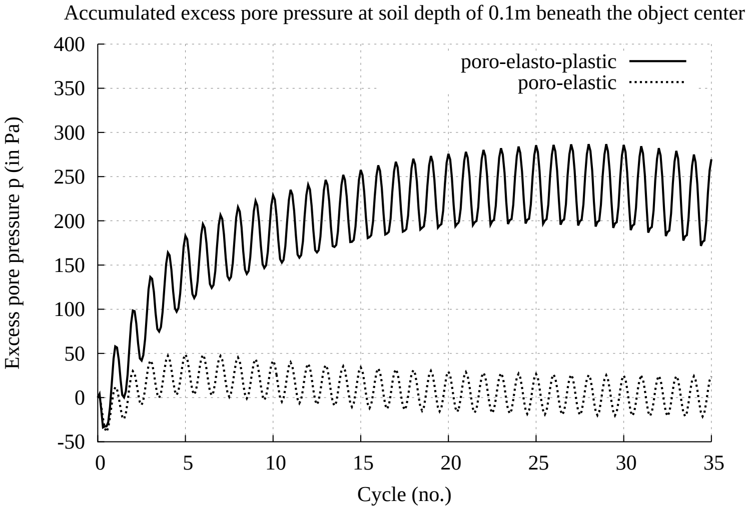


Figure 3.9: Simulated pore pressure accumulations at the soil depth of $1/10d$ underneath the object center.

door to further extension of the method to the variety of anisotropic poro-elasto-plastic models is open.

In the case studied here the original form of the generalized coupled equations is used by omitting the nonlinear plastic terms, that is:

$$\frac{n}{K'}\frac{\partial p}{\partial t} = \frac{1}{\gamma_w}\nabla \cdot (\mathbf{k} \cdot \nabla p) - \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{\text{explicit}} \tag{3.18}$$

$$\begin{aligned}\nabla \cdot [\mathbf{K} \cdot \nabla(\delta\mathbf{u})] = &- \underbrace{\nabla \cdot \left\{\mathbf{C} : \frac{1}{2}\left[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}\right]\right\} + \nabla \cdot [\mathbf{K} \cdot \nabla(\delta\mathbf{u})]}_{\text{explicit}} \\ &+ \underbrace{\nabla(\delta p)}_{\text{explicit}}\end{aligned} \tag{3.19}$$

The determination of the elastic stiffness tensors, **C** and **K**, and the permeability tensor **k** will be described in the next subsection.

3.3.1 Orthotropic linear elasticity model

We adopted an orthotropic linear elastic constitutive relation, in which the mechanical properties only differ along each coordinate axis, thus requiring nine independent elastic constants. Consider the following generalized Hook's Law written in matrix format, Demirdzic et al. (2000):

$$\delta\boldsymbol{\sigma} = \begin{pmatrix} \delta\sigma_{xx} \\ \delta\sigma_{yy} \\ \delta\sigma_{zz} \\ \delta\sigma_{xy} \\ \delta\sigma_{yz} \\ \delta\sigma_{zx} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{31} & 0 & 0 & 0 \\ A_{12} & A_{22} & A_{23} & 0 & 0 & 0 \\ A_{31} & A_{23} & A_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{66} \end{pmatrix} \begin{pmatrix} \delta\varepsilon_{xx} \\ \delta\varepsilon_{yy} \\ \delta\varepsilon_{zz} \\ \delta\varepsilon_{xy} \\ \delta\varepsilon_{yz} \\ \delta\varepsilon_{zx} \end{pmatrix} = \mathbf{C} : \delta\boldsymbol{\varepsilon} \tag{3.20}$$

where the stiffness coefficients A_{ij} are given in terms of the Young's moduli E_i, the Poisson's ratio ν_{ij} and the shear moduli G_{ij}, as:

$$\begin{aligned} &A_{11} = \frac{1 - \nu_{yz}\nu_{zy}}{JE_yE_z}, \quad A_{22} = \frac{1 - \nu_{xz}\nu_{zx}}{JE_xE_z}, \quad A_{33} = \frac{1 - \nu_{yx}\nu_{xy}}{JE_yE_x}, \\ &A_{12} = \frac{\nu_{xy} + \nu_{zy}\nu_{xz}}{JE_xE_z}, \quad A_{23} = \frac{\nu_{yz} + \nu_{yx}\nu_{xz}}{JE_xE_y}, \quad A_{31} = \frac{\nu_{zx} + \nu_{yx}\nu_{zy}}{JE_yE_z}, \\ &A_{44} = 2G_{xy}, \quad A_{55} = 2G_{yz}, \quad A_{66} = 2G_{zx} \end{aligned} \tag{3.21}$$

where

$$J = \frac{1 - \nu_{xy}\nu_{yx} - \nu_{yz}\nu_{zy} - \nu_{xz}\nu_{zx} - 2\nu_{yx}\nu_{zy}\nu_{xz}}{E_x E_y E_z} \tag{3.22}$$

$$\nu_{yx} = \nu_{xy}\frac{E_y}{E_x}, \quad \nu_{xz} = \nu_{zx}\frac{E_x}{E_z}, \quad \nu_{zy} = \nu_{yz}\frac{E_z}{E_y} \tag{3.23}$$

The "constructed" diagonal stiffness tensor **K** can therefore be expressed as:

$$\mathbf{K} = \begin{pmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{pmatrix} \tag{3.24}$$

Moreover, the anisotropic permeability tensor **k** is defined by:

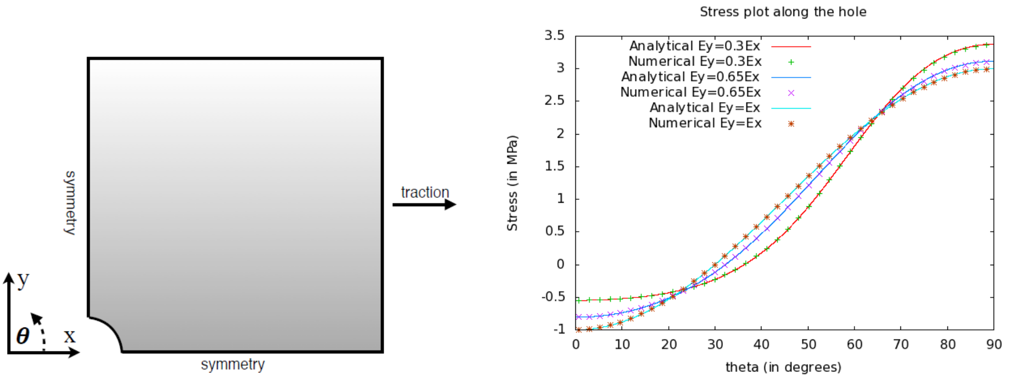
$$\mathbf{k} = \begin{pmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{pmatrix} \tag{3.25}$$

where k_x, k_y and k_z are the permeability constants measured at the corresponding axial directions.

3.3.2 Test cases

Three test cases are performed to evaluate the performance of the developed anisotropic FVM code. They are: (i) a circular hole in an orthotropic plate under tension used to verify the implemented mechanic anisotropy, (ii) a short-crested wave pressure loading on a porous seabed used to verify the implemented hydraulic anisotropy, and (iii) an example analyzing the wave-induced seabed pore pressure and stresses considering the combined mechanic and hydraulic anisotropy. The overall case setup and results for the first two test cases are illustrated in Figure 3.10 and 3.11, see Paper V for more details.

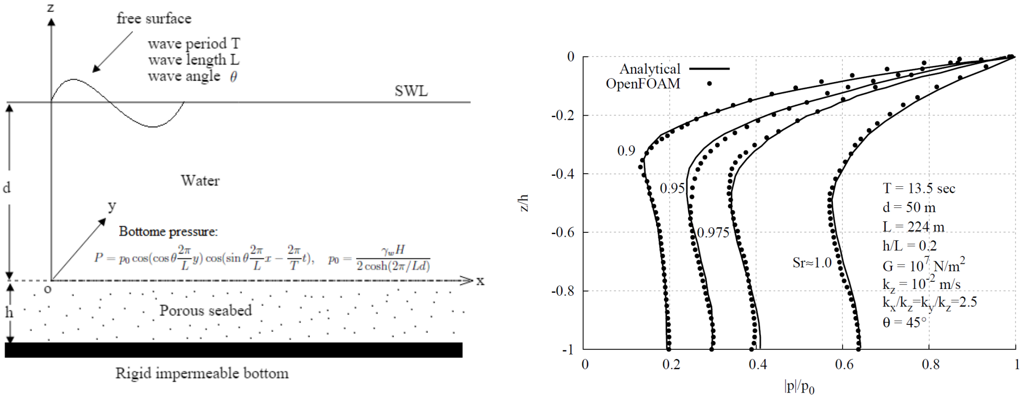
The third example studies the wave-induced seabed behavior with combined mechanic and hydraulic anisotropy in the soil, for which no exact analytical solutions exist in publications. A simple 2D seabed loaded by a standing wave pressure is considered. In parallel, an isotropic simulation was conducted for comparison. The simulation results shown in Figure 3.12 demonstrate that the standing wave-induced seabed pore pressure and the effective normal stress are symmetric with respect to the antinodes, while the shear stress is symmetric with respect to the nodes. Furthermore, it is observed that the conventional isotropic solution tends to overestimate the pore pressure while underestimating the vertical effective normal stress. These findings indeed comply with the conclusions drawn from a similar study by Jeng (1997).



(a) Orthotropic plateHole test case

(b) Hoop stress $\sigma_{\theta\theta}$ around the hole

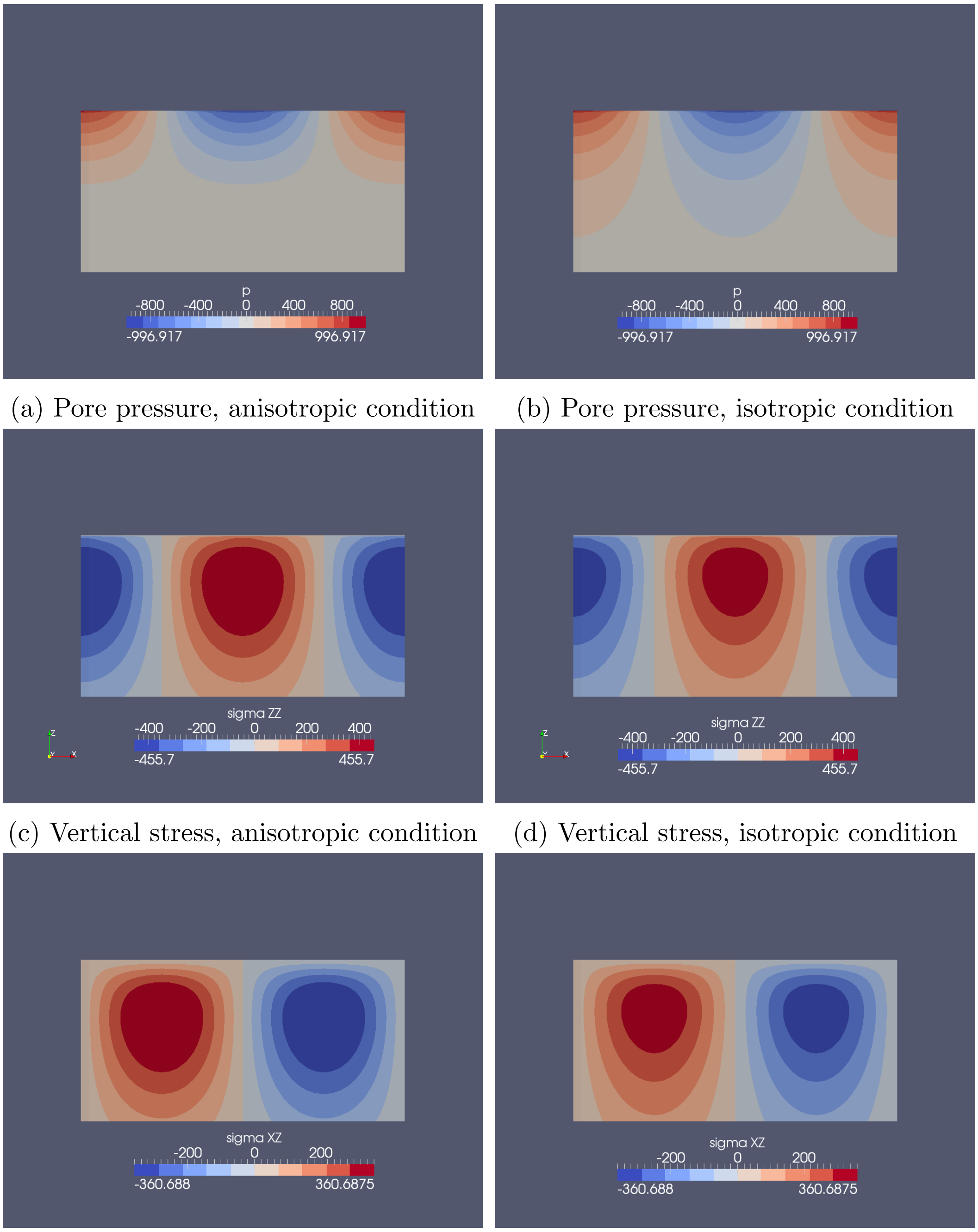
Figure 3.10: Verification of the mechanical anisotropy by comparison with the analytical solutions presented by Lekhnitskii (1981).



(a) A sketch of a short-crested wave pressure loading on a porous seabed with finite depth.

(b) Vertical distribution of the wave-induced pore pressure with varying saturation degree factor Sr.

Figure 3.11: Verification of the hydraulic anisotropy by comparison with the analytical solutions derived by Hsu and Jeng (1994).



(a) Pore pressure, anisotropic condition

(b) Pore pressure, isotropic condition

(c) Vertical stress, anisotropic condition

(d) Vertical stress, isotropic condition

(e) Shear stress, anisotropic condition

(f) Shear stress, isotropic condition

Figure 3.12: Comparison of the standing wave induced anisotropic and isotropic seabed responses. Wave input data: $p = p_0 \cos(\frac{2\pi}{L}x)\cos(\frac{2\pi}{T}t)$ with $T = 12.5$s, $L = 160$m and $p_0 = 10^3$Pa. Soil parameters: $n = 0.3$, $Sr = 0.99$, isotropy: $E = 10^7$Pa, $\nu = 0.3$, $k = 10^{-3}$m/s, anisotropy: $E_z = 10^7$Pa, $E_x = 0.6E_z$, $\nu_{xx} = \nu_{zx} = 0.3$, $G_z = 6 \cdot 10^6$Pa, $k_x = 5k_z = 10^{-3}$m/s.

3.4 Extension of the method

In previous sections, the flexibility of the segregated FVM coupled framework to handle a variety of nonlinearity and couplings in soil was demonstrated. The extension of the method can be made in a straightforward manner: it adopts a desired soil constitutive formulation and afterwards uses an associated local (nonlinear) stress integration algorithm. The global iterative solution procedure is essentially the same for all FVM soil models.

Though beyond the scope of this project, coupled analysis of (partially) saturated soft soils, such as clay, silt and peat, are of interest in geotechnical engineering. Soft soils show strong viscosity and time-dependence when exposed to loading. Due to plastic straining, they also develop gradual degradation of bonding (destructuration) and changes in fabric (induced plastic anisotropy). Constitutive modeling of these complex soft soil behaviors has been studied and interesting and promising results can be found in, e.g., Vermeer and Neher (1999); Wheeler et al. (2003); Leoni et al. (2008); Karstunen and Koskinen (2008) and Grimstad et al. (2010). Fully coupled analysis in this respect however, is relatively poorly understood. From the studies conducted in this work, it seems reasonable to assume that new knowledge can be gained also within the coupled behavior of soft soils using the proposed FVM approach.

Chapter 4

FVM Soil-water-structure interaction

With the custom-made FVM soil solvers developed, an attempt to explore the next important topic, that is, the multi-physical simulations of wave-soil-structure interaction is performed, based on a single software platform, OpenFOAM.

In principle, separate computational domains corresponding to the different physical fields are set up, and then various OpenFOAM solvers are used to solve the different partial differential equations governing each computational domain. The following is applied: an existing wave solver to compute the wave domain governed by Navier-Stokes equations; the implemented coupled soil solvers of this work to solve the seabed domain; and in the case of a deformable structure, an available elastic structural solver to calculate the displacements and stresses of the structure. The interactions in the system are considered through boundary couplings. Paper IV and V present the complete multi-physical simulations in greater detail than done here. The following sections hence focus on describing specifically how one can communicate information at the separate mesh interface boundaries, and also provide numerical examples to illustrate the applicability of the multi-physical model.

4.1 Interface coupling

In real ocean environments, the interactions between wave, seabed and offshore structure involve several coupling processes. The structure moves and vibrates slightly and transmits the wave loads down in to the soil via pressure and shear variations at the soil-structure interface. The soil also responds to the pressure variations on the seabed, both directly from the wave propagation and indirectly from the wave-induced structural response, by deformations and excess pore pressures. Furthermore, the subsequent seabed responses lead to an overall reduced soil stiffness and strength, which in turn results in enlarged displacements of the upper structure. A complete model covering all the possible couplings in the system is very complicated and expensive and beyond the objective of this work. Hence, here we make some assumptions to simplify the physical problem:

a.) The scour effects, which would alter the shape of the seabed and flow pattern near the structure, are disregarded.

b.) The dynamic response of the structure and its effect on the wave load itself is assumed to be minor, hence, it will not alter the calculated wave field;

c.) The seabed soil is considered stiff enough to support the structure in place during the simulation.

The first two aspects essentially means both the wave-soil and wave-structure couplings to be of 1-way character. These assumptions are acceptable, as the magnitude of wave-induced deformations of seabed and structure usually takes place on a much longer time scale than the wave period. The third aspect, assuming that the seabed can sustain the foundation fixed in its location, implies a 1-way soil-structure coupling. This simplification is valid for a relatively stiff soil at the pre-failure stage, namely when the soil-structure interaction force hasn't moved the structure significantly. Figure 4.1 illustrates such an overall 1-way interface coupling process graphically, using a submerged gravity foundation structure as an example. Although strong 2-way soil-structure interactions occurring under circumstances like complete liquefaction failures and large deformations in soft soils are currently not included in the analysis, however, they would be relatively easy to achieve in future work, as discussed later.

Indeed, the above discussed 1-way coupling scheme is quite efficient: it allows the fluid/solid/soil domains to be solved sequentially using different time steps and mesh sizes making convergence consideration more effective and simple. One can avoid employing very small time steps and grids that are usually required by the fluid domain on the solid/soil domains, and thus greatly reduces computational time.

The available pre- and post-processing utilities in OpenFOAM also facilitate an easy way of setting up the 1-way boundary coupling: a `sample` tool is used to collect the non-uniform wave pressure and/or structural stresses data

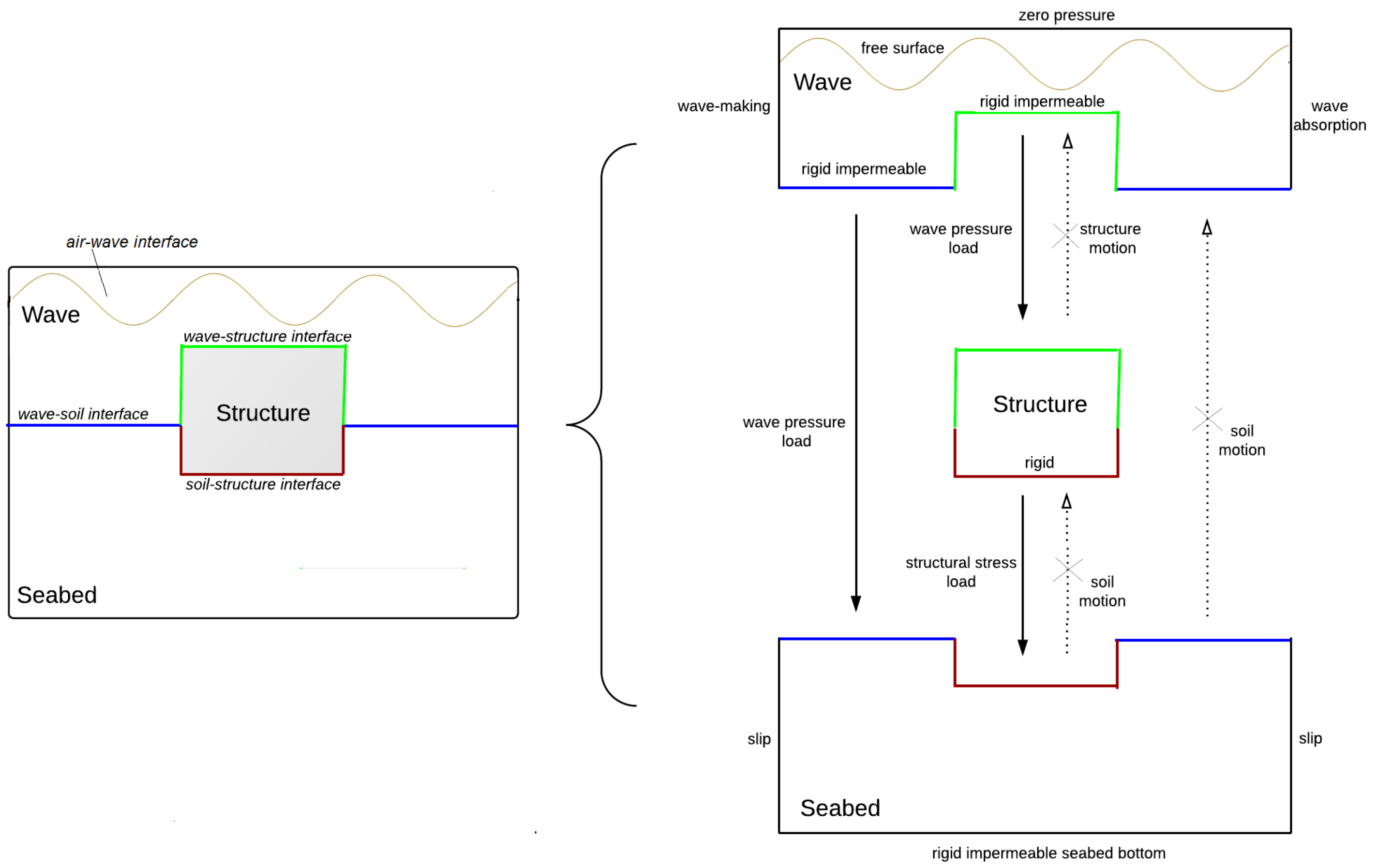


Figure 4.1: A schematic sketch of the current wave-soil-structure interaction simulation, with 1-way weak coupling.

on the interfaces at every (or certain) time points; and a primitive boundary type - `timeVaryingMappedFixedValue` - which serves to map the given spatial and temporal data onto the boundary patches with linear interpolations in both space and time, was simply "hacked" to fit to our needs of firstly mapping the wave pressure and/or structural stresses and then iteratively calculating the displacement gradient normal to the patches (as previously explained in Section 2.3).

4.2 Example simulations

As the main purpose of this part of the work is to demonstrate the new methodology of integrating FVM fluid, soil and(or) solid solvers together in OpenFOAM, some simple offshore engineering examples are considered including a buried marine pipeline and a gravity based foundation. The two examples shall then be described and discussed in the following subsections, respectively.

4.2.1 Wave-seabed-pipeline interaction

Submarine pipelines are commonly-used offshore installations, and their failures have been observed to be closely linked to the wave-induced instability of soil deposits. Interesting benchmark experiments carried out by Turcotte et al. (1984) and Teh et al. (2003), have contributed to improved knowledge of the interactions among waves, seabed and pipelines in the past, see Figure 4.2. Here simulations similar to their experiments are set up to numerically investigate the topic for comparison.

In the same case as in the experiment, a 18m long wave flume with the water depth of 0.533m is simulated by setting a wave generation inlet on the left hand side and a wave absorbing outlet on the right hand side boundary. The generated wave hight is set to be 14.3cm and the wave period is 1.75s. A rigid and impermeable pipeline with radius of 0.168m is buried (with bury depth of 0.086m) in a 4.57m long and 0.826m deep soil trench. The whole trench is placed at the mid-length of the wave flume.

The basic soil parameters available from the experimental data include elastic shear modulus $G = 6.4 \times 10^5$Pa, Poisson's ratio $\nu = 0.33$, permeability $k = 0.0011$m/s and porosity $n = 0.42$. The other plastic soil model parameters were obtained through careful calibrations and empirical relations whose values are listed in Table 4.1. One fact that needs to be pointed out is that the pipeline is only modeled as a fixed boundary in the simulation (no mesh), we focus on predicting the "potential" uplifting risks of the pipe

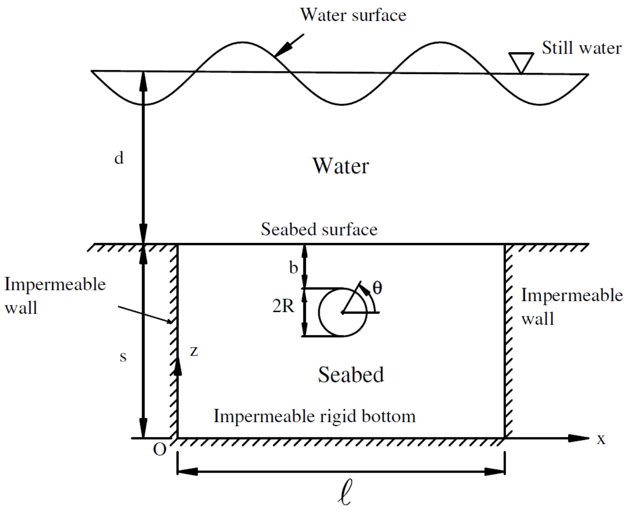


Figure 4.2: A sketch of the wave-seabed-pipeline interaction example, after Cheng and Liu (1986).

due to wave-induced soil seepage forces, hence, this may be considered as an indirect soil-structure interaction analysis in addition to the direct wave-soil interaction.

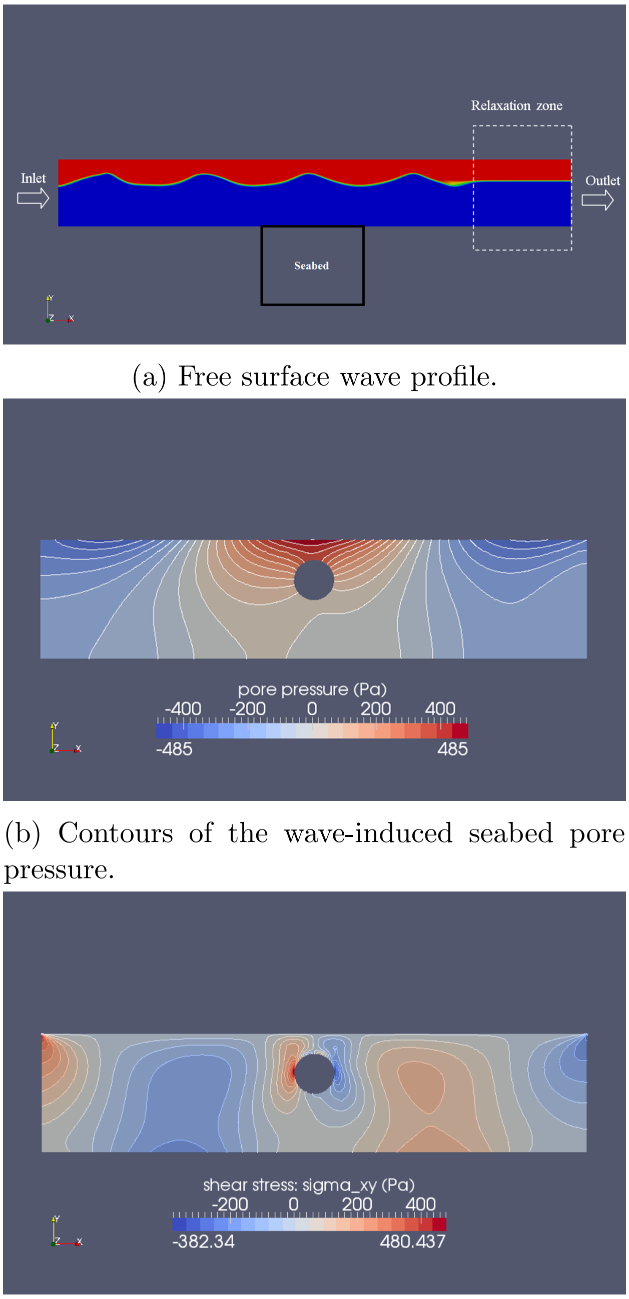
Table 4.1: Calibrated plasticity model constants for the cyclic soil constitutive model Manzari and Dafalias (1997).

Critical state		State parameter		Hardening		Dilatancy	
$M_{c(e)}$	1.15(1.035)	$k^b_{c(e)}$	3.975(2.0)	h_0	1200	A_0	6
λ	0.025	$k^d_{c(e)}$	4.2(0.07)	m, c_m	0.05, 0	F_{max}, C_f	500, 600

Figure 4.3 presents the simulated free surface wave as well as the subsequent wave-induced seabed pore pressure and shear stress contours. These transient snapshots are taken at a time $t = 20$s, at which instance the wave crest is passing right above the pipeline generating the highest amount of wave pressure. In particular, it can be found that significant shear stress concentrations on the sides of the pipeline are developed.

Figure 4.4 shows the wave-induced pore pressure amplitude around the pipeline, normalized by the base pressure. The available experiment data from Turcotte et al. (1984) are also plotted for comparison purpose. The current simulations slightly overshoot the measurements, which is mainly due to the approximation of the soil pore fluid compressibility ($Sr = 0.95$). Nevertheless, an overall match can be seen between the two.

In Figure 4.5, the seepage forces acting on the pipeline calculated by inte-



(a) Free surface wave profile.

(b) Contours of the wave-induced seabed pore pressure.

(c) Contours of the wave-induced seabed shear stress.

Figure 4.3: Snapshots of the simulated wave profile and seabed response at time $t = 20$s.

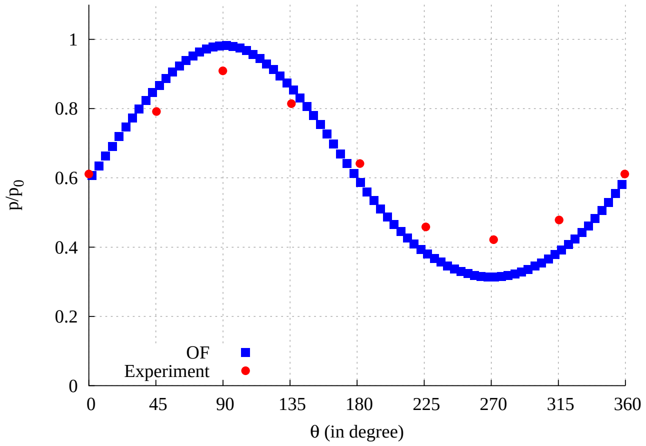


Figure 4.4: Simulated pore pressure distribution along the pipeline against the experiment measurements from Turcotte et al. (1984).

grating pore pressure along the pipeline surface are compared with available numerical solutions from Cheng and Liu (1986). The influence of varying soil permeability has been included as well. A good overall agreement between the simulation results and the solutions of Cheng and Liu (1986) is observed.

Another important parameter of importance in the pipeline design is the gradual pore pressure buildup during repeated wave cycles. Note that for this purpose a very small soil permeability has to be present so that it produces a poorly drained condition and facilitates the accumulation process. In fact, Teh et al. (2003) have used very fine soil deposits in their experiments to successfully capture the accumulation feature of the cyclic wave-induced pore pressure. The cyclic soil model is carefully calibrated against the experimental data. Figure 4.6a shows the simulated results together with the measured data at two different soil depths away from the pipeline: one at $z = 0.005$m, which is very near to the seabed surface and therefore the pore pressure change is strongly influenced by the periodic wave pressure fluctuation; and another at $z = 0.185$m, which is closer to the seabed bottom and thus pore pressure buildup is more significant due to the long drainage distance and the very low soil permeability. The performed simulations in general reproduce the experimentally observed periodic amplitudes as well as the overall accumulated trend of the pore pressure. Figure 4.6b presents the pore pressure buildups around the pipeline, where poor drainage is also created locally due to the impermeability of the pipe surface. It can be seen that the pore pressure build up rate varies along the pipeline: fastest at the

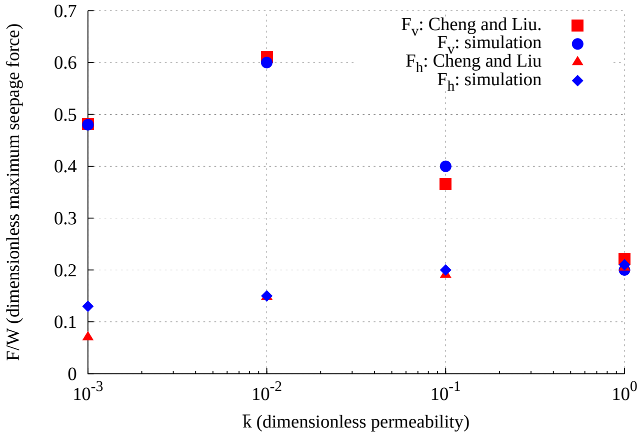


Figure 4.5: Simulated seepage forces on the pipeline against numerical solutions from Cheng and Liu (1986). $F_{v(h)}$: vertical(horizontal) seepage force, $W = \gamma\pi R^2$: the displaced water weight.

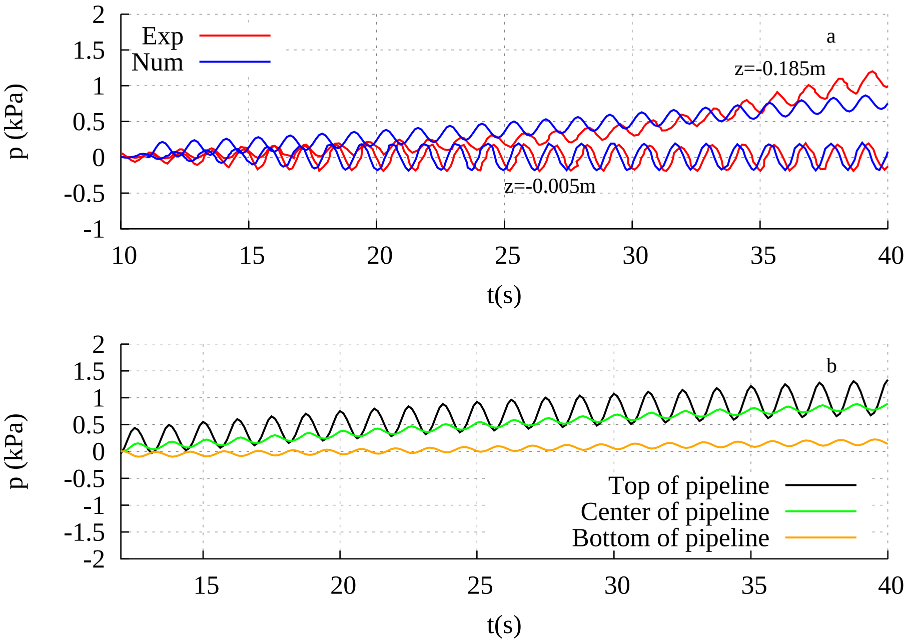


Figure 4.6: Simulated long term pore pressure variations at two different soil depths (top) and around the pipeline (bottom).

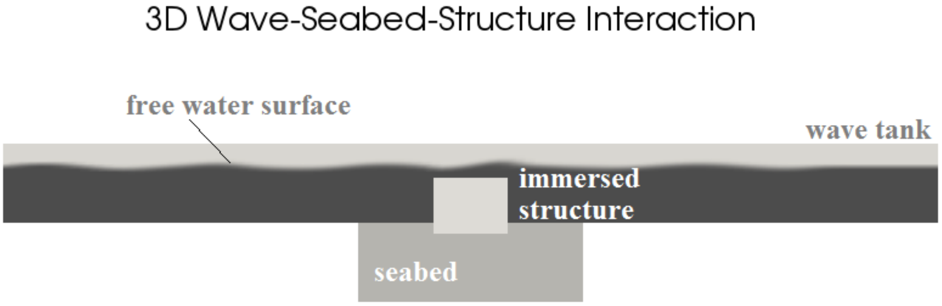


Figure 4.7: A sketch of the wave-seabed-gravity structure interaction layout (elevation view). Note that dimensions are not to scale.

top of the pipe while lowest at the bottom. This agrees with what reported in Dunn et al. (2006), that is, the liquefaction process propagates from the the top and proceeds downwards.

4.2.2 Wave-seabed-gravity structure interactions

In the second application example, we further study gravity foundation type structures which are exposed to wave loads and transmit the loads down to the soil through the soil-structure interface. Specifically, the simulations are carried out involving two cases: one is that the structure is a simple rigid body and will not move; while the other considers a more realistic structure which is deformable with small elastic displacements. In the former case, a complete structure solver is not necessary and therefore we calculate the wave-induced structural load on the soil simply by the total force balance of the structure. In the latter case, an elastic structure solver is adopted.

Case I: A simple rigid gravity structure

In this case, the numerical wave tank is a square of side length 150m with water depth 5m. Three wave-making inlets and one absorption outlet are placed at each side of the tank. The employed wave characteristics at the inlets are summed up in Table 4.2. The rigid and impermeable gravity structure (represented by a box of dimensions $20m \times 20m \times 5m$) is submerged in the middle of the wave tank with a buried depth 1m in the seabed. The porous seabed is also modeled as a square of side length 60m with soil depth 10m. The overall case geometry is illustrated in Figure 4.7.

The basic soil parameters, e.g., elastic shear modulus $G = 1.2 \times 10^7$kPa and Poisson's ratio $\nu = 0.2$, are adopted from the design code of gravity platforms in the North Sea, Kjekstad and Lunne (1981). The soil permeability is assumed to be k = 0.001m/s and the saturation factor is set to Sr = 0.98. The other plasticity model constants are kept the same as the previous studied pipeline case, for simplicity.

Table 4.2: Multi-directional wave parameters.

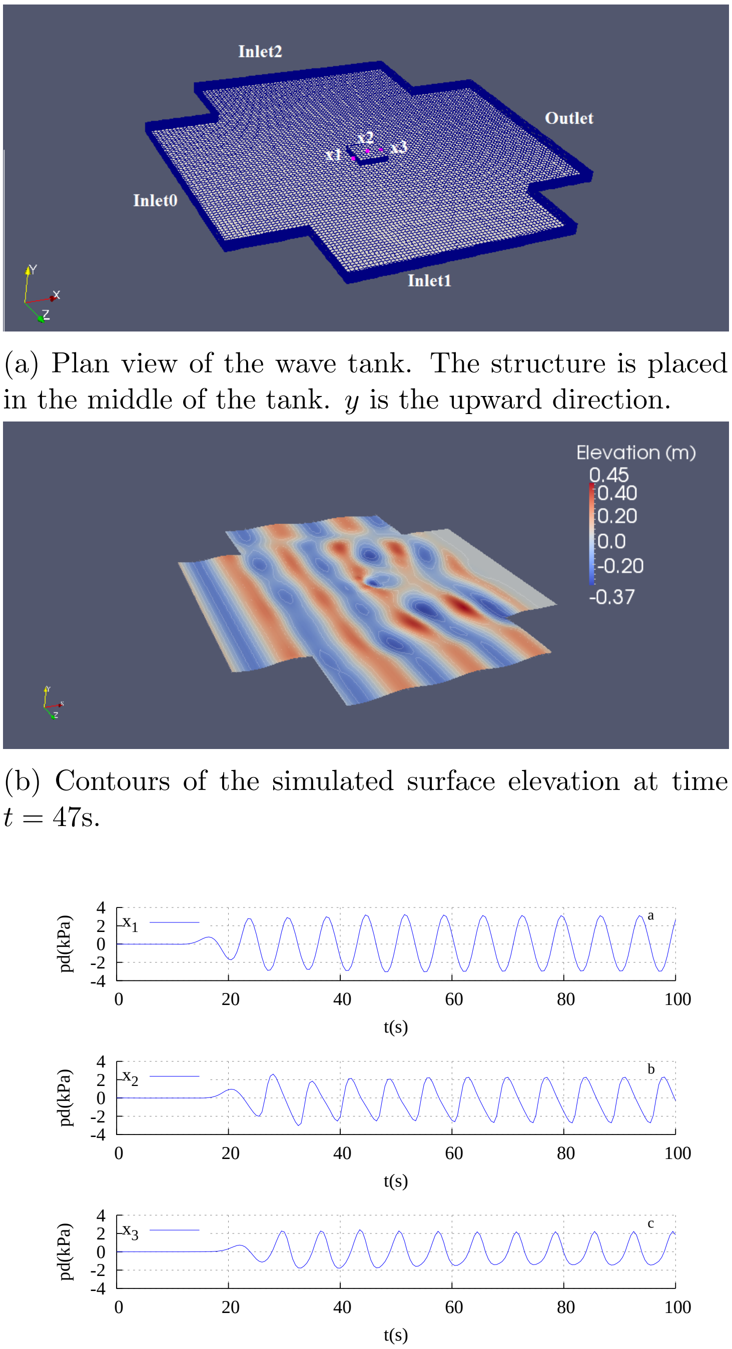
	Wave direction	Wave length	Wave period	Wave height
Inlet0	(1 0 0)	45.66m	7s	0.5m
Inlet1	(1 0 1)			
Inlet2	(2 0 -1)			

It is important to note that before switching on the wave tank simulation, a base state of the soil-structure system is computed, by simply putting the structure self-weight on the soil and running the elastic consolidation calculations until equilibrium is reached. This gives a realistic initial stress state of the seabed and the subsequent effect of the dynamic wave loads.

Figure 4.8a first provides an overview of the wave tank mesh, the inlet and outlet zones and the structure location. Figure 4.8b shows a snapshot of the modeled wave surface elevation, taken at the time t = 47s at which instance a wave crest passes on top of the structure. The modification of the structure on the wave field around it is clearly seen. Figure 4.8b also presents that the generated wave crest (0.45m) is further from the mean sea level than the wave trough (-0.37m), due to the nonlinear nature of waves. Figure 4.8c illustrates the cyclic dynamic wave pressures acting on the seabed and on the structure, sampled at three representative positions x_1, x_2 and x_3 as shown in Figure 4.8a.

Figure 4.9 presents the different wave force components and magnitude acting on the gravity structure. Those forces undergo cyclic variations along with the wave propagation. In particular, a larger pull upward force amplitude ($\approx$ 300kN) than its counterpart push downward force amplitude ($\approx$ 200kN) is simulated, as shown in the top right sub-figure of Figure 4.9. This pressure force variation complies with the previously mentioned fact that the generated wave crests are located further away from the mean sea level than the troughs.

Provided with the direct wave pressure load and the indirect structural stress load (which we obtained from the force balance of the rigid structure and uniformly applied on the soil-structure interface), the subsequent soil responses in both transient and long term scenarios will be predicted. Figure 4.10 is a snapshot of simulation results of the pore pressure distribution in the deformed seabed, taken at the time t = 47s. It is obvious that at this moment a wave crest is passing above the structure and consequently the



(a) Plan view of the wave tank. The structure is placed in the middle of the tank. y is the upward direction.

(b) Contours of the simulated surface elevation at time $t = 47$s.

(c) Simulated wave dynamic pressures acting on sea bottom and structure at the sample points x_1, x_2 and x_3 marked in (a).

Figure 4.8: Numerical wave tank simulations.

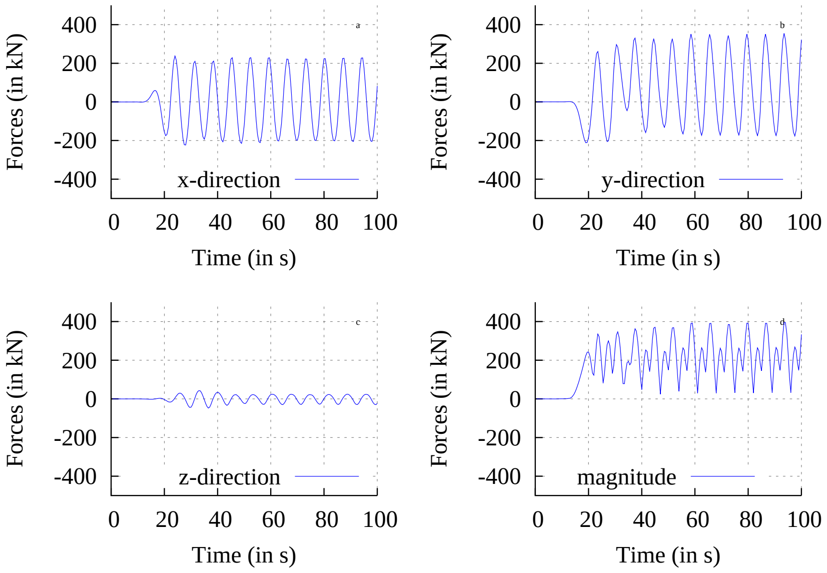


Figure 4.9: Simulated wave forces on the gravity structure.

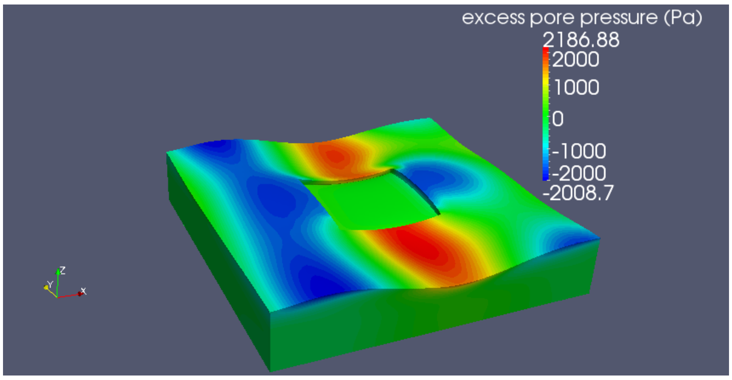


Figure 4.10: Simulated pore pressure distributions within the deformed seabed at time $t = 47$s. The deformation is exaggerated with a factor of 1000. z is the upward direction.

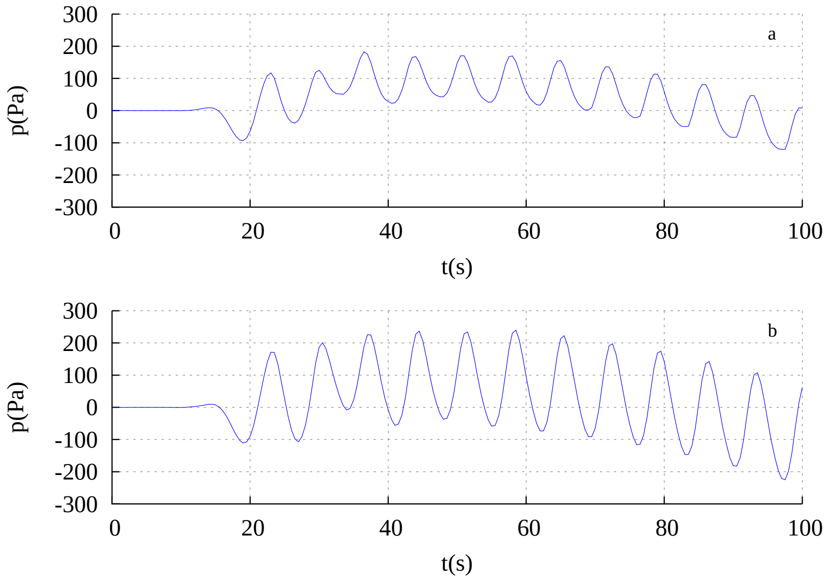


Figure 4.11: Simulated wave-induced pore pressure buildup at 0m (top) and 3m (bottom) beneath the center of the structure.

parts of soil facing the wave crest are compressed. More interesting, it is observed that only a shallow layer at the top of seabed has generated significant excess pore pressures, whereas the deeper soil layers experience almost no excess pore pressure upon the wave loading. This phenomenon is partially determined by the saturation degree (here $Sr = 0.98$ is used), because the air content in the soil makes the pressure dissipate at a very fast rate. One may expect that considerable pore pressure upward gradients can be created at the top layer under the wave trough, leading to the ultimate (transient) soil liquefaction and structure failure. Future research including 2-way soil-structure interaction designed for handling the final liquefied condition will probably give better predictions for the above discussed case.

Figure 4.11 demonstrates the long-term seabed response in terms of the generated excess pore pressure during wave cycles. It is clear that excess pore pressures (at the two soil locations underneath the impermeable structure) accumulates quickly during the initial wave cycles. However, followed by further applied cycles they start to dissipate. This prediction can be considered reasonable, since the soil underneath the structure is on one hand compressed during cyclic shearing of the soil skeleton and this generates positive pore pressure; while on the other hand the structural force pulls the soil upwards (as referring to Figure 4.9) and creates negative pore pressure.

The two above mentioned processes are competing against each other and in the end the pull upward force becomes dominant according to the simulation results obtained. It is important to keep in mind that pore pressure buildup in a seabed is always a combined effect of certian soil materials (e.g., soils with very small permeability and loose-packed density) and unfavorable wave loading conditions (like short wave periods and large to medium amplitude exposures).

Case II: A realistic deformable gravity structure

In this case, a realistic gravity-based foundation standing on the seabed for offshore wind farm is considered, see Figure 4.12. The wave is generated in the direction (1 0.3 0), in which z is the vertical axis. Full reflection on the side walls of the numerical wave tank causes a directional spreading sea state. The gravity structure is deformable and the actual displacements and stresses are computed by a linear elastic solid mechanics solver. The seabed is assumed as poro-elastic with anisotropic characteristics in mechanical and hydraulic behaviour.

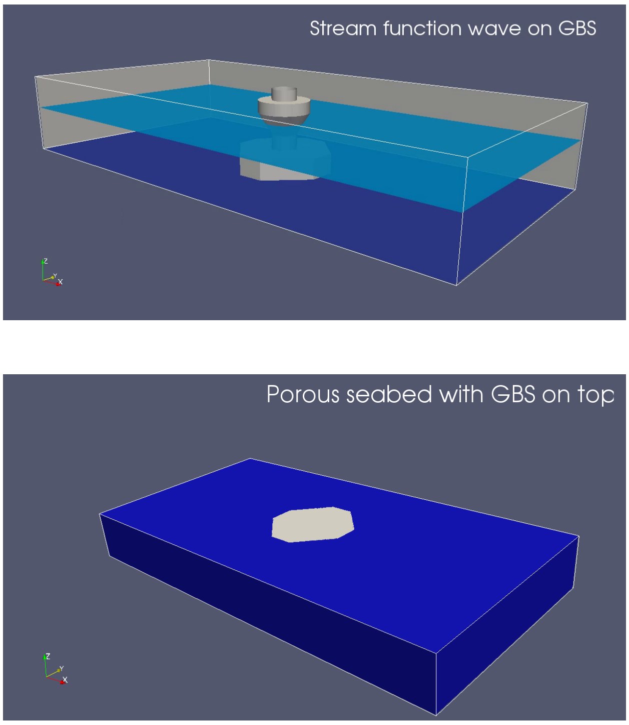


Figure 4.12: Computational domains of wave, gravity structure and seabed.

Details of the geometries of the different domains as well as the input parameters for the wave generation and dynamic structure and soil calculations are reported in Paper V. Here, the computational cost of the simulation and the major results are briefly described.

The total computational cost for solving the three domains are given in Table 4.3 below.

Table 4.3: Computational data of the 1-way integration simulation for a total time period $t = 17$s.

Domain	No. of cells	No. of CPU	CPU time	Time step
Wave	1753264	8	29067s	Adjustable, ca. 1∼3ms
Structure	86400	1	105672s	Fixed, 0.05s
Seabed	247500	1	39678s	Fixed, 0.05s

Figure 4.13 shows the calculated free surface of water waves at different times in a wave period ($T = 6.5$s). The presence of the structure has modified the wave motion around it. The exerted wave pressures acting on the seafloor (Figure 4.13) and on the gravity structure (Figure 4.14) is highly nonlinear and 3-dimensional, partly due to the disturbance effect of the structure and also due to the directional spreading of natural waves. This complex wave pressure loading environment is impossible to get from analytical solutions based on wave theory, but can readily be estimated from advanced CFD wave solvers.

Figure 4.15 illustrates the computed structural responses in terms of the equivalent shear stress measurement σ_{eq} as well as the magnitude of displacement, at a time $t = 10.5$s when a wave crest approached to the structure. The equivalent stress σ_{eq} is defined by $\sigma_{eq} = \sqrt{2/3\mathbf{s} : \mathbf{s}}$, in which $\mathbf{s}$ is the stress deviatoric tensor, i.e., $\mathbf{s} = \boldsymbol{\sigma} - 1/3(\mathrm{tr}\boldsymbol{\sigma})\mathbf{I}$. Considerable amount of shear stresses are generated around the cone part of the gravity structure as a consequence of the dynamic wave pressure variations. However, the resulting total displacement of the structure is still very small, which complies with the previously mentioned 1-way wave-structure interaction assumption.

The transient seabed behavior including the pore fluid (seepage) flow, soil deformations and shearing stresses are important factors for geotechnical foundation design. Figure 4.16 shows the pore fluid flows induced in the seabed at time $t = 14$s, at which instance a wave trough is passing the gravity structure. There are notable upwards pore fluid flows created underneath the structure. It is commonly acknowledged that when the upwards seepage forces exceed the self-weight of the soil, the momentary soil liquefaction can

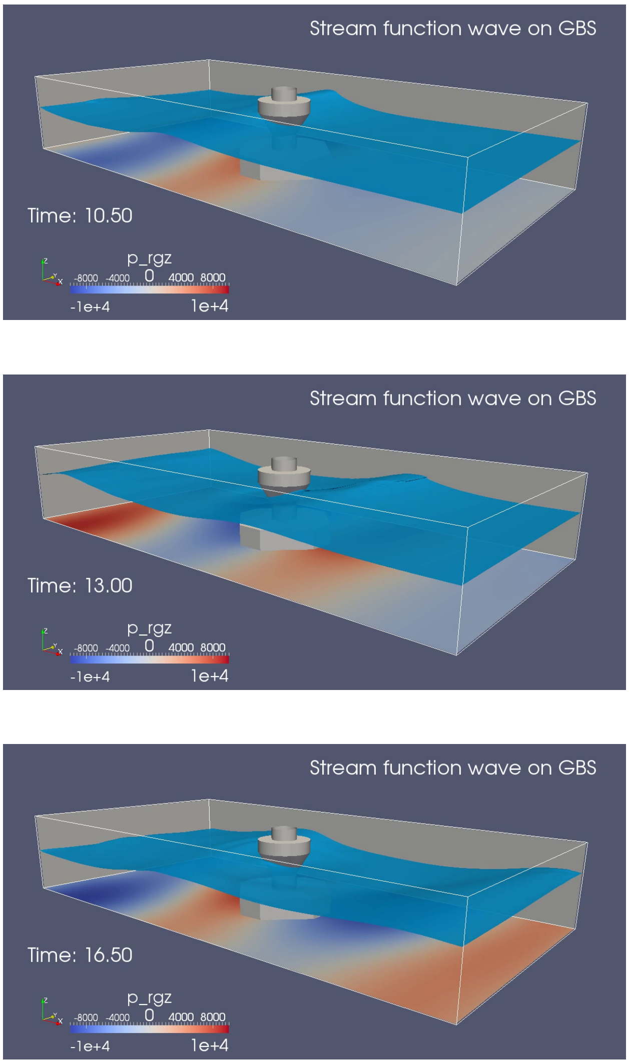


Figure 4.13: Free surface of waves passing by the gravity structure in a wave period. The resulting dynamic wave pressures p_rgz acting on the sea floor are colored from blue to red.

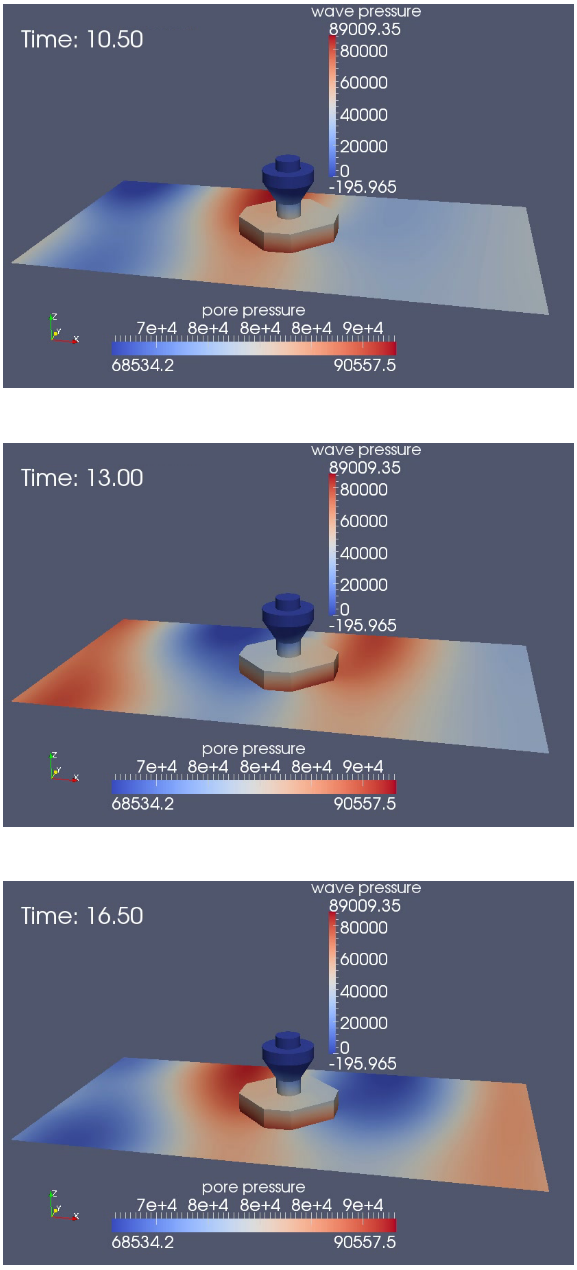


Figure 4.14: Generated wave pressure load on the gravity structure and porous seabed surface at times corresponding to Figure 4.13. Note that the pore pressure in the soil and the wave pressure on the structure use separate scales.

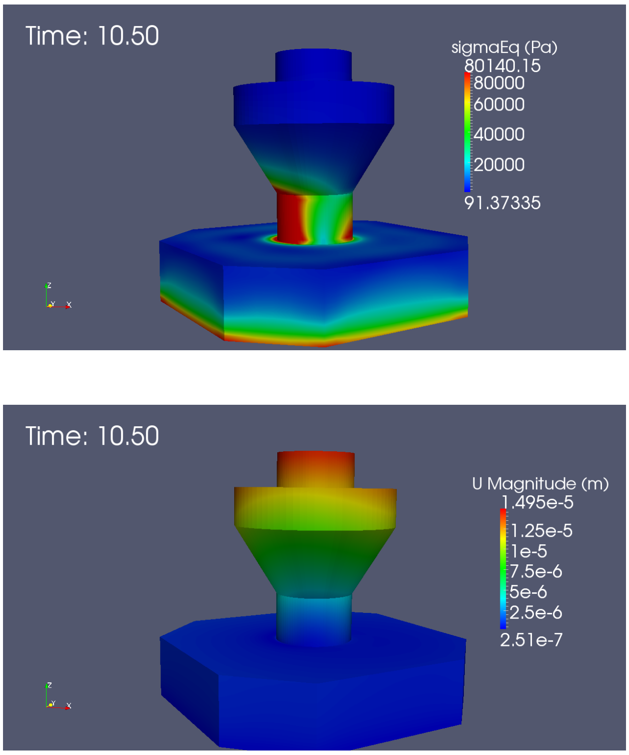


Figure 4.15: Computed structure responses, represented by the distributions of the equivalent shear stress (top) and the displacement magnitude (bottom).

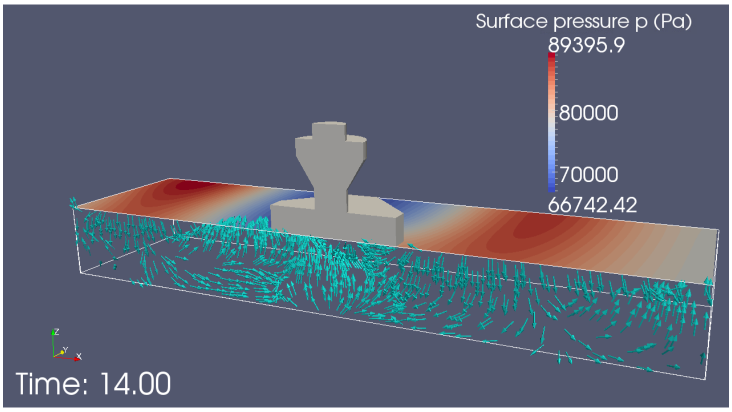


Figure 4.16: Seepage flows occurred in the seabed soil at time $t = 14$s (wave trough). The display is in a cutting plane along x-axis.

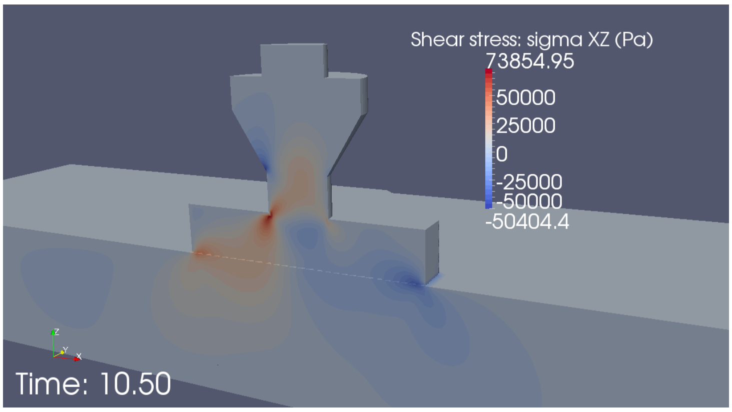


Figure 4.17: Contours of the shear stress distribution in the seabed and gravity structure at time $t = 10.50$s.

occur and thus impacts on the safety of the upper structure, Jeng (1997). This leads to a widely applied liquefaction criteria:

$$(p - p_b) \geq \frac{1 + 2K_0}{3}(\gamma_s - \gamma_w)z \tag{4.1}$$

where p_b is the pore water pressure at the seabed surface, K_0 is the coefficient of lateral earth pressure and the value of $K_0 = 0.5$ is used here, and $\gamma_{s(w)}$ is the unit weight of soil and water, respectively. The liquefaction risk was examined accordingly, and it is found that the seabed soil did not reach liquefaction failure given the wave and soil inputs in this simulation work.

Figure 4.17 presents the contours of the wave-induced shear stress, σ_{xz}, in the seabed soil and the upper gravity structure. It is interesting to see that in the field away from the structure the soil shear stress directly produced by the fluctuating wave pressures is almost negligible, compared to the concentred shearing zones around the structure edges. The latter are generated because of the wave pressures as well, however, the loads are transferred trough the gravity structure and act on the soil 'indirectly'. This implies that the presence of the structure greatly amplifies the wave shearing effects down to the underlying soil. Integrated wave-structure-seabed interaction analysis can help to give correct predictions of the various loading conditions and consequently the more realistic seabed response.

Chapter 5

Convergence considerations

The convergence issue is a key factor determining the applicability of the proposed FVM soil modeling approach. This chapter therefore studies the convergence criteria of the fixed point iterative scheme adopted in the coupled FVM soil models in some detail. Also possible stabilization and acceleration methods for the iteration procedure are discussed.

5.1 Convergence of the fixed point iteration

Table 5.1 summarizes the steps of the fixed point iteration scheme in a general schematic manner. One shall keep in mind that the function $F(x)$ becomes a complicated system of partial differential equations with nonlinearity and coupling features, and the *segregated* FVM discretization creates $G(x)$ as an explicit function that contains all the nonlinear and coupling dependencies. However, the overall iteration process is essentially the same.

Table 5.1: Algorithm - Fixed Point Iteration.

```
1. Given an equation $F(x) = 0$
2. Convert $F(x) = 0$ into the form $x = G(x)$
3. Specify the initial guess to be $x_0$
4. Do
       $x_{i+1} = G(x_i)$
   while (neither the convergence criterion C1 nor C2 is met)
```

C1: the maximum number of iterations N.

C2: $|x_{i+1} - G(x_i)|$ less than some tolerance limit.

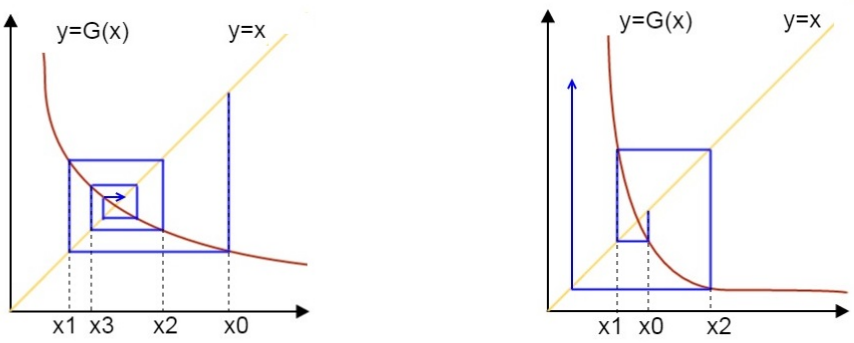
A number of research studies have been carried out by, e.g., Rhoades (1976); Weng (1991) and Hyvarinen (1999) to address the important convergence question: what choices on $G(x)$, does the fixed point iteration algorithm converge? The answer is the use of an unique criteria as follows:

If $G(x)$ is a differentiable function, and

$$|G'(x)| \leq \alpha < 1 \qquad \text{for all } x \in [a, b] \tag{5.1}$$

then $G(x)$ has exactly one fixed point l_0 in $[a, b]$ and the sequence (x_i) with a starting point $x_0 \in [a, b]$ converges to l_0. The constant α establishes the rate of convergence, the closer it is to 1 the slower the convergence will be.

Figure 5.1 provides a graphical representation of both converged and diverged types of fixed point iterations.



(a) Converged sequence, $|G'(x)| < 1$. (b) Diverged sequence, $|G'(x)| > 1$.

Figure 5.1: A graphical example of fixed point iteration convergence.

In practice, checking the above convergence condition is often not easy, mainly due to the complexity of $G(x)$ and consequently the determination of $G'(x)$. This is also the case for the coupled soil models developed in this work: the choice of $G(x)$ is a system of complicated functions that describe both the nonlinear stress condition and the soil-pore fluid coupling. It is necessary to separate these issues. In other words, one can study the convergence condition assuming that $G(x)$ merely contains the segregated nonlinear terms; and likewise one need to check the condition assuming $G(x)$ only involves the contribution from the segregated coupling. If both criteria are fulfilled, there is a better chance that the overall segregated function will lead to a reliable converged solution.

In the appendix of Paper II, details on the derivation of $G'(x)$ considering the two above mentioned segregation aspects have been reported. Here only the major findings will be present. For the case of partitioned nonlinear terms in the momentum equation: if the nonlinearity is based on simple perfect plasticity, the fixed point iteration scheme unconditionally converges, due to the fact that the implicit stiffness is always greater than its explicit counterpart; whereas for other more advanced hardening plasticity models, the iterative scheme only converges conditionally and may require stabilization methods. For the case of partitioned soil-pore fluid coupling, the convergence of the iterative scheme can be guaranteed if the following condition is met:

$$K' \leq n(2\mu + \lambda) \tag{5.2}$$

in which, the left land side term of Eq. (5.2) is the effective bulk modulus of the pore fluid, and the right hand side of Eq. (5.2) is the bulk modulus of the soil skeleton multiplied by a factor of porosity n.

For natural partially saturated soils, one percent of air in the pore volume results in a dramatic reduction of the bulk modulus of the pore fluid to $K' \approx$ 10MPa, comparable to the soil bulk modulus. Hence, the lower saturation of the soil, the better convergence of the iterative scheme.

5.2 Stabilization and acceleration

In general, the fixed point iteration method is very simple and computationally economic for dealing with nonlinear and coupled problems. There is no need for the formulation, storage and update of a Jacobian matrix as is used in the standard types of Newton-Raphson methods. Only the source terms of the equations are iteratively re-evaluated depending on the explicit approximations of the nonlinearity and coupling, and matrix-vector products are all performed with relatively small sizes as a result of the segregation.

However, there is always a price to pay, and that is, the convergence of the method is closely influenced by the specific soil features (e.g., the values of the material parameters describing hardening plasticity and pore fluid compressibility). It is common that the proposed simple iterative scheme do not have convergence when large plastic deformations and/or very strong pore pressure coupling are present in the simulations. Under such circumstances, one can usually apply a fixed under-relaxation factor $\theta \in [0, 1]$ and modify the iterative guess, so as to promote the stability of the scheme:

$$\tilde{x}_{i+1} = \theta x_{i+1} + (1 - \theta) x_i \tag{5.3}$$

In the equation above, x corresponds to the variables $\delta\mathbf{u}$, p and $\delta\boldsymbol{\varepsilon}^p$. This means that the method under-relaxes not only the two primary solution variables, $\delta\mathbf{u}$ and p, obtained from the global matrix solutions, but also the plastic strain term, $\delta\boldsymbol{\varepsilon}^p$, calculated from the local stress integration.

The fixed under-relaxation technique is very easy to incorporate into the soil solvers, and furthermore helpful for stabilization. For most of the simulations performed in this work, convergence was the case using (sufficiently) small value of θ. It should be point out that the convergence rate of the scheme turns out to be very slow in some cases. It would be better if one could use some kind of acceleration processes that adjust the value of θ in order to speed up the convergence.

One of the widely-used techniques for accelerating a convergent fixed-point sequence is the Aitken's method, Küttler and Wall (2008); Walker and Ni (2011). The principles of this method is as follows. For the first iteration, it still chooses a fixed under-relaxation factor $\theta_0 = const.$; starting from the second iteration, it uses the residual r measured by:

$$r_i = x_{i+1} - x_i \tag{5.4}$$

to adjust the value of θ by the formulation below:

$$\theta_{i+1} = -\theta_i \frac{(r_i - r_{i-1})^{\mathrm{T}} r_{i-1}}{(r_i - r_{i-1})^{\mathrm{T}}(r_i - r_{i-1})} \tag{5.5}$$

In the Aitken's method, the relaxation factor is dynamically determined based on the measured residual history (in terms of the latest two residuals). It is interesting to note that, Eq. (5.5) allows for a single averaged value of θ to under-relax the vectors of the corresponding variable over the domain. One may also calculate the relaxation factor cell by cell, and then under-relax the variable for each cell. Here the first simpler option was implemented. Indeed, also a very simple tension case using a nonlinear stress solver with both relaxation methods was tested. Table 5.2 presents the computational cost data. It is encouraging that Aitken's method can reduce the cost with a factor of two during the plastic steps, compared to the fixed under-relaxation approach.

Table 5.2: Convergence acceleration within a large deformation elasto-plastic simple tension test case.

Relaxation method	No. fixed point iterations (plastic step)	CPU time (s)
Fixed under-relaxation	618	38.34
Aitken's method	264	15.79

Another useful but more complicated acceleration method is the so-called Quasi-Newton Inverse Least Squares (QN-ILS) algorithm, Degroote et al. (2009); Haelterman et al. (2009). As revealed by its name, this technique approximates the inverse of the Jacobian matrix in the Newton-Raphson method using the Least Squares approach:

$$x_{i+1} = x_i + \underbrace{(J)^{-1}}_{-\theta I}(-r_i), \quad J = \frac{dr}{dx} \tag{5.6}$$

More simply speaking, it makes use of the complete residual and iterative histories, namely $(r_i, r_{i-1}, r_{i-2}...r_0)$ and $(x_i, x_{i-1}, x_{i-2}...x_0)$, to achieve a high-rank approximation of the Jacobian and subsequently an accurate modification of the predicted unknowns in the iteration. It is an "expensive" option, because the whole iterative series as well as the residual series have to be stored. Furthermore, Least-Squares calculations also needs to be performed for obtaining J. Due to the above mentioned facts the QN-ILS algorithm is not used in this work. However, it can be an interesting subject to explore in the future, for the sake of fast convergence in problems with highly nonlinear and strong coupling characteristics.

Chapter 6

Concluding remarks

6.1 Conclusions

In this thesis, the wave-induced seabed responses - especially in terms of pore pressure variations and nonlinear soil deformations - which are important factors influencing the overall stability of offshore structures, have been investigated. The investigation has mainly been carried out through numerical modeling, and in particular by using FVM. The developed FVM soil models are based on the modified Biot's consolidation equations where strong plastic soil-pore fluid couplings in the porous soil is considered. Such coupled soil models can effectively predict the transient flow of pore fluid and its influence on the strength of the soil skeleton and vice versa, in natural saturated soils under partially drained conditions. The implicit and explicit discretisation strategies and the associated *segregated* solution procedure are applied to treat the quite complicated, nonlinear and coupled equation sets. A simple fixed point iteration scheme along with relaxation techniques is also used to produce reliable and converged solutions.

Practically, specific examples of FVM coupled soil models were implemented and tested with the aid of OpenFOAM codes and utilities. The developed FVM models covered different soil constitutive features, such that they can be used to reproduce the distinct soil behaviors under the transient and cyclic modes of wave loading. The verification and application test cases in general demonstrated that FVM is an effective tool for the analysis of various coupled, nonlinear soil problems. Especially, within a strip footing test, the performance of the developed FVM poro-elasto-plasticity model was compared with that of a similar commercial FEM coupled model: there was found a noticeable advantage of the *segregated* FVM scheme over the FEM implicit solution in terms of computational costs. Furthermore, perhaps the most interesting part of the established FVM coupled soil framework is that it is not limited to these implemented examples; instead, it is flexible and easily extendable to include other nonlinear soil stress-strain relations, and thus generally applicable to solve other coupled soil issues in the wider field of geotechnics.

With the developed FVM soil solvers as a basis, modeling of the soil-water-structure interactions was implemented by adopting a unified FVM technology. In this part of the work, existing FVM fluid and solid models in OpenFOAM were adopted and incorporated together with the already developed FVM soil model to solve the wave, structure and seabed field. The interactions between the domains were accounted by coupling at their common boundaries. An overall 1-way coupling for the common boundaries was assumed in this work, which enabled the different fields to be solved sequentially: i.e., first the wave dynamics is solved and followed by the structure and seabed response solutions. Two types of offshore structures were examined, including the buried marine pipeline and the gravity foundations. The simulation output reasonably captures the wave-induced transient and long term soil behaviour in the seabed in the presence of a solid structure.

6.2 Future work

In this work, the interaction of the wave-soil system has been modeled 1-way by neglecting the sediment transport. This is true for time scale of interest here of a few wave periods, whereas on the time scale of many wave periods the seabed shape may gradually deform due to scour and the flow field is affected consequently. The transformation of the seabed shape will also change the domain of the consolidation and change the pore pressure distribution. The inclusion of such gradual deformation is an interesting possible extension.

The interaction of the soil-structure system has also been modeled 1-way, by assuming the seabed soil is stiff enough to support the structure in place during the simulation. However, this is not always valid. Under circumstances like complete liquefaction failures and large deformations, the soil can no longer sustain the structure and the structure will move significantly. Strong 2-way interaction of soil-structure system is necessary to be included in the future.

Bibliography

Christian LeBlanc Bakmar, O. Hededal, and Lars Bo Ibsen. *A Modified Critical State Two-surface Plasticity Model for Sand: Theory and Implementation.* DCE Technical Memorandum. Department of Civil Engineering, Aalborg University, 2008.

JP Bardet. Bounding surface plasticity model for sands. *Journal of Engineering Mechanics*, 112(11):1198–1217, 1986.

M.A. Biot. General theory of three dimensional consolidation. *Journal of Applied Physics*, 12(2):155–164, 1941.

P. Cardiff, A. Karac, and A. Ivankovic. Development of a finite volume contact solver based on the penalty method. *Computational Materials Science*, 64:283–284, 2012.

P. Cardiff, A. Karac, D. FitzPatrick, R. Flavin, and A. Ivankovic. Development of a hip joint model for finite volume simulations. *Journal Of Biomechanical Engineering-Transations Of The ASME*, 136(1):–, 2014a.

P. Cardiff, A. Karac, and A. Ivankovic. A large strain finite volume method for orthotropic bodies with general material orientations. *Computer Methods in Applied Mechanics and Engineering*, 268:318, 2014b.

Alexander H-D. Cheng and Philip L-F. Liu. Seepage force on a pipeline buried in a poroelastic seabed under wave loadings. *Applied Ocean Research*, 8 (1):22 – 32, 1986.

B. Chowdhury, G. R. Dasari, and T. Nogami. Laboratory study of liquefaction due to wave-seabed interaction. *Journal Of Geotechnical And Geoenvironmental Engineering*, 132(7):842–851, 2006.

Johan Clausen, Lars Damkilde, and Lars Andersen. An efficient return algorithm for non-associated plasticity with linear yield criteria in principal stress space. *Computers and Structures*, 85(23-24):1795–1807, 2007.

M. B. de Groot, M. D. Bolton, P. Foray, P. Meijers, A. C. Palmer, R. Sandven, A. Sawicki, and T. C. Teh. Physics of liquefaction phenomena around marine structures. *Journal Of Waterway Port Coastal And Ocean Engineering-ASCE*, 132(4):227–243, 2006a.

M. B. de Groot, M. Kudella, P. Meijers, and H. Oumeraci. Liquefaction phenomena underneath marine gravity structures subjected to wave loads. *Journal Of Waterway Port Coastal And Ocean Engineering-ASCE*, 132(4): 325–335, 2006b.

Joris Degroote, Klaus-Jürgen Bathe, and Jan Vierendeels. Performance of a new partitioned procedure versus a monolithic procedure in fluid–structure interaction. *Computers & Structures*, 87(11):793–801, 2009.

I. Demirdzic and A. Ivankovic. Finite volume stress analysis. University Lecture, 1999.

I. Demirdzic and D. Martinovic. Finite volume method for thermo-elasto-plastic stress analysis. *Computer Methods in Applied Mechanics and Engineering*, 109(3-4):331–349, 1993.

I. Demirdzic and S. Muzaferija. Finite-volume method for stress analysis in complex domains. *International Joural For Numerical Methods In Engineering*, 37(21):3751–3766, 1994.

I. Demirdzic and S. Muzaferija. Numerical method for coupled fluid flow, heat transfer and stress analysis using unstructured moving meshes with cells of arbitrary topology. *Computer Methods in Applied Mechanics and Engineering*, 125(1-4):235–255, 1995.

I. Demirdzic, S. Muzaferija, and M. Peric. Benchmark solutions of some structural analysis problems using finite-volume method and multigrid acceleration. *International Joural For Numerical Methods In Engineering*, 40(10):1893–1908, 1997.

I. Demirdzic, I. Horman, and D. Martinovic. Finite volume analysis of stress and deformation in hygro-thermo-elastic orthotropic body. *Computer Methods In Applied Mechanics And Engineering*, 190(8-10):1221–1232, 2000.

SL Dunn, PL Vun, AHC Chan, and JS Damgaard. Numerical modeling of wave-induced liquefaction around pipelines. *Journal of waterway, port, coastal, and ocean engineering*, 132(4):276–288, 2006.

J. Ferentinos. Global offshore oil and gas outlook. `http://www.gaselectricpartnership.com/HOffshore%20Infield.pdf`, 2013. ACCESSED: 2014-08-31.

A. Flora, S. Lirer, and F. Silvestri. Undrained cyclic resistance of undisturbed gravelly soils. *Soil Dynamics And Earthquake Engineering*, 43:366–379, 2012.

FP Gao and YX Wu. Non-linear wave-induced transient response of soil around a trenched pipeline. *Ocean Engineering*, 33(3-4):311–330, 2006.

FP Gao, DS Jeng, and H. Sekiguchi. Numerical study on the interaction between non-linear wave, buried pipeline and non-homogenous porous seabed. *Computers And Geotechnics*, 30(6):535–547, 2003.

Gustav Grimstad, Samson Abate Degago, Steinar Nordal, and Minna Karstunen. Modeling creep and rate effects in structured anisotropic soft clays. *Acta Geotechnica*, 5(1):69–81, 2010.

Rob Haelterman, Joris Degroote, Dirk Van Heule, and Jan Vierendeels. The quasi-newton least squares method: a new and fast secant method analyzed for linear systems. *SIAM Journal on numerical analysis*, 47(3):2347–2368, 2009.

Pablo Higuera, Javier L. Lara, and Inigo J. Losada. Three-dimensional interaction of waves and porous coastal structures using openfoam. part i: Formulation and validation. *Coastal Engineering*, 83(0):243 – 258, 2014a.

Pablo Higuera, Javier L. Lara, and Inigo J. Losada. Three-dimensional interaction of waves and porous coastal structures using openfoam. part ii: Application. *Coastal Engineering*, 83(0):259 – 270, 2014b.

J.R.C. Hsu and D.S. Jeng. Wave-induced soil response in an unsaturated anisotropic seabed of finite thickness. *International Journal for Numerical and Analytical Methods in Geomechanics*, 18(11):785–807, 1994.

Masayuki Hyodo, Hidekazu Murata, Norituki Yasufuku, and Teruhisa Fujii. Undrained cyclic shear strength and residual shear strain of saturated sand by cyclic triaxial tests. *Soils And Foundations*, 31(3):60–76, 1991.

Aapo Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *Neural Networks, IEEE Transactions on*, 10(3):626–634, 1999.

Kenji Ishihara and Mitsutoshi Yoshimine. Flow potential of sand during liquefaction. *Soils and Foundations*, 38(3):189–198, 1998.

A. Ivankovic, I. Demirdzic, J. G. Williams, and P. S. Leevers. Application of the finite volume method to the analysis of dynamic fracture problems. *International Journal of Fracture*, 66(4):357–371, 1994.

Niels G. Jacobsen, David R. Fuhrman, and Jørgen Fredsøe. A wave generation toolbox for the open-source cfd library: Openfoam. *International Journal for Numerical Methods in Fluids*, 70(9):1073–1088, 2012.

H Jasak and HG Weller. Application of the finite volume method and unstructured meshes to linear elasticity. *International journal for numerical methods in engineering*, 48(2):267–287, 2000.

Hrvoje. Jasak. *Error analysis and estimation for the finite volume method with applications to fluid flows.* PhD thesis, Imperial College London (University of London), 1996.

D. Jeng. Soil response in cross-anisotropic seabed due to standing waves. *Journal of Geotechnical and Geoenvironmental Engineering*, 123(1):9–19, 1997.

D.-S. Jeng, X.L. Zhou, J.H. Wang, D.-S. Jeng, X.D. Luo, J. Zhang, and F.P. Gao. Response of porous seabed to dynamic loadings. *Geotechnical Engineering*, 41(4), 2010.

Dong Sheng Jeng. Wave-induced sea floor dynamics. *Applied Mechanics Reviews*, 56(4):407–429, 2003.

DS Jeng and YS Lin. Wave-induced pore pressure around a buried pipeline in gibson soil: Finite element analysis. *International Joural For Numerical And Analytical Methods In Geomechanics*, 23(13):1559–1578, 1999.

DS Jeng and BR Seymour. Response in seabed of finite depth with variable permeability. *Journal Of Geotechnical And Geoenvironmental Engineering*, 123(10):902–911, 1997.

Minna Karstunen and Mirva Koskinen. Plastic anisotropy of soft reconstituted clays. *Canadian Geotechnical Journal*, 45(3):314–328, 2008.

O. Kjekstad and T. Lunne. Soil parameters used for design of gravity platforms in the north sea. *Applied Ocean Research*, 3(2):50 – 58, 1981.

M. Kudella, H. Oumeraci, M. B. de Groot, and P. Meijers. Large-scale experiments on pore pressure generation underneath a caisson breakwater. *Journal Of Waterway Port Coastal And Ocean Engineering-ASCE*, 132(4): 310–324, 2006.

Ulrich Küttler and Wolfgang A Wall. Fixed-point fluid–structure interaction solvers with dynamic relaxation. *Computational Mechanics*, 43(1):61–72, 2008.

A. Kvitrud, G. Ersdal, R. L. Leonhardsen, et al. On the risk of structural failure on norwegian offshore installations. In *Proceedings of the 11th international offshore and polar engineering conference on ISOPE*, 2001.

SG Lekhnitskii. *Theory of elasticity of an anisotropic elastic body.* Holden Day, 1981.

Martino Leoni, Minna Karstunen, and PA Vermeer. Anisotropic creep model for soft soils. *Géotechnique*, 58(3):215–226, 2008.

XK Li, JB Zhang, and HW Zhang. Instability of wave propagation in saturated poroelastoplastic media. *International Joural For Numerical And Analytical Methods In Geomechanics*, 26(6):563–578, 2002.

O.S. Madsen. Wave-induced pore pressures and effective sresses in a porous seabed. *Geotechnique*, 28(4):377–393, 1978.

W. Magda. Wave-induced cyclic pore-pressure perturbation effects in hydrodynamic uplift force acting on submarine pipeline buried in seabed sediments. *Coastal Engineering*, 39(2-4):243 – 272, 2000.

K. Maneeratana. *Development of the finite volume method for non-linear structural applications.* PhD thesis, Imperial College London, 2000.

MT Manzari and YF Dafalias. A critical state two-surface plasticity model for sands. *Geotechnique*, 47(2):255–272, 1997.

MT Manzari and R. Prachathananukit. On integration of a cyclic soil plasticity model. *International Joural For Numerical And Analytical Methods In Geomechanics*, 25(6):525–549, 2001.

A.M. Mostafa, N. Mizutani, and K. Iwata. Nonlinear wave, composite breakwater, and seabed dynamic interaction. *Journal of Waterway, Port, Coastal and Ocean Engineering*, 125(2):88–97, 1999.

Jean-Herve Prevost. Plasticity theory for soil stress-strain behavior. *Journal of the Engineering Mechanics Division*, 104(5):1177–1194, 1978.

W. Raman-Nair and G.C.W. Sabin. Wave-induced failure of poro-plastic seabed slopes: A boundary element study. *Proc, Inst of Civil Eng, Part 2*, 91:771–794, 1991.

BE Rhoades. Comments on two fixed point iteration methods. *Journal of Mathematical Analysis and Applications*, 56(3):741–750, 1976.

K. Sadeghi. Significant guidance for design and construction of marine and offshore structures. *GAU J. Soc. & Appl. Sci.*, 4(7):67–92, 2008.

S. Sassa and H. Sekiguchi. Wave-induced liquefaction of beds of sand in a centrifuge. *Geotechnique*, 49(5):621–638, 1999.

S. Sassa and H. Sekiguchi. Analysis of wave-induced liquefaction of sand beds. *Geotechnique*, 51(2):115–126, 2001.

Luc Scholtes, Bruno Chareyre, Herve Michallet, Emanuele Catalano, and Donia Marzougui. Modeling wave-induced pore pressure and effective stress in a granular seabed. *Continuum Mechanics and Thermodynamics*, 2014.

H. Sekiguchi and R. Phillips. Generation of water-waves in a drum centrifuge. *Centrifuge 91*, pages 343–350, 1991.

AK Slone, C. Bailey, and A. Cross. Dynamic solid mechanics using finite volume methods. *Applied Mathematical Modeling*, 27(2):69–87, 2003.

J. C. Small, J. R. Booker, and E. H. Davis. Elasto-plastic consolidation of soil. *International Journal of Solids and Structures*, 12(6):431–448, 1976.

B. M. Sumer, S. K. Sumer, J. Fredsøe, and F. H. Dixen. Pore pressure buildup in the subsoil under a caisson breakwater. *Proceedings of the International Offshore and Polar Engineering Conference*, pages 664–671, 2008.

B. Mutlu Sumer, Jørgen Fredsøe, S. Christensen, and M.T. Lind. Sinking/floatation of pipelines and other objects in liquefied soil under waves. *Coastal Engineering*, 38:53–90, 1999.

B. Mutlu Sumer, Özgür Kirca, and Jørgen Fredsøe. Experimental validation of a mathematical model for seabed liquefaction under waves. *International Journal of Offshore and Polar Engineering*, 22(2):133–141, 2012.

B.M. Sumer and J. Fredsøe. *The Mechanics of Scour in the Marine Enviroment.* World Scientific, River Edge, N.J., 2002.

GA Taylor, C. Bailey, and M. Cross. A vertex-based finite volume method applied to non-linear material problems in computational solid mechanics. *International Joural For Numerical Methods In Engineering*, 56(4):507–529, 2003.

T.C Teh, A.C Palmer, and J.S Damgaard. Experimental study of marine pipelines on unstable and liquefied seabed. *Coastal Engineering*, 50(1 - 2): 1 – 17, 2003.

Z. Tukovic, A. Ivankovic, and A. Karac. Finite-volume stress analysis in multi-material linear elastic body. *International Joural For Numerical Methods In Engineering*, 93(4):400–419, 2013.

B.R. Turcotte, P.L.F. Liu, and F.H. Kullhawy. *Laboratory evaluation of wave tank parameters for wave-sediment interaction.* Joseph. H. DeFrees Hydraulic Laboratory Report 84-1. School of Civil and Environmental Engineering, Cornell Univeristy, 1984.

M. B. C. Ulker, M. S. Rahman, and M. N. Guddati. Wave-induced dynamic response and instability of seabed around caisson breakwater. *Ocean Engineering*, 37(17-18):1522–1545, 2010.

M. B. C. Ulker, M. S. Rahman, and M. N. Guddati. Breaking wave-induced response and instability of seabed around caisson breakwater. *International Joural For Numerical And Analytical Methods In Geomechanics*, 36 (3):362–390, 2012.

PA Vermeer and HP Neher. A soft soil model that accounts for creep. In *Proceedings of the International Symposium "Beyond 2000 in Computational Geotechnics"*, pages 249–261, 1999.

A. Verruijt. Elastic storage of aquifers. In R.J.M. De Wiest, editor, *Flow through Porous Media*, pages 37–51. Academic Press, New York, 1969.

Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method.* Pearson Education, 2007.

Homer F Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.

Yuchen Wang and Erwin Oh. Finite difference model for assessing sandy seabed response. *Electronic Journal of Geotechnical Engineering*, 18:5449–5470, 2013.

H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Comput. Phys.*, 12(6):620–631, 1998.

Xinlong Weng. Fixed point iteration for local strictly pseudo-contractive mapping. *Proceedings of the American Mathematical Society*, 113(3):727–731, 1991.

MA Wheel. A geometrically versatile finite volume formulation for plane elastostatic stress analysis. *The Journal of Strain Analysis for Engineering Design*, 31(2):111–116, 1996.

Simon J Wheeler, Anu Näätänen, Minna Karstunen, and Matti Lojander. An anisotropic elastoplastic model for soft clays. *Canadian Geotechnical Journal*, 40(2):403–418, 2003.

D. M. Wood. *Soil Behaviour and Critical State Soil Mechanics*. Cambridge Univeristy Press, New York, NY 10011., 1990.

T. Yamamoto. Wave-induced pore pressures and effective sresses in inhomogeneous seabed foundations. *Ocean Engineering*, 8(1):1–16, 1981.

T. Yamamoto, H. L. Koning, H. Sellmeijer, and E. Van Hijum. On the response of a poro-elastic bed to water waves. *Journal of Fluid Mechanics*, 87(1):193–206, 1978.

QS Yang and HB Poorooshasb. Seabed response to wave loading. *Proceedings Of The Seventh (1997) International Offshore And Polar Engineering Conference, VOL I, 1997*, pages 689–695, 1997.

Jianhong Ye, Dongsheng Jeng, Ren Wang, and Changqi Zhu. A 3-d semi-coupled numerical model for fluid-structures-seabed-interaction (fssi-cas 3d): Model and verification. *Journal Of Fluids And Structures*, 40:148–162, 2013a.

Jianhong Ye, Dongsheng Jeng, Ren Wang, and Changqi Zhu. Validation of a 2-d semi-coupled numerical model for fluid-structure-seabed interaction. *Journal Of Fluids And Structures*, 42:333–357, 2013b.

H. Zhang and DS Jeng. An integrated three-dimensional model for wave-induced seabed response in a porous seabed. i. a sloping seabed. *Ocean Engineering*, 32(5-6):701–729, 2005.

J. S. Zhang, D. S. Jeng, and P. L. F. Liu. Numerical study for waves propagating over a porous seabed around a submerged permeable breakwater: Poro-wssi ii model. *Ocean Engineering*, 38(7):954–966, 2011a.

X. L. Zhang, D. S. Jeng, and M. T. Luan. Dynamic response of a porous seabed around pipeline under three-dimensional wave loading. *Soil Dynamics And Earthquake Engineering*, 31(5-6):785–791, 2011b.

Y. Zhang, D.-S. Jeng, J.-S. Zhang, H. Zhang, H. Zhao, and D.-S. Jeng. Three-dimensional model for wave-induced dynamic soil response around breakwaters. *Proceedings of the International Offshore and Polar Engineering CONFERENCE*, pages 1349–1355, 2012.

Chunyan Zhou, Guangxue Li, Ping Dong, Jinghao Shi, and Jishang Xu. An experimental study of seabed responses around a marine pipeline under wave and current conditions. *Ocean Engineering*, 38(1):226 – 234, 2011.

Part II

Papers

Paper I

"A coupled soil-pore fluid formulation for modeling soil liquefaction and cyclic mobility in seabed using the finite volume method"

T. Tang, J. Roenby & O. Hededal

Published in: *Proceedings of the Int'l Conference on Advances in Coupled System Mechanics (ACSM12) under umbrella of ACEM12, 2012*

The 2012 World Congress on
Advances in Civil, Environmental, and Materials Research (ACEM' 12)
Seoul, Korea, August 26-30, 2012

A coupled soil-pore fluid formulation for modeling soil liquefaction and cyclic mobility in seabed using the finite volume method

*Tian Tang[1], Johan Roenby [2] and Ole Hededal [3]

[1], [3] *Department of Civil Engineering, Technical University of Denmark, Lyngby, Denmark*
[2] *DHI, Hoersholm, Denmark*
* *tiat@byg.dtu.dk*

ABSTRACT

The stability of offshore structures, such as wind turbine foundations, breakwaters, and immersed tunnels can be strongly affected by the liquefaction and cyclic mobility phenomena in the seabed. Our goal is to develop a numerical code for analysis of these situations. For this purpose, we start by formulating the strong interactions between soil skeleton and the pore fluid via a coupled set of partial differential equations. A single bounding surface soil model capable of simulating the accumulations of pore pressures, strains, dilatancy, and strain 'softening', is then adopted for quantifying the cyclic soil constitutive relations. To deal with the high non-linearity in the equations, the finite volume (FV) method is proposed for the numerical simulation. The corresponding discretization strategies and solution algorithms, including the conventional segregated method and the more recent block matrix solver, are discussed as well. Overall, investigations in this paper provide a methodology for developing a numerical code simulating liquefaction and cyclic mobility. In future work this will be implemented in practice with the aid of the open source CFD toolbox, OpenFOAM.

1. INTRODUCTION

Soil liquefaction and cyclic mobility are two of the most important subjects in offshore engineering when the stability of offshore foundations is to be assessed. Liquefaction typically occurs in saturated loose granular soil when subject to repeated cyclic loading (e.g. waves, currents, vibration of superstructures in a marine environment). The gradual build-up of pore pressure due to compaction of the soil skeleton eventually results in a failure state of zero effective stress. Cyclic mobility, on the other hand, is characteristic for medium-dense granular soil with stabilized pore pressure and effective stresses, but with large permanent shear strains. Both of these phenomena are results of the strong interaction of the soil skeleton with the fluid present in the pore structure (generally water). Therefore, in order to model and simulate these complex phenomena more accurately, a coupled analysis of soil and pore fluid is necessary.

[1] PhD student
[2] Researcher
[3] Associate Professor

The fundamental theory describing such a coupled soil-pore fluid system was first established by Biot (1941) assuming an elastic soil skeleton and a single pore fluid phase satisfying Darcy's law. Biot's theory was later extended to cope with non-linear geomaterial behaviors by Zienkiewicz (1982), Zienkiewicz & Shiomi (1984) and Zienkiewicz et al. (1990). Three general formulations, differing in the choice of unknowns, were put forward: the *u-p-U*, *u-U*, and *u-p* formulations, where *u* symbolizes the soil skeleton displacements vector, *p* is the pore fluid pressure, and *U* is the pore fluid displacements vector. The *u-p-U* formulation containing all the three unknowns is powerful in dealing with high frequency phenomena (Oka et al. 1994). The *u-U* formulation aimed at tracking the displacements of both the soil skeleton and the fluid phase has not been widely applied due to the potential volumetric locking problem in numerical simulations (Jeremic et al. 2008). The simplistic yet capable *u-p* formulation is currently the most common description used in practice (Oka et al. 1994, Elgamal et al. 2002, Di & Sato 2004, and Taiebat et al. 2007) since it is valid for low-frequency problems in dynamic analysis and reduces the total number of degrees of freedom.

Any of the above mentioned formulations for liquefaction and cyclic mobility modeling requires a well-defined constitutive law for capturing the granular seabed behavior under cyclic loading conditions. From field measurements and cyclic loading experiments, it is evident that plastic deformations occur both under loading and under unloading-reloading processes. The resulting hysteresis loop (Fig. 1.) in each cycle indicates that the soil is unable to return all the energy put into it during loading. Hence, conventional plasticity models with isotropic hardening, though adequate for modeling monotonic loading, fail to work under cyclic conditions. During the last few decades, the limitation of classical plasticity has motivated extensive efforts towards developing appropriate cyclic plasticity models. Kinematic hardening laws have been incorporated as well as models describing realistic smooth transition from elastic to fully plastic domains. Among these models, the two most successful and widely applied cyclic plasticity concepts are the *multi-surface plasticity* proposed by Mroz (1967), and the *bounding surface plasticity* by Dafalias and Popov (1975). In the branch of multi-surface plasticity, Prevost (1985) applied a simple multi-surface J_2 theory for frictional cohesionless soils. Elgamal et al. (2002) and Elgamal et al. (2003) further updated Prevost's model and applied it to the subject of earthquake engineering. Recently, Yang & Elgamal (2008) put forward a new multi-surface model considering the Lode angle effect to capture the three-dimensional loading conditions. The bounding surface theory has also been extended by several researchers (Bardet 1986, Crouch et al. 1994, Manzari & Dafalias 1997, and Li et al. 1999) to capture more realistic soil behavior such as strain softening, state dependence, fabric anisotropy, and the behavior under multi-axial loading conditions.

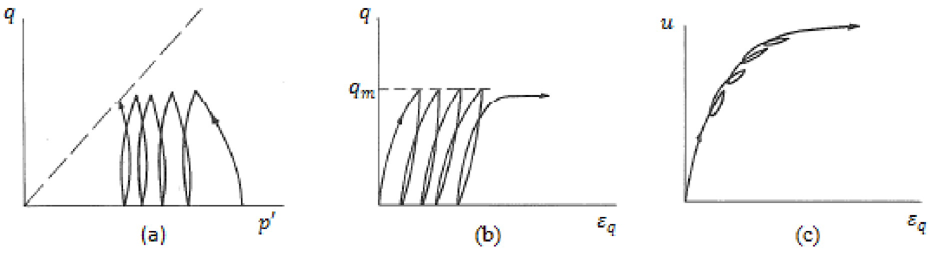


Fig. 1. Typical soil response observed during undrained cyclic loading: (a) effective stress path; (b) stress:strain response; (c) pore pressure:strain response (from Wood 1990).

Numerical simulations of hysteretic soil behavior and pore fluid pressure accumulation have mainly been conducted using the finite element method (Zienkiewicz et al. 1993, Huang & Zienkiewicz 1998, Pastor et al. 2000, and Taiebat et al. 2007), or a combination of finite element and the finite difference method (Oka et al. 1994, and Di & Sato 2003). Due to the complexity of cyclic plasticity, these coupled models are not implemented into commercial engineering software packages presently applied in engineering practice. The objective of this paper is therefore to develop a numerical code for analysis of liquefaction and cyclic mobility, which is applicable in the practical context of offshore engineering. In the following sections we will present the methodology on achieving this goal.

2. GOVERNING EQUATIONS FOR THE COUPLED SYSTEM

In this section, a mathematical framework is introduced for quantifying the behavior of the coupled soil-pore fluid system. We adopt the method of the *u-p* formulation (Zienkiewicz & Shiomi 1984, Zienkiewicz et al. 1990, Oka et al. 1994), which has the soil skeleton displacements and the pore fluid pressure as the principal unknowns. Some fundamental assumptions are made:

1) The soil grains are incompressible;
2) The soil skeleton strains remain sufficiently small;
3) The relative acceleration of the fluid phase to the solid phase is much smaller than the acceleration of the solid phase, i.e. $\dot{v}_i^f - \dot{v}_i^s \ll \dot{v}_i^s$. Where $v_i^{f(s)}, \dot{v}_i^{f(s)}$ are the velocity and acceleration of the fluid (solid) phase, respectively;

The general formulations are then derived one by one from conservation equations.

First, consider the overall momentum equation for the soil mixture:

$$(1-n)\rho_s \dot{v}_i^s + n\rho_f \dot{v}_i^f = \sigma_{ij,j} + (1-n)\rho_s b_i + n\rho_f b_i, \qquad (1)$$

Here, n is the porosity; ρ_s and ρ_f are the densities of solid and fluid phase, respectively; $\sigma_{ij,j}$ stands for the divergence of the total soil stress tensor; and b_i is the body force acceleration.

Defining the mixture density $\rho = (1-n)\rho_s + n\rho_f$ and applying the third assumption above, Eq. (1) is reduced to:

$$\rho \dot{v}_i^s = \sigma_{ij,j} + \rho b_i, \qquad (2)$$

Next, based on the mass balances of the solid phase and the fluid phase, and the state of equation for the fluid that reads $\dot{\rho}_f / \rho_f = \dot{p}/K_f$, the following equation can be derived:

$$\frac{n}{K_f}\dot{p} + v_{i,i}^s + n\left(v_i^f - v_i^s\right)_{,i} = 0, \qquad (3)$$

Here p represents the pore fluid pressure and K_f is the bulk modulus of the pore fluid. The last term in Eq. (3) is the specific discharge. It can be rewritten using the momentum equation for the pore fluid phase,

$$n\rho_f \dot{v}_i^f = np_{,i} + n\rho_f b_i + \frac{n^2\gamma_w}{k}\left(v_i^f - v_i^s\right), \qquad (4)$$

where the last term is the Darcy drag force due to friction with the skeleton with k denoting the scalar permeability coefficient and $\gamma_w$denoting the unit weight of fluid. Inserting the specific discharge from Eq. (4) in the last term of Eq. (3), and employing assumption 3) to replace $\dot{v}_i^f$ with $\dot{v}_i^s$, we obtain the equation

$$\frac{n\gamma_w}{K_f k}\dot{p} + \frac{\gamma_w}{k}v_{i,i}^s + \rho_f \dot{v}_{i,i}^s - p_{,ii} - \rho_f b_{i,i} = 0, \tag{5}$$

To further formulate the obtained governing equations - Eq. (2) and Eq. (5) in terms of our interested unknowns (u^s, p), the following relations are specified:

a) The effective stress concept: $\Delta\sigma_{ij} = \Delta\sigma'_{ij} + \alpha\delta_{ij}(\Delta p)$, where $\Delta\sigma'_{ij}$ and $\Delta\sigma_{ij}$represent, respectively, the increment of effective stress and total stress, Δp is the incremental pore fluid pressure, δ_{ij} is the Kronecker delta and α is the Biot-Willis coefficient in the range of $n \le \alpha \le 1$, (Wang 2000);

b) The nonlinear stress-strain relations of soil skeleton: $\Delta\sigma'_{ij} = D_{ijkl}(\Delta\varepsilon_{kl})$. D_{ijkl} stands for the tangential stiffness matrix defined by state variables (e.g. σ', ε), the direction of the increment and/or other considerations (loading history, strain rate, etc.). Details about this matrix will be presented in section 3;

c) The linear strain-displacement relation of soil skeleton: $\Delta\varepsilon_{ij} = \frac{1}{2}\left(\Delta u_{i,j}^s + \Delta u_{j,i}^s\right)$;

d) The velocity-displacement and acceleration-displacement relations of soil skeleton: $v_i^s = \dot{u}_i^s$ and $\dot{v}_i^s = \ddot{u}_i^s$.

Based on the above specification, since it is more convenient to keep the unknowns in incremental form due to the incremental stress-strain relationship, Eq. (2) and Eq. (5) can be rewritten as:

$$-\rho\Delta\ddot{u}_i^s + \left(D_{ijkl}\Delta u_{l,k}^s\right)_{,j} + \alpha\Delta p_{,i} + \rho\Delta b_i = 0, \tag{6}$$

$$\frac{n\gamma_w}{K_f k}\Delta\dot{p} + \frac{\gamma_w}{k}\Delta\dot{u}_{i,i}^s + \rho_f\Delta\ddot{u}_{i,i}^s - \left(\Delta p_{,i}\right)_{,i} - \rho_f\Delta b_{i,i} = 0, \tag{7}$$

With the soil skeleton displacement and pore fluid pressure present simultaneously in all the governing equations, the strong interaction in the soil-pore fluid system is modeled.

3. CONSTITUTIVE MODELING

To this point, the most crucial work left is an appropriate constitutive law describing the cyclic soil stress-strain relationship, i.e. the determination of the tangential soil stiffness matrix D_{ijkl},.

The key issues in the selection of a plasticity model for the analysis of soil-fluid interaction are a) the ability to alter the void ratio of the soil, since it is this feature that allows pore pressures to develop irreversibly, leading e.g. to liquefaction; and b) a realistic but preferably numerically simple method for modeling the cyclic behavior of soil.

During the last thirty years, soil constitutive models that are able to capture the accumulations of pore pressures and strains upon cycling have been developed largely by introducing the multi-surface plasticity and the bounding surface plasticity concepts (Prevost 1985, Bardet 1986, Manzari & Dafalias 1997, Elgamal et al. 2003, and Yang & Elgamal 2008, etc.). The multi-surface plasticity is characterized by an approximation of the actual stress-strain curve by *n* linear segments of constant stiffness moduli. The bounding surface plasticity model is featured with continuously changed stiffness moduli depending on the distance from current stress state to a correspondent image state on the prescribed bounding surface. Since both of the two cyclic plasticity theories have merits and shortcomings, a comparison is presented in Appendix I. The purpose is to select a model, which is effective enough to describe the cyclic soil behavior and yet economic enough to be implemented and applied in practice.

As a result, Bardet's single bounding surface model which has a very simple surface definition, few model constants as well as straightforward mapping technique will be chosen for the practical implementation. The numerical efficiency of Bardet's model is also considered positive since only the bounding surface must be updated in each increment, and the evolution of the surface is controlled by the volumetric strains as in classical Cam Clay models. The essential elements of this model are presented in the following.

3.1 General constitutive equations

Classically, the increment of strain resulting from a stress increment is assumed as the sum of the elastic ($\Delta\varepsilon_{ij}^{e}$) and plastic ($\Delta\varepsilon_{ij}^{p}$) incremental strains:

$$\Delta\varepsilon_{ij} = \Delta\varepsilon_{ij}^{e} + \Delta\varepsilon_{ij}^{p}, \tag{8}$$

Hence, the elastic constitutive equation can be written as:

$$\Delta\sigma'_{ij} = E_{ijkl}\,(\Delta\varepsilon_{kl}^{e}) = E_{ijkl}(\Delta\varepsilon_{kl} - \Delta\varepsilon_{kl}^{p}), \tag{9}$$

E_{ijkl} is the isotropic elastic stiffness tensor: $E_{ijkl} = K\delta_{ij}\delta_{kl} + G(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk} - \frac{2}{3}\delta_{ij}\delta_{kl})$, in which K stands for the bulk modulus and G is the shear modulus.

The increment of plastic deformation tensor can be defined through the flow rule,

$$\Delta\varepsilon_{ij}^{p} = \langle L\rangle m_{ij}. \tag{10}$$

Here, m is a dimensionless symmetric second-order tensor indicating the 'direction' of the plastic strain determined by the outward normal to a plastic potential surface; L is the loading function, and the symbol <> denotes the McCauley's bracket so that $\langle L\rangle = L$ if $L \geq 0$; otherwise $\langle L\rangle = 0$. The loading function is defined as:

$$L = \frac{1}{H}(n_{ij}\Delta\sigma'_{ij}), \tag{11}$$

In Eq. (11), $H$is the plastic modulus, and n, which is also a symmetric second-order tensor, is the outward normal to the convex yield (bounding) surface. Substituting Eq. (9)-(10) into Eq. (11), the expression for the loading function can be re-written with respect to the strain increments:

$$L = \frac{n_{ij}E_{ijkl}}{H + n_{pq}E_{pqrs}m_{rs}}\Delta\varepsilon_{kl}, \tag{12}$$

Combing Eq. (9)-(10) and Eq. (12), it is now possible to form the full stress-strain relationship and thus obtain D_{ijkl}:

$$\Delta\sigma'_{ij} = E_{ijkl}(\Delta\varepsilon_{kl} - \langle L\rangle m_{kl}) = \left\{E_{ijkl} - \frac{E_{ijpq}m_{pq}n_{rs}E_{rskl}}{H+n_{pq}E_{pqrs}m_{rs}}\right\}\Delta\varepsilon_{kl} = D_{ijkl}(\Delta\varepsilon_{kl}), \quad (13)$$

Most plastic models can fit within the above general framework. What makes a model unique is the specification of n, m and H, i.e. the yield (bounding) surface, the plastic potential and the plastic modulus. The specified features for Bardet's model are summarized in the following.

3.2 Bounding surface and radial mapping

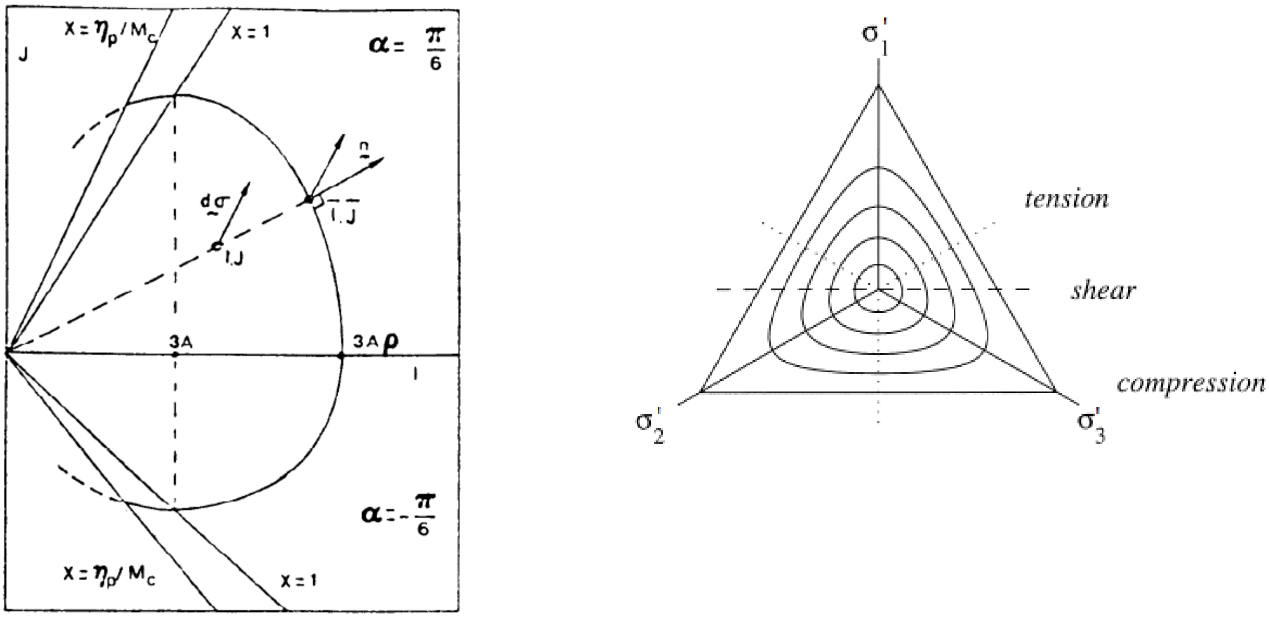


Fig. 2. left) an illustration of bounding surface, the radial mapping and image points in *I*-*J* space (Bardet, 1986); right) a smooth triangular contour of the surface in the deviatoric plane (Krenk 2000, LeBlanc et al. 2008).

In the model, the bounding surface is selected as a Cam-Clay type ellipse with $3A'$ as the coordinate of the ellipse summit on the I axis and ρ' as the aspect ratio in Fig. 2:

$$f(\tilde{I},\tilde{J},A) = \left(\frac{\frac{\tilde{I}}{3}-A'}{\rho'-1}\right)^2 + 3\left(\frac{\tilde{J}}{M(\theta)}\right)^2 - A'^2 = 0, \quad (14)$$

In Eq. (14), I is the first effective stress invariant, J is the second effective deviatoric stress invariant, the tilde ($\sim$) denotes the image stress point, and θ is the Lode's angle. The definitions of the stress invariants can be found in Appendix II. The generalization of the critical state slope *M* into the three-dimensional stress space is done by representing the deviatoric contour by a triangular shape function (Fig. 2. right):

$$M(\theta) = \frac{\cos(\gamma)M_c}{\cos\left\{\frac{1}{3}\arccos[\cos(3\gamma)\cos(3\theta)]\right\}}, \quad \gamma = \frac{\pi}{3} + \arctan\frac{1-2\frac{M_e}{M_c}}{\sqrt{3}}, \tag{15}$$

Where M_c and M_e are the critical state slope obtained from triaxial compression and extension tests, respectively.

The position of the ellipse summit is obtained similarly to the classic Cam Clay model:

$$A' = A'_0 exp\left(\frac{\Gamma - e - \kappa ln(\frac{I}{3})}{\lambda - \kappa}\right), \tag{16}$$

In above, $A'_0, \Gamma, \lambda, \kappa$ are respectively the unit pressure, critical void ratio at unit pressure, slopes of virgin loading and unloading-reloading line; and e is the current void ratio. The bounding surface may expand or shrink depending upon whether the plastic volumetric strain increases or decreases, and it must always enclose the current stress state.

The radial mapping technique generates the image stress point as the intersection of bounding surface with straight line through the origin and the current stress state. If considering an associated flow rule, it is then possible to obtain the following relationships:

$$m_{ij} = n_{ij} = \tilde{n}_{ij} = \frac{\frac{\partial f}{\partial \tilde{\sigma}'_{ij}}}{\sqrt{\frac{\partial f}{\partial \tilde{\sigma}'_{kl}} \frac{\partial f}{\partial \tilde{\sigma}'_{kl}}}}, \tag{17}$$

Notice that other considerations such as non-associated plastic potential are also possible.

3.3 Plastic modulus

The most important feature of this bounding surface model is an analytically prescribed function about the dependence of the plastic modulus H on H_b (the bounding surface plastic modulus):

$$H = H_b + S(\eta_p, \eta, M(\theta), I)\frac{\delta}{\delta_{max} - \delta}, \tag{18}$$

so that $H = +\infty$ when the distance between stress point and image point is larger than a reference distance δ_{max}, representing a purely elastic behavior; while $H = H_b$ when the two points coincide. The distance parameter δ is calculated from $\delta = \left|\tilde{\sigma}'_{ij} - \sigma'_{ij}\right|$ and the function S is an experimentally determined positive function depending on the mean stress and peak stress so as to simulate the strain-softening (Bardet 1986).

To determine the bounding surface plastic modulus H_b, two conditions are considered. Firstly, the consistency condition at the image points implies that:

$$df(\tilde{\sigma}', A') = \frac{\partial f}{\partial \tilde{\sigma}'_{ij}} d\tilde{\sigma}'_{ij} + \frac{\partial f}{\partial A'} dA' = 0, \tag{19}$$

Secondly, the stress increments at stress point and at image point on the bounding surface gives the same plastic strain increment, which leads to the expression:

$$\Delta\varepsilon^{p}_{ij} = \langle\frac{1}{H}(n_{kl}\Delta\sigma'_{kl})\rangle\, n_{ij} = \langle\frac{1}{H_b}(n_{kl}\Delta\tilde{\sigma}'_{kl})\rangle\, n_{ij}, \tag{20}$$

Recalling the volumetric hardening law for the surface evolution, and after some simple mathematical manipulations on Eq. (16), (17), (19) and (20), the following expression is obtained:

$$H_b = -\frac{vA'}{\lambda-k}\frac{\partial f}{\partial A'}\frac{1}{\frac{\partial f}{\partial\tilde{\sigma}'_{kl}}\frac{\partial f}{\partial\tilde{\sigma}'_{kl}}}\frac{\partial f}{\partial\tilde{\sigma}'_{ii}}, \tag{21}$$

In above, $v = 1 + e = 1/(1-n)$ is the current specific volume.

Up to now, the constitutive equations for the cyclic stress-strain relationships have been completed.

4. FV DISCRETIZATION & SOLUTION ALGORITHM

To obtain a numerical solution, an appropriate discretisation strategy is necessary. Conventionally, solid body problems have been tackled mainly by the finite element method (FEM); however, for this cyclic soil behavior analysis, the presence of the non-linear term – the tangential matrix D_{ijkl} complicates the system significantly that using FEM can be quite expensive. Alternatively, the present paper is attempting to apply the finite volume method (FVM) for the simulation, which is motivated by the appealing capacity of FVM in solving highly non-linear problems in computational fluid dynamics.

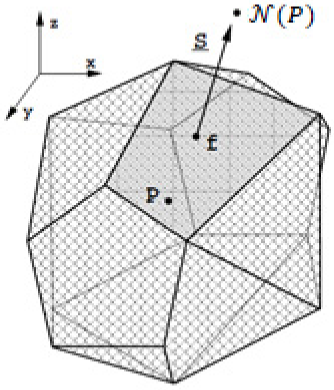


Fig. 3. Control volume: *P* is the centroid, *f* for the face centre, <u>S</u> for the face area vector, and $\mathcal{N}(P)$ for an assemblage of centroids in neighboring CVs (from Jasak & Weller) 2000).

Discretization of the computational domain consists of both temporal and spatial discretization: the time is discretized into an arbitrary number of time steps (δt), and the problem domain is subdivided into a finite number (*N*) of contiguous control volumes (CV), where the computational points lie in the centre (see illustration in Fig. 3 above).

The Eq. (6)-(7) can now be integrated over each of the N CV's giving the 4N equations

$$\int_{V_P} \underbrace{\left(D_{ijkl}\Delta u^s_{l,k}\right)_{,j}}_{S1^*} dV + \int_{V_P} \underbrace{\left(\alpha \Delta p_{,i}\right)}_{S2} dV - \int_{V_P} \underbrace{\left(\rho \Delta \ddot{u}^s_i\right)}_{T1+S3} dV + \int_{V_P} (\rho \Delta b_i) dV = 0, \qquad (22)$$

$$\int_{V_P} \underbrace{\left(\frac{n\gamma_w}{K_f k}\Delta \dot{p}\right)}_{T2+S3} dV + \int_{V_P} \underbrace{\left(\frac{\gamma_w}{k}\Delta \dot{u}^s_{i,i}\right)}_{T2+S2} dV - \int_{V_P} \underbrace{\left(\left(\Delta p_{,i}\right)_{,i}\right)}_{S1} dV + \int_{V_P} \underbrace{\left(\rho_f \Delta \ddot{u}^s_{i,i}\right)}_{T1+S2} dV -$$
$$\int_{V_P} \left(\rho_f \Delta b_{i,i}\right) dV = 0, \qquad (23)$$

The approximations of different derivative terms in above can be attained generally by applying Gauss theorem and some single time step schemes (Newmark 1959, Krenk 1999) summarized in Table 1:

Table. 1. Generalized approximation scheme

Spatial approximatio n	S1*	$\int_{V_P}\left(D_{ijkl}\Delta u^s_{l,k}\right)_{,j} dV = \oint_{S_P}\left(D_{ijkl}\Delta u^s_{l,k}\right) dS_j \cong \sum_f \left(D_{ijkl}\Delta u^s_{l,k}\right)^f S_j$	The nonlinear term $\left(D_{ijkl}\right)^f$ needs special attention.
	S1	$\int_{V_P} \left(\vartheta\left(\Delta\phi_{,i}\right)_{,i}\right) dV = \oint_{S_P}\left(\vartheta\Delta\phi_{,i}\right) dS_i \cong \sum_f \vartheta^f \left(\Delta\phi_{,i}\right)^f S^f_i$	$\Delta\phi =$ general variables $(\Delta u_i, \Delta p)$, $\vartheta =$ corresponding general coefficients, S_i =surface area vector $f =$ face $\beta_1, \beta_2 =$ scheme parameters in range: $\beta_1 \geq 1/2$ $\beta_2 \geq 1/2$ $\dot{\phi}^o, \ddot{\phi}^o$=old-time values.
	S2	$\int_{V_P} \left(\vartheta\Delta\phi_{,i}\right) dV = \oint_{S_P}(\vartheta\Delta\phi) dS_i \cong \sum_f \vartheta^f (\Delta\phi)^f S^f_i$	
	S3	$\int_{V_P}(\vartheta\Delta\phi) dV \cong \vartheta_P \Delta\phi_P V_P$	
Temporal approximatio n	T1	$\Delta\ddot{\phi} \cong \dfrac{\Delta\phi - \delta t\dot{\phi}^o - \frac{1}{2}(\delta t)^2\ddot{\phi}^o}{\beta_1(\delta t)^2}$	
	T2	$\Delta\dot{\phi} \cong \dfrac{\Delta\phi - \delta t\dot{\phi}^o}{\beta_2(\delta t)}$	

With an assumption of a linear variation of the variables over the CV, there are several existing approaches for approximating the increments of face gradient $\left(\Delta\phi_{,i}\right)^f$ and face value $(\Delta\phi)^f$ from the computing centroids ($\Delta\phi^P$) and neighboring centroids ($\Delta\phi^{\mathcal{N}(P)\cdot}$), available in the literature (Jasak & Weller 2000, Demirdzic & Martinovic 1993, Demirdzic & Muzaferija 1995). The biggest challenge for our problem lies in how to treat the nonlinear term D_{ijkl} , since it depends on the current stresses, hardening parameter, and displacements as well. It has been demonstrated that explicit treatment of D_{ijkl} tends to overestimate the stiffness and results in cumulative errors (Krenk 1993). Therefore, a convergent iterative process using Newton's procedure is necessary to correct the solutions in each time step.

At present, to discuss the solution algorithm alone, the assembled discretisation equations for the *P*P_{th} control volume are written:

$$a_{ij}^{PP}\Delta u_j^P + \sum_{Q\in\mathcal{N}(P)} a_{ij}^{PQ}\,\Delta u_j^Q + \underbrace{b_i^{PP}\Delta p^P + \sum_{Q\in\mathcal{N}(P)} b_i^{PQ}\,\Delta p^Q}_{pressure-displacement\ coupling} \ = c_i^P \tag{24}$$

$$d^{PP}\Delta p^P + \sum_{Q\in\mathcal{N}(P)} d^{PQ}\,\Delta p^Q + \underbrace{e_j^{PP}\Delta u_j^P + \sum_{Q\in\mathcal{N}(P)} e_j^{PQ}\,\Delta u_j^Q}_{displacement-pressure\ coupling} \ = f^P \tag{25}$$

Here a, b, d, e represent, respectively, the different forms of the discretized coefficients from Eq. 22 and 23; c, f stand for the source terms including the body force, old time values and boundary conditions; the superscript *Q* is another index like P to address the influences between cells, and the set of neighbor cells to P is denoted by $\mathcal{N}(P)$.

Conventionally, such an algebraic system is solved by applying the segregated method. This is an iterative solution scheme where one considers the first equation as an equation for the first component Δu_1, the second and third equations as equations for Δu_2 and Δu_3, respectively and the final equation as an equation for p. It treats the inter-component displacement coupling and the pressure coupling in Eq. (24) explicitly from guessed or previously iterated values, shifting them into source terms so as to solve each component of displacement increments successively. Afterwards, explicit displacement coupling is applied to attain the pressure increment in Eq. (25). The solution algorithm is illustrated in the figure below:

$$\begin{array}{cc} & \overbrace{\hphantom{A_1}}^{N} \\ N\left\{\vphantom{A_1}\right. & \end{array}\begin{pmatrix} A_1 & 0 & 0 & 0 \\ 0 & A_2 & 0 & 0 \\ 0 & 0 & A_3 & 0 \\ 0 & 0 & 0 & D \end{pmatrix} \cdot \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ P \end{pmatrix} = \begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ F \end{pmatrix}$$

Fig. 4. Segregated method, a variable-based sorted algorithm

Where,

$A_i = N \times N$ sparse matrix with a_{ii}^{PP} on the diagonal and a_{ii}^{PQ}on the off-diagonals;

$D = N \times N$ sparse matrix with d^{PP}on the diagonal, and d^{PQ}on the off-diagonals;

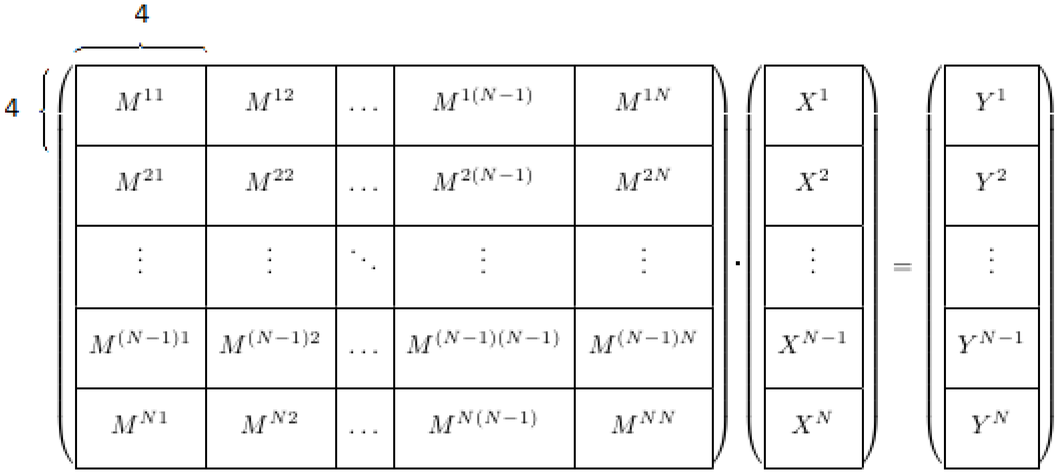
$U_i = [\Delta u_i^1, \Delta u_i^2, \cdots, \Delta u_i^N]^T$; $P = [\Delta p^1, \Delta p^2, \cdots, \Delta p^N]^T$;

$C_1 = c_1 + c_1^*(\Delta u_2^*, \Delta u_3^*, \Delta p^*)$, $C_2 = c_2 + c_2^*(\Delta u_1^*, \Delta u_3^*, \Delta p^*)$, $C_3 = c_3 + c_3^*(\Delta u_1^*, \Delta u_2^*, \Delta p^*)$;

$F = f + f^*(\Delta u_1^*, \Delta u_2^*, \Delta u_3^*)$.

It is noted that the number of off-diagonal non-zero elements in the *P_{th}* row of A_i and D equals the number of neighbors of the *P*$_{th}$ cell. The asterisk notation represents the additional source terms created from explicit treatment of couplings.

The segregated method is solving one variable for the whole domain once and then moves on to the next variables consecutively. A few iterations are needed to correct the solutions in each discretized time step. The method is economic since the large system has been split into four sub-systems for each variable and the sub-system is diagonally dominant, and thus well-suited for iterative solvers. However, the limitation of this method lies in that it can only be expected to converge for weakly coupled problems. For our strongly coupled system, this explicit treatment of coupling could cause very slow (or even no) convergence. Thus a fully implicit algorithm being able to solve all the displacement components and the pore pressures simultaneously would be more suitable. The newly developed block matrix solver (Clifford & Jasak, 2009, Kissling et al. 2010) is providing this kind of choice by treating all the coupling terms implicitly, see illustration in below:



$$\begin{pmatrix} M^{11} & M^{12} & \dots & M^{1(N-1)} & M^{1N} \\ M^{21} & M^{22} & \dots & M^{2(N-1)} & M^{2N} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ M^{(N-1)1} & M^{(N-1)2} & \dots & M^{(N-1)(N-1)} & M^{(N-1)N} \\ M^{N1} & M^{N2} & \dots & M^{N(N-1)} & M^{NN} \end{pmatrix} \cdot \begin{pmatrix} X^1 \\ X^2 \\ \vdots \\ X^{N-1} \\ X^N \end{pmatrix} = \begin{pmatrix} Y^1 \\ Y^2 \\ \vdots \\ Y^{N-1} \\ Y^N \end{pmatrix}$$

Fig. 5. Block matrix solver, a control volume-based sorted algorithm

In Fig. 5, the matrix consists of N^2 blocks of 4×4 local matrix (M), whose definition is:

$$M^{PQ} = \begin{bmatrix} a_{11}^{PQ} & a_{12}^{PQ} & a_{13}^{PQ} & b_1^{PQ} \\ a_{21}^{PQ} & a_{22}^{PQ} & a_{23}^{PQ} & b_2^{PQ} \\ a_{31}^{PQ} & a_{32}^{PQ} & a_{33}^{PQ} & b_3^{PQ} \\ e_1^{PQ} & e_2^{PQ} & e_3^{PQ} & d^{PQ} \end{bmatrix}$$

All the unknown variables are arranged cell by cell, with the same strategy for the source terms, shown below:

$$X^P = \begin{bmatrix} \Delta u_1^P \\ \Delta u_2^P \\ \Delta u_3^P \\ \Delta p^P \end{bmatrix}, \qquad Y^P = \begin{bmatrix} c_1^P \\ c_2^P \\ c_3^P \\ f^P \end{bmatrix}$$

Note that when the cell Q is not the neighbor of P, M^{PQ} is simply a zero matrix. Thus, the sparseness pattern of the matrix in Fig. 5 is the same as for A_i and D in the segregated method shown in Fig.4, with nonzero entries only where the index pair corresponds to neighbor cells. However, this block matrix system is non-symmetrical and has much more non-zero entries compared to the system of segregated method, which would then require more computational efforts.

In general, whether to choose the segregated method or the block matrix solver method is a question of trade-off between solving a reduced cheaper system repeatedly with slow convergence, and solving a much larger system with all the coupling relations preserved.

CONCLUSION

This paper proposes the fundamental formulations and methodology of developing a numerical code modeling the soil liquefaction and cyclic mobility phenomena in the marine environment. The strong interactions between soil skeleton displacements and the pore pressure variations are mathematically modeled by a set of coupled partial differential equations. A constitutive model that appropriately describes the cyclic soil stress-strain relationships is crucial for achieving the accuracy of a numerical solution. A bounding surface model has been selected and presented in detail. In order to obtain reliable numerical simulations, the finite volume method is proposed for the discretization strategy. Different solution algorithms are discussed for solving the resulted algebraic system. Generally, it is expected that this coupled soil-pore fluid formulation incorporating the strong interactions in the porous seabed can gain more accuracy in modeling soil liquefaction and cyclic mobility compared with conventionally uncoupled approaches.

APPENDIX I. Comparisons on three constitutive models for cyclic plasticity

	Multi-surface plasticity models (*Prevost 1985*, *Elgamal et al. 2003*, *Yang & Elgamal 2008*)	**Bounding surface plasticity models**	
		The critical state two-surface model (*Manzari & Dafalias 1997*)	**The single bounding surface model** (*Bardet 1986*)
Storage requirement	A field of nested yield surfaces(location, size, plastic modulus)	Yield surface + critical surface+ bounding surface+ dilatancy surface (prescribed surface evolutions)	Bounding surface (contraction/expansion)
Model constants (with unclear physical meanings)	None (optional constants to achieve more accurate dilatancy if required)	Surface quantities: k^b & k^d Kinematic hardening constant: h_0 Dilatancy constant: A	Ellipse aspect ratio (dilatancy): ρ Plastic modulus constant: h_0
Mapping rule	None	Radial mapping	
		Three image back-stress ratio points on the dilatancy, critical, and bounding surface.	A single image stress point on the bounding surface
Adequacy in modeling smoothly varied stiffness upon loading and the dramatic changes in stiffness under shear reversal	-Segments of linear stress-strain behavior; Only if the more segments are divided, smooth transitions from elastic to plastic domain can be approximated. - Stiff (elastic) behavior immediately after shear reversal, followed by decreasing stiffness in further unloading.	-Prescribed smooth variation of the stiffness upon loading by measuring the distance from current stress state to the bounding surface. - Discontinuously shifted 'image' points under shear reversal.	
Accuracy in estimating dilatancy (*D*)	$D = 0$ only with $\eta = \eta_{PT}$ $D > 0$ at $\eta < \eta_{PT}$ or $\dot{\eta} < 0$ $D < 0$ at $\eta > \eta_{PT}$ Independent on material state	$D = 0$ with $\eta = \eta_{PT}$ or $\eta = M_{c(e)}$ The sign of *D* depends directly on whether stress point within or without the dilatancy surface	$D = 0$ with $\eta = \eta_{PT} = M_{c(e)}$ Strong dilatancy defined by small aspect ratio ρ.
Capability in simulating strain softening	Without apparent softening	Softening after peak stress ratio	Bounding surface contracts after dilation, while softening of the loading occurs after the peak stress ratio.
Potential for model extension (simulating the history-dependent soils, e.g. clay)	unclear	Incorporating an extra history surface so that plastic modulus: $K_p \propto \{d_1$(current stress point → image point on history surface) + d_2(image point on history surface → image point on bounding surface)$.\}$	

APPENDIX II.

The definitions of stress invariant$s$:

$$I = \sigma'_{kk} \tag{A.1a}$$

$$J = \left(\frac{1}{2}s_{ij}s_{ij}\right)^{1/2} \tag{A.1b}$$

Where, the deviatoric stress s_{ij} is calculated by:

$$s_{ij} = \sigma'_{ij} - \frac{1}{3}\sigma'_{kk}\delta_{ij} \tag{A.2}$$

The definition of Lode's angle:

$$\theta = \frac{1}{3}sin^{-1}\left(\frac{3\sqrt{3}}{2}\frac{S^3}{J^3}\right) \tag{A.3}$$

In which, the third deviator stress invariant:

$$S = \left(\frac{1}{3}s_{ij}s_{jk}s_{ki}\right)^{1/3} \tag{A.4}$$

The stress invariants at image point based on radial mapping:

$$\tilde{I} = 3\gamma^* A' \tag{A.5a}$$

$$\tilde{J} = 3\gamma^* \eta^* A' \tag{A.5b}$$

Where η^* is the generalized stress ratio:

$$\eta^* = \frac{3\sqrt{3}J}{I} \tag{A.6}$$

The scalar γ^* can be calculated by substituting Eq.A5 and A6 into Eq.27:

$$\gamma^* = \frac{M(\theta)^2 I^2 + M(\theta)I(\rho' - 1)\sqrt{M(\theta)^2 I^2 + \rho'(\rho' - 2)J^2}}{M(\theta)^2 I^2 + 27(\rho' - 1)^2 J^2} \tag{A.7}$$

REFERENCE

Bardet, J.P., (1986). "Bounding surface plasticity model for sands", *J. Eng. Mech*, Vol.**112**(11), November.

Biot, M.A., (1941). "General theory of three dimensional consolidation"*, J. appl. Phys*, Vol.**12**, 155-164.

Clifford,. I., Jasak, H., (2009). "The application of a multi-physics toolkit to spatial reactor dynamics", *Proceedings of International Conference on Mathematics, Computational Methods & Reactor Physics (M&C 2009)*, New York.

Crouch, R.S., Wolf J.P., Dafalias Y.F., (1994). "Unified critical-state bounding surface plasticity model for soil"*, J. Eng. Mech*, Vol.**120**(11), 2251-2270.

Dafalias, Y.F., and Popov, E.P., (1975). "A model of nonlinearly hardening materials for complex loading", *Acta. Mech.,* Vol.**21**, 173-192.

Demirdzic, I., and Martinovic, D., (1993). "Finite volume method for thermo-elasto-plastic stress analysis", *Comput. Methods Appl. Mech. Engrg.,* **109**, 331-349.

Demirdzic, I., and Muzaferija, S., (1995). "Numerical method for coupled fluid flow, heat transfer and stress analysis using unstructured moving meshes with cells of arbitrary topology", *Comput. Methods Appl. Mech. Engrg.,* **125**, 235-255.

Di, Y., and Sato, T., (2004). "A practical numerical method for large strain liquefaction analysis of saturated soils", *Soil Dynamics and Earthquake Engineering*, **24**, 251-260.

Elgamal, A., Yang, Z.H., Parra, E., (2002). "Computational modeling of cyclic mobility and post-liquefaction site response", *Soil Dynamics and Earthquake Engineering*, **22**, 259-271.

Elgamal, A., Yang, Z.H., Parra, E., and Ragheb, A., (2003). "Modeling of cyclic mobility in saturated cohesionless soils", *Int. J. Plasticity*., **19**, 883-905.

Huang, M.S., and Zienkiewicz, O.C., (1998). "New unconditionally stable staggered solution procedures for coupled soil-pore fluid dynamic problems", *Int. J. Numer. Engng*. **43**, 1029-1052.

Jasak, H., and Weller, H.G., (2000). "Application of the finite volume method and unstructured meshes to linear elasticity", *Int. J. Numer. Engng.*, **48**, 267-287.

Jeremic, B., Cheng, Z., Taiebat, M., and Dafalias, Y.F., (2008). "Numerical simulation of fully saturated porous materials", *Int. J. Numer. Anal. Mech. Geomech*., **32**, 1635-1660.

Kissling, K., Springer, J., Jasak, H., Schutz, S., Urban, K., and Piesche, M., (2010). "A coupled pressure based solution algorithm based on the volume-of-fluid approach for two or more immiscible fluids", *V European Conference on Computational Fluid Dynamics*, Lisbon.

Krenk, S., (1993). "Non-linear analysis with finite elements", Alborg University, Denmark.

Krenk, S., (1999). "Dynamic analysis of structures – numerical time integration", *Course on Non-Linear Analysis of Frame Structures,* RAMBØLL, Denmark.

Li, X.S., Dafalias, Y.F., and Wang, Z.L., (1999). "State-dependent dilatancy in critical-state constitutive modeling of sand", *Can. Geotech. J.*, **36**, 599-611.

LeBlanc, C., Hededal, O., Ibsen L.B., (2008). "A modified critical state two-surface plasticity model for sand – theory and implementation strategy"*, DCE Technical Memorandum No.8.*

Manzari, M.T., and Dafalias, Y.F., (1997). "A critical state two-surface plasticity model for sands", *Geotechnique* 47, No. 2, 255-272.

Mroz, Z., (1967). "On the description of anisotropic hardening", *J. Mech. Phys. Solids*., Vol. **15**, 163-175.

Newmark, N.M., (1959). “A method of computation for structural dynamics”, *Proc. Am. Soc. Civ. Engrs*, **8**, 67-94.

Oka, F., Yashima, A., Shibata, T., Kato, M., and Uzuoka, R., (1994). “FEM-FDM coupled liquefaction analysis of a porous soil using an elasto-plastic model”, *App. Sci. Research*, 52, 209-245.

Pastor, M., Li, T., Liu, X., Zienkiewicz, O.C., and Quecedo, M., (2000). “A fractional step algorithm allowing equal order of interpolation for coupled analsis of saturated soil problems”, *Mech. Cohes.-Frict. Mater.*, **5**, 511-534.

Prevost, J.H., (1985). “A simple plasticity theory for frictional cohesionless soils”, *Soil Dynamics and Earthquake Engineering*, Vol.**4**, No.1.

Taiebat, M., Shahir, h., and Pak, A., (2007). “Study of pore pressure variation during liquefaction using two constitutive models for sand”, *Soil Dynamics and Earthquake Engineering*, **27**, 60-72.

Wang, H.F., (2000). “Theory of Linear Poroelasticity – with Applications to Geomechanics and Hydrogelogy”*, Princeton University Press*.

Wood, D.M., (1990). “Soil behavior and critical state soil mechanics”*, Cambridge University Press*.

Yang, Z.H., and Elgamal, A., (2008). “Muiti-surface cyclic plasticity sand model with Lode angle effect”, *Geotech. Geol. Eng.*, **26**, 335-348.

Zienkiewicz, O.C., (1982). “Basic formulation of static and dynamic behavior of soil and other porous material”, *Appl. Math. Mech*, Vol.**3**, No.4, Aug.

Zienkiewicz, O.C., Chan, A.H.C., Pastor, M., Paul, D.K., and Shiomi, T., (1990). “Static and dynamic behavior of soils: a rational approach to quantitative solutions. I. Fully saturated problems”*, Proc. R. Soc. Lond.*, A **429**, 285-309.

Zienkiewicz, O.C., Huang, M.S., Wu, J., and Wu, S.M., (1993). “A new algorithm for the coupled soil-pore fluid problem”, *Shock and Vibration*, Vol.**1**, No.1, 3-14.

Zienkiewicz, O.C., and Shiomi, T., (1984). “Dynamic behavior of saturated porous media; The generalized Biot formulation and its numerical solution”, *Int. J. Numer. Anal. Methods Geomech.,* Vol.**8,** 71-96.

Paper II

"On finite volume method implementation of poro-elasto-plasticity soil model"

T. Tang, O. Hededal & P. Cardiff

Submitted for publication in: *International Journal for Numerical and Analytical Methods in Geomechanics, 2014*

On finite volume method implementation of poro-elasto-plasticity soil model

Tian Tang[1,*,†,‡], Ole Hededal[1,§], and Philip Cardiff[2,¶]

[1]Civil Engineering Department,Technical University of Denmark, 2800 Kongens Lyngby, Denmark
[2]School of Mechanical and Materials Engineering, University College Dublin, Belfield, Ireland

Summary

Accurate prediction of the interactions between the nonlinear soil skeleton and the pore fluid under loading plays a vital role in many geotechnical applications. It is therefore important to develop a numerical method that can effectively capture this nonlinear soil-pore fluid coupling effect. This paper presents the implementation of a new finite volume method code of poro-elasto-plasticity soil model. The model is formulated on the basis of Biot's consolidation theory and combined with a perfect plasticity Mohr-Coulomb constitutive relation. The governing equation system is discretized in a segregated manner, namely, those conventional linear and uncoupled terms are treated implicitly, while those nonlinear and coupled terms are treated explicitly by using any available values from previous time or iteration step. The *Implicit-Explicit* descretization leads to a linearized and decoupled algebraic system, which is solved using the fixed-point iteration method. Upon the convergence of the iterative method, fully nonlinear coupled solutions are obtained. Also explored in this paper is the special way of treating traction boundary in FVM compared to FEM. Finally, three numerical test cases are simulated to verify the implementation procedure. It is shown in the simulation results that the implemented solver is capable of and efficient at predicting reasonable soil responses with pore pressure coupling under different loading situations.

KEY WORDS: finite volume method; poro-elasto-plasticity; coupled analysis; soil model; segregated solution procedure; OpenFOAM

1 Introduction

It is commonly recognized that the mechanical behavior of soils (and indeed other saturated geo-materials) is governed largely by the interaction of their solid skeleton with the fluid present in the pore structures [1, 2]. The fundamental mathematical framework describing the coupled effects in porous soil was first established by Biot in 1941, who - by assuming a linear elastic behavior of the soil skeleton and a Darcian fluid flow - formulated a coupled model with the soil skeleton displacements and the pore fluid pressure as the primary unknowns [3]. Extensive research works have in the past few decades been devoted to analytical or numerical solutions of this linear poro-elasticity theory due to its simplicity and broad applicability in different engineering problems [4]. However, increasing focus on more complex applications has triggered research aimed at extending

[*]Correspondence to: Tian Tang, Civil Engineering Department,Technical University of Denmark, 2800 Kongens Lyngby, Denmark
[†]E-mail: tiat@byg.dtu.dk
[‡]Phd Student
[§]Associate Professor
[¶]Post Doctoral Research Fellow

the original Biot theory to account for nonlinear material behaviors, i.e. the development of poro-elasto-plasticity models [5]. Often, due to the complexity of those models (from the coupling and highly nonlinear constitutive relation), analytical solutions are very difficult to obtain, and the numerical approaches therefore play an active role in nonlinear coupled analyses, such as in dynamic earthquake problems [6–8] and in foundation bearing capacity problems [9–11].

The aforementioned numerical poro-elasto-plasticity models adopted the traditional approach of continuum solid mechanics, i.e. finite element method (FEM) technique. An alternative numerical scheme is the finite volume method (FVM), which was originally applied to fluid mechanical problems. Demirdzic and co-workers [12–18] established FVM as noteworthy alternative to the FEM for linear and nonlinear continuum solid mechanics. It is pointed out that the driving force behind the development of segregated FVM stress analysis algorithms is the potential for dealing with nonlinear problems with only a marginal increase in computational cost [19]. Hence, it is interesting to explore whether FVM is also capable of and efficient at treating poro-elasto-plasticity models containing both material nonlinearity and strong soil-fliud coupling effects.

This article provides background information and documentation for a new finite volume implementation of a poro-elasto-plasticity soil model. The constitutive relation employed is a linear elastic/plastic model based on a classical, non-associated Mohr-Coulomb formulation. The selected stress integration scheme is an explicit return mapping algorithm formulated in principal stress space by Clausen et al. [20]. Emphasis is given to the introduction of the *Implicit-Explicit* split methodology, i.e. the segregated method in FVM, to deal with the nonlinear and coupling effects in the model. As a consequence of the special implicit and explicit treatment, the complicated nonlinear coupled equations can be linearized and solved sequentially using a fixed-point type iterative method, which usually provides a linear convergence. Additionally, the treatment of traction boundaries in the current FVM implementation differs much from that in FEM and thus is also given special attention in this work.

The numerical procedure has been implemented as a custom solver in OpenFOAM (version 1.6-ext), which is a free-to-use open source numerical software that has extensive CFD and multi-physics capabilities [21]. This implementation work therefore also opens a possibility of using the same numerical methodology, namely FVM, for solving the wave-seabed-structure interaction problems in the future.

The structure of this article is as follows: In Section 2, the mathematical (poro-elasto-plasticity) model for nonlinear porous soil is presented, and the key issues in modeling poro-elasto-plasticity as well as the corresponding solution strategies are proposed. In Section 3, the FVM discretisation techniques including the discretisation of the solution domain and the discretisation of equations are described in details. Section 4 outlines the global iterative solution procedure and the local return mapping stress update algorithm. A brief introduction of the relaxation method applied for the improvement of convergence is also included. Finally, several test cases for which comparable analytical solution or FEM simulation results exist are presented in Section 5 in order to assess the capability and efficiency of the the implemented code.

2 Mathematical model

The mathematical formulation of the poro-elasto-plasticity model employed in this article is an extension of the original Biot's consolidation equations [3] to account for plasticity. The derivation and how the equations are manipulated to fit FVM will be briefly explained in the following subsections.

2.1 Governing equations

The behavior of the soil-pore fluid(mainly water) mixture is governed by two types of differential equations: the total momentum balance of the soil and the conservation of the flow of water in the pores.

The momentum equation is formulated for static condition and without body force:

$$\nabla \cdot \boldsymbol{\sigma} = \nabla \cdot (\boldsymbol{\sigma}' - p\mathbf{I}) = \mathbf{0} \tag{1}$$

where $\nabla\cdot$ symbolizes the divergence operator, $\boldsymbol{\sigma}$ the total stress tensor, $\boldsymbol{\sigma}'$ the 'effective' stress tensor, p the pore fluid pressure, and $\mathbf{I}$ the identity tensor. Here, tension is assumed positive as normally employed in computational continuum mechanics. And no distinction is made between the initial and deformed configurations.

The seepage equation is expressed as:

$$\frac{k}{\gamma_w}\nabla^2 p = \frac{n}{K'}\frac{\partial p}{\partial t} + \frac{\partial}{\partial t}(\nabla \cdot \mathbf{u}) \tag{2}$$

where ∇^2 is the laplacian operator, k is the permeability, γ_w is the density of water, n is the porosity and $\mathbf{u}$ is the soil skeleton displacement vector. The effective bulk modulus of the pore fluid, K', may be estimated assuming that air present in the pores remains close to atmospheric pressure:

$$\frac{1}{K'} = \frac{Sr}{K_w} + \frac{1-Sr}{p_a} \tag{3}$$

Here, Sr is the degree of saturation, K_w the pure water bulk modulus ($\approx 2 \times 10^9 Pa$), and p_a the atmospheric pressure ($\approx 10^5 Pa$).

2.2 Constitutive relation

The elasto-plastic constitutive relation of the soil skeleton is written in incremental form, applying small-strain assumption and linear isotropic elasticity:

$$d\boldsymbol{\sigma}' = 2\mu\,(d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}^p) + \lambda \mathrm{tr}\,(d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}^p) \tag{4}$$

$$d\boldsymbol{\varepsilon} = \frac{1}{2}[\nabla(d\mathbf{u}) + \nabla(d\mathbf{u})^{\mathrm{T}}] \tag{5}$$

where $d\boldsymbol{\varepsilon}$ is the total incremental strain, $d\boldsymbol{\varepsilon}^p$ the plastic incremental strain, $d\mathbf{u}$ the incremental displacement, and ∇ the gradient operator. The material constants μ and λ are the Lamé coefficients, which can be defined by the more commonly used Young's modulus E and Poisson's ratio ν [13,22]. It is notable that the strain spilt - in elastic and plastic strains - though simple, is essential for the *segregated* nonlinear stress analysis in FVM.

The incremental plastic strain $d\boldsymbol{\varepsilon}^p$ can further be defined from classic plasticity theory as:

$$d\boldsymbol{\varepsilon}^p = \langle d\Lambda \rangle \frac{\partial g}{\partial \boldsymbol{\sigma}'}, \quad \langle d\Lambda \rangle = \begin{cases} 0, & \text{when pre-yielding } f < 0. \\ d\Lambda, & \text{when yielding } f = 0. \end{cases} \tag{6}$$

where the gradient to the plastic potential, $\partial g/\partial \boldsymbol{\sigma}$, defines the direction of the plastic strain increment, and $d\Lambda$ is the plastic multiplier defining the magnitude. For stress states on the yield surface the loading-unloading condition is given by:

$$\frac{\partial f}{\partial \boldsymbol{\sigma}'} : d\boldsymbol{\sigma}' \quad \begin{cases} < 0 & \text{elastic unloading.} \\ \geq 0 & \text{plastic loading.} \end{cases} \tag{7}$$

in which $\partial f/\partial \boldsymbol{\sigma}'$ is an outward normal to the yield surface and the symbol : represents tensor contraction operation. In this work, we adopt a non-associated Mohr-Coulomb perfect plasticity model and employ a consistent return mapping stress integration procedure proposed by Clausen et al. [20]:

$$f = (\sigma_1' - \sigma_3') + (\sigma_1' + \sigma_3')\sin\varphi - 2c\cos\varphi \tag{8}$$

$$g = (\sigma_1' - \sigma_3') + (\sigma_1' + \sigma_3')\sin\psi \tag{9}$$

$$d\Lambda = \frac{f(\boldsymbol{\sigma}_{tl}')}{H + (\partial f/\partial \boldsymbol{\sigma}') : \mathbf{C} : (\partial g/\partial \boldsymbol{\sigma}')} \tag{10}$$

where f is the yield function and g the plastic potential. The return mapping algorithm is formulated in terms of principal stresses, σ'_1, σ'_3, which are easily obtained as the maximum and minimum eigenvalue of the general effective stress tensor $\boldsymbol{\sigma}'$. The soil properties φ, c, ψ are the friction angle, cohesion and dilation angle, respectively. For determining the plastic multiplier, $f(\boldsymbol{\sigma}'_{tl})$ is the yield function evaluated at the trial elastic stress, and $\mathbf{C}$ is the elastic stiffness tensor fully defined by μ and λ. Since we are considering a non-hardening model the plastic modulus $H = 0$.

Other plasticity models, if needed, can also be integrated into Equation (6) as long as an efficient stress update procedure has been carefully selected.

2.3 Poro-elasto-plasticity model

By substituting Equations (4) and (5) into Equation (1) and rearranging the linear terms on the left side and the non-linear or coupling terms on the right side, the completed poro-elasto-plasticity model is achieved and consists of the following equations:

- Displacement equation

$$\nabla \cdot \left[\mu \nabla(d\mathbf{u}) + \mu \nabla(d\mathbf{u})^{\mathrm{T}} + \lambda \mathbf{I} \mathrm{tr}(\nabla(d\mathbf{u}))\right] = \underbrace{\nabla \cdot \left[2\mu(d\boldsymbol{\varepsilon}^p) + \lambda \mathbf{I} \mathrm{tr}\left(d\boldsymbol{\varepsilon}^p\right)\right]}_{\text{nonlinearity}} + \underbrace{\nabla(dp)}_{\text{coupling}} \tag{11}$$

- Pore pressure equation

$$\frac{k}{\gamma}\nabla^2 p - \frac{n}{K'}\frac{\partial p}{\partial t} = \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{\text{coupling}} \tag{12}$$

As shown in Equations (11) and (12), the poro-elasto-plasticity soil model has two important features: the nonlinearity, that is the plastic strain term $d\boldsymbol{\varepsilon}^p$ depends on the unknown displacement and stress level (from Equation (6)); and the strong coupling, i.e. having pore pressure gradient term present in the displacement equation and velocity divergence term present in the pore pressure equation. In order to deal with these two issues, an *Implicit-Explicit* split strategy from the FVM segregated method will be used: namely, the conventional linear, uncoupled terms categorized on the left hand side of the equations will be treated implicitly, while the nonlinear, coupled terms on the right hand side will be treated explicitly by using any available values from previous time or iteration step. This splitting of the nonlinearity and coupling in general guarantees the linear convergence of the employed global iterative solution procedure, i.e. the Fixed Point Iteration method (more details can be found in the Appendix A). In general, the iterations are performed over the systems of equations until all the explicit terms have converged.

2.4 Initial and boundary conditions

The poro-elasto-plasticity shall be further complemented with the definition of the initial and boundary conditions. The initial conditions consist of the displacements and pore pressure in the whole solution domain at the initial instant of time. The boundary conditions can be specified as either constant or time varying of the types summarized in Table 1.

Table 1: Boundary conditions (BC)

$d\mathbf{u}$:	- prescribed displacement - prescribed traction - fixedDisplZeroShear (zero value on normal direction, zero traction on tangent directions) - plane of symmetry
p :	- prescribed pressure - zero gradient (impermeable) - plane of symmetry

It should be mentioned that the current implementation of traction boundaries in FVM is not as trivial as that in FEM. In FEM, the traction is straightforwardly accounted for as an external force contribution into the source term. Whereas in our implementation, we must ensure fulfillment of the momentum balance on boundary surfaces and it thus involves some computations in solving the Cauchy's stress theorem equation on the boundary:

$$\begin{aligned} d\mathbf{t} &= \mathbf{n}\cdot(d\boldsymbol{\sigma}) \\ &= \mathbf{n}\cdot(d\boldsymbol{\sigma}' - dp\mathbf{I}) \\ &= \mu\mathbf{n}\cdot[2\mu d\boldsymbol{\varepsilon} + \lambda \mathrm{tr} d\boldsymbol{\varepsilon} - \underbrace{(2\mu d\boldsymbol{\varepsilon}^p + \lambda \mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)) - dp\mathbf{I}}_{\text{nonlinearity and coupling on BC}}] \end{aligned} \tag{13}$$

where $\mathbf{t}$ stands for the traction force vector acting on the boundary, $\boldsymbol{\sigma}$ is the total stress in the soil and $\mathbf{n}$ is the surface normal vector of the boundary.

It is clear from Equation (13) that the same nonlinearity and coupling effects exist in the traction boundary condition as the inner solution domain. It is therefore vital to apply the same *Implicit-Explicit* split strategy and iteration method on the traction boundary as for the inner domain, which will be described later.

3 Finite volume discretization

In this section, cell-centred FVM technique is applied to discretize the poro-elasto-plasticity model derived in the previous section. FVM discretization is often second order accurate by assuming a linear variation of the variables over the control volume (CV). The standard discretization procedure consists of two parts: the discretization of solution domain and discretization of equations.

3.1 Discretization of solution domain

Discretization of the solution domain consists of discretization of the time interval and discretization of space. The time interval is split into a finite number of time steps Δt and the equations are then solved in a time-marching manner using first-order accurate implicit method. The computational space domain is subdivided into a finite number of convex polyhedral CVs which are bounded by polygonal faces. Figure 1 illustrates a typical control volume V_P with the computational point P located in its centroid, the face f, the face area S_f, the face unit normal vector $\mathbf{n}_f$, the centroid N of the neighboring CV sharing the face f, the distance vector $\mathbf{d}_f$ joining P and N, and the skewness vector $\mathbf{m}_f$ pointing from the intersection point between $\mathbf{d}_f$ and the face f to the face center [23].

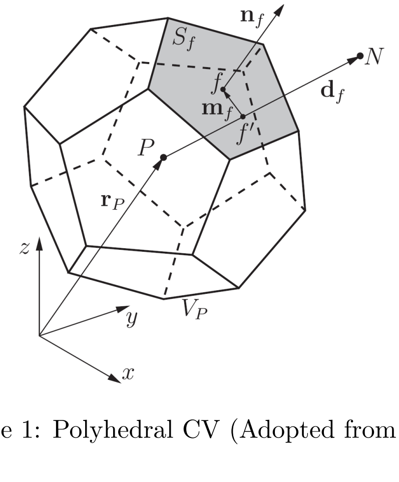


Figure 1: Polyhedral CV (Adopted from [23])

Notice, if the vectors $\mathbf{n}_f$ and $\mathbf{d}_f$ are not parallel, the mesh is denoted as non-orthogonal. Furthermore, if the vector $\mathbf{m}_f$ is non-zero, the mesh is skewed. Both features have a large effect on the accuracy of the FVM technique, and thus requires special corrections in the discretization procedure [24]. Since the soil geometry is often simple and has regular mesh, non-orthogonal and skewness corrections will not be discussed further.

3.2 Discretization of equations

For each time step, the discretization of equations uses the integral form of Equations (11) and (12) over V_P and then applies the Gauss' theorem to convert the volume integrals into surface integrals:

$$\begin{aligned}\overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot[(2\mu+\lambda)\nabla(d\mathbf{u})]}^{\text{implicit}} = & -\overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot\left\{\mu\nabla(d\mathbf{u})^{\mathrm{T}}+\lambda\mathbf{I}\mathrm{tr}[\nabla(d\mathbf{u})]-(\mu+\lambda)\nabla(d\mathbf{u})\right\}}^{\text{inter-component coupling, explicit}} \\ & + \overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot[2\mu(d\boldsymbol{\varepsilon}^p)+\lambda\mathbf{I}\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)]}^{\text{nonlinearity, explicit}} \\ & + \overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot(dp)}^{\text{pressure coupling, explicit}}\end{aligned} \tag{14}$$

$$\overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot(\frac{k}{\gamma_w}\nabla p)-\int_{V_P}\frac{n}{K'}\frac{\partial p}{\partial t}dV}^{\text{implicit}} = \overbrace{\oint_{\partial V_P} d\mathbf{s}\cdot\left(\frac{\partial\mathbf{u}}{\partial t}\right)}^{\text{displacement coupling, explicit}} \tag{15}$$

The integral equations have been split into *Implicit* and *Explicit* discretization parts in order to allow the system being solved in the *segregated* manner as mentioned in Section 2.3. Iterations will be applied until the convergence of the whole system is attained. It is worth mentioning that besides from the nonlinear and pressure-displacement coupling terms, an extra inter-component coupling part is also treated explicitly to promote faster convergence, see Jasak & Weller [22].

Let us now describe the implicit and explicit discretization parts on a term-by-term basis.

3.2.1 Implicit discretization

The implicit surface diffusion term (laplacian terms) from the momentum equation is discretized as:

$$\oint_{\partial V_P} d\mathbf{s}\cdot[(2\mu+\lambda)\nabla(d\mathbf{u})] = \sum_{f=1}^{F}(2\mu_f+\lambda_f)\mathbf{n}_f\cdot[\nabla(d\mathbf{u})]_f S_f \tag{16}$$

where F is for the number of faces of V_P. By assuming a linear variation across face f, the face-center gradient $[\nabla(d\mathbf{u})]_f$ is evaluated as:

$$\mathbf{n}_f\cdot[\nabla(d\mathbf{u})]_f = |\mathbf{n}_f|\frac{(d\mathbf{u})_N-(d\mathbf{u})_P}{|\mathbf{d}_f|} \tag{17}$$

Similarly, the pore pressure diffusion term is discretized as follows:

$$\oint_{\partial V_P} d\mathbf{s}\cdot(\frac{k}{\gamma_w}\nabla p) = \sum_{f=1}^{F}\frac{k_f}{\gamma_{w\,f}}\mathbf{n}_f\cdot(\nabla p)_f S_f = \sum_{f=1}^{F}\frac{k_f}{\gamma_{w\,f}}\left(|\mathbf{n}_f|\frac{p_N-p_P}{|\mathbf{d}_f|}\right)S_f \tag{18}$$

For constant soil properties, μ_f, λ_f, k_f and $\gamma_{w\,f}$ are simply equal to μ ,λ, k and γ_w.

The volume integral of the time derivative of p is calculated using the mid-point rule and a first order implicit Euler method:

$$\int_{V_P}\frac{n}{K'}\frac{\partial p}{\partial t}dV = \int_{V_P}\frac{n}{K'}\frac{p-p^o}{\Delta t}dV = \frac{n}{K'}\frac{p_P-p_P^o}{\Delta t}V_P \tag{19}$$

where the upper index o is representing the old-time step value.

3.2.2 Explicit discretization

The explicit surface diffusion terms (the inter-component coupling terms) in Equation (14) are firstly approximated the same way as in Equation (16):

$$\begin{aligned}&\oint_{\partial V_P} d\mathbf{s}\cdot\left\{\mu\nabla(d\mathbf{u})^{\mathrm{T}}+\lambda\mathbf{I}\mathrm{tr}[\nabla(d\mathbf{u})]-(\mu+\lambda)\nabla(d\mathbf{u})\right\}\\&=\sum_{f=1}^{F}\mathbf{n}_f\cdot\left\{\mu_f\left[\nabla(d\mathbf{u})^{\mathrm{T}}\right]_f+\lambda_f\mathbf{I}\mathrm{tr}\left[\nabla(d\mathbf{u})\right]_f-(\mu_f+\lambda_f)[\nabla(d\mathbf{u})]_f\right\}S_f\end{aligned}\tag{20}$$

But, unlike the implicit discretization, the face-center gradient $[\nabla(d\mathbf{u})]_f$ will now be calculated from linear interpolation of the cell center gradients (available from previous iteration) as follows:

$$[\nabla(d\mathbf{u})]_f=f_x[\nabla(d\mathbf{u})]_P+(1-f_x)[\nabla(d\mathbf{u})]_N\tag{21}$$

where $f_x=|\mathbf{n}_f|/|\mathbf{d}_f|$ is the interpolation factor, and the cell center gradient values, i.e. $[\nabla(d\mathbf{u})]_P$ and $[\nabla(d\mathbf{u})]_N$ can be evaluated using the least square fit approach based on the available distribution of $d\mathbf{u}$ [22].

The explicit gradient terms are related to the nonlinearity and pressure coupling terms in Equation (14). They will be discretized assuming linear variation of the values across the face (i.e. linear interpolation approach):

$$\begin{aligned}\oint_{\partial V_P} d\mathbf{s}\cdot\left[2\mu(d\boldsymbol{\varepsilon}^p)+\lambda\mathbf{I}\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)\right]=&\sum_{f=1}^{F}\mathbf{n}_f\cdot\left[2\mu_f(d\boldsymbol{\varepsilon}^p)_f+\lambda_f\mathbf{I}\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)_f\right]S_f\\=&\sum_{f=1}^{F}\mathbf{n}_f\cdot\left\{2\mu_f[f_x\,(d\boldsymbol{\varepsilon}^p)_P+(1-f_x)\,(d\boldsymbol{\varepsilon}^p)_N]\right\}S_f\\&+\sum_{f=1}^{F}\mathbf{n}_f\cdot\left\{\lambda_f\mathbf{I}\left[f_x\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)_P+(1-f_x)\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)_N\right]\right\}S_f\end{aligned}\tag{22}$$

$$\oint_{\partial V_P} d\mathbf{s}\cdot(dp)=\sum_{f=1}^{F}\mathbf{n}_f\cdot(dp)_fS_f=\sum_{f=1}^{F}\mathbf{n}_f\left[f_x(dp)_P+(1-f_x)(dp)_N\right]S_f\tag{23}$$

wherein above all the cell center values: $(d\boldsymbol{\varepsilon}^p)_P\,,(d\boldsymbol{\varepsilon}^p)_N\,,(dp)_P,(dp)_N$ are evaluated from the previous iterative values.

The explicit displacement coupling term is differentiated with respect to both time and space, and it can be approximated as follows:

$$\begin{aligned}\oint_{\partial V_P} d\mathbf{s}\cdot\left(\frac{\partial\mathbf{u}}{\partial t}\right)=\oint_{\partial V_P} d\mathbf{s}\cdot\left(\frac{\mathbf{u}-\mathbf{u}^o}{\Delta t}\right)=&\frac{1}{\Delta t}\sum_{f=1}^{F}\mathbf{n}_f\cdot\left(\mathbf{u}_f-\mathbf{u}_f^o\right)S_f\\=&\frac{1}{\Delta t}\sum_{f=1}^{F}\mathbf{n}_f\cdot\left[f_x\mathbf{u}_P+(1-f_x)\mathbf{u}_N\right]S_f\\&-\frac{1}{\Delta t}\sum_{f=1}^{F}\mathbf{n}_f\cdot\left[f_x\mathbf{u}_P^o+(1-f_x)\mathbf{u}_N^o\right]S_f\end{aligned}\tag{24}$$

where $\mathbf{u}_P,\mathbf{u}_N$ will be evaluated from the current available iterative value.

3.2.3 Traction boundary conditions

The traction boundary condition in FVM is discretized into displacement gradients of the CVs on the boundary patches based on Cauchy's stress theorem. It has been discussed previously in Section 2.4 that the same *Implicit-Explicit* split and iteration method shall be applied on the boundary as the inner solution domain, expressed by the formula below:

$$\begin{aligned}\underbrace{(2\mu+\lambda)\mathbf{n}_b\cdot[\nabla(d\mathbf{u})]_b}_{\text{implict}} &= (d\mathbf{t})_b - \underbrace{\{\mu[\nabla(d\mathbf{u})]_b\cdot\mathbf{n}_b + \lambda\mathrm{tr}\left([\nabla(d\mathbf{u})]_b\right)\mathbf{n}_b - (\mu+\lambda)\mathbf{n}_b\cdot[\nabla(d\mathbf{u})]_b\}}_{\text{explicit inter-component coupling}} \\ &+ \underbrace{\{2\mu\mathbf{n}_b\cdot(d\boldsymbol{\varepsilon}^p)_b + \lambda\mathrm{tr}(d\boldsymbol{\varepsilon}^p)_b\mathbf{n}_b\} + (dp)_b\mathbf{n}_b}_{\text{explicit nonlinear and pressure coupling}}\end{aligned} \tag{25}$$

where $\mathbf{n}_b$ is the outward-pointing boundary face area vector, $(d\mathbf{t})_b$ is the traction increment on the boundary (be either directly specified or computed using the specified total traction minus the previous time total traction), and the variables with lower index b are values on boundary face.

In this way, the traction boundary will be iteratively updated until both the convergence of the inner solution domain and convergence of the boundaries have been obtained. In general, this extra computation effort on the traction BC is marginal as the size of the internal domain is often far larger than that of the boundary patches.

4 Solution procedure

As a consequence of the above discretization, the discretized form of Equations (11) and (12) for each CV can finally be arranged as four linearized algebraic equations (one vector equation and one scalar equation):

$$a_P^{(d\mathbf{u})}(d\mathbf{u})_P + \sum_F a_N^{(d\mathbf{u})}(d\mathbf{u})_N = b_P^{(d\mathbf{u})} \tag{26}$$

$$a_P^{(p)}p_P + \sum_F a_N^{(p)}p_N = b_P^{(p)} \tag{27}$$

where F is the number of control volume faces.

The discretized coefficients $a_P^{(d\mathbf{u})}$, $a_N^{(d\mathbf{u})}$, $a_P^{(p)}$, $a_N^{(p)}$, and source terms $b_P^{(d\mathbf{u})}$, $b_P^{(p)}$ are summarized as:

$$a_P^{(d\mathbf{u})} = -\sum_F a_N^{d\mathbf{u}}, \quad a_N^{(d\mathbf{u})} = (2\mu_f+\lambda_f)\frac{|\mathbf{n}_f|}{|\mathbf{d}_f|}S_f \tag{28}$$

$$\begin{aligned} b_P^{(d\mathbf{u})} = &-\sum_F \mathbf{n}_f\cdot\left\{\mu_f\left[\nabla(d\mathbf{u})\right]_f^{\mathrm{T}} + \lambda_f\mathbf{I}\mathrm{tr}\left[\nabla(d\mathbf{u})\right]_f - (\mu_f+\lambda_f)[\nabla(d\mathbf{u})]_f\right\}S_f \\ &+\sum_F \mathbf{n}_f\cdot\left[2\mu_f(d\boldsymbol{\varepsilon}^p)_f + \lambda_f\mathbf{I}\mathrm{tr}\left(d\boldsymbol{\varepsilon}^p\right)_f\right]S_f + \sum_F \mathbf{n}_f\cdot(dp)_f S_f\end{aligned} \tag{29}$$

$$a_P^{(p)} = -\sum_F a_N^p - \frac{nV_P}{K'\Delta t}, \quad a_N^{(p)} = \frac{k_f}{\gamma_f}\frac{|\mathbf{n}_f|}{|\mathbf{d}_f|}S_f \tag{30}$$

$$b_P^{(p)} = -\frac{n}{K'}\frac{p_P^o}{\Delta t}V_P + \frac{1}{\Delta t}\sum_F \mathbf{n}_f\cdot\left(\mathbf{u}_f - \mathbf{u}_f^o\right)S_f \tag{31}$$

In above, the source terms $b_P^{(d\mathbf{u})}$ and $b_P^{(p)}$ cover the contribution from explicit nonlinear terms, explicit couplings, and boundary conditions.

Assembling Equations (26) and (27) for each CV in the mesh results in four sets of algebraic equations:

$$[A^{(d\mathbf{u})}][d\mathbf{u}] = [b^{(d\mathbf{u})}] \tag{32}$$

$$[A^{(p)}][p] = [b^{(p)}] \tag{33}$$

where $[A^{(d\mathbf{u})}]$, $[A^{(p)}]$ are the sparse $N \times N$ matrices (N stands for the total number of CVs), with coefficients $a_P^{(d\mathbf{u})}, a_P^{(p)}$ on the diagonal and F non-zero neighbor coefficients $a_N^{(d\mathbf{u})}, a_N^{(p)}$ off the diagonal (recall F as the total internal face numbers of each CV); $[d\mathbf{u}]$ the displacement increment vector consisting of $d\mathbf{u}$ at the cell center of all CVs, $[p]$ the vector of pore pressure ps at each CV center; and $[b^{(d\mathbf{u})}]$, $[b^{(p)}]$ the assembled source vectors.

Having solved the incremental system for the four primary unknown variables, $d\mathbf{u}$ and p, it is possible to compute the the dependent unknown variables, i.e. the total displacement vector $\mathbf{u}$, the plastic strain increment $d\varepsilon^p$, the effective stress σ', and the pore pressure increment dp. Overall, the segregated solution procedure with fixed point iterations is summarized in Table 3:

Table 2: Global solution procedure

```
TIME STEP: $t$, $t = n$
ITERATION: $i$, $i = 0$

1. Initialize variables with the value from previous time(load) step:
   $(d\mathbf{u})_n^0 = (d\mathbf{u})_{n-1}$, $(\mathbf{u})_n^0 = (\mathbf{u})_{n-1}$, $(p)_n^0 = (p)_{n-1}$, $(d\varepsilon^p)_n^0 = (d\varepsilon^p)_{n-1}$
2. Assemble and solve the pore pressure equation for $(p)_n^i$ by:
   $\texttt{fvm::laplacian}(k/\gamma, p) - \texttt{fvm::ddt}(n/K', p) == \texttt{fvc::div}(\texttt{fvc::ddt}(\mathbf{u}))$[1]
3. Calculate the pore pressure increment:
   $dp = p_n^i - p_{n-1}$:
4. Assemble and solve the displacement equation for $(d\mathbf{u})_n^i$ by:
   $\texttt{fvm::laplacian}(2\mu + \lambda, d\mathbf{u}) == -\texttt{fvc::div}(\text{dsigmaExp})$
                                            $+\texttt{fvc::div}(2\mu d\varepsilon^p + \lambda \mathbf{I}\text{tr}(d\varepsilon^p))$
                                            $+\texttt{fvc::div(dp)}$
   with
   $\text{dsigmaExp} = \mu * \texttt{fvc::grad}(d\mathbf{u}).\text{T}() + \lambda * \mathbf{I} * \text{tr}(\texttt{fvc::grad}(d\mathbf{u})) - (\mu + \lambda) * \texttt{fvc::grad}(d\mathbf{u})$
5. Obtain plastic corrections $(d\varepsilon^p)_n^i$ through local stress update algorithm in Table 3
6. Update the total displacement:
   $\mathbf{u}_n^i = \mathbf{u}_{n-1} + (d\mathbf{u})_n^i$
7. Return to step 2 until converged solutions.
   $i = i + 1$

EXIT: $n = n + 1$
```

Two levels of iteration are involved in the global solution procedure. The *outer iterations*, namely the repeated Step 2-6, are performed to account for the explicitly lagged nonlinear and couplings terms. While the *inner iterations* are performed during the iterative solution of the symmetric and diagonally dominant linear sparse systems in Step 2 and Step 4, typically using the incomplete Cholesky pre-conditioned conjugate gradient (ICCG) method. Due to the applied linear Mohr-Coulomb plasticity model, the local return mapping process to a linear yield plane with a linear plastic potential can be performed in one step.

In principle, it is not necessary to solve the *inner iterations* to a very tight tolerance, as the solutions will only be used to update the explicit terms in the *outer iterations.* The entire system is considered to be solved when the solution changes less than a predefined tolerance.

In addition, in order to promote the stability of the iteration method, an under-relaxation method is often necessary. In this work, we employ two levels of under-relaxation for the *outer iteration* and *inner iteration*, respectively:

1. Under-relaxation of the variables in *outer iteration*

   The variable under-relaxation can be explained briefly by the form below:

$$\tilde{x}^i = (1 - \theta)x^{i-1} + \theta x^i \tag{34}$$

   where $\tilde{x}^i$ stands for the i_{th} under-relaxed iterative value of a generalized unknown variable x, and θ the under-relaxation factor with a value between 0 to 1. The variable under-relaxation slows down the change of two successive iterations in the outer iteration and thus promotes

[1] The `fvm::` operator indicates an *implicit* discretization term in OpenFOAM, whereas the `fvc::` operator indicates an *explicit* calculus.

Table 3: Local return mapping stress update

```
INPUT: $d\mathbf{u}$, displacement increments
       $\boldsymbol{\sigma'}^A$, initial/old-time stress

1. Compute the elastic trial stress $\boldsymbol{\sigma'}^B$ by: $\boldsymbol{\sigma'}^B = \boldsymbol{\sigma'}^A + \{\mu\nabla(d\mathbf{u}) + \mu[\nabla(d\mathbf{u})]^{\mathrm{T}} + \lambda\mathbf{I}\mathrm{tr}[(d\mathbf{u})]\}$
2. Transform $\boldsymbol{\sigma}^B$ into principal space as $\boldsymbol{\sigma'}^B_{prin}$ (eigenvalues).
   Store the principal directions (eigenvectors).
3. Evaluate the yield function $f(\boldsymbol{\sigma'}^B_{prin})$:
   if $f < 0$, EXIT, $\boldsymbol{\sigma'}^C = \boldsymbol{\sigma}^B$, $d\boldsymbol{\varepsilon}^p = 0$
   if $f \geq 0$, CONTINUE
4. Determine the right stress return type. Obtain the principal plastic corrector stress $\boldsymbol{\sigma'}^C_{prin}$.
5. Reuse the preserved principal directions and transform $\boldsymbol{\sigma'}^C_{prin}$ back to the general space as $\boldsymbol{\sigma'}^C$
6. Compute the correct elastic strain increment $d\boldsymbol{\varepsilon}^e$ by:
   $d\boldsymbol{\varepsilon}^e = \dfrac{\mathrm{iso}(\boldsymbol{\sigma'}^C - \boldsymbol{\sigma'}^A)}{3\lambda + 2\mu} + \dfrac{\mathrm{dev}(\boldsymbol{\sigma'}^C - \boldsymbol{\sigma'}^A)}{2\mu}$
7. Calculate the plastic strain increment $d\boldsymbol{\varepsilon}^p$ by: $d\boldsymbol{\varepsilon}^p = \dfrac{1}{2}[\nabla(d\mathbf{u}) + \nabla(d\mathbf{u})^{\mathrm{T}}] - d\boldsymbol{\varepsilon}^e$

OUTPUT: $\boldsymbol{\sigma'}^C$, $d\boldsymbol{\varepsilon}^p$
```

the stabilization of the solution method. The relaxation factor θ can be specified either as constant value (fixed under-relaxation) or more efficiently as adaptive value based on the iteration sequences (e.g. Aitken's method, Anderson acceleration, etc.). In this paper, the simple fixed under-relaxation has been adopted.

2. Under-relaxation of equations in *inner iteration*

   The equation under-relaxation is dealing with the linear algebraic system from Step 2 and Step 4 in the global solution procedure. If we take Equation (26) as an example for explanation, it may be written as:

$$\frac{a_P^{(d\mathbf{u})}}{\beta}(d\mathbf{u})_P^i = -\sum_F a_N^{(d\mathbf{u})}(d\mathbf{u})_N^i + b_P^{(d\mathbf{u})} + (1-\beta)\frac{a_P^{(d\mathbf{u})}}{\beta}(d\mathbf{u})_P^{i-1} \tag{35}$$

   where β is another under-relaxation factor between 0 and 1. It enhances the diagonal dominance of the linear system and thus improves the convergence of most iterative linear algebraic solvers [13, 25].

The segregated fixed point iteration algorithm combined with the under-relaxation has been implemented as a custom solver `nonlinearBiotFoam` in the open-source FVM code OpenFOAM-1.6-ext.

5 Numerical examples

In order to assess the performance of the `nonlinearBiotFoam` solver, three geotechnical problems with comparable solutions in either analytical or numerical (FEM) forms have been simulated: 1-D consolidation of a poro-elastic soil column; the bearing capacity of a strip footing; and the uplift capacity of a circular suction caisson foundation. Additionally, the computational cost for each case is presented to demonstrate the efficiency of the solver.

5.1 Consolidation of poro-elastic soil column

The first simple case is aimed to verify the elastic soil-pore fluid coupling. Figure 2 outlines the case definition: a saturated soil column with height h is subjected to a surface step loading (T) of

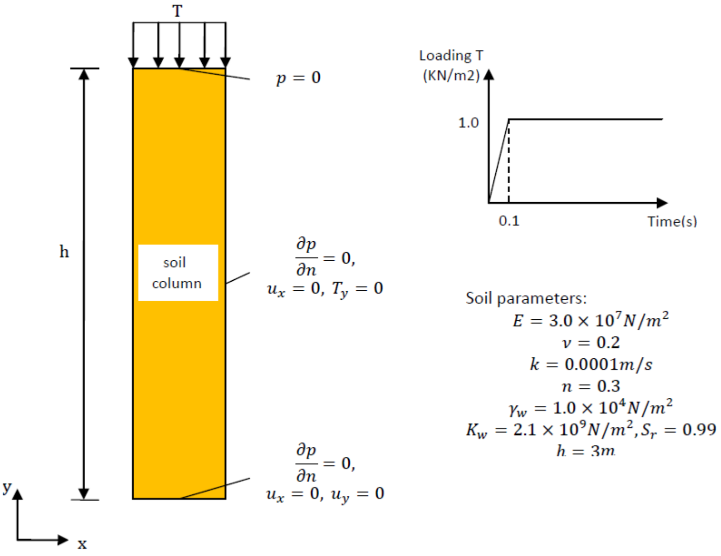

T
$p = 0$
h
soil column
$\frac{\partial p}{\partial n} = 0,$
$u_x = 0,\ T_y = 0$
$\frac{\partial p}{\partial n} = 0,$
$u_x = 0,\ u_y = 0$
y
x
Loading T (KN/m2)
1.0
0.1
Time(s)
Soil parameters:
$E = 3.0 \times 10^7 N/m^2$
$v = 0.2$
$k = 0.0001 m/s$
$n = 0.3$
$\gamma_w = 1.0 \times 10^4 N/m^2$
$K_w = 2.1 \times 10^9 N/m^2, S_r = 0.99$
$h = 3m$

Figure 2: A poro-elastic soil column subjected to a surface step loading.

$1kN/m^2$ applied over a time of $0.1s$. The boundary conditions and soil material properties have also been shown in the figure itself.

A comparison between FVM prediction, the analytical solution of Terzaghi's 1-D consolidation theory [26], and FEM simulation in Abaqus is shown in Figure 3. The horizontal axis represents the pore pressure normalized by an initial pore pressure built up immediately after loading, and the vertical axis stands for the depth of the soil column normalized by the total height. The five data series correspond to different dimensionless consolidation time, $(c_v t/h^2)$, where c_v is the consolidation coefficient. Overall, the FVM results agree very well to both the analytical and Abaqus solutions.

The OpenFOAM FVM simulations were carried out on a laptop computer with Intel Core i7 1.5GHz processor and 8GB RAM, and the total computation cost is listed in Table 4.

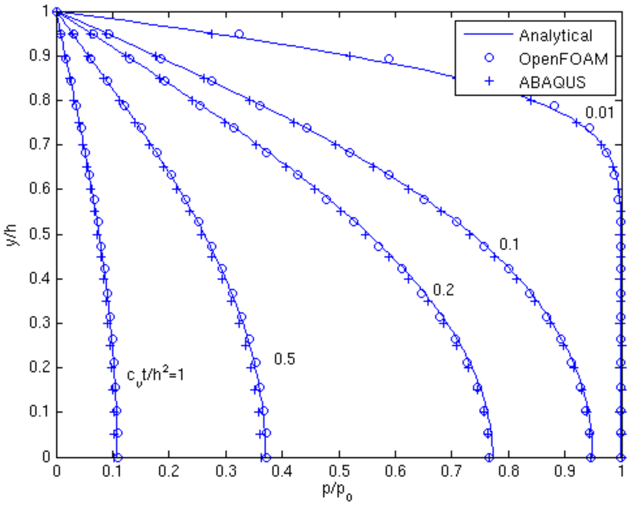


Figure 3: Numerical and analytical excess pore pressure dissipation isochrones (sampled along the center line of soil column)

Table 4: FVM computation cost of the poro-elastic consolidation case

No. CV	No. time step	No. outer iterations per time step	Total CPU time (s)
40	110	12	10.59

5.2 Bearing capacity of strip footing

The second test aims at demonstrating the elasto-plastic soil-pore fluid coupling. Following the work done by Small et al. [11], a similar elasto-plastic consolidation validation case is examined here: a smooth, perfectly flexible, uniformly loaded, permeable strip footing acting on a layer of soil resting on a smooth rigid base. In order to completely define the problem, it is also assumed that there is no horizontal force on any vertical section. The original case geometry is sketched in Figure 4 and the OpenFOAM case set-up with regular mesh and soil properties is shown in Figure 5. Plain strain condition has been considered. To demonstrate the computational efficiency of the proposed FVM technique, a comparable coupled Mohr-Coulomb FEM simulation is run in Abaqus (version 6.11-1).

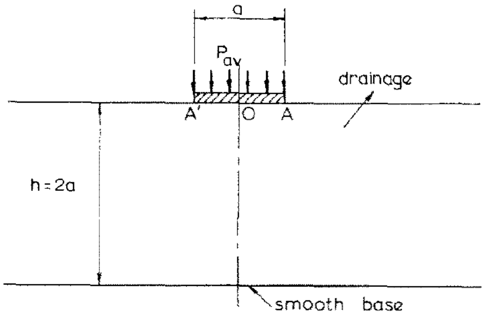
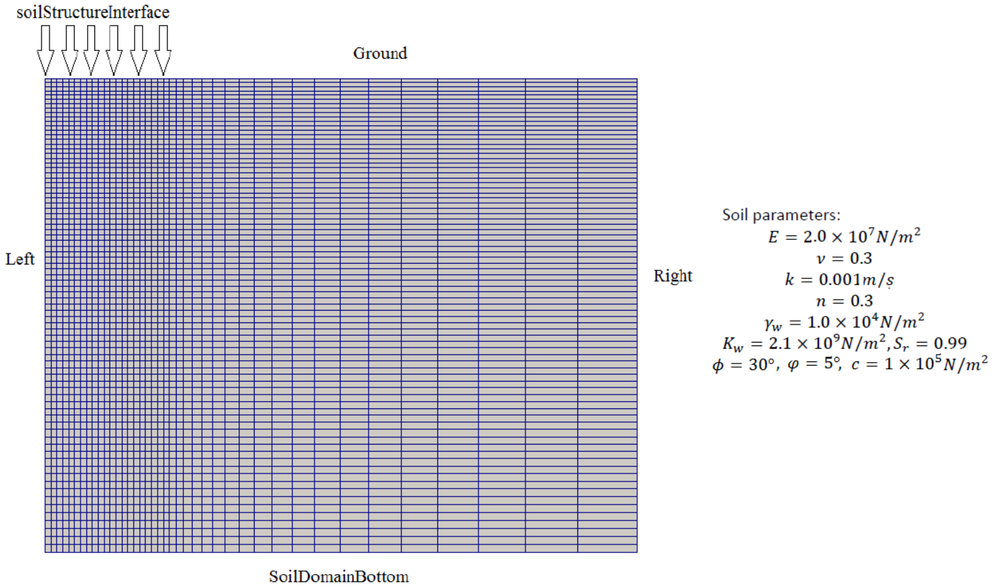


Figure 4: Illustration of case geometry (Adopted from [11])



Soil parameters:

$$E = 2.0 \times 10^7 N/m^2$$
$$\nu = 0.3$$
$$k = 0.001 m/s$$
$$n = 0.3$$
$$\gamma_w = 1.0 \times 10^4 N/m^2$$
$$K_w = 2.1 \times 10^9 N/m^2, S_r = 0.99$$
$$\phi = 30°, \ \varphi = 5°, \ c = 1 \times 10^5 N/m^2$$

Figure 5: Finite volume mesh used for footing (half domain)

The boundary conditions for the displacement increment ($d\mathbf{u}$) and pore fluid pressure (p) in

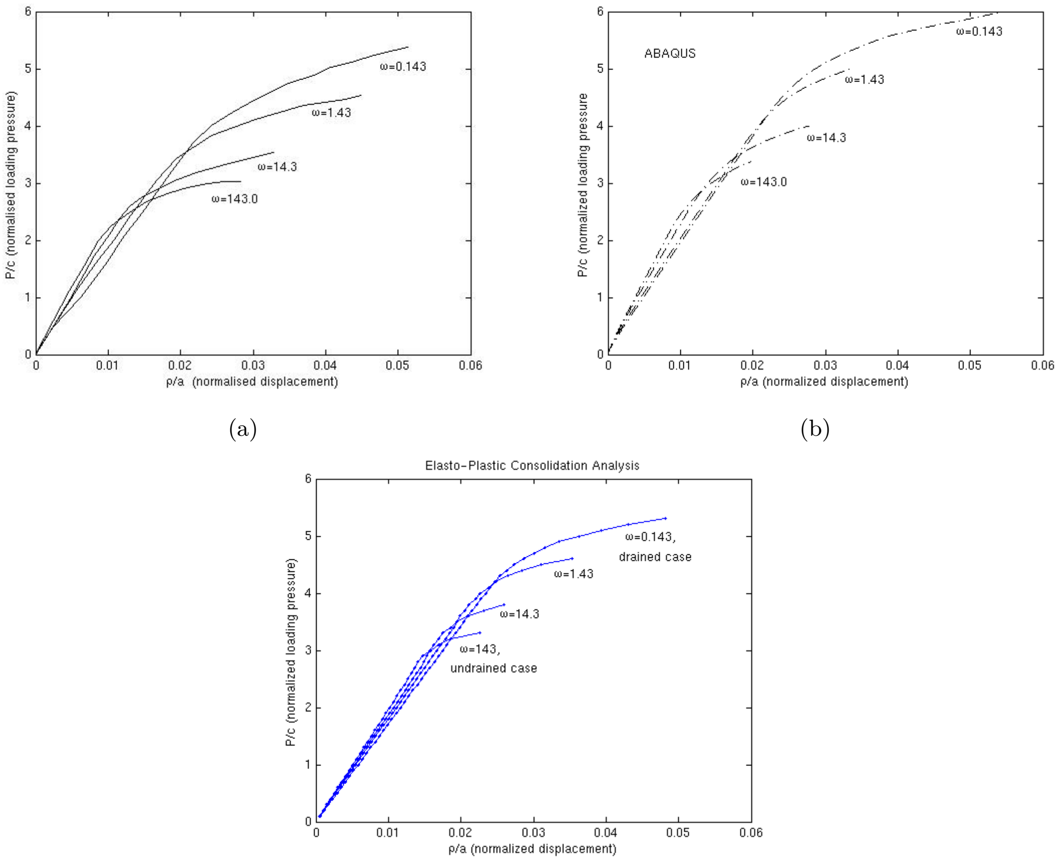
OpenFOAM are listed in Table 5. Correspondingly, the Abaqus model input of the same boundary and loading conditions is summed in Table 6.

Table 5: The boundary conditions used in OpenFOAM

	soilStructureInterface	Ground	Left	Right	soilDomainBottom
$d\mathbf{u}$	timeVaryingTraction	zeroTraction	symmetryPlane	zeroTraction	fixedDispZeroShear $(du_z = 0, t_x = 0)$
p	fixedValue $(p = 0)$	fixedValue $(p = 0)$	symmetryPlane	zeroGradient	zeroGradient

Table 6: The prescribed boundary and load conditions in Abaqus

	soilStructureInterface	Ground	Left	Right	soilDomainBottom
Displacement BC			XSYMM		YSYMM
pore pressure BC	$p = 0$	$p = 0$			
load condition	Uniform pressure				



(a)

(b)

(c)

Figure 6: Bearing capacities under different loading rates a). FEM resluts of Small et al.; b). FEM results in Abaqus; c). `nonlinearBiotFoam` simulations

In the beginning, it is assumed that there is no initial stress and pore pressure in the domain. A load rate parameter ω is adopted from [11]:

$$\omega = \frac{d(P/c)}{d(T_v)}, \qquad T_v = \frac{c_v t}{a^2} \tag{36}$$

Here, P/c represents the external load pressure (normalized by soil cohesion), T_v the dimensionless time defined by c_v which is the aforementioned one-dimensional consolidation coefficient, and a the width of strip footing.

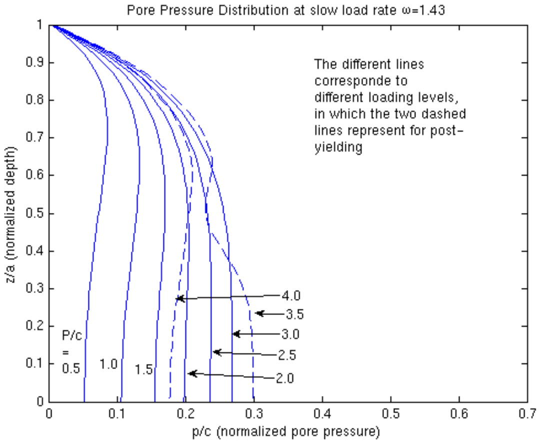


Figure 7: Pore pressure distributions along the footing center line with slow load rate $\omega = 1.43$

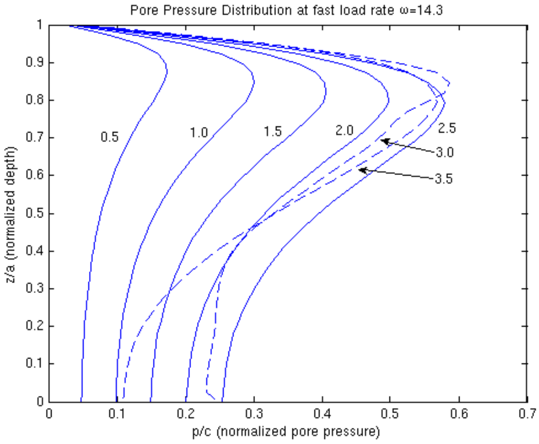


Figure 8: Pore pressure distributions along the footing center line with fast load rate $\omega = 14.3$

In general, the smaller the load parameter ω, the slower the load rate acting on the soil layer. It is suggested in [11] that $\omega <= 0.143$ could represent drained condition, $\omega >= 143$ for fully undrained condition, and the two intermediate rates of $\omega = 1.43$ and $\omega = 14.3$ for a slow load rate and a fast load rate in partially drained conditions, respectively. For comparison, Figure 6a shows the original results of Small et al, Figure 6b presents the Abaqus predictions, while Figure 6c shows the simulation results of `nonlinearBiotFoam` in OpenFOAM.

It can be clearly seen that the `nonLinearBiotFoam` solver successfully predicts the different failure load levels under various load rates (i.e. stronger response at slower load rate), consistent with the results of Small et al and Abaqus. Moreover, Figure 7 and Figure 8 show the pore pressure distributions along the footing center line at different load levels, for the slow load rate $\omega = 1.43$ and for the fast load rate $\omega = 14.3$, respectively. The numbers next to the lines are indicating the load level (P/c). At low load levels in the beginning, the whole soil domain behaves elastically (solid lines). Whereas, when load pressure increases to high levels, soil reaches failure (dotted line). It is interesting to observe from the simulation results that the excess pore pressure in the soil starts to dissipate fast as long as plastic deformation occurs and results in dilation. This demonstrates well the capability of `nonlinearBiotFoam` to effectively model the interactions between the pore pressure development and the nonlinear soil behavior.

The OpenFOAM simulation was performed under parallel computing using 8 cores on an IBM HPC cluster (2.66 GHz Intel Xeon E5430 cores). Comparably, the Abaqus simulations were each run on 6 cores in parallel on an SGI ICE X cluster (2.4 GHz Intel Xeon E5-2695 v2 Ivy Bridge cores).

The overall computational costs for FVM and FEM analysis are summarized in Table 7 and Table 8, respectively.

Table 7: FVM OpenFOAM computation cost of the strip footing case

Case	No. CV	No. time steps/ plasticity occurs at	Max. No. outer iter. per plastic time step	Total wall time (hr)
drained	3200	53/38	3.4×10^4	00:17:23
slow load rate	3200	46/34	1.0×10^5	00:42:43
fast load rate	3200	38/27	1.2×10^4	00:18:58
undrained	3200	33/25	1.0×10^5	00:33:04

Table 8: FEM Abaqus computation cost of the strip footing case

Case	No. elements/ No. nodes per element	No. time steps/ plasticity occurs at	Max. No. equil iter. per plastic time step	Total wall time (hr)
drained	4800/8	$1.150 \times 10^4/396$	16	14:50:00
slow load rate	4800/8	$6.142 \times 10^3/358$	16	08:07:00
fast load rate	4800/8	$9.558 \times 10^3/263$	16	13:21:00
undrained	4800/8	$1.752 \times 10^4/257$	16	06:50:00

It is necessary to note that the proposed *segregated* FVM simulations use constant time steps with large amounts of fixed-point iterations, while the implicit FEM predictions in Abaqus employ automatic (small) time step with Newton-type iterations. The two methods encounter the same challenge - convergence rate decreases during plastic steps and requires special algorithm handling. We therefore applied under-relaxation methods in OpenFOAM and automatic stabilization with fixed damping factor in Abaqus to help for convergence.

In particular, the Abaqus soil analysis only assumes fully saturated condition (i.e. incompressible pore fluid) using standard FE coupled formulation (e.g. [27, sec. 6.8]). Element type with hybrid formulation or reduced integration has been used to alleviate the volumetric locking problem (in this case, we applied the 8-node quadratic elements with hybrid formulation CPE8PH[2]). Furthermore, the fully-saturated assumption yields singularities in the stiffness matrix and significantly slows down the convergence. The solution of the large implicit matrix system in Abaqus also takes much more RAM and longer computational time. Whereas, the FVM `nonlinearBiotFoam` solver is formulated on the basis of unsaturated condition, in which true bulk modulus of pore fluid (comparable to that of soil skeleton) is considered and the convergence is improved. The staggered solution procedure also enables efficient iterative solver to be used for solving the small matrix system. Alternatives to the Abaqus implicit solver are explicit FEM/hybrid-FEM solvers (with optimized algorithms as

[2] CPE8PH: an 8-node plane strain quadratic, biquadratic displacement, bilinear pore pressure, hybrid, linear pressure stress

for the proposed FVM), however they are not available for fully coupled poro-elasto-plastic stress analysis in the commercial software packages.

Additionally, if comparing the two methods in terms of traction boundary condition treatment: the FEM Abaqus analysis only requires a simple external load input, while the proposed FVM uses user-defined boundaries which do iterative computation from the external traction to displacement on the patches. This drawback of FVM may be acceptable as long as consistent staggered iterative method has been applied on the boundary and the inner solution domain, which provides an overall efficiency of the algorithm.

5.3 Uplift capacity of suction caisson

The third test case is representing a problem that draws increasing applications in the offshore geotechnical field: namely, the vertical uplift capacity of suction caisson foundations. A suction caisson, open at the bottom and closed at the top, is designed to penetrate to the sea floor by its own weight and also by creating an internal under-pressure relative to the external water pressure [28]. The developed passive suction inside the soil plug significantly enhances the pullout capacity of the suction caisson.

Figure 9 illustrates a cylindrical suction caisson with an overall diameter d and an embedded length L, subjected to an uplift load at the top of the caisson.

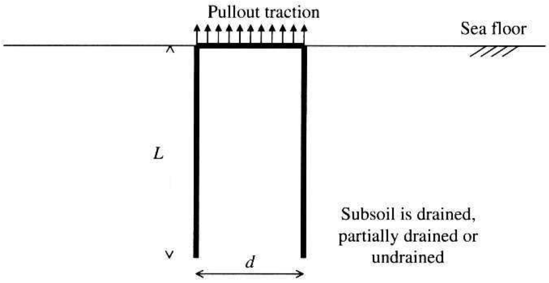


Figure 9: Suction caisson foundation under vertical uplift loading (Adopted from [28]).

The above geometry are then set up in OpenFOAM using the same rectangular FVM mesh type as in Figure 5. A 'wedge' type boundary in OpenFOAM is also applied to simulate the case as 2-D axi-symmetric. It has to be mentioned that the entire suction caisson is assumed to be lifted uniformly under constant rate (as a prescribed displacement boundary). All the parameters involved in the simulations have been listed in Table 9.

Table 9: Parameters employed in the suction caisson case

Component	Property	Symbol	Value(s)
Caisson geometry	Embedment depth	L	1m
	Diameter	d	2m
Soil properties	Young's modulus	E	2×10^7 Pa
	Poisson's ratio	ν	0.3
	Permeability	k	0.001m/s
	Porosity	n	0.2
	Saturation factor	Sr	0.98
	Friction angle	φ	30
	Cohesion	c	1×10^5 Pa
	Dilation angle	ψ	0
	Bulk modulus of water	K_w	2.1×10^9 Pa
	Specific weight of water	γ_w	1×10^4 Pa
Loading rate	Pullout velocity	v	2×10^{-4} to 2×10^{-1} m/s

Our focus is to evaluate the effects of different loading rate on the uplift capacity. It is revealed in Figure 10 that faster load rates give greater pullout capacities, which comes from the ability to build up and sustain a significant negative pore pressure inside the bucket, Figure 11. This captured phenomenon agrees well to those simulation results in the references [28, 29] with FEM analysis, although they are dealing with less permeable clay.

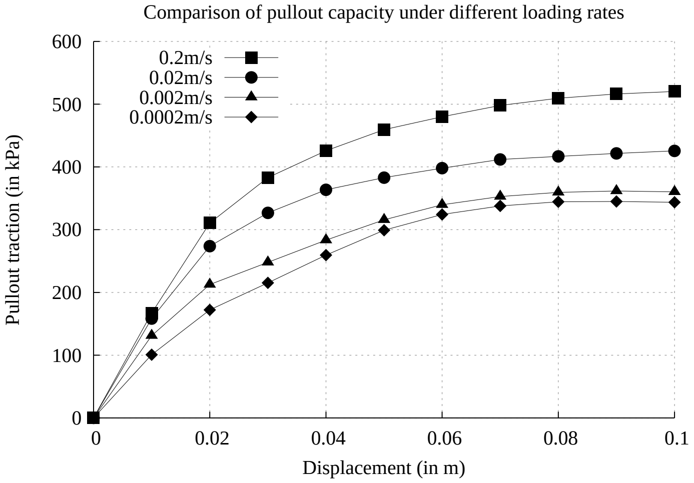


Figure 10: Pullout load-displacement behavior under various loading rate

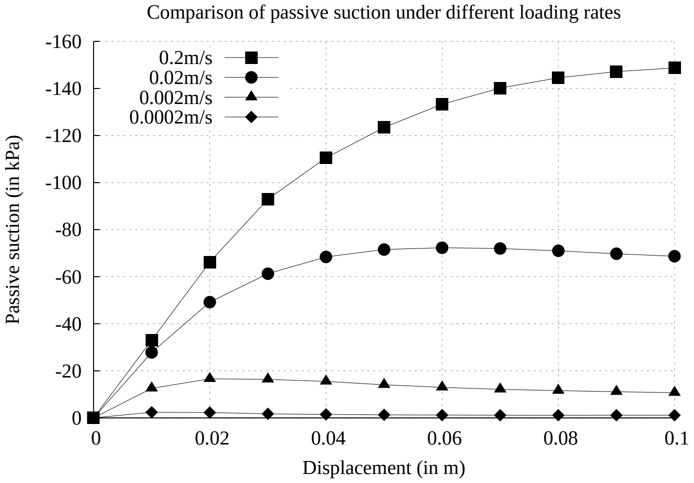


Figure 11: Passive suction development under various loading rate

The simulation was carried out under parallel computation using 8 cores of a single computation node which has 2 XEON 5550 2.66 GHZ quad core CPUs and 24 GB RAM. The overall cost is listed in Table 10.

Table 10: Computation cost of the suction caisson

Case	No. CV	No. time steps	Max. No. outer iterations per time step	Total used wall time (hr)
0.2m/s	1840	10	1×10^4	00:18:58
0.02m/s	1840	10	1×10^4	00:20:43
0.002m/s	1840	10	1×10^4	00:20:55
0.0002m/s	1840	10	1×10^4	00:23:31

6 Conclusion

A new numerical method for poro-elasto-plastic soil is presented, based on a finite volume method discretization and segregated solution procedure. The main features of the method are:

(i) It is computationally efficient in terms of both memory and CPU time-wise, as the *Implicit-Explicit* split of discretization and linearization result in a decoupled system of linear algebraic system with sparse diagonally dominant coefficient matrices, solvable by efficient iterative solvers.

(ii) The solution algorithm is neat and economic, because any nonlinearity and coupling can be handled with one single *outer iteration* process (i.e. the fixed point iteration method), thus requiring no or little additional cost. Moreover, it has been demonstrated in the convergence study of the iteration method (in Appendix A.1 and A.2): careful splitting of the nonlinear and coupling terms guarantees a linear convergence rate for unsaturated soil problems. For fully saturated situations, the selection of time step and mesh density needs to fulfill the convergence criteria, namely Eq. 47.

(iii) The implementation work is time-saving under the help of OpenFOAM C++ library where rich FVM classes and functions are freely available.

(iv) The model itself is easily extendable, as other nonlinear constitutive relations can be integrated into the model with the same global solution procedure.

(v) It has a lot of application potentials to model many geotechnical problems in a simple and effective manner, as illustrated by the test cases.

Acknowledgement

We would like to thank University College Dublin OpenFOAM Simulation Group, especially Professor Alojz Ivankovic, for all the support and helpful discussions during the first author's 3-month external research stay there. Our gratefulness also goes to Dr. Johan Roenby for his important inputs on developing the ideas present in this work. Moreover, the authors would like to acknowledge the SFI/HEA funded Irish Centre for High End Computing for the provision of computational resources and support.

References

1. Zienkiewicz OC, Chan AHC, Pastor M, Paul DK, Shiomi T. Static and dynamic behaviour of soils: a rational approach to quantitative solutions. I. Fully saturated problems. *Proceedings of the Royal Society London A* 1990; **429**:285-309.

2. Zienkiewicz OC, Taylor RL. *The Finite Element Method*, vol. 2 (4th edn). McGraw-Hill: New York, 1991.

3. Biot MA. General theory of three-dimensional consolidation. *Journal of Applied Physics* 1941; **12(2)**: 155-164.

4. Wang HF. *Theory of Linear Poroelasticity with Applications to Geomechanics and Hydrogeology.* Princeton University Press: New Jersey, 2000.

5. Zienkiewicz OC. Basic formulation of static and dynamic behaviours of soil and other porous media. *Applied Mathematics and Mechanics* 1982; **3(4)**:457-468.

6. Zienkiewicz OC, Shiomi T. Dynamic behaviour of saturated porous media: The generalized Biot formulation and its numerical solution. *International Journal for Numerical and Analytical Methods in Geomechanics* 1984; **8(1)**:71-96.

7. Elgamal A, Yang Z, Parra E. Computational modeling of cyclic mobility and post-liquefaction site response. *Soil Dynamics and Earthquake Engineering* 2002; **22(4)**:259-271.

8. Jeremic B, Cheng Z, Taiebat M, Dafalias Y. Numerical simulation of fully saturated porous materials. *International Journal for Numerical and Analytical Methods in Geomechanics* 2008; **32(13)**:1635-1660.

9. Huang M, Zienkiewicz, OC. New unconditionally stable staggered solution procedures for coupled soil-pore fluid dynamic problems. *International Journal for Numerical Methods in Engineering* 1998; **43(6)**:1029-1052.

10. Carter JP, Booker JR, Small JC. Analysis of finite elasto-plastic consolidation. *International Journal for Numerical and Analytical Methods in Geomechanics* 1979; **3(2)**:107-129.

11. Small JC, Booker JR, Davis EH. Elasto-plastic consolidation of soil. *International Journal of Solids and Structures* 1976; **12(6)**:431-448.

12. Demirdzic I, Martinovic D, Ivankovic A. Numerical simulation of thermal deformation in welded workpiece (in croatian), *Zavarivanje* 1988; **31**:209-219.

13. Demirdzic I, Martinovic D. Finite volume method for thermo-elasto-plastic stress analysis. *Computer Methods in Applied Mechanics and Engineering* 1993; **109(3-4)**:331-349.

14. Demirdzic I, Martinovic D. Finite volume method for stress analysis in complex domains. *International Journal for Numerical Methods in Engineering* 1994; **37**:3751-3766.

15. Demirdzic I, Muzaferija S. Numerical method for coupled fluid flow, heat transfer and stress analysis using unstructured moving meshes with cells of arbitrary topology. *Computer Methods in Applied Mechanics and Engineering* 1995; **125(1-4)**:235-255.

16. Demirdzic I, Muzaferija S. Benchmark solutions of some structural analysis problems using the finite-volume method and multgrid acceleration. *International Journal for Numerical Methods in Engineering* 1997; **40**:1893-1908.

17. Bijelonja I, Demirdzic I, Muzaferija S. A finite volume method for large strain analysis of incompressible hyperelastic materials. *International Journal for Numerical Methods in Engineering* 2005; **64(12)**:1594-1609.

18. Demirdzic I, Horman I, Martinovic D. Finite volume analysis of stress and deformation in hygro-thermo-elastic orthotropic body. *Computer Methods in Applied Mechanics and Engineering* 2000; **190**:1221-1232.

19. Jasak H, Weller H. Finite volume methodology for contact problems of linear elastic solids. In *Proceedings of 3rd International Conference of Croatian Society of Mechanics*, Cavtat/-Dubrovnik, Crotatia, 2000; 253-260.

20. Clausen J, Damkilde L, Andersen L. An efficient return algorithm for non-associated plasticity with linear yield criteria in principal stress space. *Computers and Structures* 2007; **85(23-24)**:1795-1807.

21. Weller HG, Tabor G, Jasak H, Fureby C. A tensorial approach to computational continuum mechanics using object-orientated techniques. *Computers in Physics* 1998; **12(6)**:620-631.

22. Jasak H, Weller HG. Application of the finite volume method and unstructured meshes to linear elasticity. *International Journal for Numerical Methods in Engineering* 2000; **48(2)**:267-287.

23. Tukovic Z, Ivankovic A, Karac A. Finite-volume stress analysis in multi-material linear elastic body. *International Journal for Numerical Methods in Engineering* 2013; **93(4)**:400-419.

24. Jasak H. Error analysis and estimation for finite volume method with applications for fluid flows. *Ph.D. Thesis*, Imperial College, University of London, 1996.

25. Maneeratana K. Development of the finite volume method for non-linear structural applications. *Ph.D. Thesis*, Imperial College London 2000.

26. Verruijt A. *Soil Mechanics.* Delft University of Technology, 2001.

27. Abaqus Analysis User's Manual. Online resources: `http://www.tu-chemnitz.de/projekt/abq_hilfe/docs/v6.11/books/usb/default.htm.`, accessed in May 2014.

28. Deng W, Carter JP. A theoretical study of the vertical uplift capacity of suction caissons. *International Journal of Offshore and Polar Engineering* 2002; **12(2)**:89-97.

29. Cao J, Phillips R, Popescu R, Audibert JME, Al-Khafaji Z. Numerical analysis of the behavior of suction caissons in clay. In *Proceedings of the International Offshore and Polar Engineering Conference* 2002; **12**:795-799.

30. Gradinaru DM. *Iterative Methods for Non-Linear Systems of Equations.* Lecture Notes. Online Website: `http://www.math.ethz.ch/education/bachelor/lectures/fs2012/other/nm\_pc/NPch1.pdf`, accessed in September 2013.

31. Turska E, Schrefler BA. On convergence conditions of partitioned solution procedures for consolidation problems. *Computer Methods in Applied Mechanics and Engineering* 1993; **106(1-2)**:51-63.

32. Zohdi TI. An adaptive-recursive staggering strategy for simulating multifield coupled processes in microheterogeneous solids. *International Journal for Numerical Methods in Engineering* 2002; **53(7)**:1511-1532.

A Remarks on fixed point iteration method

The fixed-point (FP) iteration method in FVM simply uses the following recursive form:

$$x^{i+1} = G\left(x^i\right), \quad i = 0, 1, 2, \cdots \tag{37}$$

The FP method usually results in linear convergence [30], slightly inferior to the commonly applied Newton-type iteration method in FEM that gives quadratic convergence. However, the important feature of FP method lies in that, it avoids the formation of the large non-symmetric Jacobian matrix, and instead it produces several small symmetric matrices which can be very efficiently solved by standard fast iterative algebraic solvers. The main problem of applying FP method comes out that that the convergence condition is dependent on the choice of function $G(x)$, which shall in principle be differentiable and fulfill $|G'(x)| \leq 1$ in some operator norm [31]. In our case, the definition of function $G(x)$ is closely related to the method of partitioning the nonlinear terms in the momentum equation and the soil-pore fluid coupling operators. The following paragraphs will then discuss these two aspects separately.

A.1 The partitioning of nonlinear terms in momentum equation

If we are solving a nonlinear system equation $A(x)x = b$, the splitting of linear and nonlinear terms by writing $A(x) = A_l - A_{nl}(x)$, results in:

$$A_l x = A_{nl}(x)x + b \tag{38}$$

The FP iteration method is then:

$$x^{i+1} = A_l^{-1}(A_{nl}(x^i)x^i + b) \tag{39}$$

where, x symbolizes the displacement increment $d\mathbf{u}$ in the momentum equation Equation (14), A_l relates to the linear (implicit) elastic stiffness matrix, and $A_{nl}(x)$ the nonlinear (explicit) plastic stiffness matrix. In this way, the function $G(x)$ can simply be written out as $G(x) = A_l^{-1}(A_{nl}(x)x+b)$ and its differential is:

$$G'(x) = A_l^{-1}A_{nl}(x) + A_l^{-1}A'_{nl}(x)x \tag{40}$$

Particularly, as we used a perfect plasticity model without any hardening or softening effect in the constitutive relation, the derivative of plastic stiffness $A'_{nl}(x)$ is equivalent to zero and therefore the second term on the right hand side of Equation (40) simply vanishes. In the end, the sufficient convergence condition for the partitioning scheme turns to be $|G'(x)| = \|A_l^{-1}A_{nl}(x)\| \leq 1$. We may prove such convergence condition in a qualitative way in stead of complicated matrix norm manipulation. The elastic stiffness contribution A_l is the maximum stiffness that the system can provide, in other words the plastic stiffness A_{nl} is always weaker than the pre-yielding elastic stiffness A_l, which means that $|A_l^{-1}A_{nl}(x)\| \leq 1$ always holds. In general, as long as the implicit stiffness part is stronger than its counterpart explicit stiffness part, the system of momentum equation will converge. While for other advanced hardening/softening plastic models, more terms will be involved in Equation (40) and hence careful evaluation might be needed.

A.2 The partitioning of soil-pore fluid coupling

Consider a simple linearly coupled equation system in the following form:

$$\begin{aligned} aw_1 &= w_2 \\ b\dot{w}_2 + cw_2 &= \dot{w}_1 \end{aligned} \tag{41}$$

The above system has the same coupling feature as our poro-elasto-plasticity model, i.e. Equations (11) and (12), but is formed in a much simpler form for demonstration purpose. Reference [32] has pointed out that the sufficient condition for convergence of a coupled system is related to the spectral radius of the coupled operator, which is indeed equivalent to the value of aforementioned $|G'(x)|$. We here adopted the same procedure of defining the coupled spectral radius:

Firstly, use the backward Euler scheme and the coupled system Equation (41) is discretized in time as,

$$\begin{bmatrix} 1 & -1/a \\ -1/(b+c\Delta t) & 1 \end{bmatrix} \begin{bmatrix} w_1^{n+1,i+1} \\ w_2^{n+1,i+1} \end{bmatrix} = \begin{bmatrix} 0 \\ bw_2^n - w_1^n \end{bmatrix} \tag{42}$$

where the superscript n stands for the time step counter and i the iterative counter.

Then apply the FP splitting, Equation (43) can be written as,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} w_1^{n+1,i+1} \\ w_2^{n+1,i+1} \end{bmatrix} = \begin{bmatrix} 0 \\ bw_2^n - w_1^n \end{bmatrix} + \begin{bmatrix} (1/a)w_2^{n+1,i} \\ (1/(b+c\Delta t))w_1^{n+1,i} \end{bmatrix} \tag{43}$$

Further rewrite Equation (43) into the following form as,

$$\begin{bmatrix} w_1^{n+1,i+1} \\ w_2^{n+1,i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1/a \\ 1/(b+c\Delta t) & 0 \end{bmatrix}}_{\mathbf{G}} \begin{bmatrix} w_1^{n+1,i} \\ w_2^{n+1,i} \end{bmatrix} + \begin{bmatrix} 0 \\ bw_2^n - w_1^n \end{bmatrix} \tag{44}$$

The spectra radius of the coupled operator is the eigenvalues of the matrix $\mathbf{G}$,

$$\rho(\mathbf{G}) = eig \begin{bmatrix} 0 & 1/a \\ 1/(b + c\Delta t) & 0 \end{bmatrix} = \pm\sqrt{1/(a(b + c\Delta t))} \tag{45}$$

The convergence condition requires that $\rho(\mathbf{G})$ shall be less than 1. Let us now examine whether this criteria could be fulfilled in our case. Recall the model Equations (11) and (12) and ignore the nonlinear terms in this stage, we could figure out the expression of the coefficients a, b, c in terms of the soil properties and the mesh and time step sizes:

$$a = \frac{2\mu + \lambda}{h}, b = \frac{nh}{K'}, c = \frac{k\Delta t}{\gamma h} \tag{46}$$

where h could represent the generalized mesh size. The convergence criteria now can be written out as:

$$a(b + c\Delta t) = \frac{2\mu + \lambda}{K'/n} + \frac{(2\mu + \lambda)k}{\gamma_w} \cdot \frac{\Delta t}{h^2} \geq 1 \tag{47}$$

the above criteria implies that, once the bulk modulus of the soil skeleton is stronger than the effective bulk modulus of the fluid normalized by the porosity, i.e. $2\mu + \lambda \geq K'/n$, the convergence can be guaranteed without limitation on time and mesh conditions. It therefore reveals that the poro-elasto-plastic model might fail to simulate the case when pure water presents in the soil pore, as pure water is often considered as incompressible compared to the soil skeleton. However favorably, for natural soil that even only contains one percent of air in the pore, the fluid bulk modulus is reduced dramatically by a factor of 200 to $10^7 Pa$, which often turns out to be small enough compared to the solid bulk modulus and thus fulfills Equation (47).

Paper III

"Simulation of pore pressure accumulation under cyclic loading using finite volume method"

T. Tang & O. Hededal

Published in: *Numerical Methods in Geotechnical Engineering, 2014*

Numerical Methods in Geotechnical Engineering – Hicks, Brinkgreve & Rohe (Eds)
© 2014 Taylor & Francis Group, London, 978-1-138-00146-6

Simulation of pore pressure accumulation under cyclic loading using finite volume method

T. Tang
Department of Civil Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark

O. Hededal
Department of Civil Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark
Marine & Foundation Engineering, COWI, Kgs. Lyngby, Denmark

ABSTRACT: This paper presents a finite volume implementation of a porous, nonlinear soil model capable of simulating pore pressure accumulation under cyclic loading. The mathematical formulations are based on modified Biot's coupled theory by substituting the original elastic constitutive model with an advanced elasto-plastic model suitable for describing monotonic as well as cyclic loading conditions. The finite volume method is applied to discretize these formulations. The resulting set of coupled nonlinear algebraic equations are then solved by a 'segregated' solution procedure. An efficient return mapping algorithm is used to calculate the stress and strain relation in each control volume level. Test cases show very good performance of the model.

1 INTRODUCTION

In offshore engineering, it is acknowledged that the coupled interactions between the soil skeleton and pore fluid under loading may lead to built-up of pore pressure, yielding material softening and shear strength degradation. Therefore, it poses considerable risks on the stability of offshore structures and/or foundations (Sumer & Fredsøe 2002, Jeng 2003). Numerical analysis of this phenomenon however is a challenging task, due to the inherently strong coupling effects inside the two-phase physical system, and lack of reliable constitutive models that capture soil mechanical behaviors realistically under cyclic loading. For this purpose, a suitable formulation based on the modified Biot's theory (Biot 1941, Zienkiewicz 1982), which takes both the soil-pore fluid coupling and material nonlinearity into account, is employed to simulate porous soil behavior in this paper. A critical state two-surface plasticity model originally proposed by Manzari and Dafalias (1997), which can reproduce cyclic stress-strain behaviors with several important soil features including phase transformation, critical state and peak failure etc., is incorporated as a proper constitutive relation for the soil skeleton.

Moreover, the Finite Volume Method (FVM) is creatively proposed to solve the two-phase soil equation system. FVM is a numerical approach that has been successfully applied in the field of computational fluid dynamics for solving various flow problems. It was Demirdzic and Martinovic (1993) and Jasak and Weller (2000) who first started to employ FVM to successfully solve a number of solid mechanical problems. Possibly, the most attractive feature of FVM revealed by their research is the simplicity and efficiency in dealing with coupling and nonlinearity of the equations through a 'segregated' solution procedure, where each unknown variable is solved sequentially and the coupling and nonlinear terms are lagged. Often, a fixed point iteration method is combined with the segregated procedure to achieve full convergence of the nonlinearly coupled system.

Hence, the main goal of this paper is to present the formulation and FVM implementation of a nonlinear porous soil model capable of simulating the pore pressure accumulation. Numerical tests and application are also included to demonstrate the performance of the implemented model.

2 GENERAL FORMULATION FOR THE TWO-PHASE POROUS SOIL MODEL

In the classic Biot's coupled theory for two-phase porous media, the pore fluid and the solid skeleton are considered compressible obeying linear elasticity, while the flow in the pores is assumed governed by Darcy's law. Here, we will extend the theory to account for more realistic nonlinear material behaviors of the soil skeleton.

The governing equation of the pore fluid flow is formulated as follows:

$$\frac{k}{\gamma_w}\nabla^2 p = \frac{n}{K'}\frac{\partial p}{\partial t} + \frac{\partial}{\partial t}(\nabla \cdot \mathbf{u}) \tag{1}$$

where the unknown variable p is the pore pressure and $\mathbf{u}$ the displacement vector of the soil skeleton. The material properties: k, γ_w, n, K' are the soil permeability, density of water, soil porosity and effective bulk modulus of the pore fluid, respectively. A possible estimation of bulk modulus of the pore fluid K' has been made through the following expression: $1/K' = Sr/K_w + (1 - Sr)/[b(p + p_b) + p_a]$, with Sr being the degree of saturation, K_w the pure water bulk modulus ($\approx 2 \times 10^9$Pa), p_b the back pressure, p_a the atmospheric pressure ($\approx 10^5$Pa) and b a model constant (Bian & Shahrour 2009).

In Eq. (1), the left hand side term represents the rate of pore fluid seepage, and the first and second terms on the right hand side are the rate of volume change for the pore fluid and the soil skeleton phase, respectively. It is hence very clear to see the embedded interactions of pore fluid seepage and the volume change of the soil skeleton from the formulation.

The governing equation of the soil skeleton phase is based on total momentum equilibrium in incremental form. By incorporating the stress-strain relation, small-strain split, and strain-displacement relation below, it is conveniently written as:

$$\begin{cases} \nabla \cdot (\delta\boldsymbol{\sigma}) = \nabla \cdot (\delta\boldsymbol{\sigma}' - \delta p\mathbf{I}) = \mathbf{0} \\ \delta\boldsymbol{\sigma}' = 2G\delta\boldsymbol{\varepsilon}^e + (K - \frac{2}{3}G)\mathrm{tr}(\delta\boldsymbol{\varepsilon}^e) \\ \delta\boldsymbol{\varepsilon}^e = \delta\boldsymbol{\varepsilon} - \delta\boldsymbol{\varepsilon}^p \\ \delta\boldsymbol{\varepsilon} = \frac{1}{2}\left[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}\right] \end{cases}$$

$$\Downarrow$$

$$\nabla \cdot \left[G\nabla(\delta\mathbf{u}) + G\nabla(\delta\mathbf{u})^{\mathrm{T}} + (K - \tfrac{2}{3}G)\mathbf{I}\nabla \cdot (\delta\mathbf{u})\right] \\ -\nabla \cdot \left[K\left(\delta\boldsymbol{\varepsilon}_v^p\right) + 2G\left(\delta\boldsymbol{\varepsilon}_d^p\right)\right] - \nabla(\delta p) = \mathbf{0} \qquad (2)$$

where $\delta\boldsymbol{\sigma}, \delta\boldsymbol{\sigma}'$ are the incremental total stress and incremental effective stress. The strain variables: $\delta\boldsymbol{\varepsilon}, \delta\boldsymbol{\varepsilon}^e, \delta\boldsymbol{\varepsilon}^p$ are the incremental total strain, incremental elastic strain, incremental plastic strain respectively. The primary unknown variable $\delta\mathbf{u}$ symbolizes the incremental displacement vector of the soil skeleton, and δp the incremental pore pressure. The other two variables: $\delta\boldsymbol{\varepsilon}_v^p, \delta\boldsymbol{\varepsilon}_d^p$ are the volumetric and deviatoric part of incremental plastic strain, respectively. The soil skeleton properties K, G are the elastic bulk and shear modulus. $\mathbf{I}$ is the identity tensor and $^{\mathrm{T}}$ the tensor transformation operator. Here, tension is assumed positive as normally employed in computational continuum mechanics.

Eq. (2), which represents the momentum balance, may look different from its equivalent forms employed in other works e.g. (Zienkiewicz 1982), as it has been particularly formulated to fit the 'segregated' solution procedure in FVM. The first term in the equation can be considered as the 'predicted' internal force due to soil skeleton deformation with purely elastic response, while the second term denotes a 'correcting' internal force due to soil skeleton that might undergo plastic deformation, and the third term is the internal force due to pore fluid flow. Again, it is very clear to see from the equation that the influence of pore fluid pressure on the nonlinear deformation of the soil skeleton is captured.

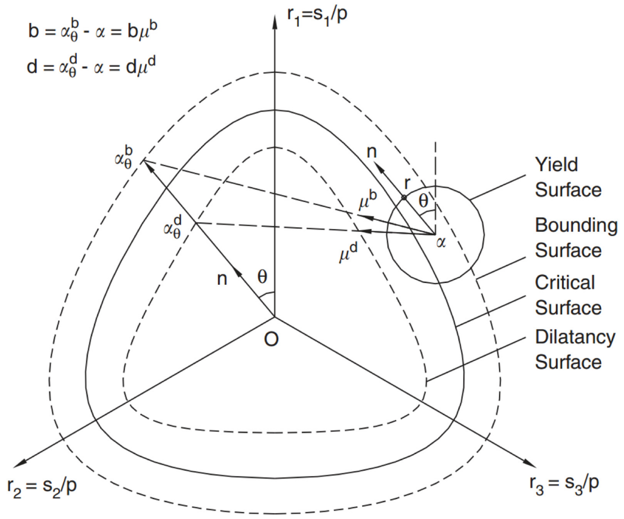


Figure 1. Schematic illustration of different surfaces, adopted from (Manzari & Dafalias 1997).

Eqs. (1–2) make a set of four equations with four primary unknowns – pore pressure p and incremental displacement vector $\delta\mathbf{u}$. To complete the mathematical model, specifications for initial conditions and boundary conditions such as impermeable and traction boundaries are also required.

3 THE CRITICAL STATE TWO-SURFACE PLASTICITY CONSTITUTIVE MODEL

Determination of the 'correcting' internal force due to plastic deformations, namely $\delta\boldsymbol{\varepsilon}_v^p$ and $\delta\boldsymbol{\varepsilon}_d^p$, has to be done with a proper soil constitutive model. Here, we briefly summarize the essential features of the chosen critical state two-surface plasticity constitutive model. A detailed description is found in Manzari and Dafalias (1997).

This model is based on the framework of critical state soil mechanics (Wood 1990), and combines a state parameter, ψ, originally proposed by Been and Jefferies (1985). The state parameter ψ defines the soil state based on combined effects of initial density (void ratio) and confining stress. It is then used to determine the bounding (peak) and dilatancy surfaces, see a schematic illustration in Fig. (1) below. All the surfaces have been formulated in multiaxial stress space by considering the effect of Lode angle θ.

The yield surface is a cone-type shape with circular cross-section in π-plane, determined by two internal variables: m and $\boldsymbol{\alpha}$, i.e. the radius and position of the axis of the cone. The evolution of parameter $\boldsymbol{\alpha}$ allows the model to describe plastic deformations in both reverse and forward loading.

Moreover, mean stress dependent elastic bulk and shear modulus have also been included in the model.

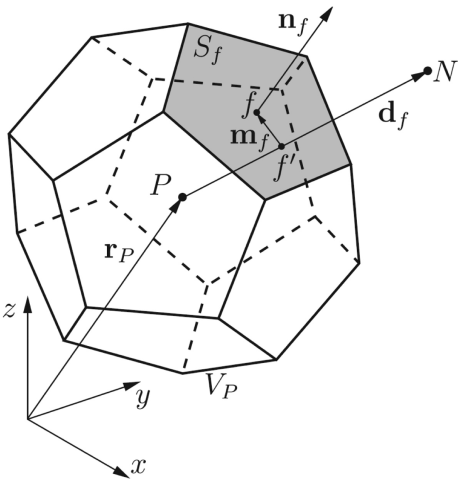


Figure 2. Polyhedral control volume (CV) in FVM, adopted from (Jasak & Weller 2000).

4 NUMERICAL METHOD AND IMPLEMENTATION

In this section, the cell-centered FVM is applied to discretise the aforementioned equation system, i.e. Eqs. (1–2). The method firstly requires a subdivision of the solution domain into a finite number of contiguous, non-overlapping control volumes (CVs), with a computational node in its center, as shown in Fig. 2.

For the equation discretisation, we describe our special strategy tailored to the soil model only, as details of finite volume discretisation itself are available in the references (Demirdzic & Martinovic 1993, Jasak & Weller 2000):

$$\underbrace{\frac{k}{\gamma_w}\nabla^2 p}_{Implicit} = \underbrace{\frac{n}{K'}\frac{\partial p}{\partial t}}_{Implicit} + \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{Explicit} \tag{3}$$

$$\begin{aligned} &\underbrace{\nabla \cdot [(K + \tfrac{4}{3}G)\nabla(\delta\mathbf{u})]}_{Implicit} - \underbrace{\nabla \cdot [(K + \tfrac{1}{3}G)\nabla(\delta\mathbf{u})]}_{Explicit} \\ &+ \underbrace{\nabla \cdot [G\nabla(\delta\mathbf{u})^{\mathrm{T}} + (K - \tfrac{2}{3}G)\mathbf{I}\nabla \cdot (\delta\mathbf{u})]}_{Explicit} \\ &- \underbrace{\nabla \cdot [K\,(\delta\boldsymbol{\varepsilon}_v^p) + 2G\,(\delta\boldsymbol{\varepsilon}_d^p)]}_{Explicit} - \underbrace{\nabla(\delta p)}_{Explicit} = \mathbf{0} \end{aligned} \tag{4}$$

As marked in the equations, two types of discretisation are selected for each term: *Implicit* discretisation and *Explicit* discretisation. The *Implicit* discretisation constructs the matrix coefficients and unknown variable vectors of the large algebraic equation system, while the *Explicit* discretisation uses the current available value of the variables directly and therefore contributes to the source terms.

It can be noted that these coupled terms: $\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})$ in the pore fluid flow equation and $\nabla(\delta p)$ in the momentum equation, and the nonlinear terms: $\nabla \cdot [K(\delta\boldsymbol{\varepsilon}_v^p) + 2G(\delta\boldsymbol{\varepsilon}_d^p)]$, have all been treated as *explicit.* Extra *explicit* portions in the momentum equation are the displacement cross-component coupling term: $\nabla \cdot \left[G\nabla(\delta\mathbf{u})^{\mathrm{T}} + (K - \frac{2}{3}G)\mathbf{I}\nabla \cdot (\delta\mathbf{u})\right]$ and an over-relaxed term $\nabla \cdot \left[(K + \frac{1}{3}G)\nabla(\delta\mathbf{u})\right]$. Using the above *Implicit-Explicit* split, the nonlinear coupled model reduces to a set of small linear decoupled equations, easily solvable through a segregated solution procedure, see Section 4.1.

Note that in this model since the elastic moduli, K and G, and the soil porosity n are dependent on soil mean stress level and volumetric deformations, the *Implicit* discretisation terms – $\nabla \cdot \left[(K + \frac{4}{3}G)\nabla(\delta\mathbf{u})\right]$ in the momentum equation and $\frac{n}{K'}\frac{\partial p}{\partial t}$ in the flow equation – contain nonlinearity and therefore indeed can not be treated fully implicitly. However, the adopted iterative segregated approach allows these nonlinear coefficients to be dealt with in the same natural way as the other coupling and nonlinear terms: namely they will be updated explicitly through iterations.

4.1 *Segregated solution procedure*

As a result of above FVM discretization, a system of linear algebraic equations:

$$\mathbf{Ax} = \mathbf{b} \tag{5}$$

is created for each of four primary unknowns – the pore pressure p and three components of incremental displacement vector $\delta\mathbf{u}$. Where $\mathbf{A}$ is an $N \times N$ symmetric, sparse and diagonally dominant matrix, vector $\mathbf{x}$ contains values of unknown variables at N nodal points, and $\mathbf{b}$ the source vector covering those explicit contributions from coupling and nonlinearity.

Eq. (5) are solved sequentially for each unknown following the order: first the pore pressure p and then the three components of $\delta\mathbf{u}$ in parallel. After solution of all four unknowns is performed, one iteration is completed and unless a converged solution is obtained, the coefficient $\mathbf{A}$ and source $\mathbf{b}$ are updated using the new solutions and next iteration proceeds. Fig. (3) depicts such solution algorithm.

It may be argued that these iterative procedures require large computational effort. However, a great advantage lies in that the matrix systems in each iteration are ideally suitable for efficient linear iterative solvers. And in fact, there is no need to solve these equations to a tight tolerance, as sources and coefficients are only approximated (based on the previous iteration). Hence, the overall simplicity and relative efficiency of this segregated solution procedure makes it a proper choice for solving our complicated nonlinear coupled soil model.

4.2 *Stress-strain update*

Inside each global iteration, after the displacement vector of the soil skeleton in the whole domain has been solved, it is important to calculate the corresponding stress and strain, especially the plastic strains $\delta\boldsymbol{\varepsilon}_v^p$ and $\delta\boldsymbol{\varepsilon}_d^p$, at each CV center based on the chosen constitutive model. An efficient return mapping

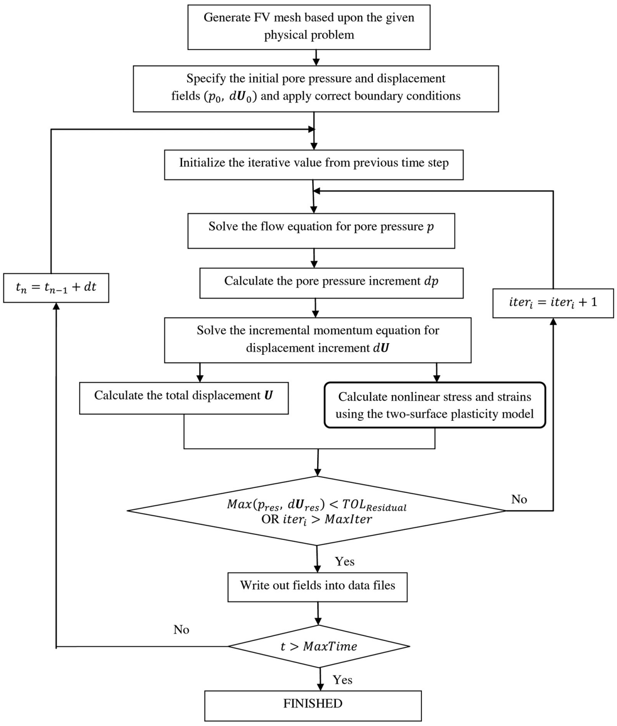

Generate FV mesh based upon the given physical problem
Specify the initial pore pressure and displacement fields (p_0, $d\boldsymbol{U}_0$) and apply correct boundary conditions
Initialize the iterative value from previous time step
Solve the flow equation for pore pressure p
Calculate the pore pressure increment dp
Solve the incremental momentum equation for displacement increment $d\boldsymbol{U}$
Calculate the total displacement $\boldsymbol{U}$
Calculate nonlinear stress and strains using the two-surface plasticity model
$Max(p_{res}, d\boldsymbol{U}_{res}) < TOL_{Residual}$ OR $iter_i > MaxIter$
No
Yes
Write out fields into data files
$t_n = t_{n-1} + dt$
$iter_i = iter_i + 1$
No
$t > MaxTime$
Yes
FINISHED

Figure 3. The global iterative 'segregated' solution procedure.

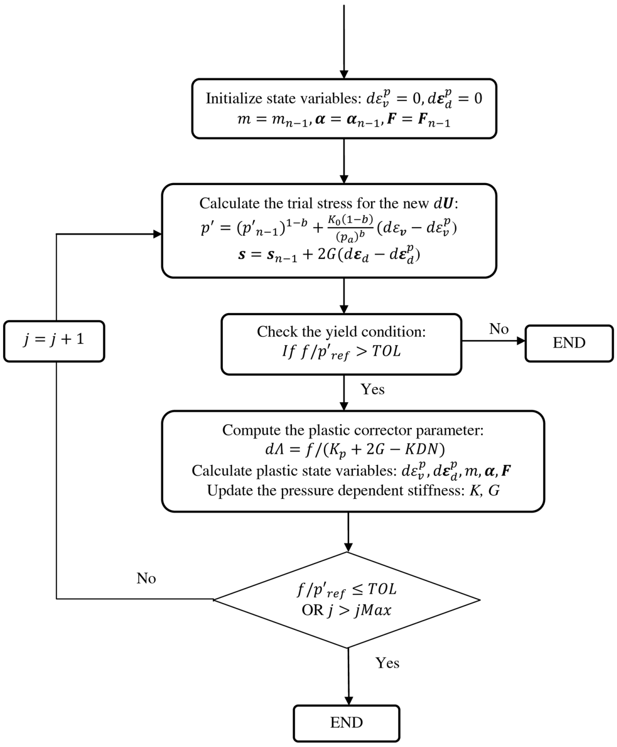

Initialize state variables: $d\varepsilon_v^p = 0, d\boldsymbol{\varepsilon}_d^p = 0$
$m = m_{n-1}, \boldsymbol{\alpha} = \boldsymbol{\alpha}_{n-1}, \boldsymbol{F} = \boldsymbol{F}_{n-1}$
Calculate the trial stress for the new $d\boldsymbol{U}$:
$p' = (p'_{n-1})^{1-b} + \frac{K_0(1-b)}{(p_a)^b}(d\varepsilon_v - d\varepsilon_v^p)$
$\boldsymbol{s} = \boldsymbol{s}_{n-1} + 2G(d\boldsymbol{\varepsilon}_d - d\boldsymbol{\varepsilon}_d^p)$
$j = j + 1$
Check the yield condition:
$If\ f/p'_{ref} > TOL$
No
END
Yes
Compute the plastic corrector parameter:
$d\Lambda = f/(K_p + 2G - KDN)$
Calculate plastic state variables: $d\varepsilon_v^p, d\boldsymbol{\varepsilon}_d^p, m, \boldsymbol{\alpha}, \boldsymbol{F}$
Update the pressure dependent stiffness: K, G
No
$f/p'_{ref} \leq TOL$
OR $j > jMax$
Yes
END

Figure 4. Local stress-strain update procedure at each CV center.

algorithm, adopted from LeBlanc et al. (2008), has therefore been tailored to achieve the goal. Fig. (4) below summarizes the essential steps of this algorithm.

4.3 *Implementation*

The described discretisation, global solution procedure and stress-strain update have been implemented as a custom OpenFOAM solver named twoSurfacePlastBiotFoam, where OpenFOAM is an open source software having various FVM classes freely accessible in C++ language.

Table 1. Soil properties used in the simulations, with the plasticity model parameters adopted from (Manzari & Dafalias 1997).

Elasticity		Pore fluid			
G_0 (MPa)	31.4	k (m/s)	0.0001	p_a (kPa)	100
K_0 (MPa)	31.4	K_w (GPa)	2.1	γ_w (kPa)	10
a	0.6	S_r	0.90~1.00		
Critical state		State parameter			
M_c	1.2	k_c^b	3.975		
M_e	0.857	k_e^b	2.0		
λ	0.025	k_c^d	4.2		
$(e_c)_{\text{ref}}$	1.2	k_e^d	0.07		
Dilatancy		Hardening			
A_0	0.6	h_0	800		
C_f	100	m	0.05		
$F_{\max}$	100	c_m	0.0		

5 TESTS AND APPLICATION OF THE METHOD

In order to validate the implemented code, first, simulations of small-scale undrained triaxial tests on full 3D saturated soil samples are presented. Thereafter a hypothetical test case with wave pressure loading on large-scale porous seabed is performed. For simplicity, all the simulations use the same soil properties summarized in Table 1.

5.1 *Monotonic and cyclic undrained triaxial tests*

Undrained monotonic compression and extension tests are simulated for two soil samples at different void ratios at a confining pressure of 160 kPa yielding $e_c = 0.80$. The distinctly different responses of a soil state looser than critical ($e_0 = 0.85$) and denser than critical ($e_0 = 0.79$) are well predicted in the simulation results presented in Fig. 5. Loose soil is showing softening, while dense soil is showing hardening behavior, which conforms with experimental evidence. Large impact of degree of saturation on soil strength is predicted in the analysis (Fig. 5 right) as also predicted by Bian and Shahrour (2009).

Simulations of cyclic triaxial tests are then conducted for initial void ratios of 0.65 and 0.85, respectively. The soils are firstly isotropically consolidated to a mean confining stress of $p' = 80$ kPa and thereafter loaded by cyclic shear stress with the amplitude of 30~40 kPa and the frequency of 1 Hz. As it can be clearly seen from Fig. 6, the well-known butterfly-shape stress path is observed for the dense soil that undergoes 'cyclic mobility' failure with accumulated large shear strains. Likewise, the complete loss of

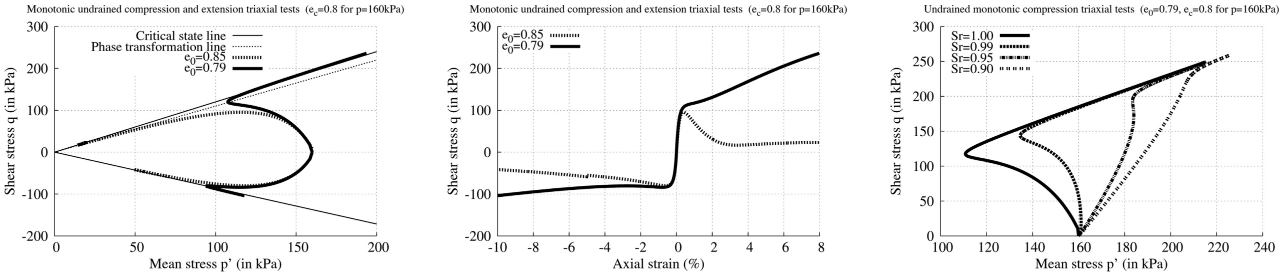


Figure 5. Simulated stress paths (left, right) and stress-strain curve (middle) in monotonic undrained triaxial tests on full soil sample.

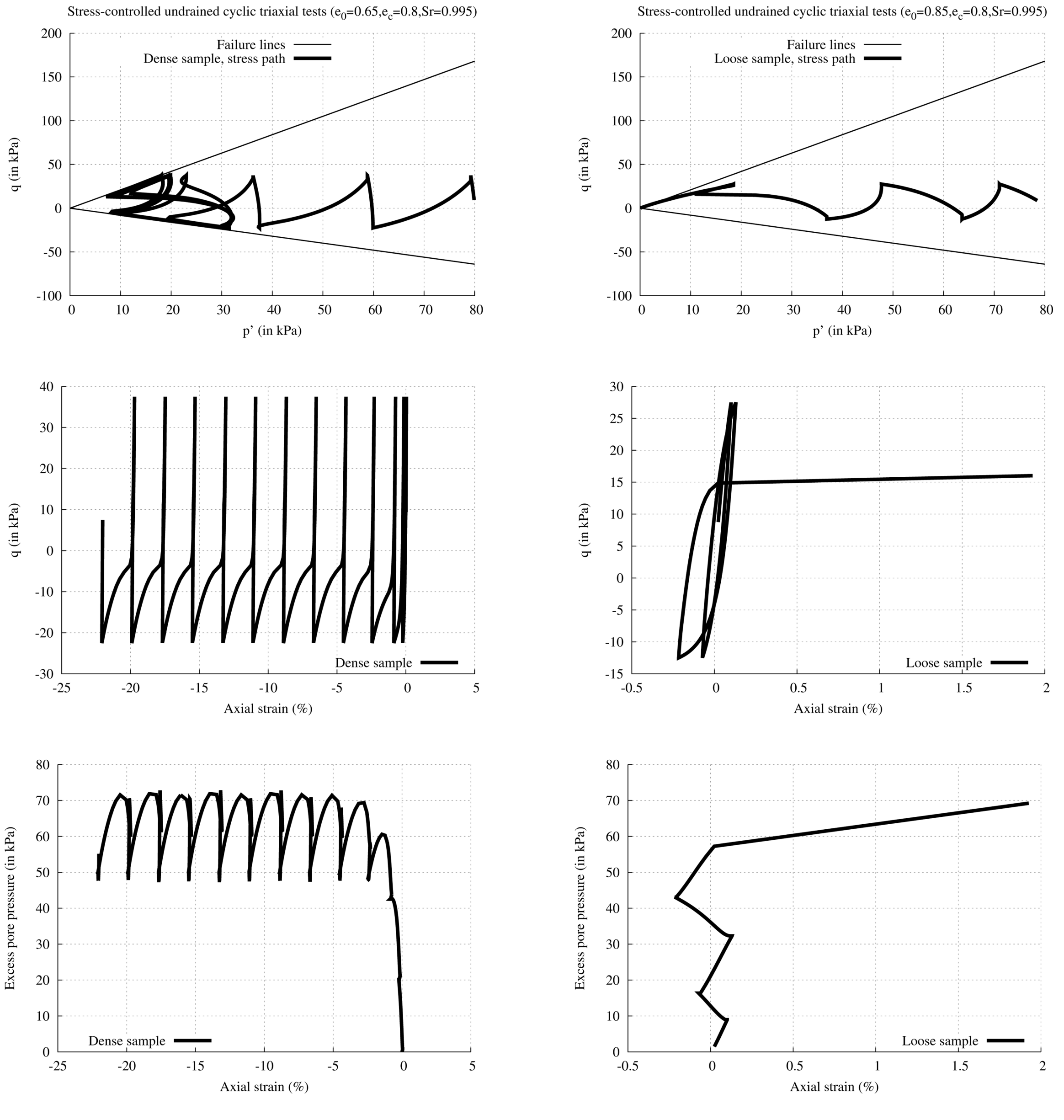


Figure 6. Simulation of stress-controlled cyclic undrained triaxial tests on a dense soil sample (left) and a loose soil sample (right).

mean effective stress of a loose sample leading to liquefaction failure is captured successfully.

5.2 *Standing wave pressure loading on seabed*

A hypothetical case of standing wave pressure acting on top of porous seafloor is also explored here. The objective of this example is to demonstrate the capability of predicting accumulated excess pore pressure inside the seabed, due to a combined effect of cyclic wave loads and presence of a impermeable object on top resulting in a partially undrained condition. The generated seabed mesh is shown in Fig. (7) with lateral dimensions equal to the applied wave

Table 2. Boundary conditions.

Patch	Boundary type
Seabed sides, bottom and the object	impermeable & zero displacement
Seabed top	standing wave pressure: amplitude = 5 kPa, period = 0.1 s & free traction

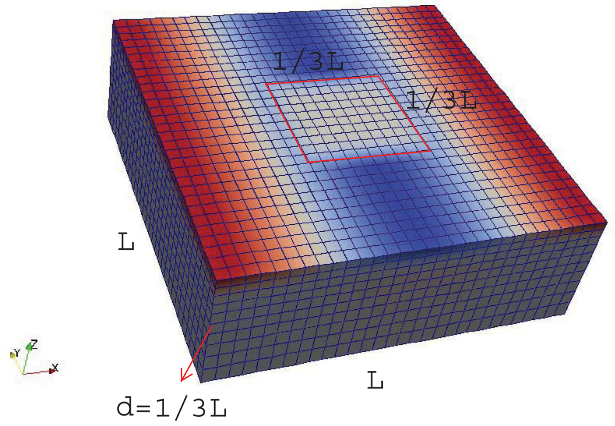


Figure 7. Mesh setup of the porous seabed case with dimensions.

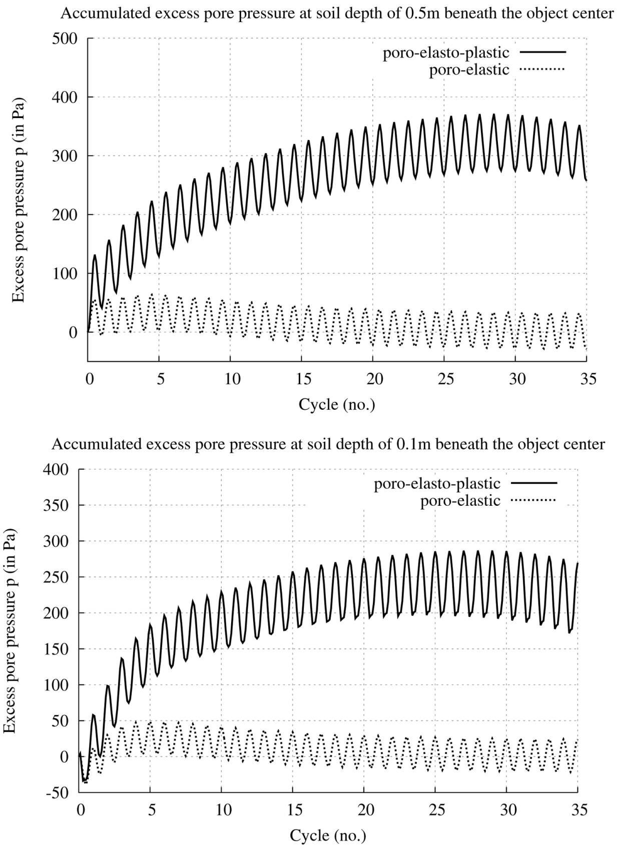


Figure 8. Simulated pore pressure accumulations at soil depth of 0.5 m (top) and 0.1 m (bottom) underneath the object center.

length ($l = L$) and vertical depth od one third of the wave length ($d = 1/3L$). The impermeable rectangular object occupies one ninth of the whole seafloor.

The boundary conditions are presented in Table 2.

The simulated excess pore pressure variations, cycle by cycle, at two different soil depths have been plotted in Fig. 8. Results from the classic Biot's theory, i.e. the poro-elastic model, are also presented for comparison. The poro-elasto-plastic model developed in this paper predicts a gradual accumulation of excess pore pressure inside the soil corresponding to a gradual decrease of pore volume, while the simple poro-elastic model – which has constant pore volume – only captures the steady state pore pressure variation. Analysis of the poro-elasto-plastic simulation results implies that, shear strains induced by the standing wave pressure gradually rearrange the soil grains at the expense of the pore volume of the soil. The latter pressurizes the pore water, and as a result, lead to the built-up of pore pressure.

6 CONCLUSIONS

In this study, the described FVM discretisation strategy and iterative 'segregated' solution procedure offers an interesting avenue to model complicated porous soil material, namely having soil – pore fluid coupled interactions as well as highly nonlinear constitutive behavior, in a simple and effective manner.

ACKNOWLEDGEMENT

We would like to thank Dr. Johan Roenby and Dr. Philip Cardiff for their supports during the first author's OpenFOAM implementation.

REFERENCES

Been, K. & M. Jefferies (1985). A state parameter for sands. *Geotechnique 35 (2)*, 99–112.

Bian, H. & I. Shahrour (2009). Numerical model for unsaturated sandy soils under cyclic loading: Application to liquefaction. *Soil Dyna. Earth. Engng. 29*, 237–244.

Biot, M. (1941). General theory of three dimensional consolidation. *J. Appl. Phys. 12 (2)*, 155–164.

Demirdzic, I. & D. Martinovic (1993). Finite volume method for thermo-elasto-plastic stress analysis. *Com. Meth. Appl. Mech. Engng. 109 (3–4)*, 331–349.

Jasak, H. & H. Weller (2000). Application of the finite volume method and unstructured meshes to linear elasticity. *Int. J. Num. Meth. Engng. 48 (2)*, 267–287.

Jeng, D. (2003). Wave-induced sea floor dynamics. *Applied Mechanics Reviews 56 (4)*, 407–429.

LeBlanc, C., O. Hededal, & L. Ibsen (2008). A modified critical state two-surface plasticity model for sand – theory and implementation. *DCE Technical Memorandum Aalborg Uni. 8*.

Manzari, M. & Y. Dafalias (1997). A critical state two-surface plasticity model for sands. *Geotechnique 47 (2)*, 255–272.

Sumer, B. & J. Fredsøe (2002). *The Mechanics of Scour in the Marine Enviroment*. River Edge, N.J.: World Scientific.

Wood, D. (1990). *Soil Behaviour and Critical State Soil Mechanics*. New York: Cambridge Uni. Press.

Zienkiewicz, O. (1982). Basic formulation of static and dynamic behaviours of soil and other porous media. *Appl. Math. Mech. 3 (4)*, 457–468.

Paper IV

"A FVM multi-physics simulation of wave-seabed-structure interaction using OpenFOAM"

T. Tang, B. Johannesson & J. Roenby

Published in: *Proceedings of the 9th OpenFOAM Workshop, 2014*

9th OpenFOAM® Workshop
June 23- 26, 2014
Zagreb, CROATIA

A FVM multi-physics simulation of wave-seabed-structure interaction using OpenFOAM®

Tian Tang*[1], Björn Johannesson[2] and Johan Roenby[3]

[1,2] *Civil Engineering Department, Technical University of Denmark.*
Anker Engelunds Vej 1, 2800 Kgs. Lyngby, Denmark.
[3] *Ports and Offshore Technology, DHI*
Agern Alle 5, 2970 Horsholm, Denmark.

Abstract

The open source finite volume based C++ code library OpenFOAM®can solve a variety of problems within computational continuum mechanics due to its tensorial approach and object-oriented nature. In this paper, we demonstrate how OpenFOAM®can be used as a platform for multi-physics simulations involving wave-seabed-structure interaction. Our primary concern is the seabed soil response to ocean waves in the vicinity of offshore structures. Hence a wave tank sub domain is modeled with the aid of the wave generation toolbox `waves2Foam`. The resulting pressure field on the seabed varies both in space and time and is used as an input boundary condition for the underlying soil sub-domain. For the soil simulation a newly developed nonlinear porous soil solver, `twoSurfacePlastBiotFoam`, is applied in order to assess the possibility of pore pressure build-up and risk of soil failure. Test cases demonstrate the performance and applicability of the simulation method.

1. INTRODUCTION

When designing offshore structures such as pipelines, breakwaters and wind turbine foundations it is essential to be able to assess their geotechnical stability. This involves evaluation of wave-induced pore pressure, soil stresses and groundwater flow. Typically in conventional geotechnical soil modeling simplified wave pressure fields derived from analytical wave theories are applied on top of the seabed [1, 2]. When structures of complicated geometry are present in the water such analytical expressions are not available. Better approximations can be achieved by using multi-physics numerical models where, the porous soil model for the seabed domain is augmented by computational fluid dynamics (CFD) simulations yielding accurate pressure modifications due to the presence of the structure.

Recently, the freely available C++ finite volume method (FVM) library, OpenFOAM®, has gained popularity in a wide range of fields within computational continuum mechanics, ranging from complex fluid flow problems, to solid mechanics and electromagnetism. OpenFOAM®possesses great potentials for multi-physics numerical simulations due to its general structure. Currently, the officially released versions do not include solvers directly dedicated to porous seabed modeling. However, the tensorial approach and object-oriented nature of OpenFOAM®[3] makes it relatively easy to implement new tailor-made soil solvers for porous seabed problems. Indeed, Liu and Garcia [4] successfully made the first effort on

[*] Corresponding author: Tian Tang (tiat@byg.dtu.dk)

investigating seabed response under waves using OpenFOAM®. They applied the original OpenFOAM solver for two immiscible incompressible fluids (water and air) to produce a wave field, and implemented a new poro-elastic soil solver based on Biot's consolidation theory for the seabed behaviour. Still, a key element lacking within their research is the incorporation of a realistic soil stress-strain constitutive relation: The Biot equations are not capable of predicting the nonlinear soil skeleton behaviour and the gradual built-up of excess pore pressure under cyclic loading. Implementation and validation of a soil solver that include these important phenomena is the main goal of the present work.

The structure of this paper is as follows. In section 2 we briefly review the governing equations for the two different domains - sea and seabed. The adopted numerical solvers for solving the two systems are introduced in Section 3, where a detailed account of the solution algorithm in the newly implemented cyclic nonlinear porous soil solver is given. In Section 4, test cases are provided to demonstrate the overall performance of the simulations. Finally in Section 5 discussions and conclusion are presented.

2. GOVERNING EQUATIONS

The governing equations for the free surface water waves and the nonlinear porous seabed are briefly described below.

2.1 Wave domain: Reynolds averaged Navier-Stokes equations

The combined flow of air and water above the seabed is governed by the incompressible Reynolds averaged Navier-Stokes (RANS) equations:

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}^{\mathrm{T}}) = -\nabla p^* - (\mathbf{g}\cdot\mathbf{x})\nabla\rho + \nabla\cdot(\mu\nabla\mathbf{v} + \rho\tau) \tag{1}$$

$$\nabla\cdot\mathbf{v} = \mathbf{0} \tag{2}$$

where, $\mathbf{v}$ is the velocity vector field in Cartesian coordinates $\mathbf{x}$, p^* is the pressure in excess of the hydrostatic pressure ($p^* = p^{tot} - \rho\mathbf{g}\cdot\mathbf{x}$), $\rho = \rho(\mathbf{x})$ is the density taking the values 1 and 1000 kg/m^3 at regions occupied by air and water, respectively. Likewise the viscosity μ takes the values $1.48\cdot 10^{-5}$ and 10^{-3} kg/ms in air and water, respectively. The constant vector $\mathbf{g}$ is the gravity acceleration. The specific Reynolds stress tensor τ is defined as $\tau = \frac{2}{\rho}\mu_t\mathbf{S} - \frac{2}{3}k\mathbf{I}$ and $\mathbf{S} = \frac{1}{2}(\nabla\mathbf{v} + \nabla\mathbf{v}^{\mathrm{T}})$, with μ_t standing for the dynamic eddy viscosity, $\mathbf{S}$ being the strain rate tensor and k the turbulent kinetic energy.

The two immiscible fluids are tracked using a scalar field γ, which is the cell averaged density shifted and normalized to a volume fraction taking the value 0 for air filled cells, 1 for water filled and intermediate values for partially water filled cells. The transportation equation for γ is modeled by:

$$\frac{\partial\gamma}{\partial t} + \nabla\cdot(\mathbf{v}\gamma) + \nabla\cdot[\mathbf{v}_r(1-\gamma)] = 0 \tag{3}$$

where $\mathbf{v}_r$ is a relative velocity [5]. Using γ, the spatial variation of any fluid property Φ (such as viscosity μ and density ρ) can be given by $\Phi = \gamma\Phi_{\text{water}} + (1-\gamma)\Phi_{\text{air}}$.

Turbulence is modeled using a $k-\omega$ closure model:

$$\frac{\partial\rho k}{\partial t} + \nabla\cdot(\rho\mathbf{v}k) = \mathrm{P}_k - \beta^* k\omega + \nabla\cdot[(\mu + \sigma^*\mu_t)\nabla k] \tag{4}$$

$$\frac{\partial\rho\omega}{\partial t} + \nabla\cdot(\rho\mathbf{v}\omega) = \alpha\mathrm{P}_\omega - \beta\omega^2 + \frac{\sigma_d}{\omega}\nabla k\cdot(\nabla\omega)^{\mathrm{T}} + \nabla\cdot[(\mu + \sigma_\omega\frac{k}{\omega})\nabla\omega] \tag{5}$$

$$\mu_t = \rho\frac{k}{\tilde{\omega}}, \quad \tilde{\omega} = \max(\omega, \frac{C_{lim}}{\sqrt{\beta^*}}\sqrt{2\mathbf{S}:\mathbf{S}}) \tag{6}$$

where, ω is the characteristic frequency for the turbulence. The closure coefficients: β^*, σ^*, α, β, σ_d, σ_ω and C_{lim} take the standard values from literature. The production terms, P_k and P_ω, can be determined via standard formulations, e.g. on the basis of the rotation of velocity field [6].

2.2 Seabed domain: Modified Biot's consolidation equations

The nonlinear porous seabed (filled with water and/or air in the pores) is governed by the modified Biot's consolidation equations. The term 'modified' is used since the original elastic soil constitutive model has been substituted with an advanced cyclic plasticity model suitable for describing both monotonic and cyclic loading condition:

$$\frac{k}{\gamma_w}\nabla^2 p = \frac{n}{K'}\frac{\partial p}{\partial t} + \frac{\partial}{\partial t}(\nabla \cdot \mathbf{u}) \tag{7}$$

$$\nabla \cdot \left[G\nabla(\delta\mathbf{u}) + G\nabla(\delta\mathbf{u})^{\mathrm{T}} + (K - \frac{2}{3}G)\mathbf{I}\nabla\cdot(\delta\mathbf{u})\right] - \nabla\cdot\left[K\left(\delta\varepsilon_v^p\right) + 2G\left(\delta\varepsilon_d^p\right)\right] - \nabla(\delta p) = \mathbf{0} \tag{8}$$

Here p is the pore pressure, $\mathbf{u}$ is the soil displacement vector and $\varepsilon^p_{v(d)}$ is the volumetric (deviatoric) plastic strain. The δ symbolizes the increment of any of the aforementioned variables. The material properties are the permeability, k, the specific weight of water, γ_w, the porosity, n, the true bulk modulus of the pore fluid, K', and the bulk and shear moduli of the soil skeleton, K and G, respectively. For unsaturated soil the combined flow of water and air in the pores is simply considered as a single fluid, with the approximated bulk modulus $K' = \frac{Sr}{K_w} + \frac{(1-Sr)}{p_0}$, where Sr is the saturation factor, K_w is the pure water bulk modulus ($\approx$ 2GPa), and p_0 the absolute (not excess) pore-water pressure.

Eq. 7 describes the mass balance of the pore fluid and Eq. 8 represents the total momentum balance of the soil mixture. The total momentum equation is written in incremental form so as to incorporate the selected nonlinear cyclic plasticity constitutive model. A brief description of the model formulations is listed as following:

$$\text{Yield function:}\quad f = \sqrt{(\mathbf{s} - p'\alpha) : (\mathbf{s} - p'\alpha)} - \sqrt{2/3}mp' = 0 \tag{9}$$

$$\text{Hardening law:}\quad \delta\alpha = \tilde{\alpha}\delta\lambda, \quad \tilde{\alpha} = h_0\frac{|\mathbf{b}:\mathbf{n}|}{b_{ref} - |\mathbf{b}:\mathbf{n}|}\mathbf{b} \tag{10}$$

$$\delta m = \tilde{m}\delta\lambda, \quad \tilde{m} = c_m(1+e_0)D \tag{11}$$

$$\text{Flow rule:}\quad \delta\varepsilon_v^p = D\delta\lambda, \quad D = A_0(1 + \langle\mathbf{F}:\mathbf{n}\rangle)\mathbf{d}:\mathbf{n} \tag{12}$$

$$\delta\varepsilon_d^p = \mathbf{n}\delta\lambda \tag{13}$$

$$\delta\lambda = \frac{2G\mathbf{n}:\delta\varepsilon_d - KN\delta\varepsilon_v}{K_p + 2G - KDN}, \quad K_p = p'(\mathbf{n}:\tilde{\alpha} + \sqrt{2/3}\tilde{m}) \tag{14}$$

Here $\mathbf{s}$ is the deviatoric stress tensor and p' is the effective mean stress. The internal-state variables denoted by m and α determine the 'size' and the position of the axis of the yield surface (cone-type), respectively. The yield function f describes the bounded region in a general stress space within which elastic and recoverable deformations occur. The yield surface can also undergo dynamical expansion (hardening of the soil during plastic deformations), specified by the hardening law. In the yielding process, the flow rule determines the relative magnitude of various components (i.e. volumetric and deviatoric) of plastic deformations. The volumetric plastic deformation $\delta\varepsilon_v^p$ is calculated from the volumetric part D (the dilatancy coefficient) of a defined scalar plastic multiplier $\delta\lambda$, while the deviatoric plastic strain $\delta\varepsilon_d^p$ is from the deviatoric part $\mathbf{n}$ of $\delta\lambda$. Detailed explanations on $\delta\lambda$, the volumetric part N of the normal to the yield surface, the plastic modulus K_p, the 'distance' vectors $\mathbf{b}$ and $\mathbf{d}$, the fabric tensor $\mathbf{F}$, and the model parameters h_0, b_{ref}, c_m and A_0 - can be found in [7, 8].

In the work presented here the offshore structures are simply assumed to be rigid and fixed bodies and hence no governing equations are specified for the structure domain.

3. NUMERICAL METHODS

This section presents the numerical procedures applied to solve the governing equations in the two domains.

3.1 `waves2Foam`

`waves2Foam` is a freely available package in OpenFOAM®developed by Jacobsen [9] for generating and absorbing free surface water waves. It contains a solver, `waveFoam`, for free surface Newtonian flows using the RANS equations coupled with the volume of fluid method (Eq. 3). Its main features are the wave generation options at the inlet boundary supported by a large range of wave theories, plus the custom geometrical-shape relaxation zones at the inlet and outlet boundaries for wave absorption. Examples of application and validation of the `waves2foam` toolbox can be found in [6].

3.2 `twoSurfacePlastBiotFoam`

`twoSurfacePlastBiotFoam` is a newly developed nonlinear porous soil solver implemented by the first author of this article on the basis of the FVM discretization library in OpenFOAM®[10]. It solves the aforementioned modified Biot's consolidation equations supplemented with the nonlinear two surface cyclic plasticity constitutive model (Eq. 9-14).

There are two main challenges in the solution procedure for Eqs. 7 and 8: one is the strong nonlinearity in the total momentum equation caused by the cyclic plasticity constitutive relation, and the other is the strong pressure-displacement coupling due to the presence of the volume change term in the storage equation (Eq. 7) and the presence of the pore pressure gradient term in the momentum equation (Eq. 8). The 'segregated' solution algorithm associated with FVM, (which is also the common approach used in most of CFD codes), is adopted to deal with these challenges. Thus, Eq. 7 and 8 are first linearised and decoupled by use of the latest available values of the nonlinear plastic terms and the coupling terms (explicit discretization). This allows the equations to be solved one by one for each solution variable. It is important to apply a certain kind of iteration method – the fixed point iteration method – together with this 'segregated' procedure to recover the full nonlinearity and coupling of the original equations.

The above solution strategy results in four small linear algebraic systems of equations being solved iteratively. Iterations are stopped once a desired convergence has been achieved. When the soil is yielding, the plastic deformations will dominate and significantly impact on the convergence. It may even destroy the convergence as the explicit discretization terms overweight their implicit counterparts. Explicit under-relaxation methods are then used to improve or recover convergence. Appointing a fixed under-relaxation factor or dynamic relaxation factors from e.g. the Aitken's method on the nonlinear plastic terms in each iteration helps to fix the convergence problems for most cases.

Another important issue is the nonlinear stress calculation process. After the displacement field has been solved in the whole soil domain one must compute the corresponding stresses and (total and plastic) strains for each computational cell using the constitutive relation. As previously discussed, the plastic strain terms are calculated using explicitly approximated values inside each global momentum iteration. Applying costly complex implicit stress calculation algorithm is unnecessary. Instead `twoSurfacePlastBiotFoam` uses a simple and economic return-mapping method for the nonlinear stress-strain calculations.

The interested reader can find more details regarding to the solution algorithm and verifications of the `twoSurfacePlastBiotFoam` solver in [10]. The solver itself has also been uploaded as a public repository on Bitbucket for free download, the link can be found through [11].

3.3 Simulation procedure

In the present work we simulate the interactions among the wave, seabed and structure as a one-way coupling process. The wave motion field can be altered by the presence of offshore structure. However the structure itself is assumed rigid and will not move. Also, we focus on investigating the impact from wave loads on the seabed, that is, excess pore pressures, soil displacements and stresses. The feedback from the soil deformations on to the wave field is neglected. From a offshore engineering perspective this is acceptable since generally the magnitude of wave-induced deformations of seabed takes place on a longer time scale than the wave period. The one-way coupling assumption also allows us to solve the wave and seabed domain separately using different time steps and mesh sizes, saving extra computational costs.

The overall simulation procedure used in this paper is as follows: The wave domain is first solved using `waves2Foam` with appropriate boundary condition settings, such as wave inlet(s), outlet(s), and rigid impermeable sea bottom and/or structure. After the wave solution has been obtained for the prescribed total time period, a `sample` utility in OpenFOAM®is applied to extract the wave pressure on the sea bottom and output the data (point positions and pressure values) into files named after the corresponding time step. In cases where a structure is present, the `forces` utility is also used to calculate forces (pressure integration along the structure surface) and output a data file listing wave forces and moments at each time step. Having the above calculated wave load input - i.e. bottom wave pressure and/or structure force, `twoSurfacePlastBiotFoam` is used to solve the corresponding seabed response. A common assumption is made concerning the top of the seabed in which the pore pressure is equal to the wave bottom pressure and that the soil effective stresses vanish [1]. Moreover, simple force balance of the fixed structure is considered to transfer the total wave force on the structure to an uniformly distributed structural force acting on the seabed.

A summary of the different boundary conditions employed for the wave and seabed domains is given in Table 1.

Table 1: Primitive and derived OpenFOAM®boundary conditions used in the simulation

(a) wave domain: `waves2Foam`

	p : *pressure*	$\mathbf{v}$: *velocity*
inlet(s)	`zeroGradient`	`waveVelocity`[1]
outlet(s)	`zeroGradient`	`fixedValue`
sea bottom	`zeroGradient`	`slip`
structure	`zeroGradient`	`slip`
atmosphere	`totalPressure`	`pressureInletOutletVelocity`

[1] `waveVelocity` is a derived wave generation boundary in `waves2Foam`.

(b) seabed domain: `twoSurfacePlastBiotFoam`

	p : *pore pressure*	$\delta\mathbf{u}$: *incremental displacement*
top	`timeVaryingMappedFixedValue`[2]	`zeroTraction`
lateral	`zeroGradient`	`fixedValue` or `slip`
bottom	`zeroGradient`	`fixedValue` or `slip`
structure	`zeroGradient`	`fixedValue` or `timeVaryingTraction`[3]

[2] `timeVaryingMappedFixedValue` is a time-varying, non-uniform OpenFOAM®boundary, it can read those sampled bottom wave pressure data files and assign the corresponding values to the pore pressure field.

[3] `timeVaryingTraction` is a derived traction boundary in `twoSurfacePlastBiotFoam`, it reads the structural force data file and computes the compatible displacement gradient boundary.

4. TEST CASES

Validation of `waves2Foam` as a toolbox for modeling of free surface waves and `twoSurfacePlastBiotFoam` as a nonlinear soil solver for modeling of porous seabed has already been carried out in [6] and [10], respectively. Thus, the presented test cases focus on demonstrating the interaction modeling using the two solvers. Two numerical examples are considered: (i) a validation case of a marine pipeline buried in a porous seabed under wave loadings, and (ii) an application case with a submerged gravity-based structure partially buried in the seabed under a multi-directional wave field.

4.1 Modeling of wave-seabed-pipeline interaction

A marine pipeline is a type of common-used offshore installations. Here we simulate a test case similar to an experimental study conducted at Cornell University [12]. Fig. 1 presents the schematic view of the experiment. The details of this experiment can be found in the original paper. In this test, an uniform, homogeneous porous seabed with a buried pipeline is exposed to a progressive wave loading.

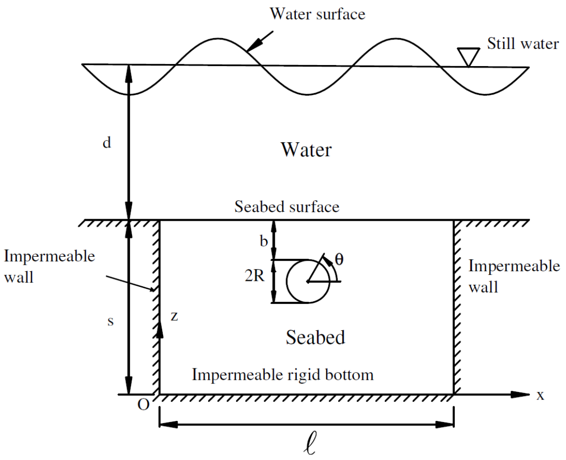


Figure 1: A sketch of the wave-seabed-pipeline interaction example, after [12]

In the simulation, a 2-D wave flume field is modeled by setting a wave generation inlet on the left and a wave absorbing outlet on the right vertical boundary. A finite soil (seabed) trench is placed at the mid-length of the wave flume. On the bottom and sides of the seabed trench all displacements and normal water flux are assumed to vanish. For the pipe, zero displacement and zero flux conditions are applied.

Fig. 2 shows the numerical results of the test case with Fig. 2a showing a snapshot of the simulated free surface water wave, and Figs. 2b-2c showing, respectively, pore pressure contours and shear stresses contours in the seabed. There are clear signs of shear stress concentrations on the sides of the pipeline. These transient plots are all taken at a representative time $t = 20s$ when the wave crest is passing right above the pipeline.

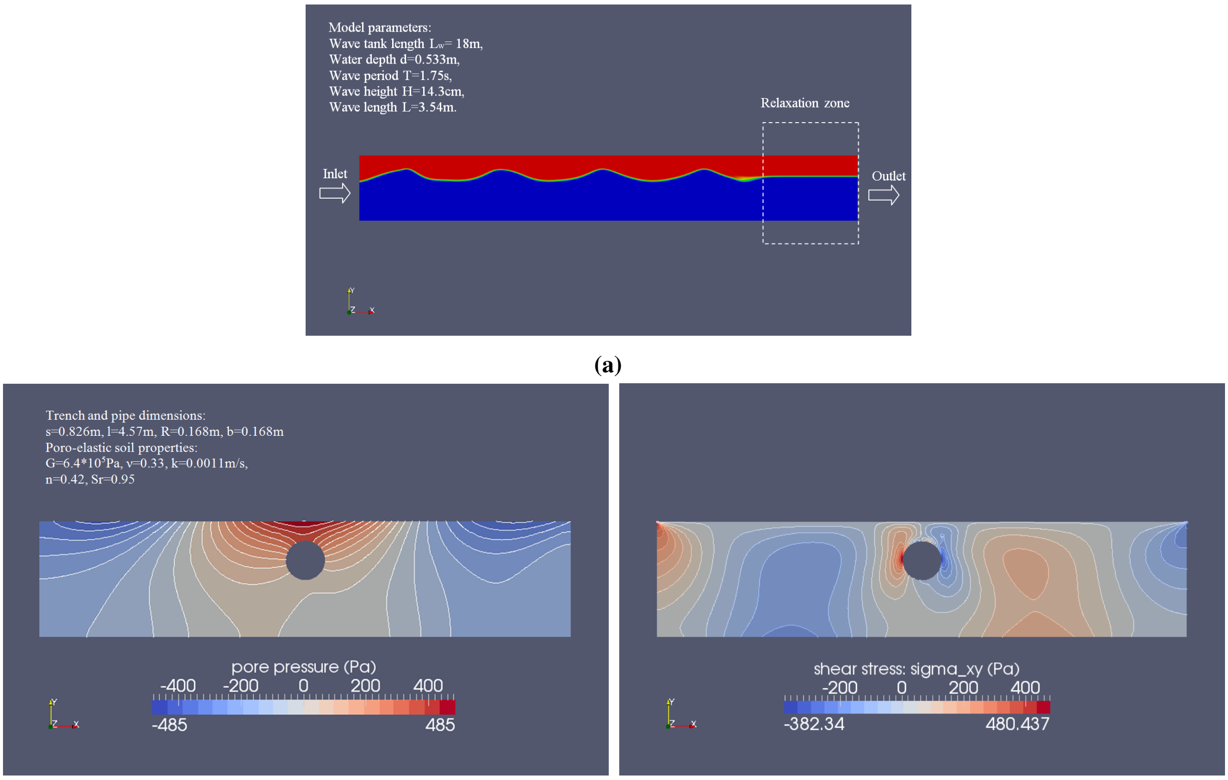
In Fig. 3a, the simulated pore pressure amplitude around the pipeline is plotted against the experimentally measured data [12]. Although the simulation results seem to slightly overshoot the experiment data mainly due to the approximation of the true pore fluid bulk modulus, an overall match can be seen between the two. In Fig. 3b, the calculated seepage forces acting on the pipeline (obtained by integrating pore pressure along the pipeline surface) are compared with available solutions from Cheng and Liu [12], where the influence of varying soil permeability has been considered. We observe a good agreement between our simulation results and their solutions.

Besides from the above transient response (e.g. amplitude damping and phase lag of the pore pressure), another quantity of interest to the pipeline design is the residual (built-up) nature of pore pressure under cyclic wave shearing. As there was no information revealed on the latter feature in the original Cornell's experiments, we now compare our results to another similar experiment conducted later by Teh et al [13], where the gradual built-up of the seabed pore pressure is investigated as well. In their experiment, very fine silt was used as the seabed material so that it could restrain the dissipation of pore pressure and facilitate the built-up process. Hence, a corresponding small soil permeability, $k = 2 \times 10^7$m/s, is used in our simulation. Typical values of loose silt were adopted and the plasticity model constants for the constitutive model [7] were also carefully calibrated to obtain a reasonable fit to the measurements. A summary of the used plasticity model parameters is given in Table 2.

Fig. 4 presents the gradual increase of pore pressure in the seabed at different locations due to cyclic wave loading. In Fig. 4a, the simulated results are first compared with experimental data at two

Table 2: Calibrated plasticity model constants used in the residual response simulation.

Elastic		Critical state		State parameter		Hardening		Dilatancy	
K_0, G_0	2, 0.866MPa	$M_{c(e)}$	1.15(1.035)	$k^b_{c(e)}$	3.975(2.0)	h_0	1200	A_0	6
a	0	λ	0.025	$k^d_{c(e)}$	4.2(0.07)	m, c_m	0.05, 0	F_{max}, C_f	500, 600

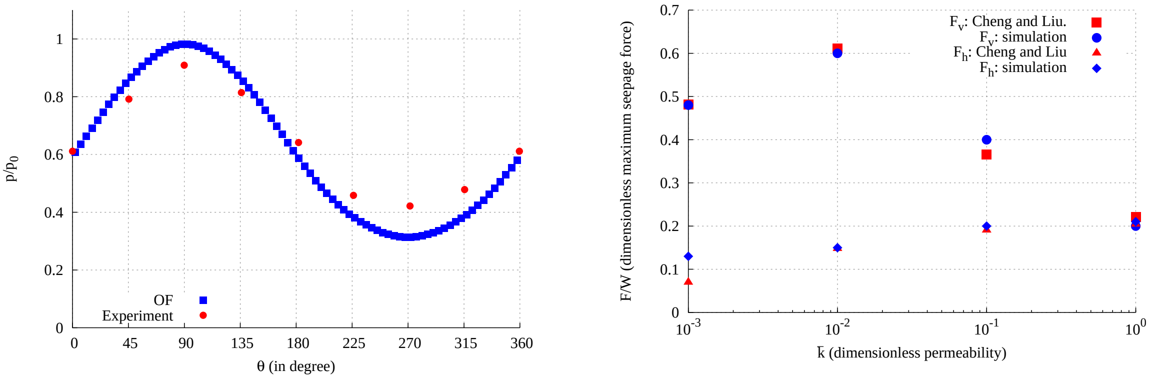


(a)

(b)

(c)

Figure 2: Numerical results of the wave-seabed-pipeline interaction test case. Upper panel: view of the free surface wave profile. Lower panels: contours of the wave-induced pore pressure (left) and shear stress (right) inside the seabed domain.



(a) **(b)**

Figure 3: Validation of transient simulation results. Left panel: comparison of pore pressure distribution along the pipeline between the experiment measurements [12] and the OpenFOAM®simulation. Right panel: comparison of the simulated seepage forces acting on the pipeline with solutions from Cheng & Liu [12], where $F_{v(h)}$=Vertical(horizontal) seepage force, $W = \gamma \pi R^2$ the displaced water weight.

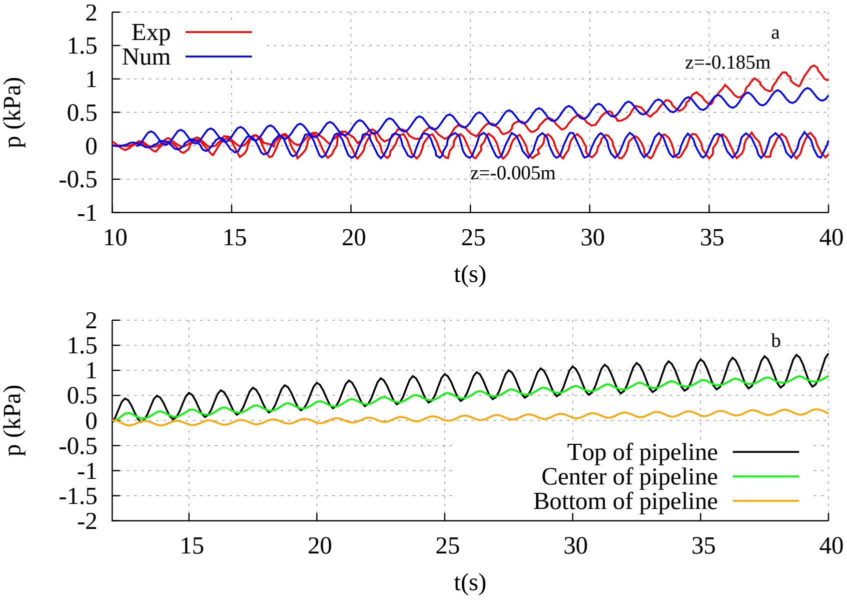


Figure 4: Validation of the residual seabed response. Upper panel: comparison of simulated pore pressure variation at two different soil depths with experiment data [13]. Lower panel: simulated pore pressure buildup around the pipeline.

different soil depths without the presence of the pipeline: one at $z = -0.005$m near to the seabed surface, where pore pressure change is strongly influenced by the periodic wave pressure fluctuation; another at $z = -0.185$m near to the seabed bottom, where pore pressure build-up is more significant due to the long drainage distance plus the very low soil permeability. Although the match with experiments is not perfect, the general trend of pore pressure accumulation is well captured in the simulation. With the same soil data set, Fig. 4b plots the pore pressure build-up in the presence of an impermeable pipeline, which locally creates a poor drainage condition. It can be seen that the pore pressure build up rate varies along the pipeline: fastest at the top of the pipe while lowest at the bottom. This agrees with the observation by Dunn et al. [14] that the liquefaction process propagates from the the top and downwards.

4.2 Modeling of wave-seabed-gravity structure interaction

To further demonstrate the applicability of the new solver in the context of practical offshore engineering, a test case of 3-D seabed response in a numerical wave tank with a submerged gravity based structure is carried out. The wave tank is a square of side length 150m and the water depth is 5m. The gravity structure (represented by a rigid box of dimensions 20m×20m×5m) is placed in the middle of the wave tank and partially buried in the seabed (buried depth = 1m). The porous seabed domain is 60m in width and length, and 10m deep. Realistic ocean wave conditions are simulated by imposing waves from three different directions. The wave characteristics are: wave hight $H_0 = 0.5$m, wave period $T = 7$s, and wave length $L = 45.66$m. The elastic soil parameters used for design of gravity platforms in the North Sea [15] are adopted here for the seabed domain: e.g. elastic shear modulus $G = 1.2 \times 10^7$kPa, Poisson's ratio $\nu = 0.2$. The soil permeability is assumed to be $k = 0.001$m/s and the saturation factor is set to $Sr = 0.98$. The other plasticity model constants are kept the same as the pipeline case.

The overall simulation process is divided into two steps: First an initial consolidation stage after the installation of gravity structure on the seabed (static wave assumption). Then follows a cyclic wave loading stage where wave-induced seabed response around and beneath the structure are of the main interests.

4.2.1 Consolidation stage

The structure is assumed to consist of concrete with density $\rho = 2.4$kg/m^3. This corresponds to a vertical load $t_v = 1.2 \times 10^5$Pa acting upon the seabed-structure interface. The seabed domain is assumed initially in the geostatic stress state with the lateral earth pressure coefficient $K_0 = 0.5$. Once the structure is in place the seabed consolidates under its load until all excess pore pressure has been dissipated. It is assumed that there are no waves present during the consolidation process.

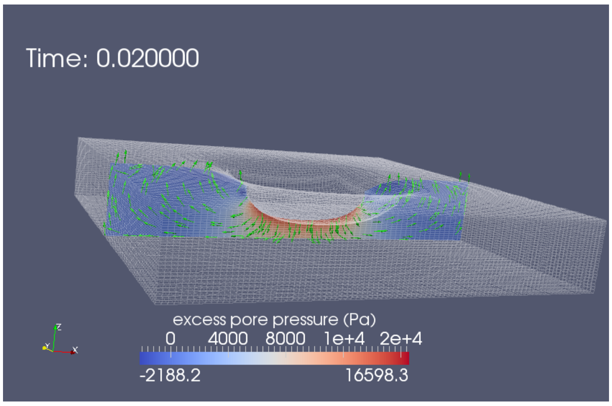
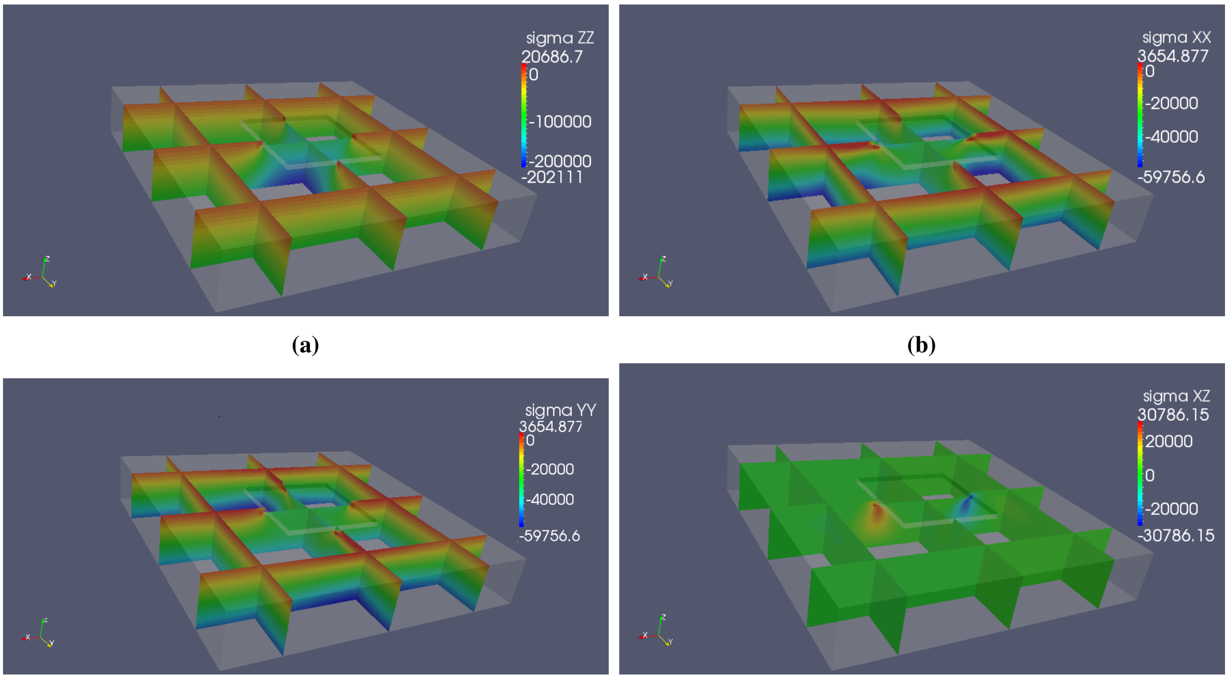


Figure 5: A slice view of the excess pore pressure distribution within the deformed seabed (exaggerated a factor 100). The green arrows represent the pore flow direction.

Fig. 5 shows a snapshot of the pore fluid flow profile in the seabed right after the installation of the structure. It can be seen that large amount of excess pore pressure is generated beneath the structure, and



(a)

(b)

(c)

(d)

Figure 6: Distribution of the effective stresses in seabed at the final stage of consolidation process

the pore fluid flows from the high pressure zone (underneath the structure) to the lower pressure zones at the seabed surface where there is free drainage. Moreover, the structure itself settles downwards during the consolidation process and the maximum settlement occurs at the middle part of the structure.

Fig. 6 illustrates the distribution of the different components of effective stresses in the seabed when the consolidation is finished. In Fig. 11a, the vertical effective stress (sigmaZZ) is increased significantly under the structure as expected. In Fig. 11b and 11c the two horizontal effective stress components (sigmaXX and sigmaYY) are shown to be reduced beneath the structure. This is because the soil has been squeezed toward the two lateral sides due to overburden compression. Fig. 6d shows the shear stress concentration zones underneath the edges of the structure.

The above final consolidated stress state in the seabed is used as the initial condition for the wave loading stage, while the displacement field is simply reset to be zero.

4.2.2 Cyclic wave loading stage

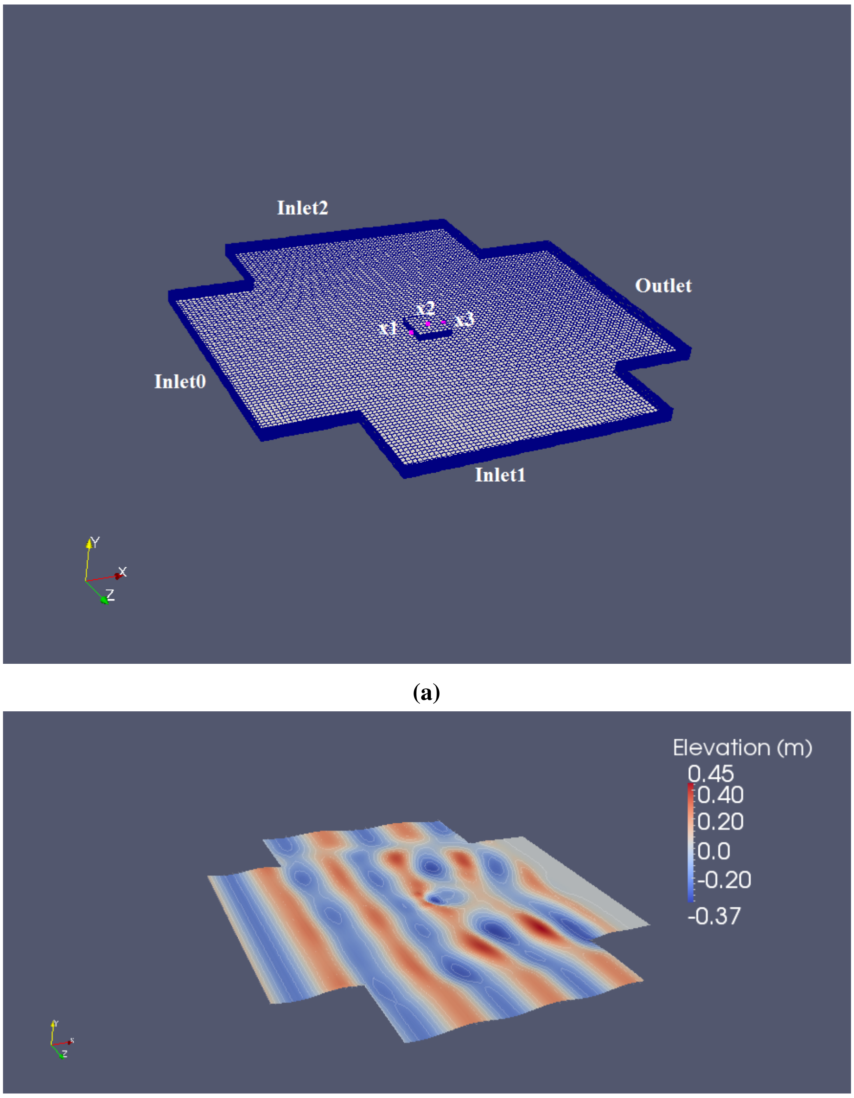
After completion of the consolidation stage, the waves are turned on, and the wave-induced seabed response is studied. Three wave generation inlet zones are placed at three sides to generate the desired wave motion inside the wave tank. On the remaining side a wave absorption outlet zone is set to absorb the wave energy. The parameters for generating the multi-directional waves are listed in Table 3 below.

Fig. 7a shows the wave tank geometry with the inlet and outlet placements. Fig. 7b plots the generated surface elevation of waves at time $t = 47$s when a wave crest passes on top of the structure. The modification of the structure on the wave field around it is clearly seen. Another feature visible in Fig. 7b is that the generated wave crest (0.45m) is further from the mean sea level than the wave trough (-0.37m), which represents a general feature of the nonlinear nature of waves.

Fig. 8 plots the generated periodic wave dynamic pressure acting on the sea bottom and the structure. Three representative positions from Fig. 7a are probed: a point on the sea bottom in front of the structure $\mathbf{x}_1 = (65, -5, 75)$, a point on the structure $\mathbf{x}_2 = (75, -1, 75)$, and a point on the sea bottom behind the structure $\mathbf{x}_3 = (85, -5, 75)$. Fig. 9 illustrates the periodic wave force components and magnitude acting

Table 3: Multi-directional wave parameters

	direction	wave number	frequency	wave height(m)
inlet0	$(\ 1 \quad 0 \quad 0\ )$	$(\ 0.137622 \quad 0 \quad 0\ )$	0.897598	0.5
inlet1	$(\ 1 \quad 1 \quad 0\ )$	$(\ 0.0973138 \quad 0.0973138 \quad 0\ )$		
inlet2	$(\ 2 \quad -1 \quad 0\ )$	$(\ 0.123093 \quad -0.0615466 \quad 0\ )$		



(a)

(b)

Figure 7: Snapshot of the 3-D wave profile. Upper panel: view of the wave tank mesh geometry (structure presenting in the middle of the tank). Lower panel: contours of the surface elevation at time $t = 47$s.

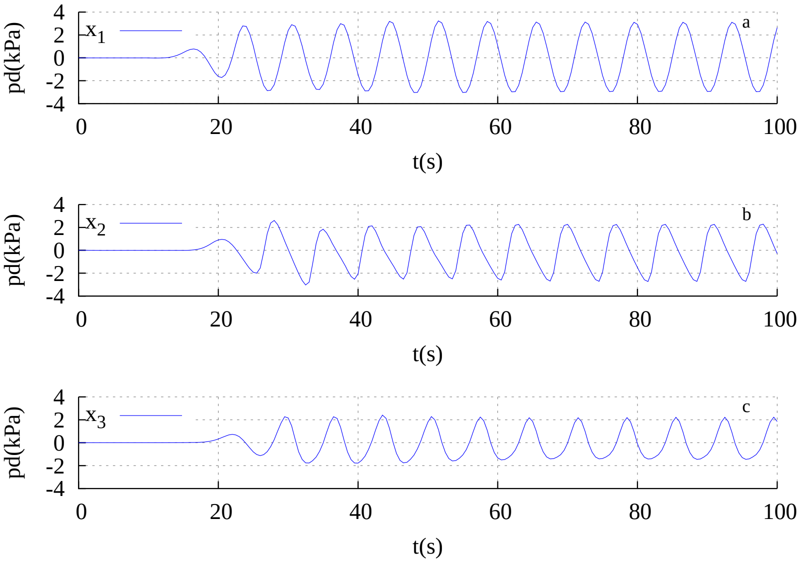


Figure 8: History of wave dynamic pressure acting on sea bottom and structure. Upper panel: x_1. Middle panel: x_2. Lower Panel: x_3.

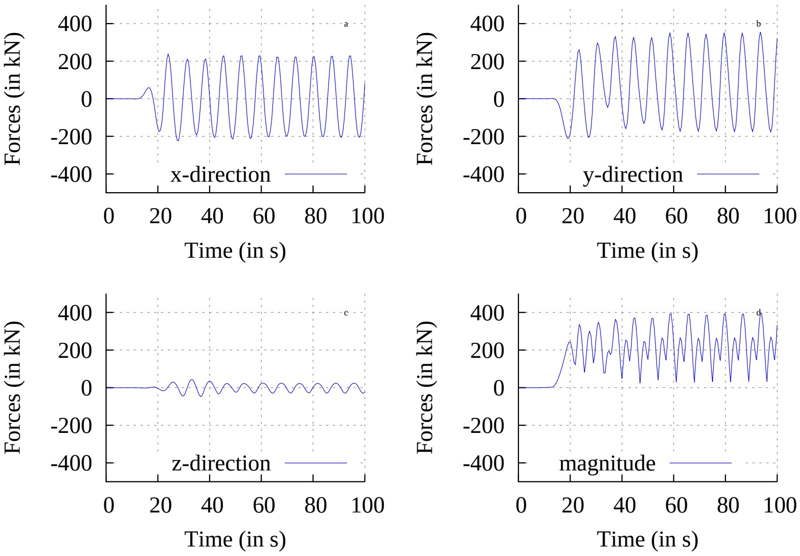
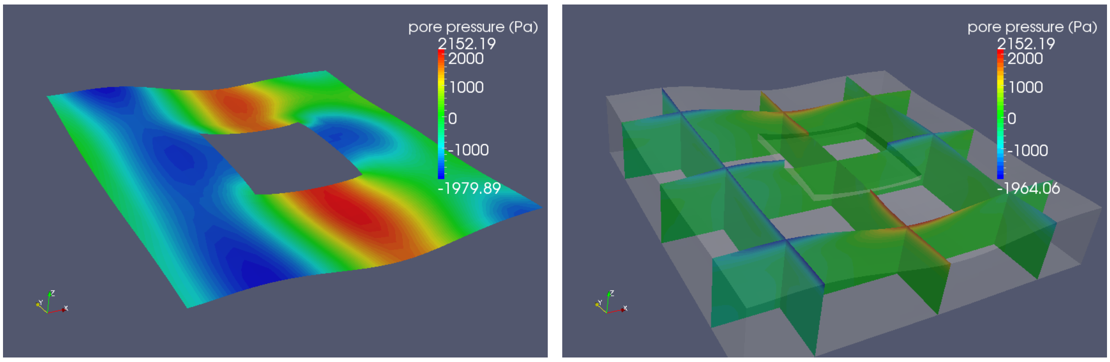


Figure 9: History of wave force acting on the gravity structure

on the gravity structure. It is seen from the upper panel of Fig. 9 that the pull upwards force ($\sim$ 300kN) is larger than its counterpart push downwards force (~ -200kN). This wave pressure force variation complies with the previously observed fact that the generated wave crests are further from the mean sea level than the troughs.



(a)

(b)

Figure 10: Wave-induced pore pressure distribution in seabed at time point $t = 47$s. Left panel: top surface view. Right panel: slices view. The deformed configuration is exaggerated a factor of 1000.

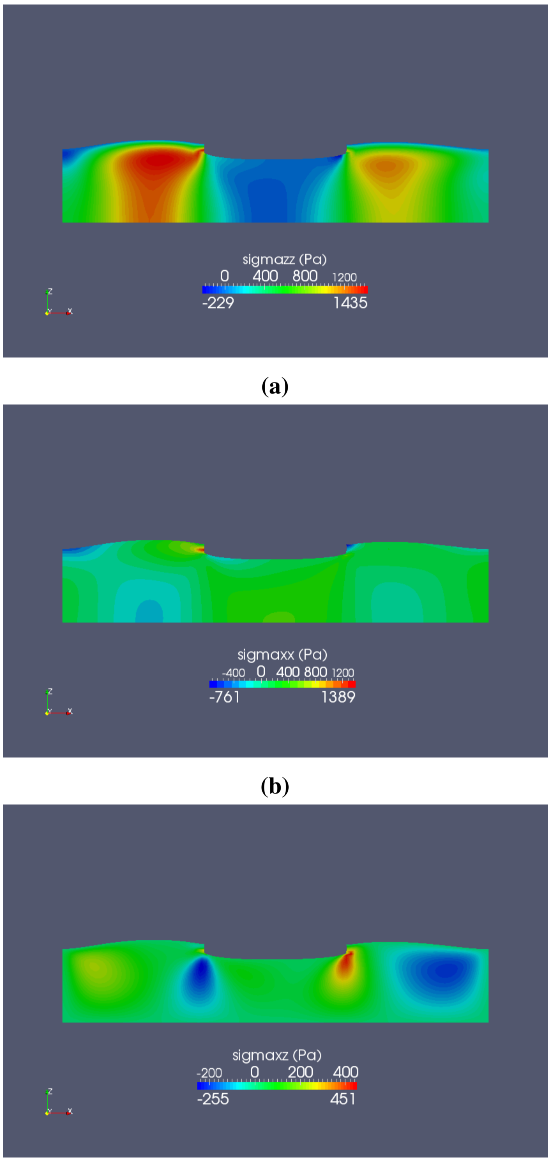
In Fig. 10, the wave-induced pore pressure distribution in the seabed at time $t = 47$s is illustrated. At this moment a wave crest is passing above the structure and consequently the parts of soil facing the wave crest are compressed. It is also observed that the soil beneath the structure only undergoes minor deformations compared to the soil just next to the structure. Fig. 10b also demonstrates the vertical penetration of the wave-induced pore pressure down through the seabed. Only a shallow layer of soil is found to be affected, while the deeper soil layers experience no excess pore pressure. This layer thickness is partially determined by the saturation degree, because the air content in the soil will make the pressure pressure dissipate at a very fast rate.

Fig. 11 shows the distribution of the wave-induced effective stress components in the seabed at time $t = 47s$. It is observed from Fig. 11a that extension stress is generated in the seabed at the two sides of the structure because a wave trough is passing by, while compression stress is generated beneath the structure because of the passage of a wave crest. Moreover, a significant amount of shear stress is observed under the edges of the structure as shown in Fig. 11c. Note that the effective stresses in response to the gravity structure weight have been subtracted from the results to demonstrate the pure effect of waves.

Fig. 12 shows the evolution of the generated pore pressure in seabed under wave loading within several wave cycles. Two positions in the seabed are selected - one below the center of structure and the other 3m underneath its center. Partial drainage are created in those places due to the impermeability of the structure. It is shown that the pore pressure is accumulated in the initial cycles after which dissipation takes over. We consider this prediction quite reasonable, since what occurs to the soil underneath the structure is that: on one hand the cyclic shearing compresses the soil skeleton and thus generates positive pore pressure, while on the other hand the structural force pulls the soil upwards (see Fig. 9) and so creates negative pore pressure. The two processes are competing against each other and in the end the pull upwards becomes dominant in this case.

5. CONCLUSION AND DISCUSSION

The present work has demonstrated in general that it is feasible to use OpenFOAM®as a platform for multi-physics modeling of the interactions between ocean waves, porous seabed and an offshore structure. For the seabed domain in particular, a new nonlinear cyclic porous soil solver is applied, which allows investigation of the different mechanism involved in seabed response to transient and cyclic wave loading condition. The object-oriented nature of the OpenFOAM®code library also allows the implementation



(a)

(b)

(c)

Figure 11: Wave-induced effective normal and shear stresses distributions in a cutting plane view (x-direction) of the seabed at time point $t = 47s$.

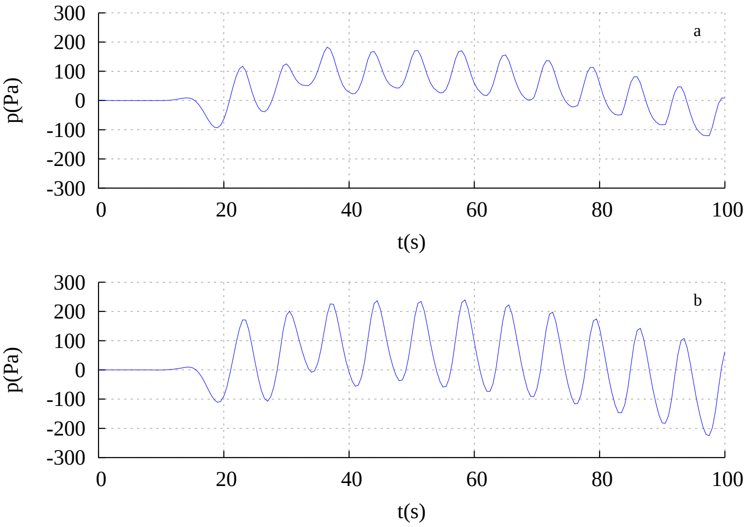


Figure 12: Wave-induced residual response of the seabed in terms of pore pressure buildup. Upper panel: 0m beneath the center of the structure. Lower panel: 3m under the center of the structure.

of other user-desired soil solvers. It is the authors' view that OpenFOAM®opens a way for offshore geotechnical engineers to incorporate more realistic wave loading conditions (retrieved from CFD models), and meanwhile to use FVM soil solvers effectively dealing with the strong nonlinearity and pressure coupling in the equation system by using the 'segregated' solution strategy.

There are also some extensions which can widen the applicability of the presented simulation method. For instance, in reality the offshore structures may not be rigid and stay in the place, instead they undergo deformation and/or rotation due to the wave loads. Therefore realistic structural response could be modeled e.g. by employing a solid mechanics solver. In this way, the soil-structure interaction can be included by imposing continuity of the displacement on the interface between the soil and structure.

REFERENCES

[1] M. Luan, P. Qu, D. S. Jeng, Y. Guo, and Q. Yang. Dynamic response of a porous seabed-pipeline interaction under wave loading: Soil-pipeline contact effects and inertial effects. *Computers and Geotechnics*, 35(2):173–186, 2008.

[2] J. H. Ou and A. H. C. Chan. 3d numerical modelling of the wave-induced response around the circular caisson founded on the seabed. In *Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE*, volume 4, pages 697–704, 2008.

[3] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, 12(6):620–631, 1998.

[4] X. F. Liu and M. H. Garcia. Numerical simulation of sea bed response under waves with coupled solver of biot consolidation equations and free surface water flow. In *Proceedings of the Seventh (2006) ISOPE Pacific/Asia Offshore Mechanics Symposium (ISOPE PACOMES-2006)*, pages 127–134, 2006.

[5] S. S. Deshpande, L. Anumolu, and M. F. Trujillo. Evaluating the performance of the two-phase flow solver interfoam. *Comput. Sci. Disc.*, 5:014016, 2012.

[6] N. G. Jacobsen, D. R. Fuhrman, and J. Fredsøe. A wave generation toolbox for the open-source cfd library: Openfoam. *International Journal for Numerical Methods in Fluids*, 70(9):1073–1088, 2012.

[7] M. T. Manzari and Y. F. Dafalias. A critical state two-surface plasticity model for sands. *GEOTECHNIQUE*, 47(2):255–272, 1997.

[8] C. L. Bakmar, O. Hededal, and L. B. Ibsen. *A Modified Critical State Two-surface Plasticity Model for Sand*. DCE Technical Memorandum. Department of Civil Engineering, Aalborg University, 2008.

[9] N. G. Jacobsen. The OpenFOAM wikki for waves2foam. `http://openfoamwiki.net/index.php/Contrib/waves2Foam`, 2014. [Online; accessed 19-Feburary-2014].

[10] T. Tang and O. Hededal. Simulation of pore pressure accumulation under cyclic loading using finite volume method. In *Proceedings of 8th European Conference on Numerical Methods in Geotechnical Engineering*, 2014.

[11] T. Tang. The solver repository hosted in Bitbucket website. `https://bitbucket.org/tiantang/twosurfaceplastbiotfoam`, 2014. [Online; accessed 09-May-2014].

[12] A. H. D. Cheng and P. L. F. Liu. Seepage force on a pipeline buried in a poroelastic seabed under wave loadings. *Applied Ocean Research*, 8(1):22–32, 1986.

[13] T. C. Teh, A. C. Palmer, and J. S. Damgaard. Experimental study of marine pipelines on unstable and liquefied seabed. *Coastal Engineering*, 50(1-2):1–17, 2003.

[14] S. L. Dunn, P. L. Vun, A. H. C. Chan, and J. S. Damgaard. Numerical modeling of wave-induced liquefaction around pipelines. *Journal of Waterway, Port, Coastal, and Ocean Engineering-ASCE*, 132(4):276–288, 2006.

[15] O. Kjekstad and T. Lunne. Soil parameters used for design of gravity platform in the north sea. *Publikasjon - Norges Geotekniske Institutt*, (127):1–18, 1979.

Paper V

"An integrated finite volume based approach for computing wave-structure-seabed interaction"

T. Tang, J. Roenby & B. Johannesson

Submitted for publication in: *Computers and Geotechnics, 2014*

An integrated finite volume based approach for computing wave-structure-seabed interaction

Tian Tang[a], Johan Roenby[b], Björn Johannesson[a,*]

[a]*Department of Civil Engineering, Technical University of Denmark, Anker Englelunds Vej 1, 2800 Kgs. Lyngby, DENMARK*
[b]*Ports and Offshore Technology, DHI, Agern Alle 5, 2970 Hoersholm, DENMARK*

Abstract

We present a new method for computing the dynamic seabed soil behaviour beneath and around an offshore structure exposed to ocean waves. Such calculations can help in the offshore foundation design process to assess the ability of the soil to resist and counter balance the dynamic environmental loads. The method proceeds in three steps. First detailed free surface CFD calculations are executed to obtain a realistic wave field around a structure standing on the seabed. Then the dynamic structural response, due to the motions in the surrounding water, are calculated using a linear elasticity solver. Finally, the direct wave loads on the seabed and the indirect wave loads on the seabed-structure interface through the structure are provided as input for a dynamic soil response simulation. The latter simulation is performed with a newly developed code solving the Biot consolidation equations extended to account for soil anisotropy. All three solvers are based on the finite volume method and implemented in the open source CFD code environment OpenFOAM. A number of calculations are presented to verify the validity of the newly developed anisotropic soil solver and to demonstrate the accuracy and applicability of the integrated simulation approach.

Keywords: Finite volume method, Wave-structure-seabed interaction, Multi-physics, Soil anisotropy, OpenFOAM

[*]Principal corresponding author

Email addresses: tiat@byg.dtu.dk (Tian Tang), jro@dhigroup.com (Johan Roenby), bjo@byg.dtu.dk (Björn Johannesson)

1. Introduction

Offshore geotechnical engineers are putting increasing efforts into the study of dynamic interaction processes between the three components in marine construction: water, soil, and structure. When ocean waves pass by an offshore structure, the presence of the structure alters the incoming wave field, which in turn exerts forces on the structure. These external wave forces should at any time be counter balanced by the stresses at the soil-structure interface to keep the structure in place. The underlying soil responds to the pressure and shear variations on the seabed with small displacements and changes in pore pressure as a result. Improved knowledge of this type of interaction process is essential for engineers when it comes to prevent future offshore structure failures of geotechnical nature.

Traditionally, a large effort has been put into modeling one of the three components accurately with somewhat oversimplified models for the other two subdomains. For instance, advanced CFD models have been applied to calculate the wave loads on structures for complex ocean wave conditions in coastal engineering, but using simple assumptions such as rigid and impermeable seabed and structure conditions Chen et al. (2014). Others employ soil models to estimate wave-induced seabed responses in geotechnical engineering, Sassa and Sekiguchi (2001) and Geremew (2013), but with simplified wave loading conditions derived from linear wave theories. Only a few attempts have been made to integrate numerical models of all three components Jeng et al. (2013) and Ye et al. (2013). These studies follow the tradition of using the finite volume method (FVM) to model the waves and a separate software based on the finite element method (FEM) for the solid domains i.e. soil and structure.

In the present study, we develop an approach for modeling the soil-water-structure interaction, in which we incorporate all subdomains in the same FVM based coding framework. Adopting this approach, the usage of additional software for data transfer between different numerical methods (and/or software) can be avoided. Benefits of using FVM based numerics for soil modeling has recently been discussed in Tang et al. (2014) and Tang and Hededal (2014). The method has proven well-suited for handling the strong coupling effect between the soil skeleton and the pore fluid, as well as the soil material nonlinearity. Integrating the wave, soil and structure solvers in the same FVM based code framework opens up for an efficient way for handling the interaction modelings. Our work may be regarded as an extension of Liu and Garcia (2007) who combined a CFD solver with an FVM isotropic poroelastic soil solver to study wave-induced seabed response in the presence of a rigid offshore structure. The new features in the present work are 1) the use of more realistic directionally spread wave field imposed on the structure, 2) the inclusion of displacements and stresses in the structure, and finally 3) the extension of the soil model to account for anisotropy. The latter extension is important since most nature soils display some degree of anisotropy, having different elastic and hydraulic properties in different directions, due to the deposition mode, particle shape, and loading history, Hsu and Jeng (1994).

2. Mathematical models of the wave-structure-seabed system

Three mathematical models governing the three different components in the interacting system are briefly presented below. We start with the description of the hydrodynamic wave calculation which provides the input pressure for the remaining sub-domains. Then we move on to the structural solver, and finally we present the soil solver in some detail.

2.1. Free surface Navier-Stokes wave model

The water waves and the air above them are modeled using the incompressible flow type of Navier-Stokes (NS) equations - including one momentum equation and one continuity equation:

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}^{\mathrm{T}}) = -\nabla p + \rho \mathbf{g} + \nabla \cdot (\mu \nabla \mathbf{v}) \tag{1}$$

$$\nabla \cdot \mathbf{v} = \mathbf{0} \tag{2}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure, and $\mathbf{g}$ is the gravity vector. The density ρ and dynamic viscosity μ of a fluid particle are assumed to be constant in time with the values $\rho_w \approx 1000\text{kg/m}^3$ and $\mu_w \approx 1 \cdot 10^{-3}\text{Pa s}$ in the water phase and the values $\rho_a \approx 1.2\text{kg/m}^3$ and $\mu_a \approx 18 \cdot 10^{-6}\text{Pa s}$ in the air phase. The passive advection equation governing the evolution of the density field reads

$$\frac{\partial \rho}{\partial t} + \mathbf{v} \cdot \nabla \rho = 0 \tag{3}$$

We note that this equation is trivially solved everywhere except at the air-water interface where it is numerically challenging due to the sharp jump in density causing $\nabla \rho$ to be a Dirac delta function. A similar equation could be written for the viscosity, but this clearly contains no new information, due to the assumption that both ρ and μ are constant everywhere except for the jump at the air-water interface.

In the FVM method the above governing equations are volume integrated over each computational cell and all terms are approximated via the values of dependent variables, $\mathbf{v}$, p and ρ, in the cell and its neighbour cells. With 5 dependent variables and N cells this gives $5N$ coupled equations that must be solved to propagate the flow in time. The details of the solution algorithm for the CFD is beyond the scope of the present paper. We only mention that the pressure-velocity coupling is solved with a PISO based algorithm and the air-water interface capturing is based on the VOF method with numerical interface compression to keep the interface sharp. The reader is referred to Anumolu et al. (2012) for further details.

2.2. Linear elastic structure model

The structure response is governed by one momentum balance equation supplemented with constitutive equation in terms of an isotropic linear elastic stress-strain relation:

$$\begin{aligned}\nabla \cdot \boldsymbol{\sigma} =& \nabla \cdot [2\mu\boldsymbol{\varepsilon} + \lambda \mathrm{tr}(\boldsymbol{\varepsilon})\mathbf{I}] \\ =& \nabla \cdot [\mu\nabla\mathbf{u} + \mu(\nabla\mathbf{u})^{\mathrm{T}} + \lambda\mathrm{tr}(\nabla\mathbf{u})\mathbf{I}] = 0 \end{aligned} \tag{4}$$

where $\boldsymbol{\sigma}$ stands for the stress tensor and $\boldsymbol{\varepsilon}$ is the small strain tensor which can be defined by $\boldsymbol{\varepsilon} = 1/2(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}})$ with $\mathbf{u}$ being the structure displacement vector. The elastic material properties μ and λ are Lamé's coefficients. The two parameters are related to the more widely used Young's modulus E and Poisson's ratio ν. The quasi-static assumption used here is valid for stiff offshore structures, such as gravity-based structure (GBS), which tend to be less sensitive to acceleration effects.

Even with the simplest isotropic linear elastic stress-strain relation, the three displacement components are still coupled with each other through the terms - $\mu(\nabla\mathbf{u})^{\mathrm{T}} + \lambda\mathrm{tr}(\nabla\mathbf{u})\mathbf{I}$ - in Eq. (4). In FVM structure analysis, such coupling is handled using the *segregated* strategy, in which one separates the inter-coupling terms into 'explicit' discretisation. Eq. (4) is split into the following format:

$$\underbrace{\nabla \cdot [(2\mu + \lambda)\nabla\mathbf{u}]}_{\text{implicit}} = \underbrace{-\nabla \cdot [\mu(\nabla\mathbf{u})^{\mathrm{T}} + \lambda\mathbf{I}\mathrm{tr}(\nabla\mathbf{u}) - (\mu+\lambda)\nabla\mathbf{u}]}_{\text{explicit}} \tag{5}$$

The above segregated momentum equation will be solved iteratively until the explicit terms essentially become implicit based on a fixed-point iteration algorithm, Jasak and Weller (2000).

According to the derivation of Jasak and Weller (2000), the over-relaxed implicit part on the left hand side of Eq. (5) is the maximum consistent implicit contribution to the component-wise contribution, whereas the explicit part on the right represents a pure rotation. For problems with little or no rotation, such as the cases studied here, the splitting described in Eq. (5) results in smooth and rapid convergence as the implicit part carry more information than its explicit counterpart.[1]

2.3. Anisotropic Biot's porous soil model

The behavior of seabed soil is largely governed by the interaction of the solid skeleton with the fluid(s) present in the pore structures. We adopt the classical Biot's consolidation equations, Biot (1941), to describe the coupled soil behavior. The first governing equation is one quasi-static momentum balance equation for the soil mixture, in which the constitutive behavior of the solid phase (soil skeleton) applied is anisotropic elasticity:

$$\nabla \cdot \left[\mathbf{C} : \frac{1}{2} \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}} \right) \right] - \nabla p = 0 \tag{6}$$

where $\mathbf{u}$ is the soil (skeleton) displacement, p is the pore fluid pressure and $\mathbf{C}$ is the forth-order elastic stiffness tensor. In particular, the orthotropic anisotropy will be considered here, as it effectively captures the degrees of anisotropy that soil materials display in nature. The orthotropic elastic stress strain relation may be expressed in the 6×6 matrix notation:

$$\boldsymbol{\sigma}' = \begin{pmatrix} \sigma'_{xx} \\ \sigma'_{yy} \\ \sigma'_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{31} & 0 & 0 & 0 \\ A_{12} & A_{22} & A_{23} & 0 & 0 & 0 \\ A_{31} & A_{23} & A_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{66} \end{pmatrix} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{pmatrix} = \mathbf{C} : \boldsymbol{\varepsilon} \tag{7}$$

[1] For structural problems where rotation is dominant, strong under-relaxation of the explicit part are required for convergence consideration, or even full implicit block matrix solution procedure is necessary when explicit relaxation fails to work.

where according to Demirdžić et al. (2000), the 9 independent coefficients A_{ij} can be given in terms of Young's moduli E_i, Poisson's ratio ν_{ij}, and shear modulus G_{ij} as follows:

$$\begin{aligned}
&A_{11} = \frac{1-\nu_{yz}\nu_{zy}}{JE_yE_z}, \quad A_{22} = \frac{1-\nu_{xz}\nu_{zx}}{JE_xE_z}, \quad A_{33} = \frac{1-\nu_{yx}\nu_{xy}}{JE_yE_x}, \\
&A_{12} = \frac{\nu_{xy}+\nu_{zy}\nu_{xz}}{JE_xE_z}, \quad A_{23} = \frac{\nu_{yz}+\nu_{yx}\nu_{xz}}{JE_xE_y}, \quad A_{31} = \frac{\nu_{zx}+\nu_{yx}\nu_{zy}}{JE_yE_z}, \\
&A_{44} = 2G_{xy}, \quad A_{55} = 2G_{yz}, \quad A_{66} = 2G_{zx}
\end{aligned} \tag{8}$$

with

$$J = \frac{1-\nu_{xy}\nu_{yx}-\nu_{yz}\nu_{zy}-\nu_{xz}\nu_{zx}-2\nu_{yx}\nu_{zy}\nu_{xz}}{E_xE_yE_z} \tag{9}$$

The second soil governing equation is one mass balance equation of the pore fluid based on Darcy's law condition:

$$\frac{n}{K'}\frac{\partial p}{\partial t} - \frac{1}{\gamma_w}\nabla\cdot(\mathbf{k}\cdot\nabla p) + \frac{\partial}{\partial t}(\nabla\cdot\mathbf{u}) = 0 \tag{10}$$

where n is the soil porosity, γ_w is the specific weight of water, and $\mathbf{k}$ is the diagonal permeability tensor with values k_x, k_y and k_z in the diagonal. The bulk modulus of the compressible pore flow K' can be approximated by the formulation $\frac{1}{K'} = \frac{1}{K_w} + \frac{1-S_r}{p_a}$ Verruijt (1969), where S_r is the degree of saturation, K_w the bulk modulus of pure water ($\approx$ 2GPa), and p_a is the absolute pore water pressure.

The momentum and mass balance equations are generally strongly coupled due to the the presence of u_y, u_z and p in the equation for u_x and so forth. In the context of FVM, one traditionally solve the equations in a *segregated* manner where in each iteration loop we solve the equations one after another using the u_y, u_z and p values from the previous iteration in the equation for u_x and so forth. In other words, the equations are split into different 'implicit' and 'explicit' discretization parts, where the 'explicit' parts contain all the coupling effect from the other variables and shall be evaluated from the previous iteration or the initial condition. Here we introduce how to rearrange Eqs. (6) and (10) into the FVM implicit-explicit format:

$$\underbrace{\nabla\cdot(\mathbf{K}\nabla\mathbf{u})}_{\text{implicit}} = \underbrace{-\nabla\cdot\left[\mathbf{C}:\frac{1}{2}\left(\nabla\mathbf{u}+\nabla\mathbf{u}^{\mathrm{T}}\right)\right] + \nabla\cdot(\mathbf{K}\nabla\mathbf{u}) + \nabla p}_{\text{explicit}} \tag{11}$$

$$\underbrace{\frac{n}{K'}\frac{\partial p}{\partial t} - \frac{1}{\gamma_w}\nabla \cdot (\mathbf{k} \cdot \nabla p)}_{\text{implicit}} = \underbrace{-\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{\text{explicit}} \tag{12}$$

in the above, the new 'implicit' stiffness $\mathbf{K}$ in the momentum equation is a second-order diagonal tensor defined by its three non-zero diagonals: K_1, K_2, K_3. Their values are according to $\sigma'_{ii} = K_i \varepsilon_{ii}$ (i=1,2,3 and ε = soil strain).

3. Numerical solvers and coupling procedure

Numerical codes used in this work for solving the soil, water and structure models are based on OpenFOAM, which is an open-source C++ library for continuum mechanics, Weller et al. (1998). One of the main advantages of OpenFOAM is that the partial differential equation and tensor operations syntax closely resembles the equations being solved. For FVM, OpenFOAM provides two distinct types of tensor-derivative classes: `finiteVolumeCalculus` or `fvc`, which performs an 'explicit' evaluation from predetermined data; and `finiteVolumeMethod` or `fvm`, which constructs appropriate matrices using the 'implicit' finite-volume discretisation. Taking advantage of these FVM classes in OpenFOAM, different types of custom solvers can easily be implemented.

In the following subsections, we briefly describe the OpenFOAM solvers applied in this work (either developed by ourself or adopted from the released OpenFOAM solver library), the coupling procedure for modeling the interaction in the system, and the corresponding boundary conditions specified.

3.1. OpenFOAM Solvers

`anisoBiotFoam` is a newly developed anisotropic porous soil mechanics solver. It is capable of computing transient pore pressure, elastic soil displacement, strains and stresses accounting for mechanical and hydraulic soil anisotropy. An extract of the code is given in Appendix (A), which demonstrates the straight forwardness of implementing the governing equations, i.e. Eqs. (11) and (12). In the implementation, the fixed-point iteration method is applied to improve the convergence of the *segregated* solution approach, and under-relaxation factors are adopted for stabilization.

`InterFoam` is a standard OpenFOAM solver for 2 incompressible fluids. It solves the NS equations and tracks the interface using the PISO scheme and VOF approach, as described in Section 2.1.

`stressedFoam` is a linear elastic stress analysis solver released in OpenFOAM-1.6-ext. It is capable of predicting elastic stresses and displacements in solid bodies. The two solvers can be downloaded freely at the OpenFOAM Extend Project website.

3.2. Coupling procedure

There are 3 separate mesh domains used in the computation: one for the soil, one for the wave and one for the structure. The coupling between different domains, which indicates the interactions in the multi-physics system, is the information transfer via the common boundaries, i.e. water-soil interface, water-structure interface and soil-structure interface. In general, two kinds of coupling exist: strong coupling with 2-way data exchange, and weak coupling with only 1-way data exchange. In this work, we will consider only the latter type.

The 1-way coupling procedure for the soil-water-structure system, also illustrated in Fig. 1, is explained as follows. The water wave domain is solved first and the pressure on the seabed and on the structure is mapped to the soil domain and the structure domain, respectively. The elastic structure response is then computed based on the wave pressure loading on the water-structure interface and the structural stresses acting on the soil-structure interface are mapped to the soil domain. In the end, the soil dynamics is solved given the wave pressure as well as the stress load from the structure. This 1-way approach can be acceptable since the magnitude of wave-induced deformation of seabed and marine structure displacement is minor compared to the wave length. We note that this is true for time scale of interest here of a few wave periods, whereas on the time scale of many wave periods the seabed shape may gradually deform due to scour. The inclusion of such gradual deformation is an interesting possible extension which is outside of the scope of the current paper.

3.3. Boundary conditions

When solving the integrated mathematical models, appropriate boundary conditions at external boundaries and internal interfaces for the three computational domains are required, see Fig. 1. We note that the interface boundaries specified here are on a basis of the 1-way coupling assumption.

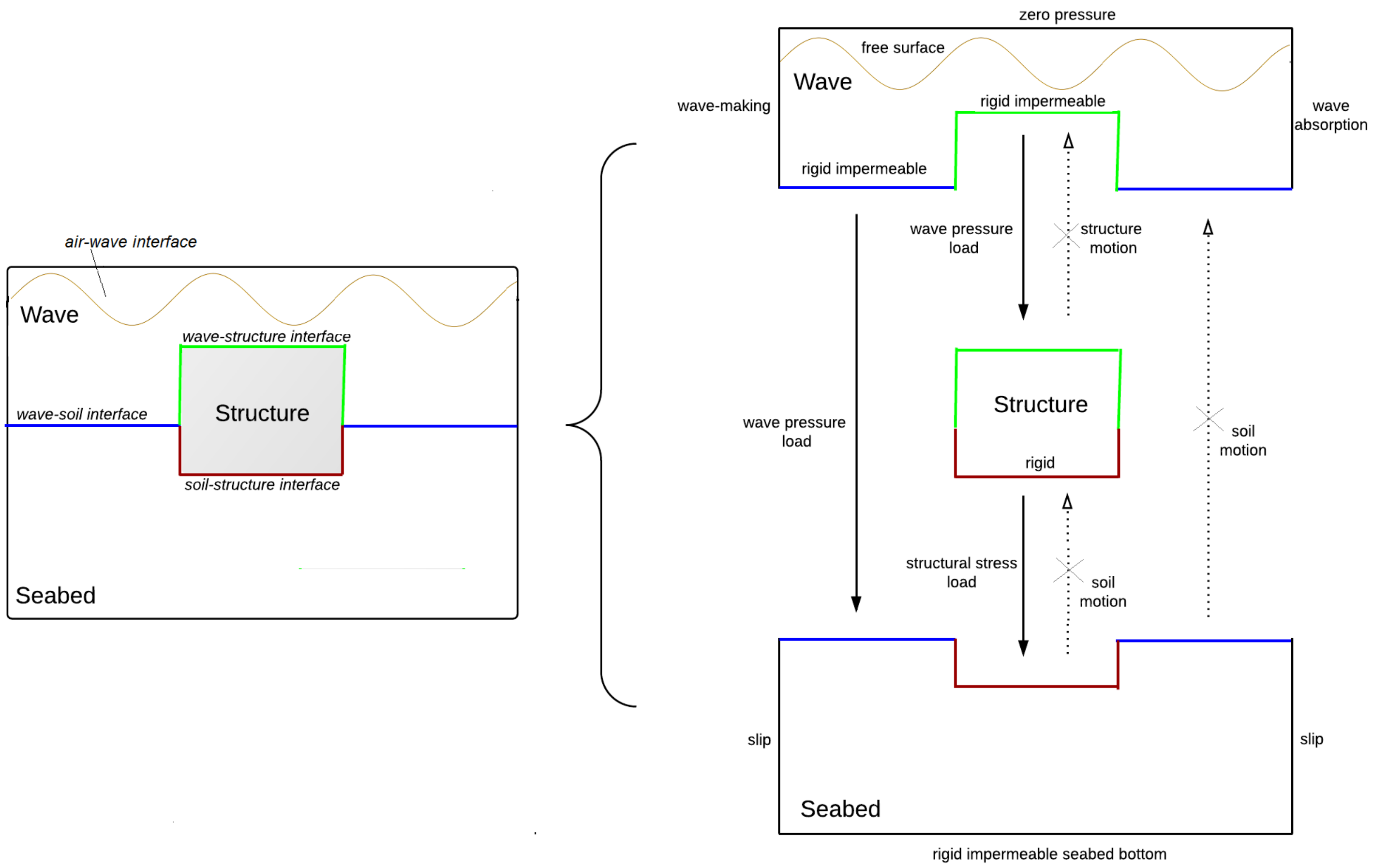


Figure 1: A schematic diagram for the 1-way weak coupling process adopted.

For 2-way coupling, the interface boundary conditions for the wave and structure solver parts respectively, should be modified.

In the wave domain, an implemented stream function wave-making boundary (referring to the theoretical description in Fenton (1999)) is used in the inlet. An active absorption boundary similar to the work of Higuera et al. (2013) is located in the outlet to dissipate the wave energy. The slip condition is adopted at the solid surfaces such as sea bottom, sidewalls and surface of structure. Zero dynamic pressure is set at the top atmosphere boundary and the outlet, and a 'fixedFluxPressure' boundary condition is applied for the remaining boundaries. The fixedFluxPressure boundary sets the pressure gradient to the provided value such that the flux on the boundary is that specified by the velocity boundary condition.

In the structure domain, a time varying nonuniform pressure boundary is specified at the wave-structure interface. This type of boundary condition maps the wave pressure data produced by the CFD calculations on the structure at different times and spatial positions to the structure model with proper interpolations. At the soil-structure interface, a fixed zero displacement boundary is prescribed for the structure, assuming the soil material is stiff to support the structure in place during the simulations. If given a strong 2-way soil-structure interaction assumption, the displacement on this boundary shall no longer be fixed but updated iteratively according to the actual soil displacement upon calculations.

In the seabed domain, it is commonly accepted that effective soil stresses vanish at the wave-soil interface (i.e. zero traction), and that the pore pressure equals the hydrodynamic pressure on the seabed obtained from the wave calculation. At the soil-structure interface it is assumed that the structure is impermeable to the pore water flow. From the Darcy flow equation this implies that the pressure condition should be zero normal gradient for the pore pressure. The displacement on the soil-structure interface is assumed to be dominated by the wave-induced structural forces. Hence a time varying nonuniform traction force boundary is specified, in which the traction forces are derived from the structure computations using patch to patch interpolations. At the bottom and lateral boundaries of the soil domain the soil skeleton is allowed to slip and the normal pore pressure gradient vanishes in accordance with the impermeability condition.

4. Test cases: verification and application

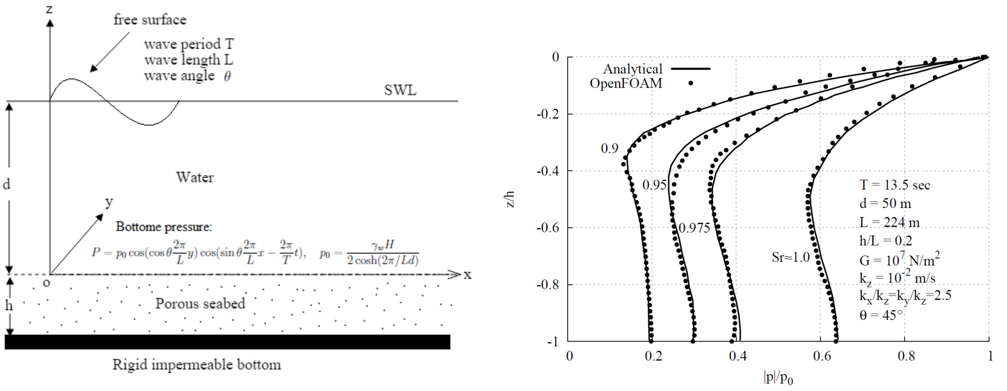
4.1. Verification of FVM anisotropic soil solver

Thorough verifications and validations of the standard OpenFOAM solvers - `interFoam` and `stressedFoam` - have already been well documented in Deshpande et al. (2012) and Jasak and Weller (2000), respectively; thus, the verification work here will be limited to the performance of the newly implemented `anisoBiotFoam` solver.

The verification of the `anisoBiotFoam` is achieved by examining two separate test cases and comparing the numerical predictions to the available analytical solutions. The first test case investigates the wave-induced pore pressures in unsaturated and hydraulically anisotropic seabed, similar to that of Hsu and Jeng (1994) where analytical solutions were derived. We apply the same case geometry and wave and soil conditions as in Hsu and Jeng's paper for direct comparison, see Fig. 2a. The wave loading condition (based on linear wave theory) employed is a 2-D intersecting wave (with wave obliquity $\theta = 45°$) of period 13.5 seconds in a water depth $d = 50$m (or wavelength $L = 224$m). A coarse sand layer with thickness $h = 48$m is assumed as the seabed, with cross-anisotropic permeability $k_z = 10^{-2}$ m/s, $k_x/k_z = k_y/k_z = 2.5$. Some other important parameters for the soil are: Poisson's ratio $\nu = 1/3$, shear modulus $G = 10^7$Pa (here mechanical isotropy is considered), and porosity $n = 0.3$. The condition calculated is for unsaturated soils with the degree of saturation S_r over a range of 0.9 (partially saturated) to approximately 1 (nearly fully saturated).

In Fig. 2b, the numerical results of vertical distribution of pore pressure under different degree of saturation condition in seabed are illustrated. The analytical solutions of Hsu and Jeng (1994) are also plotted. It can be seen that the present numerical solutions are in good agreement with the analytical solutions. The results also show that the degree of saturation has a large impact on the pore pressure development: the partially saturated soils have much lower pore pressure than a nearly fully saturated soil, which has been commonly observed in experiments, Sumer and Fredsøe (2002).

The second test case consists of an orthotropic square plate with a circular hole, as illustrated in Fig. 3a (only a quarter of the physical domain is modeled for reasons of symmetry). This test case examines the implemented mechanical anisotropy feature of `anisoBiotFoam` alone. Plane stress condition is assumed. The plate width to the hole radius ratio is 200:1. The material properties employed are the varied directional Young's modu-

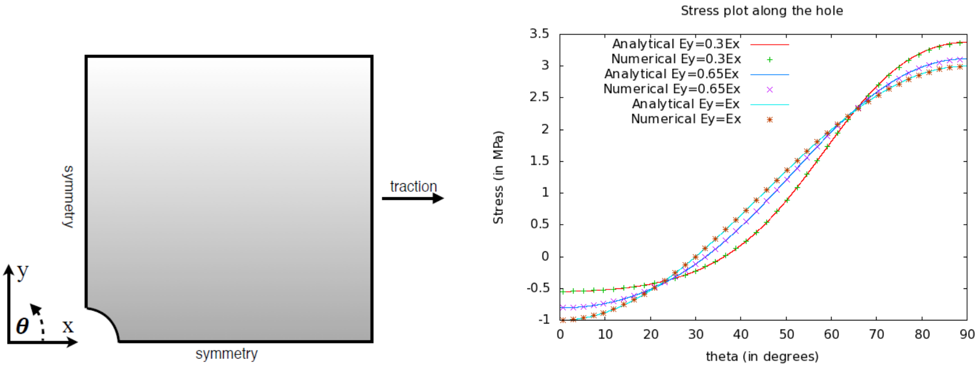


(a) Definition sketch of a short-crested wave pressure loading on a porous seabed with finite depth.

(b) Vertical distribution of the wave-induced pore pressure with varying saturation degree factor Sr.

Figure 2: Verification of the hydraulic anisotropy of `anisoBiotFoam` with the analytical solutions derived by Hsu and Jeng (1994).

lus $E_x = 200GPa$, $E_y = 60GPa, 130GPa, 200GPa$, Poisson's ratio $\nu_{xy} = 0.3$, and shear modulus $G_{xy} = 76.9GPa$. A uniform tension of 1MPa is applied on the right boundary of the plate.



(a) Orthotropic plateHole test case

(b) Hoop stress $\sigma_{\theta\theta}$ around the hole

Figure 3: Verification of the mechanical anisotropy of `anisoBiotFoam` with the analytical solutions presented by Lekhnitskii (1981).

Fig. 3 plots the numerically predicted hook stress $\sigma_{\theta\theta}$ around the circumference of the hole, compared to the analytical solutions by Lekhnitskii (1981). A close agreement can be seen between the numerical predictions and the analytical solutions for all the orthotropic property variations.

4.2. Application on wave-structure-seabed interaction

The application test case demonstrates the applicability of the integrated approach to a realistic offshore structure. This test case, as illustrated in Fig. 4, consists of three computational domains: a numerical wave tank 86m long and 42.5m wide, a grounded gravity base structure (with geometries listed in Table 1), and an underlying porous seabed 10m thick.

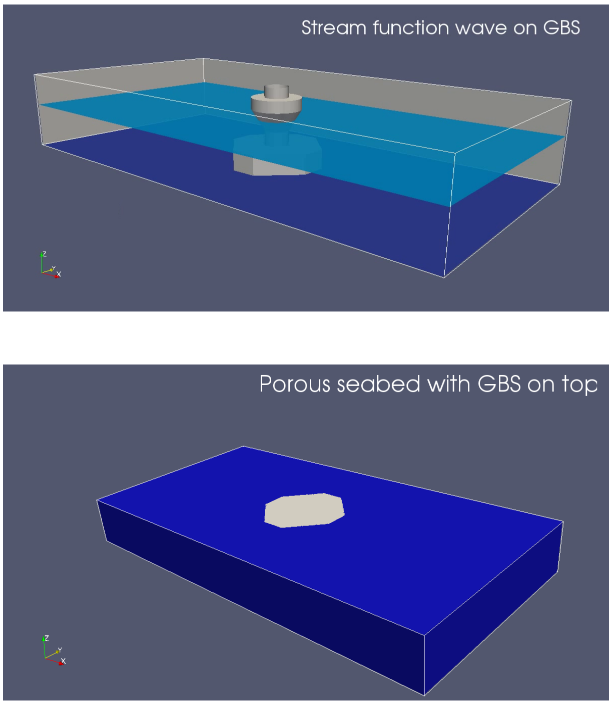


Figure 4: Computational domains of wave, gravity structure and seabed.

The inputs for the wave generation are wave height 2.6m, water depth 8.08261m, wave length 53.3697m and wave period 6.7s. The wave is generated in the direction (1 0.3 0), in which z is the vertical axis. Full reflection on

Table 1: A gravity-based foundation geometry for offshore wind farm, Denmark

Foot height	3.45m	Middle cylinder height	2.65m
Cone height	3.8m	Top cylinder height	2.1m
Tower section height	1.9m	Short side length	4.65m
Foot diameter at short side	17m	Middle cylinder diameter	4.3m
Top cylinder diameter	9m		

the side walls of the wave tank causes a directional spread sea state. The elastic properties of the structure used are $E = 20$GPa and $\nu = 0.2$. The porous seabed parameters are extracted from typical values of the North sea soil measurements, Kjekstad and Lunne (1981), together with reasonable assumptions concerning the anisotropic conditions. They are: the varied directional Young's modulus $E_z = 20$MPa, $E_x = E_y = 12$MPa, directional Poisson's ratios $\nu_{xy} = 0.2$ and $\nu_{xz} = \nu_{yz} = 0.24$, and the shear modulus $G_{yz} = G_{zx} = 12$MPa and $G_{xy} = 5$MPa. The soil permeability is assumed to be $k_z = 0.001$m/s in vertical direction and $k_x = k_y = 5k_z = 0.005$m/s in horizontal directions, the soil porosity $n = 0.3$ and the saturation factor is set to $S_r = 0.98$.

The computations start from a reference state, that is, the gravity structure is already installed in place and the seabed soil has adjusted itself in equilibrium with the massive weight of the gravity structure. This is because the main purpose of this work is to study the dynamic wave effects on the structure and seabed and the interactions.

The total computational cost for running the three domains are given in Table 2 below.

Table 2: Computational data of the 1-way integration simulation for a total time period $t = 17$s.

Domain	No. of cells	No. of CPU	CPU time	Time step
Wave	1753264	8	29067s	Adjustable, ca. 1∼3ms
Structure	86400	1	105672s	Fixed, 0.05s
Seabed	247500	1	39678s	Fixed, 0.05s

Fig. 5 shows the calculated free surface of water waves at different times in a wave period. It can be seen that the presence of the structure significantly

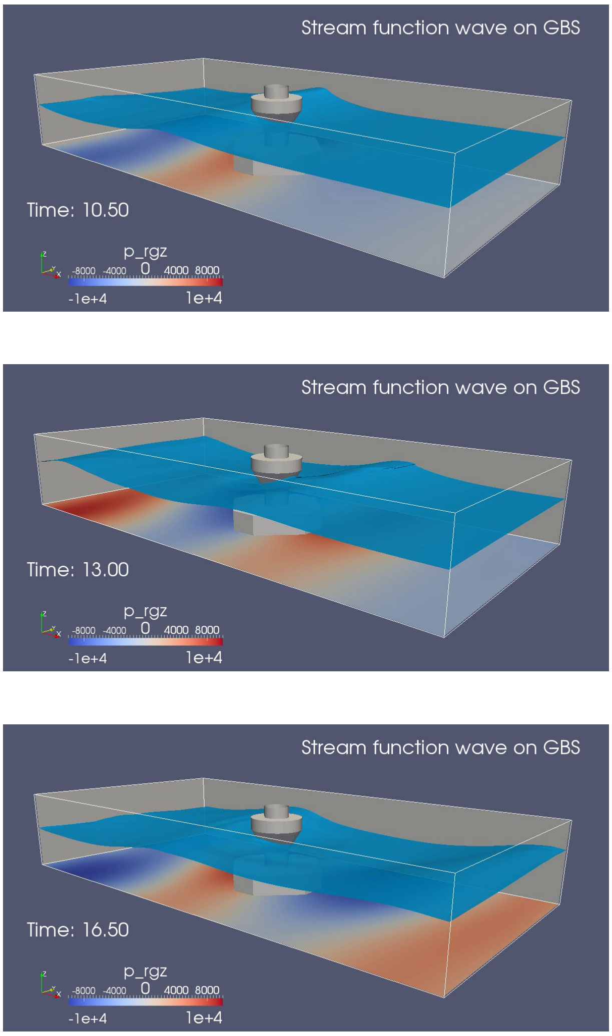


Figure 5: Free surface of waves passing by the gravity structure in a wave period. The resulting dynamic wave pressures p_rgz acting on the sea floor are colored from blue to red.

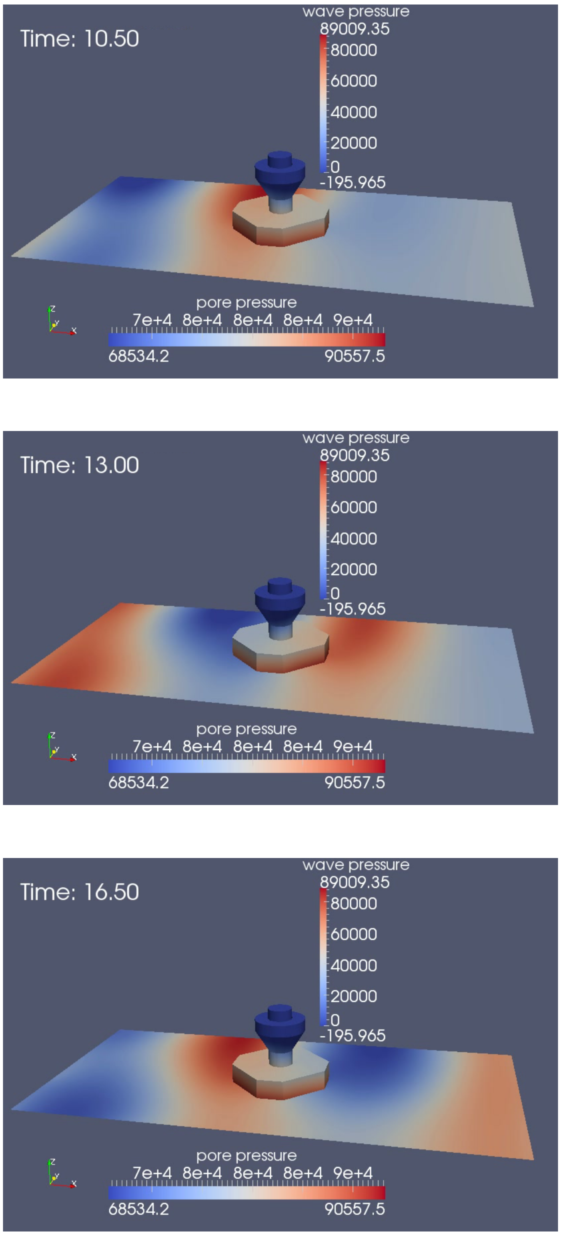


Figure 6: Generated wave pressure load on the gravity structure and porous seabed surface at times corresponding to Fig. 5. Note that the pore pressure in the soil and the wave pressure on the structure use separate scales.

modified the wave motion around it. The exerted wave pressures acting on the seafloor (Fig. 5) and on the gravity structure (Fig. 6) is highly nonlinear and 3-dimensional, partly due to the disturbance effect of the structure and also due to the directional spreading of natural waves. This complex wave pressure loading environment is impossible to get from analytical solutions based on wave theory, but can readily be estimated from advanced CFD wave solvers.

Fig. 7 provides an insight of the computed structural responses in terms of the equivalent shear stress measurement σ_{eq} as well as the magnitude of displacement, at a time $t = 10.5$s when a wave crest approached to the structure. The equivalent stress σ_{eq} is defined by $\sigma_{eq} = \sqrt{2/3\mathbf{s} : \mathbf{s}}$, in which $\mathbf{s}$ is the stress deviatoric tensor, i.e., $\mathbf{s} = \boldsymbol{\sigma} - 1/3(\mathrm{tr}\boldsymbol{\sigma})\mathbf{I}$. Considerable amount of shear stresses are generated around the cone part of the gravity structure as a consequence of the dynamic wave pressure variations. However, the resulting total displacement of the structure is still very small, which complies with the previously mentioned 1-way wave-structure interaction assumption. In Fig. 8, the total forces created at the bottom of the structure, namely the dynamic wave loads transferred through the structure to the underlying soil, are plotted. The magnitudes of the shearing forces - F_x denoted by red and F_y by green - are quite comparable to that of the normal force F_z (blue). This demonstrates that the soil underneath the structure is subjected to significant combined loading effects from the structure response.

Let us now analyze the transient seabed behavior including the pore fluid (seepage) flow, soil deformations and shearing stresses, which are important factors for geotechnical foundation design. Fig. 9 illustrates the pore fluid flows induced in the seabed at time $t = 14$s, when a wave trough is passing the gravity structure. There are notable upwards pore fluid flows created underneath the structure. It is commonly acknowledged that when the upwards seepage forces exceed the self-weight of the soil, the momentary soil liquefaction can occur and thus impacts on the safety of the upper structure, Jeng (1997). This leads to a widely applied liquefaction criteria:

$$(p - p_b) \geq \frac{1 + 2K_0}{3}(\gamma_s - \gamma_w)z \tag{13}$$

where p_b is the pore water pressure at the seabed surface, K_0 is the coefficient of lateral earth pressure and the value of $K_0 = 0.5$ is used here, and $\gamma_{s(w)}$ is the unit weight of soil and water, respectively. We have examined the liquefaction risk accordingly, and found that the seabed soil did not reach

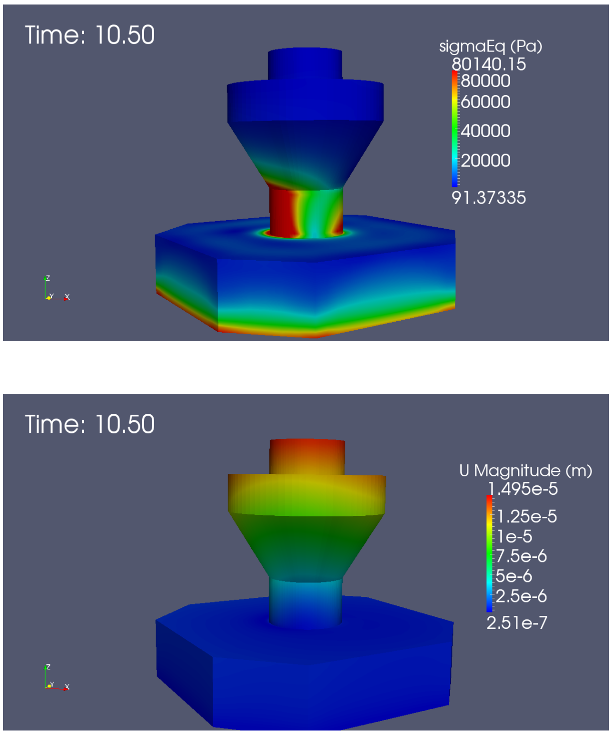


Figure 7: Computed structure responses, represented by the distributions of the equivalent shear stress (top) and the displacement magnitude (bottom).

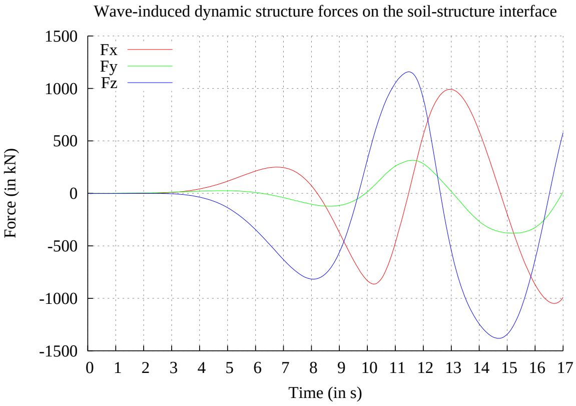


Figure 8: Combined structure force loading on the soil-structure interface.

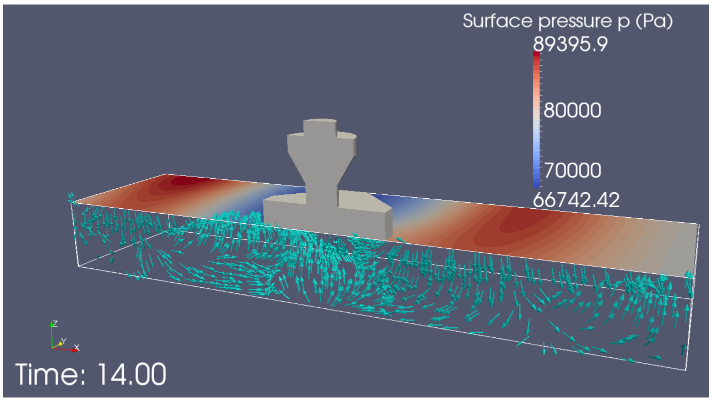


Figure 9: Seepage flows occurred in the seabed soil at time $t = 14$s (wave trough). The display is in a cutting plane along x-axis.

liquefaction failure given the wave inputs in this work.

Fig. 10 presents the calculated vertical and lateral components of the soil displacement at $t = 10.5$s. As a wave crest is crossing by the structure at that time, the soil underneath the structure is compressed accordingly, and the compressions increase from the center towards the edges of the structure. Significant amount of lateral soil displacements, particularly along the x-direction, are generated below the structure as a result of those horizontal structural traction forces shown in Fig. 8.

Fig. 11 shows the contours of the wave-induced shear stress measurement σ_{xz} in the seabed soil and the upper gravity structure. It is interesting to see that in the field away from the structure the soil shear stress directly produced by the fluctuating wave pressures is almost negligible, compared to the concentred shearing zones around the structure edges. The latter are generated because of the wave pressures as well, however, the loads are transferred trough the gravity structure and act on the soil 'indirectly'. This implies that the presence of the structure greatly amplifies the wave shearing effects down to the underlying soil. Integrated wave-structure-seabed interaction analysis can help to give correct predictions of the various loading conditions and consequently the more realistic seabed response.

5. Conclusion

In this work, a new integrated finite volume approach is proposed to model the interactions between wave, structure and seabed for offshore engi-

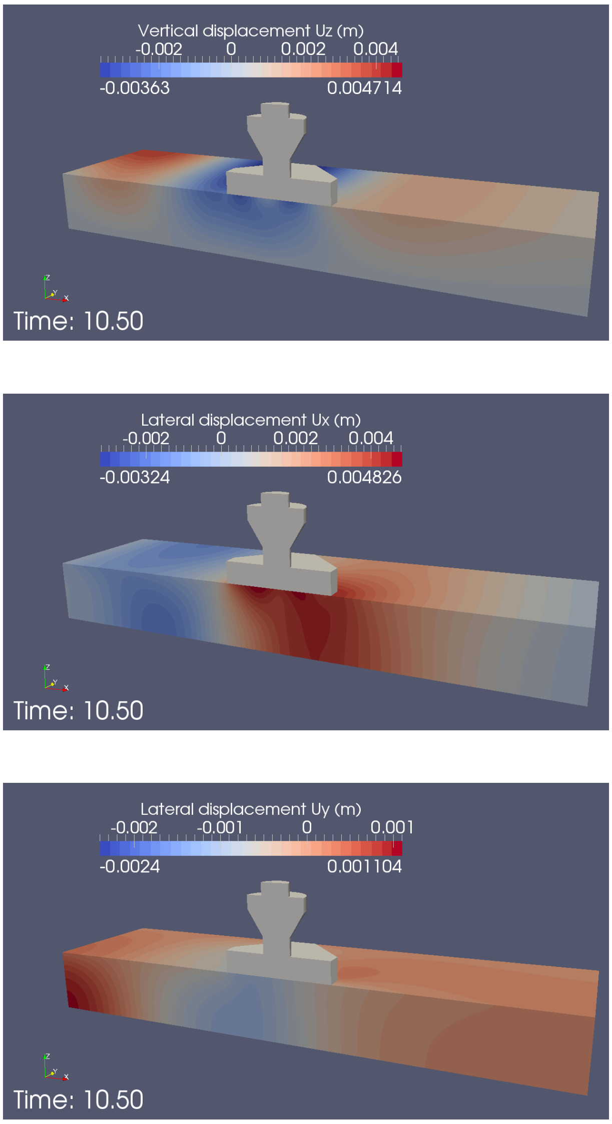


Figure 10: Contours of the calculated soil displacements at time $t = 10.50$s.

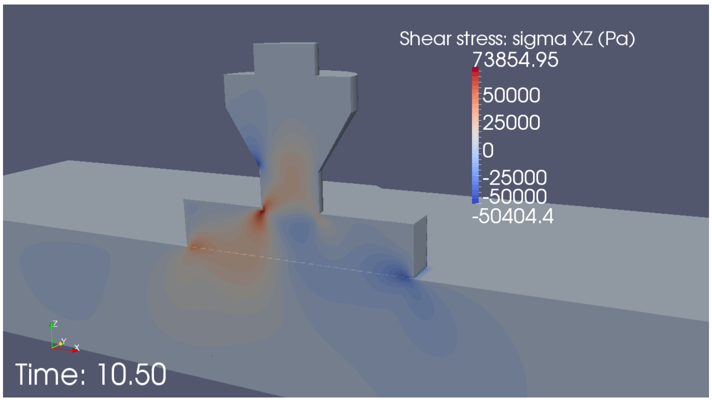


Figure 11: Contours of the shear stress distribution in the seabed and gravity structure at time $t = 10.50$s.

neering purpose. Three FVM solvers, which are dedicated to solve problems of fluid, solid and porous medium, respectively, are employed to compute the 3D wave motion, as well as the dynamic structure and seabed responses. The interactions in the system are modeled through 1-way data transfer via the interface boundaries in terms of pressures and tractions. The computations are done in the open source C++ numeric library, OpenFOAM, where preprocessing tools (e.g. FVM mesh generation), modifiable solvers, and postprocessing tools (e.g. data sampling, patch interpolation) are all freely available.

Two verification tests are first carried out to verify the newly implemented anisotropic coupled soil solver. Afterwards the integrated FVM approach is applied to solve a realistic offshore example involving a gravity base foundation. Simulation results demonstrate that the interaction analysis can provide improved wave loading environments for geotechnical assessment of the seabed soil.

An appealing feature of this work applying FVM to solve the multi-physics system is the computational efficiency. The FVM and its associated segregate solution procedure only solve small matrix system with a simple fixed iteration method for any nonlinearly coupled fluid or solid problems. As a result, in future studies where even 2-way information transfer is required for the coupling and the three subsystems shall be solved simultaneously, FVM can produce very efficient solvers without requiring heavy computational resources.

In general, the integrated modeling approach proposed in this work can be used to enhance the design of offshore structures and foundations, by providing the information on the wave-generated pore fluid flows, liquefaction risks and stress distributions.

Acknowledgements

The implementation support on the anisotropic features in OpenFOAM from Dr. Philip Cardiff at University College Dublin, Ireland, is gratefully acknowledged.

Appendix A.

An extract of the code from the developed `anisoBiotFoam` solver is shown in Listing 1. The `fvm::` operator indicates an implicit term, operator `fvc::` indicates an explicit term, operator `&` indicates a dot product, and operator `&&` indicates a double dot product. Comments given describe the different steps taken. A custom fourth order tensor class and the required operators (e.g. double dot product) implemented by Cardiff et al. (2014) have been adopted.

```
// Start the time loop
   for (runTime++; !runTime.end(); runTime++)
    {
        int iCorr = 0;
    // Start the FPI scheme
        do
        {
            p.storePrevIter();
            // Construct storage equation
      fvScalarMatrix pEqn
      (
          fvm::ddt(p) == fvm::laplacian(Dp, p) - fvc::div(fvc::
   ddt(Dp2,U))
      );
            // Solve storage equation
      pResidual = pEqn.solve().initialResidual();
            // Under-relaxation of the pore pressure
      p.relax();

            U.storePrevIter();
            // Construct momentum equation
            fvVectorMatrix UEqn
            (
                fvm::d2dt2(rho, U)
             ==
                fvm::laplacian(K, U, "laplacian(DU,U)")

              + fvc::div(C && epsilon)

              - fvc::div(K & gradU)

              - fvc::grad(p)
            );
            // Solve momentum equation
            UResidual = UEqn.solve().initialResidual();
            U.relax();
            // Recalulate displacement gradient and strain
            gradU = fvc::grad(U);
            epsilon = symm(gradU);

            initialResidual = max(pResidual,UResidual);

} while // Iterate until the explicit terms become implicit
        (initialResidual > convergenceTolerance && ++iCorr <
 nCorr);
      // Write stress field and evaluate liquefaction depth
      #       include "calculateStress.H"
      #       include "evaluateLiquefactionRisk.H"
   }
```

Listing 1: anisoBiotFoam.C

References

Anumolu, L., Trujillo, M. F., Deshpande, S. S., 2012. Evaluating the performance of the two-phase flow solver interfoam. Computational Science and Discovery 5 (1), 014016.

Biot, M., 1941. General theory of three dimensional consolidation. Journal of Applied Physics 12 (2), 155–164.

Cardiff, P., Karač, A., Ivanković, A., 2014. A large strain finite volume method for orthotropic bodies with general material orientations. Computer Methods in Applied Mechanics and Engineering 268 (0), 318 – 335.

Chen, L., Zang, J., Hillis, A., Morgan, G., Plummer, A., 2014. Numerical investigation of wave-structure interaction using openfoam. Ocean Engineering 88 (0), 91 – 109.

Demirdžić, I., Horman, I., Martinović, D., 2000. Finite volume analysis of stress and deformation in hygro-thermo-elastic orthotropic body. Computer methods in applied mechanics and engineering 190 (8), 1221–1232.

Deshpande, S. S., Anumolu, L., Trujillo, M. F., 2012. Evaluating the performance of the two-phase flow solver interfoam. Computational Science & Discovery 5 (1), 014016.

Fenton, J. D., 1999. Numerical methods for nonlinear waves. Advances in coastal and ocean engineering 5, 241–324.

Geremew, A., 2013. Pore-water pressure development caused by wave-induced cyclic loading in deep porous formation. International Journal of Geomechanics 13 (1), 65–68.

Higuera, P., Lara, J. L., Losada, I. J., 2013. Realistic wave generation and active wave absorption for navierstokes models: Application to openfoam. Coastal Engineering 71 (0), 102 – 118.

Hsu, J., Jeng, D., 1994. Wave-induced soil response in an unsaturated anisotropic seabed of finite thickness. International Journal for Numerical and Analytical Methods in Geomechanics 18 (11), 785–807.

Jasak, H., Weller, H., 2000. Application of the finite volume method and unstructured meshes to linear elasticity. International Joural for Numerical Methods in Engineering 48 (2), 267–287.

Jeng, D., 1997. Soil response in cross-anisotropic seabed due to standing waves. Journal of Geotechnical and Geoenvironmental Engineering 123 (1), 9–19.

Jeng, D. S., Ye, J. H., Zhang, J.-S., Liu, P. L. F., 2013. An integrated model for the wave-induced seabed response around marine structures: Model verifications and applications. Coastal Engineering 72, 1–19.

Kjekstad, O., Lunne, T., 1981. Soil parameters used for design of gravity platforms in the north sea. Applied Ocean Research 3 (2), 50–58.

Lekhnitskii, S., 1981. Theory of elasticity of an anisotropic elastic body. Holden Day.

Liu, X., Garcia, M. H., 2007. Numerical investigation of seabed response under waves with free-surface water flow. International Joural of Offshore and Polar Engineering 17 (2), 97–104.

Sassa, S., Sekiguchi, H., 2001. Analysis of wave-induced liquefaction of sand beds. Geotechnique 51 (2), 115–126.

Sumer, B., Fredsøe, J., 2002. The Mechanics of Scour in the Marine Environment. World Scientific, River Edge, N.J.

Tang, T., Hededal, O., 2014. Simulation of pore pressure accumulation under cyclic loading using finite volume method. Numerical Methods in Geotechnical Engineering - Proceedings of the 8th European Conference on Numerical Methods in Geotechnical Engineering, NUMGE 2014 2, 1301–1306.

Tang, T., Hededal, O., Cardiff, P., 2014. On finite volume method implementation of poro-elasto-plasticity soil model. International Journal for Numerical and Analytical Methods in Geomechanics(submitted).

Verruijt, A., 1969. Elastic storage of aquifers. In: Wiest, R. D. (Ed.), Flow through Porous Media. Academic Press, New York, pp. 37–51.

Weller, H. G., Tabor, G., Jasak, H., Fureby, C., Nov. 1998. A tensorial approach to computational continuum mechanics using object-oriented techniques. Comput. Phys. 12 (6), 620–631.

Ye, J., Jeng, D., Wang, R., Zhu, C., 2013. A 3-d semi-coupled numerical model for fluidstructuresseabed-interaction (fssi-cas 3d): Model and verification. Journal of Fluids and Structures 40 (0), 148 – 162.

Part III

Appendix

Advanced Solid Body Stress Analysis In OpenFOAM

Implementation Guide

based on OpenFOAM® Technology version 1.6-ext

Contents

1 Introduction

This Guide describes the technology of implementing various solid body stress analysis solvers using the Open Source Field Operation and Manipulation (OpenFOAM) C++ libraries.

> *One of the strengths of OpenFOAM is that new solvers and utilities can be created by its users with some pre-requisite knowledge of the underlying method, physics and programming techniques involved.*
>
> ———*OpenFOAM User's Guide*[1]

The key part of my Phd study focuses on soil modeling, which involved the understanding of several important soil mechanical features and the selected numerical method itself, namely the Finite Volume Method (FVM). Equipped with the above knowledge, I delved into the OpenFOAM library [2] and enjoyed my implementations with the vast flexibility that it provides.

> *Our intention is to make it as easy as possible to develop reliable and efficient computational continuum-mechanics codes: this is achieved by making the top-level syntax of the code as possible to conventional mathematical notation for tensors and partial differential equations.*
>
> ———*Founders of OpenFOAM*[3]

Hence this Guide is prepared with my hopes to generally demonstrate the great potentials of the OpenFOAM library to do computational solid continuum mechanics, and also to share in details my learning-process of building up new user-developed solvers in OpenFOAM.

The overall structure of the Guide is shown in Figure 1.1. I will first present a brief review on the standard released linear elastic solver - the `stressedFoam` - in chapter 2. Afterwards, I will describe the four advanced new stress solvers developed by myself using OpenFOAM. These new solvers cover material nonlinearity (i.e. elasto-plasticity), anisotropy, large strain and pore fluid coupling (poro-elasticity) feature, respectively. They are corresponding to the `epStressedFoam` in chapter 3, the `anisoFoam` in chapter 4, the `totalLagrangianFoam` in chapter 5 and the `biotFoam` in chapter 6.

For each solver, I shall present it through the following process:

physical model- - ➝numerical strategy- - ➝implementation- - ➝application

In this way, readers can easily follow up the whole story of the solver and will be freely able to either add any desired modifications or build their own specific solvers.

In chapter 7, I will further describe some more advanced stress solvers having multiple mechanical features. And in chapter 8, a couple of extra topics including the relaxation method and the block matrix solver algorithm which help for the numerical convergence, will be discussed.

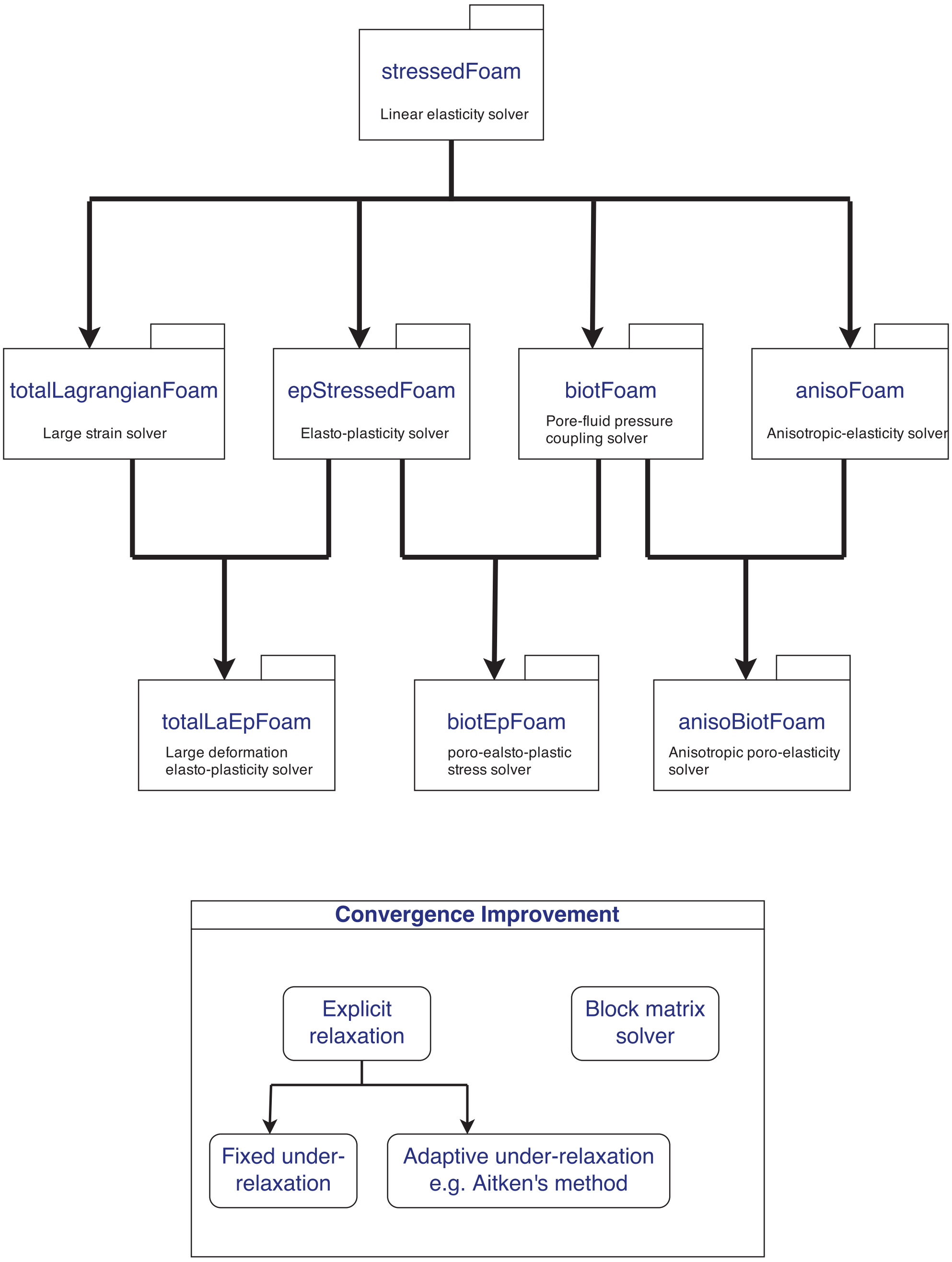


Figure 1.1: Overview of the implementation Guide

2 A short review of stressedFoam

`stressedFoam` is a linear elastic stress analysis solver released in OpenFOAM-1.6-ext. Although it is based on the simplest linear elasticity model, it possesses the following key characteristics of computational solid mechanics (CSM) using FVM framework:

- The segregated solution procedure: each component of the displacement vector is solved separately by treating the cross-component coupling explicitly.
- The fixed point iteration: the explicit cross-component coupling is updated iteratively using a simple first-order iterative method [4].
- The traction boundary condition: the force balance on the boundary is solved iteratively along with the inner solution domain.

The spirit of this linear elasticity FVM solver, namely applying cheaper iterative solvers with the necessary iteration over the explicit cross-component coupling, is very appealing. Because we can simply follow the same idea and use FVM to treat other complicated, coupled and non-linear differential equations, widely present in solid mechanics problems. As the mathematical model becomes more complex, FVM turns to be an interesting and powerful tool in CSM.

As a start, let's have a look at how `stressedFoam` is built in OpenFOAM. Shown in Figure 2.1, the `stressedFoam` solver contains the `Make` compilation subdirectory, the `tractionDisplacement` boundary condition subdirectory, the main `stressFoam.C` file and several included header files (with `.H` extension). Detailed explanation on the codes of each file is not provided here, as readers can refer to my two previous works [5, 6]. Through the following chapters, I will present the newly developed solvers, which uses more advanced mathematical models, however, still keeps the same numerical strategy and implementation structure.

```
stressedFoam
|__Make ...  This directory holds
|  |         compilation info (header
|  |         file locations and
|  |         libraries).
|  |__files
|  |__options
|__tractionDisplacement ...  This directory holds the
|  |                         traction force boundary
|  |                         condition.
|  |__tractionDisplacementFvPatchVectorField.C
|  |__tractionDisplacementFvPatchVectorField.H
|__calculateStress.H ...  This file calculates the
|                         linear elastic stress.
|__createFields.H ...  This file initializes
|                      the displacement
|                      vector field and/or
|                      the temperature field.
|__readMechanicalProperties.H ...  This file reads the
|                                  mechanical properties
|                                  from user input.
|__readStressedFoamControls.H ...  This file reads the
|                                  convergence control
|                                  parameters from user
|                                  input.
|__readThermalProperties.H ...  This file reads the
|                               thermal properties from
|                               user input.
|__stressedFoam.C ...  This file is the main
                       file that solves the
                       momentum equation.
```

Figure 2.1: The directory structure of `stressedFoam`.

3 The elasto-plasticity solver

Plasticity is one of the most important topics in solid body stress problems. Due to the high non-linearity of the differential equations, finite element method (FEM) plasticity solvers often turn to be very complicated and computationally expensive, as they produce large block-matrices usually with high condition numbers, and as a consequence rely on direct solvers. This chapter therefore presents the new FVM plasticity solver in OpenFOAM, where the non-linearity is treated in an iterative way and creates diagonally dominant matrices well suited for efficient iterative solvers.

3.1 Elasto-plasticity theory

Firstly, a brief summary of the governing equations for an elasto-plastic solid body is described as following:

$$\begin{cases} \dfrac{\partial^2[\rho(\delta\mathbf{u})]}{\partial t^2} - \nabla\cdot(\delta\boldsymbol{\sigma}) = \mathbf{0} \\ \delta\boldsymbol{\sigma} = 2\mu\delta\boldsymbol{\varepsilon}^e + \lambda\mathrm{tr}(\delta\boldsymbol{\varepsilon}^e) \\ \delta\boldsymbol{\varepsilon}^e = \delta\boldsymbol{\varepsilon} - \delta\boldsymbol{\varepsilon}^p \\ \delta\boldsymbol{\varepsilon} = \frac{1}{2}\left[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^\mathrm{T}\right] \end{cases}$$

$$\Downarrow$$

$$\frac{\partial^2[\rho(\delta\mathbf{u})]}{\partial t^2} - \nabla\cdot\left[\mu\nabla(\delta\mathbf{u}) + \mu\nabla(\delta\mathbf{u})^\mathrm{T} + \lambda\mathbf{I}\mathrm{tr}(\nabla\delta\mathbf{u})\right] + \nabla\cdot\left[2\mu(\delta\boldsymbol{\varepsilon}^p) + \lambda\mathbf{I}\mathrm{tr}\left(\delta\boldsymbol{\varepsilon}^p\right)\right] = \mathbf{0} \tag{3.1}$$

where, $\delta\mathbf{u}$: displacement increment;
$\delta\boldsymbol{\sigma}$: stress increment;
$\delta\boldsymbol{\varepsilon}$, $\delta\boldsymbol{\varepsilon}^e$, $\delta\boldsymbol{\varepsilon}^p$: total, elastic, and plastic strain increment, respectively;
ρ: density;
μ, λ: material elasticity properties (Lame's coefficients).

The plastic strain increment $\delta\boldsymbol{\varepsilon}^p$ shall be further determined by the given elasto-plastic constitutive model:

$$\delta\boldsymbol{\varepsilon}^p = \langle d\Lambda\rangle\frac{\partial g}{\partial\boldsymbol{\sigma}}, \quad \langle d\Lambda\rangle = \begin{cases} 0, & \text{when pre-yielding } f < 0. \\ d\Lambda, & \text{when yielding } f = 0. \end{cases} \tag{3.2}$$

where the yield function, f, defines whether the material undergoes recoverable elastic deformation or irrecoverable plastic deformation. The gradient to the plastic potential,

$\partial g/\partial\boldsymbol{\sigma}$, defines the direction of the plastic strain increment, and $d\Lambda$ is the plastic multiplier defining the magnitude. Classic plasticity models are von Mises and Drucker-Prager criterion. For details of those different plasticity model formulations, readers are suggested to refer to standard plasticity books, e.g. [7].

3.2 Numerical strategy

In order to solve the non-linear governing equation - Eq. (3.1), a split of implicit and explicit FVM discretization strategy adopted from [8, 9, 10] is applied:

$$\begin{aligned}\underbrace{\frac{\partial^2[\rho(\delta\mathbf{u})]}{\partial t^2}}_{implicit} - \underbrace{\nabla\cdot[(2\mu+\lambda)\nabla(\delta\mathbf{u})]}_{implicit} = \underbrace{\nabla\cdot\Big[\mu\nabla(\delta\mathbf{u})^{\mathrm{T}} + \lambda\mathbf{I}\mathrm{tr}(\nabla\delta\mathbf{u}) - (\mu+\lambda)\nabla(\delta\mathbf{u})\Big]}_{explicit} \\ - \underbrace{\nabla\cdot[2\mu(\delta\boldsymbol{\varepsilon}^p) + \lambda\mathbf{I}\mathrm{tr}\,(\delta\boldsymbol{\varepsilon}^p)]}_{explicit} \qquad (3.3)\end{aligned}$$

In above, the first explicit part corresponds to the relaxed cross-component coupling, and the second explicit part is the non-linearity. By simply treating all the coupling and non-linear terms explicitly, the resulted small linear matrix systems are solved easily by iterative solvers. Remember, we must iterate over the governing momentum equation until these explicit terms essentially become implicit, so as to guarantee fully converged solution.

3.3 epStressedFoam Implementation

In the following, I shall present the implementation techniques of the elasto-plastic stress solver based on aforementioned theories. The new solver is named `epStressedFoam`, as we are going to reuse the elastic solver codes and add the elasto-plastic features. Firstly, inside your local `solvers` directory[1], simply copy the original `stressedFoam` solver here and change the corresponding file names. We shall do the following commands in a terminal[2]:

```
// copy the stressedFoam from the installed OpenFOAM library, rename it
// as epStressedFoam.
cd of16ext
cp -r $WM_PROJECT_DIR/applications/solvers/stressAnalysis/stressedFoam \
.
mv stressedFoam epStressedFoam
// rename the main .C file.
cd epStressedFoam
mv stressedFoam.C epStressedFoam.C
```

[1] Assume that a local `solvers` directory has already been created for placing all the user-defined solvers.

[2] The command `sed -i ’s/some texts/another texts/g’ file-name(s)` replaces the word "`some texts`" with "`another texts`" in the file `file-name`. The \ symbol is used for line breaking

```
// create an empty plasticity file.
echo > correctPlasticity.H
// rename relevant header files.
mv calculateStress.H calculateEpStress.H
mv readStressedFoamControls.H readEpStressedFoamControls.H
// modify the included .H file-names in the epStressedFoam.C file.
sed -i 's/calculateStress.H/calculateEpStress.H/g' epStressedFoam.C
sed -i 's/readStressedFoamControls.H/readEpStressedFoamControls.H/g' \
epStressedFoam.C
// rename the traction boundary dictory and files.
mv tractionDisplacement epTractionDisplacement
cd epTractionDisplacement
mv tractionDisplacementFvPatchVectorField.H \
epTractionDisplacementFvPatchVectorField.H
mv tractionDisplacementFvPatchVectorField.C \
epTractionDisplacementFvPatchVectorField.C
sed -i 's/tractionDisplacement/epTractionDisplacement/g' \
epTractionDisplacementFvPatchVectorField.*
cd ..
// edit info in the compilation file.
sed -i 's/stressedFoam/epStressedFoam/g' Make/files
sed -i 's/tractionDisplacement/epTractionDisplacement/g' Make/files
sed -i 's/FOAM_APPBIN/FOAM_USER_APPBIN/g' Make/files
// re-compile the solver to make sure all the basic set-ups are correct.
wclean
wmake
```

After typing the above command lines, we shall have created the directory structure for the new `epStressedFoam` solver[3], shown in below:

```
epStressedFoam
 ├── Make
 │    ├── files
 │    └── options
 ├── epTractionDisplacement
 │    ├── epTractionDisplacementFvPatchVectorField.C
 │    └── epTractionDisplacementFvPatchVectorField.H
 ├── calculateEpStress.H
 ├── correctPlasticity.H
 ├── createFields.H
 ├── readEpStressedFoamControls.H
 ├── readMechanicalProperties.H
 └── epStressedFoam.C
```

Figure 3.1: The directory structure of `epStressedFoam`.

[3] Notice that, this Guide is only focusing on implementing isothermal mechanical stress, all the codes relating to the thermal effect has therefore been deleted.

Now we are ready to dig into each file and add those desired modifications. Any of the standard text editor programs, e.g. emacs or gedit, is fine to use.

- `createFields.H`
  We start with the `createFields.H` file, as it initializes all the unknown fields of our interests. By reusing the original file, we only need to add the new primitive variable, namely the incremental displacement $\delta\mathbf{u}$, and also other two dependent variables: the symmetric stress tensor $\boldsymbol{\sigma}$ and the (symmetric) plastic strain tensor $\boldsymbol{\varepsilon}$.

```
    Info<< "Reading field U\n" << endl;

    volVectorField U
    (
        IOobject
        (
            "U",
            runTime.timeName(),
            mesh,
            IOobject::READ_IF_PRESENT, // optionally read from user input when non-zero initial displacement condition
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedVector("U", dimLength, vector::zero)
    );

    volVectorField dU
    (
        IOobject
        (
            "dU",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ, // must read from user specified boundary condition
            IOobject::AUTO_WRITE
        ),
        mesh
    );

    volSymmTensorField sigma
    (
        IOobject
        (
            "sigma",
            runTime.timeName(),
            mesh,
            IOobject::READ_IF_PRESENT, // read from user input when non-zero initial stress condition
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("sigma", dimForce/dimArea, symmTensor::zero)
    );

    volSymmTensorField dEpsP
    (
        IOobject
        (
            "dEpsP",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ, // assume no plasticity in the beginning of simulation
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("dEpsP", dimless, symmTensor::zero)
    );
```

Listing 3.1: `createFields.H`

- `readMechanicalProperties.H`
  Then we are going to modify the `readMechanicalProperties.H` file, so as to add those plastic properties present in the specific plasticity model we have chosen. Worth to mention that, in the original file the Young's modulus parameter is normalized with respect to density as later the whole governing equation will be normalized by the density. However, here we will not follow this way, the elastic parameters are kept as they are without any normalization. The code snippet of the modified `readMechanicalProperties.H` file is present below:

```
    Info<< "Reading mechanical properties\n" << endl;

    IOdictionary mechanicalProperties
    (
        IOobject
        (
            "mechanicalProperties",
            runTime.constant(),
            mesh,
            IOobject::MUST_READ,
            IOobject::NO_WRITE
        )
    );

    dimensionedScalar rho(mechanicalProperties.lookup("rho"));
    dimensionedScalar E(mechanicalProperties.lookup("E")); //
```

```
       Young's Modulus
    dimensionedScalar nu(mechanicalProperties.lookup("nu")); // Poisson's ratio

    Info<< "Calculating Lame's coefficients\n" << endl;

    dimensionedScalar mu = E/(2.0*(1.0 + nu));
    dimensionedScalar lambda = nu*E/((1.0 + nu)*(1.0 - 2.0*nu));

    Switch planeStress(mechanicalProperties.lookup("planeStress"));

    if (planeStress)
    {
        Info<< "Plane Stress\n" << endl;

        //- change lambda for plane stress
        lambda = nu*E/((1.0 + nu)*(1.0 - nu));
    }
    else
    {
        Info<< "Plane Strain\n" << endl;
    }
    Info<< "mu = " << mu.value() << " Pa/rho\n";
    Info<< "lambda = " << lambda.value() << " Pa/rho\n";

    Info<< "Reading plastic properties\n";
    dimensionedScalar sigmaY(mechanicalProperties.lookup("sigmaY")); // Yielding stress
   /*
    dimensionedScalar varPhi(mechanicalProperties.lookup("varPhi")); // Friction angle
    dimensionedScalar c(mechanicalProperties.lookup("c")); // Cohesion
    dimensionedScalar varPsi(mechanicalProperties.lookup("varPsi")); // Dilation angle
   */
```

Listing 3.2: `readMechanicalProperties.H`

- `readEpStressedFoamControls.H`

  The `readEpStressedFoamControls.H` file contains information on convergence control parameters. Here we only need to edit two places using `sed` commands (you could also do direct edition by using text editor):

```
sed -i 's/stressedFoam/epStressedFoam/g' readEpStressedFoamControls.H
sed -i 's/U/dU/g' readEpStressedFoamControls.H
```

  The new `readEpStressedFoamControls.H` file shall look like as following:

```
    const dictionary& stressControl =
        mesh.solutionDict().subDict("epStressedFoam");

    int nCorr(readInt(stressControl.lookup("nCorrectors"))); // maximum iteration numbers
    scalar convergenceTolerance(readScalar(stressControl.lookup("dU"))); // desired residual
```

Listing 3.3: `readEpStressedFoamControls.H`

- `epStressedFoam.C`

  Now we can edit the main `epStressedFoam.C` file to solve the governing equation - Eq. (7.1). As in our case we would like to solve the displacement increment $\delta\mathbf{u}$, instead of the total displacement $\mathbf{u}$ in `stressedFoam.C`, we could firstly replace all the `U` text with `dU` by a single command line:

  ```
  sed -i 's/U/dU/g' epStressedFoam.C
  ```

  Then we can start to add those explicit plasticity terms into the equation. Remember that, we need to iteratively update the plastic strain, which means the plasticity computation file - `correctPlasticity.H` file has to be placed inside the momentum loop. The final `epStressedFoam.C` file is present in Listing 3.4:

```
#include "fvCFD.H"
#include "Switch.H"

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

int main(int argc, char *argv[])
{

#   include "setRootCase.H"

#   include "createTime.H"
#   include "createMesh.H"
#   include "readMechanicalProperties.H" // read material properties
#   include "createFields.H" // initialize all the computing varaibles

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    Info<< "\nCalculating displacement field\n" << endl;

    for (runTime++; !runTime.end(); runTime++) // time loop
    {
        Info<< "Iteration: " << runTime.timeName() << nl << endl;

#       include "readEpStressedFoamControls.H" // read convergence control parameters
```

```

        int iCorr = 0;
        scalar initialResidual = 0;

        do // momentum loop, using the simple fixed point iteration
        {
            volTensorField graddU = fvc::grad(dU); // compute the current displacement gradient field

            fvVectorMatrix dUEqn // construct the momentum matrix
            (
                fvm::d2dt2(rho, dU) // implicit time derivative term
             ==
                fvm::laplacian(2*mu + lambda, dU, "laplacian(DdU,dU)") // implicit over-relaxed laplacian term

              + fvc::div
                (
                    mu*graddU.T() + lambda*(I*tr(graddU)) - (mu + lambda)*graddU,
                    "div(sigmaE)"
                ) // explicit cross-component coupling terms (after over-relaxation)
              - fvc::div
                (
                    2.0*mu*dEpsP + lambda*(I*tr(dEpsP)),
                    "div(sigmaP)"
                ) // explicit non-linear plastic terms
            );

            initialResidual = dUEqn.solve().initialResidual(); // solve the momentum equation and store the residual

#       include "correctPlasticity.H" // based on the adopted elasto-plastic constitutive model, calculate the plastic strain for each control volume using the currently solved dU value.

        } while (initialResidual > convergenceTolerance && ++iCorr < nCorr); // evaluate the convergence condition

        U += dU; // update the total displacement field

#       include "calculateEpStress.H" // calculate the correct stress upon converged solution

        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
```

```
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
    }

    Info<< "End\n" << endl;

    return(0);
}
```

Listing 3.4: `epStressedFoam.C`

- `correctPlasticity.H`
  Up to this stage, the `correctPlasticity.H` file is still empty, with no plasticity information incorporated. Here I select to implement a simple von Mises J_2 plasticity model for demonstration purpose. Any other advanced plasticity model can be implemented as well. Depending on the plasticity model itself, implicit or explicit stress integration procedure shall be carefully examined, so that the local integration is consistent to the global momentum solution procedure. In our case of the simple von Mises model, a radial return algorithm is chosen for the implementation. Readers are invited to refer to the tutorial [11], which gives detailed information on the J2 plasticity formulation and the radial return method. The code snippet of the `correctPlasticity.H` file is shown in Listing 3.5. Notice that, it is more accurate that we calculate the plastic strain not only in the internal domain but also along all the boundary patches.

```
        const volSymmTensorField& sigma_old = sigma;

     // Get the trial updated stress
        volSymmTensorField sigmaNew =sigma_old+ 2.0*mu*symm(graddU) + lambda*I*tr(graddU);

     // Check the yield condition
        volScalarField sqrtJ2 = sqrt((3.0/2.0)*magSqr(dev(sigmaNew)));
        volScalarField fac = sqrtJ2/sigmaY;

        symmTensorField& sigmaI = sigmaNew.internalField();
        symmTensorField& dEpsPI = dEpsP.internalField();

        const symmTensorField& sigma_oldI = sigma_old.internalField();
        const scalarField& facI = fac.internalField();
        const tensorField& graddUI = graddU.internalField();

        forAll(facI, celli) // iterate all the internal cells
        {
            if (facI[celli] > 1.0)         // Plasticity occurs
             {
                sigmaI[celli] = 1.0/3.0*I*tr(sigmaI[celli]) + dev(sigmaI[celli])/facI[celli];
                symmTensor dsigma = sigmaI[celli] - sigma_oldI[celli];
                tensor deps_e = 1.0/3.0*I*tr(dsigma)/(3.0*lambda+2.0*mu).value()
                                + dev(dsigma)/(2.0*mu.value());
                tensor deps = 1.0/2.0*(graddUI[celli] + graddUI[celli].T());
                dEpsPI[celli] = symm(deps - deps_e);
             }
             else                            // only elasticity
             {
                dEpsPI[celli] = symmTensor::zero;
             }
         }

       forAll(fac.boundaryField(), patchI) // iterate all the boundary cells
       {
        symmTensorField& sigmaPatch = sigmaNew.boundaryField()[patchI];
        symmTensorField& dEpsPPatch = dEpsP.boundaryField()[patchI];

        const symmTensorField& sigma_oldPatch = sigma_old.boundaryField()[patchI];
        const scalarField& facPatch = fac.boundaryField()[patchI];
        const tensorField& graddUPatch = graddU.boundaryField()[patchI];

         forAll(facPatch, facei)
        {
            if (facPatch[facei] > 1.0)          //Plasticity occurs
             {
                sigmaPatch[facei] = 1.0/3.0*I*tr(sigmaPatch[facei]) + dev(sigmaPatch[facei])/facPatch[facei];
                symmTensor dsigma = sigmaPatch[facei] - sigma_oldPatch[facei];
                tensor deps_e = 1.0/3.0*I*tr(dsigma)/(3.0*lambda+2.0*mu).value()
                                + dev(dsigma)/(2.0*mu.value());
                tensor deps = 1.0/2.0*(graddUPatch[facei] + graddUPatch[facei].T());
                dEpsPPatch[facei] = symm(deps - deps_e);
             }
             else                            // only elasticity
             {
                dEpsPPatch[facei] = symmTensor::zero;
             }
         }
       }
```

Listing 3.5: `correctPlasticity.H`

- `calculateEpStress.H`
  During each time/load step, we calculate the stress condition using converged displacement solution (by placing the `calculateEpStress.H` file out of the momentum loop in Listing 3.4). The content of the `calculateEpStress.H` file is shown below.

```
        volTensorField graddU = fvc::grad(dU); // calculate the total strain increment

        sigma += 2.0*mu*(symm(graddU)-dEpsP)
                 + lambda*I*tr(symm(graddU)-dEpsP); // update the stress field with plastic correction

        volScalarField sigmaEq
        (
            IOobject
            (
                "sigmaEq",
                runTime.timeName(),
                mesh,
                IOobject::NO_READ,
                IOobject::AUTO_WRITE
            ),
            sqrt((3.0/2.0)*magSqr(dev(sigma))) // calculate the von-Mises stress
        );

        Info<< "Max sigmaEq = " << max(sigmaEq).value()
            << endl;
```

Listing 3.6: `calculateEpStress.H`

- `epTractionDisplacement/epTractionDisplacementFvPatchVectorField.C`
  So far we have completed the main part of the `epStressedFoam` solver. The only task left is the nonlinear traction boundary. We will now implement the nonlinear `epTractionDisplacement` boundary using force balance equation on the boundary. As the force balance equation has the same cross-component coupling and nonlinearity as the global momentum equation, we have to apply similar segregated implicit-explicit treatment, see Eq. (3.4). Regarding to the implementation, we only need to modify the `updateCoeffs()` function, as it updates the normal gradient of

the displacement increments based on the specified traction force on the boundary.

$$\begin{cases} \delta\mathbf{t} = (\delta\boldsymbol{\sigma})\cdot\mathbf{n}_b \\ \delta\boldsymbol{\sigma} = 2\mu\delta\boldsymbol{\varepsilon}^e + \lambda\mathrm{tr}(\delta\boldsymbol{\varepsilon}^e) \\ \delta\boldsymbol{\varepsilon}^e = \delta\boldsymbol{\varepsilon} - \delta\boldsymbol{\varepsilon}^p \\ \delta\boldsymbol{\varepsilon} = \frac{1}{2}\left[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}}\right] \end{cases}$$

$$\Downarrow$$

$$\begin{aligned} \underbrace{(2\mu+\lambda)\mathbf{n}_b\cdot[\nabla(\delta\mathbf{u})]_b}_{\text{implict}} =& (d\mathbf{t})_b \\ &- \underbrace{\{\mu[\nabla(\delta\mathbf{u})]_b\cdot\mathbf{n}_b + \lambda\mathrm{tr}\left([\nabla(\delta\mathbf{u})]_b\right)\mathbf{n}_b - (\mu+\lambda)\mathbf{n}_b\cdot[\nabla(\delta\mathbf{u})]_b\}}_{\text{explicit cross-component coupling}} \\ &+ \underbrace{\{2\mu\mathbf{n}_b\cdot(\delta\boldsymbol{\varepsilon}^p)_b + \lambda\mathrm{tr}(\delta\boldsymbol{\varepsilon}^p)_b\mathbf{n}_b\}}_{\text{explicit nonlinearity}} \end{aligned} \tag{3.4}$$

where $\delta\mathbf{t}$: traction increment;
$\mathbf{n}_b$: boundary surface normal vector;

The corresponding implementation code is displayed in Listing 3.7.

```
// Update the coefficients associated with the patch field
void epTractionDisplacementFvPatchVectorField::updateCoeffs()
{
    if (updated())
    {
        return;
    }

    const dictionary& mechanicalProperties =
        db().lookupObject<IOdictionary>("mechanicalProperties");

    dimensionedScalar E(mechanicalProperties.lookup("E"));
    dimensionedScalar nu(mechanicalProperties.lookup("nu"));

    dimensionedScalar mu = E/(2.0*(1.0 + nu));
    dimensionedScalar lambda = nu*E/((1.0 + nu)*(1.0 - 2.0*nu));

    Switch planeStress(mechanicalProperties.lookup("planeStress"))
       ;

    if (planeStress)
    {
        lambda = nu*E/((1.0 + nu)*(1.0 - nu));
    }

    vectorField n = patch().nf(); // store the boundary surface
       normal vector
    vectorField dTraction(n.size(),vector::zero); // initialize the traction increment field

    const fvPatchField<tensor>& graddU =
        lookupPatchField<volTensorField, tensor>("grad(dU)"); // copy the current displacement gradient field

    const fvPatchField<symmTensor>& sigma =
        lookupPatchField<volSymmTensorField, symmTensor>("sigma"); // copy the stress field, notice that here the stress is from the previous time step, not the iteration step

    const fvPatchField<symmTensor>& dEpsP =
        lookupPatchField<volSymmTensorField, symmTensor>("dEpsP"); // copy the current plastic strain field

    dTraction =  (traction_ - pressure_*n) - (n & sigma); // calculate the traction increment = user specified current traction - calculated previous time step traction

    gradient() =
    (
      dTraction
      - (n & (mu.value()*graddU.T() - (mu + lambda).value()*graddU))
      - n*tr(graddU)*lambda.value() // explicit cross-component coupling
      + 2.0*mu.value()*(n & dEpsP) + n*tr(dEpsP)*lambda.value() // explicit non-linearity
    )/(2.0*mu + lambda).value();

    fixedGradientFvPatchVectorField::updateCoeffs();
}
```

Listing 3.7: `epTractionDisplacement/epTractionDisplacementFvPatchVectorField.C`

Now we can compile all the codes by typing `wmake` command. After successful compilation, the new solver `epStressedFoam` is ready to apply for elasto-plastic solid problems.

3.4 Application

Here I will shortly describe general case set-up in OpenFOAM. As illustrated in Fig. 3.2, any OpenFOAM case contains 3 main directories:

0 The `0` directory includes several files, which are usually entitled with the corresponding unknown variable name. Inside the file, initial and boundary condition is specified.

constant The `constant` directory contains the mesh generation file and some material properties libraries.

system The `system` directory mainly set up the time step controls, the linear system solvers, and the numerical schemes.

For more information regarding to the case setup, please refer to the OpenFOAM User's Guide [1].

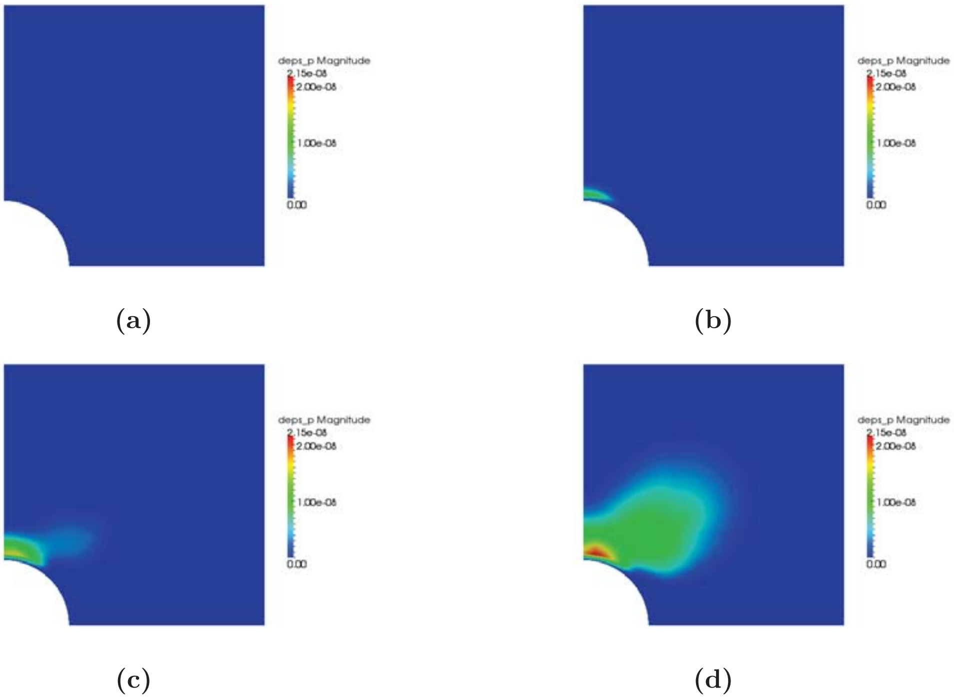
```
OpenFOAM case
 |__0
 |   |__variable1
 |   |__variable2
 |   |__variableX
 |__constant
 |   |__polyMesh
 |   |   |__blockMeshDict
 |   |   |__boundary
 |   |__materialLibrary1
 |   |__materialLibraryX
 |__system
     |__controlDict
     |__fvSchemes
     |__fvSolution
```

Figure 3.2: Basic OpenFOAM case set-up.

For the `epStressedFoam` solver, let's run two simple cases to demonstrate its performance.

- Plastic plate hole
  We first run a `plasticPlateHole` case, which is simply based on the original `plateHole` case in OpenFOAM. The test condition is set that the right hand side edge of the plate is pulled with increasing traction force (constant increasing speed). It is expected that plastic deformations will be built up with time around the hole. Figure 6.5 represents reasonable simulation results of the `epStressedFoam` solver.

- Triaxial test
  A simple triaxial compression test with cubic geometry is also created. The top and bottom sides are compressed uniformly under constantly increasing traction forces, while all the horizontal sides are left with zero traction. Worth to mention that, we have simply modified the J2 plasticity model into a non-associated Mohr Coulomb plasticity model [12] so as to test the flexibility of the `epStressedFoam` solver. The simulation results are shown in Figure 3.4, with well captured material failure behavior and the dilation effect. It is believed that the `epStressedFoam` solver has large potentials to incorporate any other advanced plasticity model.



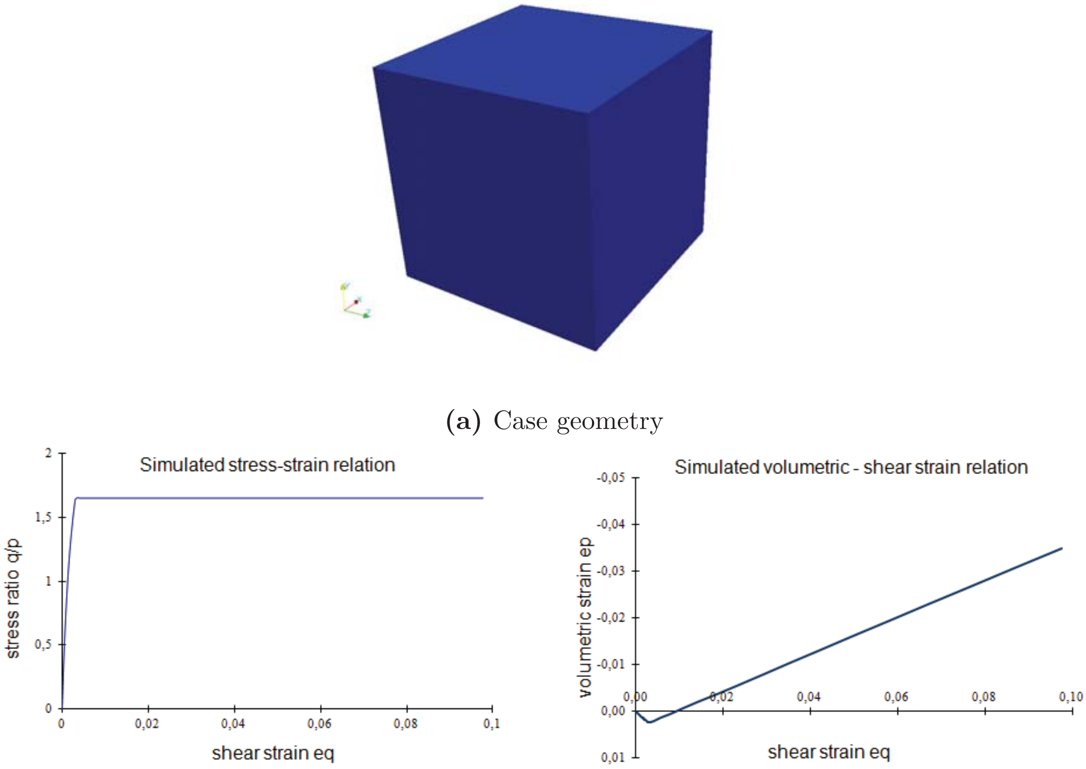
(a)

(b)

(c)

(d)

Figure 3.3: The development of plastic zones, represented by `mag(dEpsP)`. Traction=$t \cdot 10^3 Pa$, $\sigma_Y = 10^3 Pa$



(a) Case geometry

(b)

Figure 3.4: Triaxial test simulation reulsts. Employed material properties: friction angle=40°, cohesion=5kPa, dilation angle=10°

4 The anisotropic elasticity solver

Anisotropy, in material science, is a material's directional dependence of a physical property. Most materials exhibit anisotropic behavior, namely, they are stiffer when loaded along some material directions than others. Such materials are said to be anisotropic, and cannot be modeled using the conventional isotropic stress analysis procedures in OpenFOAM.

In this chapter, we are therefore going to implement a new anisotropic elasticity solver, capable of simulating solid bodies with directional dependent elastic properties. As you might expect, modeling full anisotropy is very difficult and complicated, requiring 21 independent elastic constants. Here we will focus on the so-called orthotropic solid body, so that its mechanical properties are, in general, only different along each axis (thus requiring 9 independent elastic constants).

The methodology of doing orthotropic stress analysis with finite volume discretization was firstly proposed by Demirdzic et al. 2000 [13] and recently extended by Cardiff et al. 2013 [14] into large strain problems with general material orientations. Readers are therefore kindly invited to read these articles, so as to gain better knowledge on the orthotropic elasticity using FVM.

4.1 Orthotropic elasticity theory

In this section, I will shortly present the orthotropic elasticity formulations.

Consider the following generalized Hook's Law:

$$\boldsymbol{\sigma} = \mathbf{C} : \boldsymbol{\varepsilon} \tag{4.1}$$

Where
$\boldsymbol{\sigma}$, the second-order symmetric stress tensor;
$\mathbf{C}$, the fourth-order constitutive tensor of elastic constants;
$\boldsymbol{\varepsilon}$, the second-order symmetric strain tensor;
:, the double dot operator.

For visualization purpose, we could arrange these second order symmetric tensors into vectors format and similarly transform the forth order elasticity tensor into second order matrix, Hook's Law can therefore be rewritten as following for an orthotropic solid body:

$$\boldsymbol{\sigma} = \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{31} & 0 & 0 & 0 \\ A_{12} & A_{22} & A_{23} & 0 & 0 & 0 \\ A_{31} & A_{23} & A_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{66} \end{pmatrix} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{pmatrix} = \mathbf{C} : \boldsymbol{\varepsilon} \tag{4.2}$$

where the stiffness coefficients A_{ij} are given in terms of Young's moduli E_i, Poisson's ratio ν_{ij}, and shear moduli G_{ij}:

$$\begin{aligned} & A_{11} = \frac{1-\nu_{23}\nu_{32}}{JE_2E_3}, \quad A_{22} = \frac{1-\nu_{13}\nu_{31}}{JE_1E_3}, \quad A_{33} = \frac{1-\nu_{21}\nu_{12}}{JE_2E_1}, \\ & A_{12} = \frac{\nu_{12}+\nu_{32}\nu_{13}}{JE_1E_3}, \quad A_{23} = \frac{\nu_{23}+\nu_{21}\nu_{13}}{JE_1E_2}, \quad A_{31} = \frac{\nu_{31}+\nu_{21}\nu_{32}}{JE_2E_3}, \\ & A_{44} = 2G_{12}, \quad A_{55} = 2G_{23}, \quad A_{66} = 2G_{31}, \\ & J = \frac{1-\nu_{12}\nu_{21}-\nu_{23}\nu_{32}-\nu_{13}\nu_{31}-2\nu_{21}\nu_{32}\nu_{13}}{E_1E_2E_3} \end{aligned} \tag{4.3}$$

In general, the global momentum balance equation can be written in the form below:

$$\frac{\partial^2(\rho\mathbf{u})}{\partial t^2} - \nabla\cdot\boldsymbol{\sigma} = \mathbf{0} \quad \Rightarrow \quad \frac{\partial^2(\rho\mathbf{u})}{\partial t^2} = \nabla\cdot(\mathbf{C}:\boldsymbol{\varepsilon}) \tag{4.4}$$

The above formulation is still quite simple, however it is significantly different from the isotropic model solved before in OpenFOAM, as the elasticity tensor **C** becomes complex forth-order tensor and can no longer be represented by only two Lame's constants, i.e. μ and λ. We need a cleverer way to deal with this high-order momentum equation with more complicated cross component coupling. The following section will introduce the resolution.

4.2 Numerical method

To deal with the cross-component coupling inside Eq. (4.4), the right hand side part can be decomposed to implicit and explicit components:

$$\mathbf{C}:\boldsymbol{\varepsilon} = \underbrace{\mathbf{K}\cdot\nabla\mathbf{u}}_{implicit} + \underbrace{\mathbf{C}:\boldsymbol{\varepsilon} - \mathbf{K}\cdot\nabla\mathbf{u}}_{explicit} \tag{4.5}$$

where the 3 × 3 diagonal tensor **K** is given by:

$$\mathbf{K} = \begin{pmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{pmatrix} \tag{4.6}$$

Eq. (4.5) can easily be verified by canceling out the implicit $\mathbf{K} \cdot \nabla \mathbf{u}$ with the explicit $\mathbf{K} \cdot \nabla \mathbf{u}$.

Hence, to allow the FVM segregated solution procedure, the orthotropic governing momentum equation is solved by the following numerical strategy:

$$\underbrace{\frac{\partial^2(\rho \mathbf{u})}{\partial t^2}}_{implicit} = \nabla \cdot (\underbrace{\mathbf{K} \cdot \nabla \mathbf{u}}_{implicit} + \underbrace{\mathbf{C} : \varepsilon - \mathbf{K} \cdot \nabla \mathbf{u}}_{explicit}) \tag{4.7}$$

Now the only challenge left is how to calculate the explicit $(\mathbf{C} : \varepsilon)$ term, since currently no forth-order tensor class exists in OpenFOAM library yet, nor the double dot operator between a forth-order tensor and a second-order tensor is implemented. I shall introduce how these difficulties are solved in the following implementation section.

4.3 anisoFoam Implementation

The new solver is named `anisoFoam`. Again, we can start with the original `stressedFoam` solver codes to save some work.

Similarly, by typing the command lines like what we have done to create the basic set up for the `epStressedFoam` in Page. 6, we can firstly build the following basic file structure of the new `anisoFoam` solver:

```
anisoFoam
 ├── Make
 │    ├── files
 │    └── options
 ├── tractionOrtho
 │    ├── tractionOrthoFvPatchVectorField.C
 │    └── tractionOrthoFvPatchVectorField.H
 ├── calculateSigmaEpsilonOrtho.H
 ├── createFields.H
 ├── readAnisoFoamControls.H
 ├── readMechanicalProperties.H
 └── anisoFoam.C
```

Figure 4.1: The directory structure of `anisoFoam`.

Now I shall describe those modifications inside each file as well as the contents of the new `calculateSigmaEpsilonOrtho.H` file.

- `createFields.H`
  In this case, a strain field is newly created for the convenience of explicit calculation of $(\mathbf{C} : \varepsilon)$ inside the momentum equation.

```
    Info<< "Reading field U\n" << endl;
    volVectorField U
    (
        IOobject
        (
            "U",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ,
            IOobject::AUTO_WRITE
        ),
        mesh
    );

    volTensorField gradU = fvc::grad(U);

    volSymmTensorField epsilon // the symmetric strain tensor
    (
        IOobject
        (
            "epsilon",
            runTime.timeName(),
            mesh,
            IOobject::READ_IF_PRESENT, // read from user input when initial strain condition available
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("zero", dimless, symmTensor::zero)
    );

    volSymmTensorField sigma
    (
        IOobject
        (
            "sigma",
            runTime.timeName(),
            mesh,
            IOobject::READ_IF_PRESENT, // read from user input when initial stress condition available
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("zero", dimForce/dimArea, symmTensor::zero)
    );
```

Listing 4.1: `createFields.H`

- `readMechanicalProperties.H`
  As the orthotropic elasticity contains 9 independent elastic constants comparing to the simple isotropic elasticity with only 2 constants, the `readMechanicalProperties.H` files turn to be a little bit more complicated this time. Each coefficient of the forth-order elasticity tensor **C** is calculated according to Eq. (4.3).

```
    Info<< "Reading mechanical properties\n" << endl;

    IOdictionary mechanicalProperties
    (
        IOobject
        (
            "mechanicalProperties",
            runTime.constant(),
            mesh,
            IOobject::MUST_READ,
            IOobject::NO_WRITE
        )
    );

    dimensionedScalar rho(mechanicalProperties.lookup("rho"));

    // Initialize Young's modulus at different directions
    scalar Ex = 0;
    scalar Ey = 0;
    scalar Ez = 0;

    // Initializez Poisson's ratio at different directions
    scalar vxy = 0;
    scalar vyz = 0;
    scalar vzx = 0;
    scalar vyx = 0;
    scalar vxz = 0;
    scalar vzy = 0;

    // Initialize shear modulus at different directions
    scalar Gxy = 0;
    scalar Gyz = 0;
    scalar Gzx = 0;

    // Create a switch for simulating 2D problem or 3D problem
    Switch model2d(mechanicalProperties.lookup("model2D"));

if(model2d) // 2D condition
  {
     Ex = readScalar(mechanicalProperties.lookup("Ex"));
     Ey = readScalar(mechanicalProperties.lookup("Ey"));
     vxy = readScalar(mechanicalProperties.lookup("nuxy"));
     Gxy = readScalar(mechanicalProperties.lookup("Gxy"));

     //- material contraints
     vyx = vxy*Ey/Ex;

     Info << "2D model (z is assumed to be the empty direction)" << nl
          << "Linear elastic orthotropic properties are:\n"
          << "Ex " << Ex/1e9 << " GPa" << nl
          << "Ey " << Ey/1e9 << " GPa" << nl
          << "nuxy " << vxy << nl
          << "Gxy " << Gxy/1e9 << " GPa" << endl;
  }
 else // 3D condition
   {
     Ex = readScalar(mechanicalProperties.lookup("Ex"));
     Ey = readScalar(mechanicalProperties.lookup("Ey"));
     Ez = readScalar(mechanicalProperties.lookup("Ez"));
     vxy = readScalar(mechanicalProperties.lookup("nuxy"));
     vyz = readScalar(mechanicalProperties.lookup("nuyz"));
     vzx = readScalar(mechanicalProperties.lookup("nuzx"));
     Gxy = readScalar(mechanicalProperties.lookup("Gxy"));
     Gyz = readScalar(mechanicalProperties.lookup("Gyz"));
     Gzx = readScalar(mechanicalProperties.lookup("Gzx"));

     //- material contraints
     vyx = vxy*Ey/Ex;
     vxz = vzx*Ex/Ez;
     vzy = vyz*Ez/Ey;

     Info << "3D model" << nl
          << "Linear elastic orthotropic properties are:\n"
          << "Ex " << Ex/1e9 << " GPa" << nl
          << "Ey " << Ey/1e9 << " GPa" << nl
          << "Ez " << Ez/1e9 << " GPa" << nl
          << "nuxy " << vxy << nl
          << "nuyz " << vyz << nl
          << "nuzx " << vzx << nl
          << "Gxy " << Gxy/1e9 << " GPa" << nl
          << "Gyz " << Gyz/1e9 << " GPa" << nl
          << "Gzx " << Gzx/1e9 << " GPa" << endl;
   }

    //- components of C - fourth order elastic constants tensor
scalar J = 0;
scalar A11 = 0;
scalar A22 = 0;
scalar A33 = 0;
scalar A12 = 0;
scalar A31 = 0;
scalar A23 = 0;
scalar A44 = 0;
scalar A55 = 0;
scalar A66 = 0;

scalar A21 = 0;

if(model2d)
  {
    J = 1 / (1 - vxy*vyx);
    A11 = J*Ex;
    A22 = J*Ey;
    A12 = J*vyx*Ex;
    A21 = J*vxy*Ey;
    A44 = 2*Gxy;

    Info << "\n2D stiffness coefficients are:" << nl
         << "A11 " << A11 << nl
         << "A22 " << A22 << nl
         << "A12 " << A12 << nl
         << "A21 " << A21 << nl
         << "A44 " << A44 << endl;
  }
 else
   {
      J = (1.0 - vxy*vyx - vyz*vzy - vzx*vxz - 2*vyx*vzy*vxz) / (Ex*Ey*Ez);
      A11 = (1.0 - vyz*vzy)/(J*Ey*Ez);
      A22 = (1.0 - vxz*vzx)/(J*Ex*Ez);
      A33 = (1.0 - vyx*vxy)/(J*Ey*Ex);
      A12 = (vxy + vzy*vxz)/(J*Ex*Ez);
      A31 = (vzx + vyx*vzy)/(J*Ey*Ez);
      A23 = (vyz + vyx*vxz)/(J*Ex*Ey);
      A44 = 2*Gxy;
      A55 = 2*Gyz;
      A66 = 2*Gzx;

      Info << "\n3D stiffness coefficients:" << nl
           << "A11 " << A11 << nl
           << "A22 " << A22 << nl
           << "A33 " << A33 << nl
           << "A12 " << A12 << nl
           << "A31 " << A31 << nl
           << "A23 " << A23 << endl;
   }

tensor Ku(
          A11,  0 ,  0 ,
             0  , A22,  0 ,
             0  ,  0 ,  A33
          );

Info << "\nThe implicit stiffness tensor K is " << Ku << endl;

volTensorField K
(
 IOobject
  (
   "K",
   runTime.timeName(),
   mesh,
   IOobject::NO_READ,
   IOobject::NO_WRITE
  ),
 mesh,
 dimensionedTensor
  (
   "zero",
   dimensionSet(1,-1,-2,0,0,0,0),
   Ku
  ),
 zeroGradientFvPatchScalarField::typeName
 );

 K.correctBoundaryConditions();

/*
// Create the full forth-order elasticity tensor, if the symmTensor4thOrder class has been defined
 volSymmTensor4thOrderField C
      (
       IOobject
       (
        "C",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
        ),
       mesh,
       dimensionedSymmTensor4thOrder("zero", dimForce/dimArea,
                            symmTensor4thOrder(A11, A12, A31,
                                                    A22, A23,
                                                         A33,
                                                         A44,
                                                         A55,
                                                         A66)
                            ),
        zeroGradientFvPatchSymmTensor4thOrderField::typeName
      );

 C.correctBoundaryConditions();
*/
```

Listing 4.2: `readMechanicalProperties.H`

- `readStressedFoamControls.H`
  The definition of convergence control parameters is kept the same as before shown in Listing 4.3.

```
    const dictionary& stressControl =
        mesh.solutionDict().subDict("anisoFoam");

    int nCorr(readInt(stressControl.lookup("nCorrectors")));
    scalar convergenceTolerance(readScalar(stressControl.lookup("U
       ")));
```

Listing 4.3: `readAnisoFoamControls.H`

- `anisoFoam.C`
  Having all the variable fields and model parameters set ready, we can now proceed to the main `anisoFoam.C` file. The differential equation with the segregated strategy present in Eq. (4.5) is implemented here. And in particular, a fixed under-relaxation procedure is applied for stabilizing the convergence. Details regarding to the relaxation method will be described in Chapter 8.

  A new `calculateSigmaEpsilonOrtho.H` file is included in the main file, so as to specially compute the explicit $(\mathbf{C} : \boldsymbol{\varepsilon})$ term and leave the main codes neat.

```
int main(int argc, char *argv[])
{

#   include "setRootCase.H"

#   include "createTime.H"
#   include "createMesh.H"
#   include "readMechanicalProperties.H"
#   include "createFields.H"

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      * * * * * //

    Info<< "\nCalculating displacement field\n" << endl;

    for (runTime++; !runTime.end(); runTime++)
    {
        Info<< "Iteration: " << runTime.timeName() << nl << endl;

```

```
#       include "readAnisoFoamControls.H"

        int iCorr = 0;
        scalar initialResidual = 0;

#       include "calculateSigmaEpsilonOrtho.H"

//Altertive file, if given the 4th—order tensor and the double dot operator && have been defined
//#       include "calculateSigmaEpsilonOrthoNew.H"

        do
        {
        // Store the previous iterative value for relaxation later
            U.storePrevIter();

        // Construct the equation matrix
            fvVectorMatrix UEqn
            (
                fvm::d2dt2(rho, U)
             ==
                fvm::laplacian(K, U, "laplacian(DU,U)")

              + fvc::div(sigma) // sigma=C:epsilon

              — fvc::div(K & gradU)
            );

            initialResidual = UEqn.solve().initialResidual();

         // Under—relax the displacement field, helpful for convergence. The relaxation factor will read from user input
            U.relax();
            gradU = fvc::grad(U);

         // Explicitly calculate the sigma field using the orthotropic Hook’s Law
#       include "calculateSigmaEpsilonOrtho.H"
//Altertive file, if given the 4th—order tensor and the double dot operator && have been defined
//#       include "calculateSigmaEpsilonOrthoNew.H"

        } while (initialResidual > convergenceTolerance && ++iCorr < nCorr);

        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << "
```

```
                  s"
             << nl << endl;
    }
```

Listing 4.4: `anisoFoam.C`

- `calculateSigmaEpsilonOrtho.H`
  Now we shall deal with the difficulty, namely how to calculate the double dot product between the forth-order elasticity tensor and the second-order strain tensor without such higher-order tensor class and associated function built in OpenFOAM? If looking at Eq. (4.2) which writes out the tensors in vector format, we can find the solution: though forth-order tensor class and its double dot operator are currently missing in OpenFOAM, the usual tensorial approach in OpenFOAM is not possible, we could still easily do the calculation in a term-by-term manner. Rewriting Eq. (4.2) as:

$$\begin{aligned}
\sigma_{xx} &= A_{11}\cdot\varepsilon_{xx} + A_{12}\cdot\varepsilon_{yy} + A_{31}\cdot\varepsilon_{zz}\\
\sigma_{yy} &= A_{12}\cdot\varepsilon_{xx} + A_{22}\cdot\varepsilon_{yy} + A_{23}\cdot\varepsilon_{zz}\\
\sigma_{zz} &= A_{31}\cdot\varepsilon_{xx} + A_{23}\cdot\varepsilon_{yy} + A_{33}\cdot\varepsilon_{zz}\\
\sigma_{xy} &= A_{44}\cdot\varepsilon_{xy}\\
\sigma_{yz} &= A_{55}\cdot\varepsilon_{yz}\\
\sigma_{zx} &= A_{66}\cdot\varepsilon_{zx}
\end{aligned} \tag{4.8}$$

  The following code explains the above explicit calculation procedure.

```
//- calculate epsilon and sigma for an orthotropic material

//- epsilon
epsilon = symm(gradU);

//- sigma = C:epsilon
forAll(sigma.internalField(), celli) // calulate sigma at each inner cell
{
  const scalar& e11 = epsilon.internalField()[celli].component(symmTensor::XX);
  const scalar& e22 = epsilon.internalField()[celli].component(symmTensor::YY);
  const scalar& e33 = epsilon.internalField()[celli].component(symmTensor::ZZ);
  const scalar& e12 = epsilon.internalField()[celli].component(symmTensor::XY);
  const scalar& e23 = epsilon.internalField()[celli].component(symmTensor::YZ);
  const scalar& e31 = epsilon.internalField()[celli].component(symmTensor::XZ);

  if(model2d) // 2D condition
    {
      sigma.internalField()[celli].component(symmTensor::XX) = A11*e11 + A12*e22;
      sigma.internalField()[celli].component(symmTensor::YY) = A21*e11 + A22*e22;
      sigma.internalField()[celli].component(symmTensor::XY) = A44*e12;
    }
  else // 3D condition
    {
      sigma.internalField()[celli].component(symmTensor::XX) = A11*e11 + A12*e22 + A31*e33;
      sigma.internalField()[celli].component(symmTensor::YY) = A12*e11 + A22*e22 + A23*e33;
      sigma.internalField()[celli].component(symmTensor::ZZ) = A31*e11 + A23*e22 + A33*e33;
      sigma.internalField()[celli].component(symmTensor::XY) = A44*e12;
      sigma.internalField()[celli].component(symmTensor::YZ) = A55*e23;
      sigma.internalField()[celli].component(symmTensor::XZ) = A66*e31;
    }
}

forAll(mesh.boundary(), patchi) // calculate sigma at each boundary patch cell
{
  forAll(sigma.boundaryField()[patchi], facei)
    {
      const scalar& e11 = epsilon.boundaryField()[patchi][facei].component(symmTensor::XX);
      const scalar& e22 = epsilon.boundaryField()[patchi][facei].component(symmTensor::YY);
      const scalar& e33 = epsilon.boundaryField()[patchi][facei].component(symmTensor::ZZ);
      const scalar& e12 = epsilon.boundaryField()[patchi][facei].component(symmTensor::XY);
      const scalar& e23 = epsilon.boundaryField()[patchi][facei].component(symmTensor::YZ);
      const scalar& e31 = epsilon.boundaryField()[patchi][facei].component(symmTensor::XZ);

      if(model2d) // 2D condition
        {
          sigma.boundaryField()[patchi][facei].component(symmTensor::XX) = A11*e11 + A12*e22;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::YY) = A21*e11 + A22*e22;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::XY) = A44*e12;
        }
      else // 3D condition
        {
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::XX) = A11*e11 + A12*e22 + A31*e33;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::YY) = A12*e11 + A22*e22 + A23*e33;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::ZZ) = A31*e11 + A23*e22 + A33*e33;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::XY) = A44*e12;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::YZ) = A55*e23;
          sigma.boundaryField()[patchi][facei].component(
              symmTensor::XZ) = A66*e31;
        }
    }
}
```

Listing 4.5: `calculateSigmaEpsilonOrtho.H`

You might wonder, why not we create a new forth-order tensor class and then implement the aforementioned calculation procedure as a double dot operator member function, in that way we keep the nice tensorial approach spirit of OpenFOAM? The answer is positive. It is challenging to do this, since thorough knowledge of c++ programming itself and the 'architecture' of OpenFOAM are necessary. However, it is definitely possible. I have learned my way from Dr. Philip Cardiff during my external stay in the OpenFOAM group at University College Dublin. In the following I will only introduce the main idea of building the new `symmTensor4thOrder` class and the double dot operator `&&`, leaving the details of modifying hundreds of files in OpenFOAM away. readers are invited to contact me or Philip to get the source files directly.

All the basic tensor classes are defined in the `src/OpenFOAM/primitives`. We can copy the codes from other similar tensor directory, e.g. the `symmTensor` and then create the new `SymmTensor4thOrder` class with the following basic constructor:

```
// Construct given nine Cmpts
template <class Cmpt>
inline SymmTensor4thOrder<Cmpt>::SymmTensor4thOrder
(
            const Cmpt txxxx, const Cmpt txxyy, const Cmpt txxzz,
                              const Cmpt tyyyy, const Cmpt tyyzz,
                                                const Cmpt tzzzz,
                                                const Cmpt txyxy,
                                                const Cmpt tyzyz,
                                                const Cmpt tzxzx
)
{
    this->v_[XXXX] = txxxx; this->v_[XXYY] = txxyy; this->v_[XXZZ] = txxzz;
                            this->v_[YYYY] = tyyyy; this->v_[YYZZ] = tyyzz;
                                                    this->v_[ZZZZ] = tzzzz;
                                                    this->v_[XYXY] = txyxy;
                                                    this->v_[YZYZ] = tyzyz;
                                                    this->v_[ZXZX] = tzxzx;
}
```

The `symmTensor4thOrder` class has 9 data members, corresponding to the 9 independent coefficients of the elasticity tensor. An over-loaded operator function, `&&`, is representing the double dot product. The `&&` operator takes a symmetric forth-order tensor as the first argument and a symmetric second-order tensor as the second argument, and returns a symmetric second-order tensor. The code snippet of the operator function is shown below:

```
//- Double-dot-product between a fourth order symmetric tensor and
    a symmetric tensor
template <class Cmpt>
inline SymmTensor<Cmpt>
operator&&(const SymmTensor4thOrder<Cmpt>& t4th, const SymmTensor<
   Cmpt>& st)
 {
  //- s_ij = C_ijkl E_kl
    return SymmTensor<Cmpt>
    (
     t4th.xxxx()*st.xx() + t4th.xxyy()*st.yy() + t4th.xxzz()*st.zz
         (),
     t4th.xyxy()*st.xy(),
     t4th.zxzx()*st.xz(),

     t4th.xxyy()*st.xx() + t4th.yyyy()*st.yy() + t4th.yyzz()*st.zz
         (),
      t4th.yzyz()*st.yz(),

     t4th.xxzz()*st.xx() + t4th.yyzz()*st.yy() + t4th.zzzz()*st.zz
         ()
     );
}
```

By defining the forth-order symmetric tensor class `symmTensor4thOrder` and its corresponding geometric tensor fields - the `symmTensor4thOrderField` class and the `volSymmTensor4thOrderField` class, we can now simplify our implementation codes.

First, the symmetric forth-order elasticity tensor **C** can be generated at the end of the `readMechanicalProperties.H` file as follows,

```
volSymmTensor4thOrderField C
     (
      IOobject
      (
       "C",
       runTime.timeName(),
       mesh,
       IOobject::NO_READ,
       IOobject::NO_WRITE
       ),
      mesh,
      dimensionedSymmTensor4thOrder("zero", dimForce/dimArea,
                           symmTensor4thOrder(A11, A12, A31,
                                                   A22, A23,
                                                        A33,
                                                        A44,
                                                        A55,
                                                        A66)
                           ),
      zeroGradientFvPatchSymmTensor4thOrderField::typeName
    );

C.correctBoundaryConditions();
```

Then we replace the original `calculateSigmaEpsilonOrtho.H` file with a new simply coded `calculateSigmaEpsilonOrthoNew.H` file using the `&&` operator.

```
//- calculate epsilon and sigma for an orthotropic material

//- epsilon
epsilon = symm(gradU);

//- sigma
sigma = (C && epsilon);
sigma.correctBoundaryConditions();
```

Listing 4.6: `calculateSigmaEpsilonOrthoNew.H`

In general, the above two approaches (Listing 4.5 and Listing 4.6) are equal. For those readers who do not wish to change large amount of files inside the OpenFOAM library, the first one is recommended.

- `tractionOrtho/tractionOrthoFvPatchVectorField.C`
  The last task left is the implementation of the traction boundary condition. For an orthotropic solid, we have the force balance equation:

$$\mathbf{t} = \mathbf{n} \cdot \boldsymbol{\sigma} \tag{4.9}$$

where
t, traction force;

$\mathbf{n}$, surface normal vector;
$\boldsymbol{\sigma}$, total stress.
As in OpenFOAM, we always apply the implicit and explicit split treatment to deal with the cross-component coupling inside the governing equation, the stress can be decomposed into two parts as follows:

$$\boldsymbol{\sigma} = \underbrace{\mathbf{K} \cdot \nabla\mathbf{u}}_{implicit} + \underbrace{\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u}}_{explicit} \tag{4.10}$$

Combine Eq. (4.9) and Eq. (4.10),

$$\mathbf{t} = \mathbf{n} \cdot (\mathbf{K} \cdot \nabla\mathbf{u}) + \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u}) \tag{4.11}$$

Rearrange in terms to obtain the implicit gradient:

$$\mathbf{n} \cdot (\mathbf{K} \cdot \nabla\mathbf{u}) = \mathbf{t} - \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u}) \tag{4.12}$$

Since what we would like to get is the normal gradient boundary, namely, $(\mathbf{n} \cdot \nabla\mathbf{u})$, some mathematical manipulations of Eq. (4.12) are necessary. Multiply by $\mathbf{n}$ on both sides,

$$\mathbf{n}[\mathbf{n} \cdot (\mathbf{K} \cdot \nabla\mathbf{u})] = \mathbf{n}[\mathbf{t} - \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u})] \tag{4.13}$$

Now on the left-hand-side we have a tensor which is similar to KDotGradU except it only has information in the n direction, but all the tensors on the right hand side only have information in the n direction, so it is valid to rewrite the equation:

$$\begin{aligned}
&(\mathbf{K} \cdot \nabla\mathbf{u}) = \mathbf{n}[\mathbf{t} - \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u})] \\
\rightarrow &\nabla\mathbf{u} = \mathbf{K}^{-1} \cdot \{\mathbf{n}[\mathbf{t} - \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u})]\} \\
\rightarrow &\mathbf{n} \cdot \nabla\mathbf{u} = \mathbf{n} \cdot \Big[\mathbf{K}^{-1} \cdot \{\mathbf{n}[\mathbf{t} - \mathbf{n} \cdot (\mathbf{C} : \boldsymbol{\varepsilon} - \mathbf{K} \cdot \nabla\mathbf{u})]\}\Big]
\end{aligned} \tag{4.14}$$

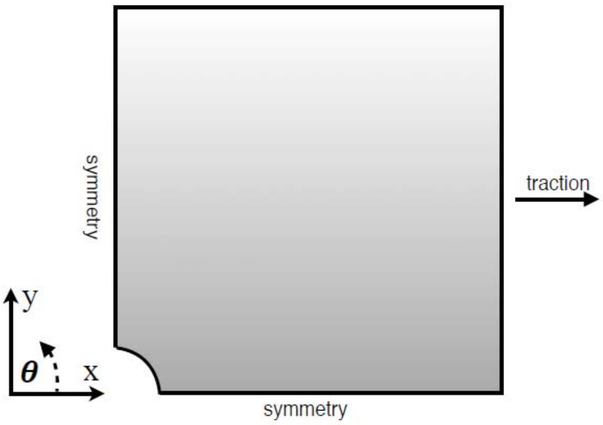
The implementation codes in Listing 4.7 below represent the above calculation procedure.

```
// Update the coefficients associated with the patch field
void tractionOrthoFvPatchVectorField::updateCoeffs()
{
    if (updated())
    {
        return;
    }

    const fvPatchField<tensor>& K =
        patch().lookupPatchField<volTensorField, tensor>("K");

    const tensorField Kinv = inv(K);

    vectorField n = patch().nf();

    const fvPatchField<tensor>& gradU =
        patch().lookupPatchField<volTensorField, tensor>("grad(U)");
    const fvPatchField<symmTensor>& sigma =
        patch().lookupPatchField<volSymmTensorField, symmTensor>("sigma");

    vectorField Traction(n.size(), vector::zero);
    tensorField sigmaExp(n.size(), tensor::zero);

    Traction = (traction_ - n*pressure_);

    sigmaExp = (n*(n & sigma)) - (K & gradU);

    gradient() =
      n &
      (
       Kinv & ( n*(Traction) - sigmaExp )
       );

    //- The following code does not work
    //scalarField sigmaExp = sigma - (K & gradU);
    //scalarField Kn = n & ( n & K );
    //gradient() = (1/Kn)*(Traction - sigmaExp);

    fixedGradientFvPatchVectorField::updateCoeffs();
}
```

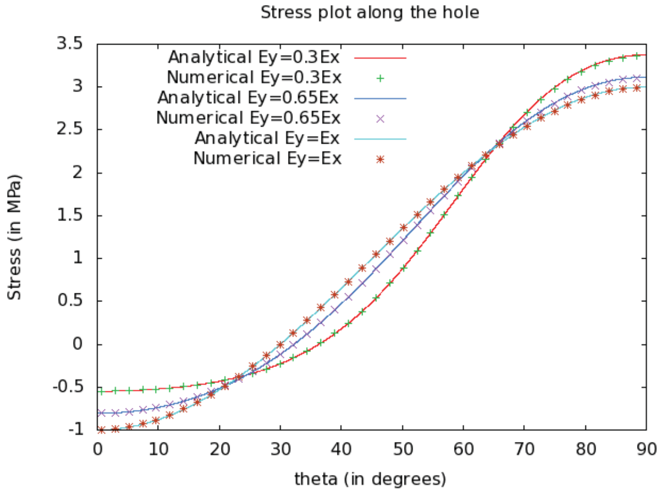
Listing 4.7: `tractionOrtho/tractionOrthoFvPatchVectorField.C`

4.4 Application

The implemented `anisoFoam` solver is applied for a test case, in which a circular hole in an orthotropic plate is subjected traction. Plane stress condition is assumed. The material properties employed are $E_x = 200GPa$, $E_y = 60GPa, 130GPa, 200GPa$, $\nu_{xy} = 0.3$, and $G_{xy} = 76.9GPa$. The traction applied on the right hand side is 1MPa. Good agreement is achieved between analytical solutions [15] and the simulation results as shown in Figure 4.2.



(a) Geometry and boundary condition



(b) Hoop stress $\sigma_{\theta\theta}$ around the hole

Figure 4.2: The Orthotropic plateHole test case.

5 The large strain solver

Another important non-linear stress analysis is involving large displacements and large strains. The so-called large strain theory, or large deformation theory, deals with deformations in which both rotations and strains are arbitrarily large, i.e. invalidates the assumptions inherent in infinitesimal strain theory. In this case, the un-deformed and deformed configurations of the continuum are significantly different and a clear distinction has to be made between them.

It is interesting to develop a proper FVM large strain solver in OpenFOAM to deal with such kind of issue in solid continuum mechanics. The following sections will describe the general theoretical background and corresponding numerical and implementation techniques.

5.1 Total lagrangian large strain formulation

Basically, two different approaches have been pursued in the incremental non-linear stress analysis involving large strains: the updated lagrangian formulation which is referred to an updated configuration in each load step, and the total lagrangian formulation which is referred to the initial configuration. Here, we will consider the second approach - the total lagrangian large strain formulation.

Let's first look at the momentum balance in integral form (static state):

$$\oint_a \mathbf{n} \cdot \boldsymbol{\sigma} |d\mathbf{a}| = \mathbf{0} \quad \text{current configuration} \tag{5.1}$$

$$\Rightarrow \oint_{a_o} \mathbf{n}_o \cdot (\mathbf{S} \cdot \mathbf{F}) |d\mathbf{a}_o| = \mathbf{0} \quad \text{reference(initial) configuration} \tag{5.2}$$

where
a and a_o, the surface area in current and reference configuration, respectively;
$\mathbf{S}$, the second Piola-Kirchhoff stress. $\mathbf{S} = J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T}$, with $J = det(\mathbf{F})$;
$\mathbf{F}$, the deformation gradient.

The incremental momentum balance is then obtained from the 'difference' form of Eq. (5.2):

$$\oint_{a_o} \mathbf{n}_o \cdot (\delta\mathbf{S} \cdot \mathbf{F} + \mathbf{S} \cdot \delta\mathbf{F} + \delta\mathbf{S} \cdot \delta\mathbf{F}) |d\mathbf{a}_o| = \mathbf{0} \tag{5.3}$$

with

$$\mathbf{F} = \mathbf{I} + \nabla\mathbf{u}, \quad \delta\mathbf{F} = \nabla(\delta\mathbf{u}) \tag{5.4}$$

Substituting Eq. (5.4) into Eq. (5.3) and rearrange the terms, we obtain:

$$\oint_{a_o} \mathbf{n}_o \cdot [\delta\mathbf{S} \cdot (\mathbf{I} + \nabla\mathbf{u}) + (\mathbf{S} + \delta\mathbf{S}) \cdot \nabla(\delta\mathbf{u})]|d\mathbf{a}_o| = \mathbf{0} \tag{5.5}$$

Now recall the linear elastic constitutive law and (green) strain-displacement relation as follows:

$$\delta\mathbf{S} = 2\mu\delta\mathbf{E} + \lambda\text{tr}(\delta\mathbf{E})\mathbf{I} \tag{5.6}$$

$$\delta\mathbf{E} = \frac{1}{2}[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u} \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}}] \tag{5.7}$$

where
$\delta\mathbf{E}$, the increment of green strain.

Using Eq. (5.6) and Eq. (5.7), we can further rewrite Eq. (5.5) as:

$$\begin{aligned}&\oint_{a_o} \mathbf{n}_o \cdot \{\mu\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u} \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\text{tr}(\delta\mathbf{E})\}|d\mathbf{a}_o| \\ &+ \oint_{a_o} \mathbf{n}_o \cdot [\delta\mathbf{S} \cdot \nabla\mathbf{u} + (\mathbf{S} + \delta\mathbf{S}) \cdot \nabla(\delta\mathbf{u})]|d\mathbf{a}_o| = \mathbf{0}\end{aligned} \tag{5.8}$$

As you might notice that due to the higher order nonlinear terms in the green strain, the governing equation turns to be quite complicated. We shall apply as usual the segregated implicit and explicit strategy to deal with the cross-component coupling, higher-order strain terms, and the stress terms present in the equation.

5.2 Numerical strategy

Let's write the Eq. (5.8) back into differential format which OpenFOAM usually solves:

$$\begin{aligned}&\nabla \cdot \left\{\mu\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u} \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\text{tr}(\delta\mathbf{E})\right\} \\ &+ \nabla \cdot [\delta\mathbf{S} \cdot \nabla\mathbf{u} + (\mathbf{S} + \delta\mathbf{S}) \cdot \nabla(\delta\mathbf{u})] = \mathbf{0}\end{aligned} \tag{5.9}$$

We can apply the following segregated numerical strategy to handle the high non-linearities in Eq. (5.9).

$$\begin{aligned}&\underbrace{(2\mu + \lambda)\nabla^2(\delta\mathbf{u})}_{implicit} \\ &+ \underbrace{\nabla \cdot \left\{-(\mu + \lambda)\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u} \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u}) \cdot \nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\text{tr}(\delta\mathbf{E})\right\}}_{explicit} \\ &+ \underbrace{\nabla \cdot [\delta\mathbf{S} \cdot \nabla\mathbf{u} + (\mathbf{S} + \delta\mathbf{S}) \cdot \nabla(\delta\mathbf{u})]}_{explicit} = \mathbf{0}\end{aligned} \tag{5.10}$$

where the implicit term corresponds to the over-relaxed laplacian. The first explicit term is related to the cross-component coupling and the nonlinearity inside the strain definition. The second explicit nonlinear term is due to the stress change.

5.3 totalLagrangianFoam implementation

This section aims to describe the implementation procedure of a new `totalLagrangianFoam` solver which basically solves Eq. (5.10).

Similar to how we built the basic set up of our previous solvers, we firstly copy and rename the original `stressedFoam`, change the corresponding files' name, modify the keywords inside the files and do the initial compilation (see command lines in Page. 6 as an example). The `totalLagrangianFoam` solver has the following directory structure:

```
totalLagrangianFoam
├── Make
│   ├── files
│   └── options
├── totalLagrangianTraction
│   ├── totalLagrangianTractionFvPatchVectorField.C
│   └── totalLagrangianTractionFvPatchVectorField.H
├── createFields.H
├── readMechanicalProperties.H
├── readTotalLagrangianFoamControls.H
├── totalLagrangianFoam.C
└── writeFields.C
```

Figure 5.1: The directory structure of `totalLagrangianFoam`.

Now I shall explain what to do for each file.

- `createFields.H`
  Except from the displacement and increment of displacement fields, we will initialize the incremental and total second Piola-Kirchhoff stress and the total and incremental green strain tensor. Optionally, we might also define the true engineering stress and non-uniform density field of interests. The code excerpt is shown in Listing 5.1. Worth to mention that, calculation of true stress from the second Piola-Kirchhoff stress is based on the formulation below:

$$\boldsymbol{\sigma} = J^{-1}\mathbf{F}^T\mathbf{S}\mathbf{F} \tag{5.11}$$

```
   volTensorField gradU = fvc::grad(U);

   volSymmTensorField epsilonG // the green strain
    (
        IOobject
        (
            "epsilonG",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        symm(gradU)+0.5*symm(gradU & gradU.T())
    );

   volTensorField graddU = fvc::grad(dU);

   volSymmTensorField dEpsilonG // increment of green strain
    (
        IOobject
        (
            "dEpsilonG",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),

        symm(graddU)+0.5*symm(graddU & gradU.T())
                    +0.5*symm(gradU & graddU.T())
                    +0.5*symm(graddU & graddU.T())
    );

   volSymmTensorField sigmaPK2 //the second Piola-Kirchhoff stress
    (
        IOobject
        (
            "sigmaPK2",
            runTime.timeName(),
            mesh,
            IOobject::READ_IF_PRESENT, // read if initial stress condition
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("zero", dimForce/dimArea, symmTensor::zero)
    );

    volSymmTensorField dSigmaPK2 // increment of the second Piola-Kirchhoff stress
    (
        IOobject
        (
            "dSigmaPK2",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedSymmTensor("zero", dimForce/dimArea, symmTensor::zero)
    );

    volTensorField F = I+gradU+graddU; // deformation gradient F+=dF

    volSymmTensorField sigmaTrue // true stress (Cauchy stress)
     (
        IOobject
        (
            "sigmaTrue",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        1.0/det(F)*symm(F.T() & sigmaPK2 & F)
    );
```

Listing 5.1: `createFields.H`

- `readMechanicalProperties.H` and `readTotalLagrangianFoamControls.H`
  As we use the same isotropic elastic constants - μ and λ - and the same convergence control parameters - the maximum iteration number and the residual tolerance, the two `readMechanicalProperties.H` and `readTotalLagrangianFoamControls.H` file remain unchanged.

- `totalLagrangianFoam.C`
  The main file `totalLagrangianFoam.C` contains the solution procedure of Eq. (5.10).

```
    for (runTime++; !runTime.end(); runTime++) // Time loop
    {
        Info<< "Iteration: " << runTime.timeName() << nl << endl;

#       include "readTotalLagrangianFoamControls.H" // Read convergence control

      int iCorr = 0;
      scalar initialResidual = 0;
      lduMatrix::solverPerformance solverPerf;

      lduMatrix::debug=0;

        do // momentum loop
        {
            dU.storePrevIter(); // store previous iteration for
```

```
               later relaxation

            fvVectorMatrix dUEqn
            (
                fvm::laplacian(2*mu + lambda, dU, "laplacian(DU,dU)")

             == - fvc::div
                 (
                    - ((mu + lambda)*graddU)
                    + (mu*graddU.T())
                    + (mu*(graddU & gradU.T()))
                    + (mu*(gradU & graddU.T()))
                    + (mu*(graddU & graddU.T()))
                    + (lambda * tr(dEpsilonG) * I ) // first explicit part

                    + (dSigmaPK2 & gradU)
                    + ((sigmaPK2+dSigmaPK2) & graddU), // second explicit part
                    "div(sigma)"
                 )
            );

          solverPerf = dUEqn.solve();// solve matrix

          initialResidual = solverPerf.initialResidual();

          dU.relax(); // fixed under-relaxation
          graddU = fvc::grad(dU);

          dEpsilonG = symm(graddU)+0.5*symm(graddU & gradU.T())
                      +0.5*symm(gradU & graddU.T())
                      +0.5*symm(graddU & graddU.T());// compute the increment of green strain
          dSigmaPK2 = 2.0*mu*dEpsilonG + lambda*tr(dEpsilonG)*I;// compute the increment of 2nd Piola-Kichhoff stress

        } while (initialResidual > convergenceTolerance && ++iCorr < nCorr);

        // Upon convergent solution, Update all the dependent variables of interests
        U += dU; // total displacement
        gradU = fvc::grad(U);
        epsilonG += dEpsilonG; // green strain
        sigmaPK2 += dSigmaPK2; //  2nd Piola-Kichhoff
        volTensorField F = I + gradU + graddU; // deformation gradient
        volScalarField J = det(F);// Jacobian
        sigmaTrue = (1/J) * symm(F.T() & sigmaPK2 & F);// true engineering stress

#       include "writeFields.H" // calculate stress invariants

        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
```

Listing 5.2: `totalLagrangianFoam.C`

- **`writeFields.H`**
  The included **`writeFields.H`** simply calculates a stress invariant - von Mises stress (also called equivalent stress) - for both 2nd Piola-Kirchhoff stress and true engineering stress.

```
    if (runTime.outputTime())
    {
        volScalarField sigmaPK2Eq // equivalent 2nd Piola−Kirchhoff
        (
            IOobject
            (
                "sigmaPK2Eq",
                runTime.timeName(),
                mesh,
                IOobject::NO_READ,
                IOobject::AUTO_WRITE
            ),
            sqrt((3.0/2.0)*magSqr(dev(sigmaPK2)))
        );

        Info<< "Max sigmaPK2Eq = " << max(sigmaPK2Eq).value()
            << endl;

        volScalarField sigmaTrueEq //equivalent true stress
      (
       IOobject
       (
        "sigmaTrueEq",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
        ),
       sqrt((3.0/2.0)*magSqr(dev(sigmaTrue)))
       );

    Info<< "Max sigmaTrueEq = " << max(sigmaTrueEq).value()
        << endl;

    runTime.write();
```

Listing 5.3: `writeFields.H`

- `totalLagrangianTraction/totalLagrangianTractionFvPatchVectorField.C`
  We also need to adjust the traction boundary so that large strain effect is accounted. Let's first write down the force balance equation which is referred to the initial configuration:

$$\begin{aligned}\delta\mathbf{t} &= \mathbf{n}_o \cdot \delta\mathbf{S} \\ &= \mathbf{n}_o \cdot [2\mu\delta\mathbf{E} + \lambda \mathrm{tr}(\delta\mathbf{E})\mathbf{I}]\end{aligned} \tag{5.12}$$

where
$\delta\mathbf{t}_o$, increment of traction referred to the initial configuration.
$\mathbf{n}_o$, the boundary surface normal vector referred to the initial configuration.

Recall the green strain definition:

$$\delta\mathbf{E} = \frac{1}{2}[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u}\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}}] \tag{5.13}$$

As our aim is to get the implicit normal displacement gradient $\mathbf{n}_o \cdot \nabla(\delta\mathbf{u})$, we can apply the following split:

$$\begin{aligned}\delta\mathbf{t}_o =& \mathbf{n}_o \cdot \{\mu[\nabla(\delta\mathbf{u}) + \nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u}\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\mathrm{tr}(\delta\mathbf{E})\mathbf{I}\} \\ =& \mathbf{n}_o \cdot \{\underbrace{(2\mu+\lambda)\nabla(\delta\mathbf{u})}_{implicit} - \underbrace{(\mu+\lambda)\nabla(\delta\mathbf{u})}_{explicit} \\ &+ \underbrace{\mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u}\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}}]}_{explicit} \\ &+ \underbrace{\lambda\mathrm{tr}(\delta\mathbf{E})}_{explicit}\} \\ &\Downarrow\end{aligned}$$

$$\mathbf{n}_o\cdot\nabla(\delta\mathbf{u}) = \frac{\delta\mathbf{t}_o - \mathbf{n}_o\cdot\Big\{-(\mu+\lambda)\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u}\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\mathrm{tr}(\delta\mathbf{E})\Big\}}{2\mu+\lambda} \tag{5.14}$$

where the incremental traction force $\delta\mathbf{t}_o$ that refers to the reference configure can be computed from user-specified total Cauchy traction $\mathbf{t}_b$:

$$\delta\mathbf{t}_o = \underbrace{J|\mathbf{F}^{-1}\cdot\mathbf{n}_o|\mathbf{T}_b\cdot\mathbf{F}^{-1}}_{newTimeTrac} - \underbrace{\mathbf{n}_o\cdot\mathbf{S}}_{oldTimeTrac} \tag{5.15}$$

The corresponding implementation is present in Listing 5.4.

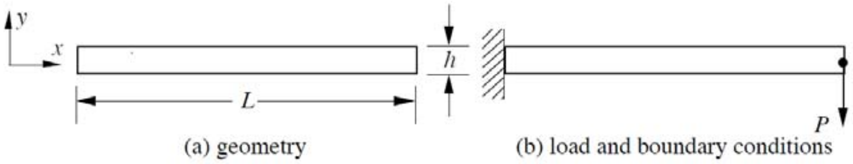
```
// Update the coefficients associated with the patch field
void totalLagrangianTractionFvPatchVectorField::updateCoeffs()
{
    if (updated())
    {
        return;
    }

    const dictionary& mechanicalProperties =
        db().lookupObject<IOdictionary>("mechanicalProperties");

    dimensionedScalar E(mechanicalProperties.lookup("E"));
    dimensionedScalar nu(mechanicalProperties.lookup("nu"));

    dimensionedScalar mu = E/(2.0*(1.0 + nu));
    dimensionedScalar lambda = nu*E/((1.0 + nu)*(1.0 - 2.0*nu));

    Switch planeStress(mechanicalProperties.lookup("planeStress"));

    if (planeStress)
    {
        lambda = nu*E/((1.0 + nu)*(1.0 - nu));
    }

    vectorField n = patch().nf();
    vectorField Traction(n.size(),vector::zero);

    const fvPatchField<tensor>& graddU =
        patch().lookupPatchField<volTensorField, tensor>("grad(dU)");
     const fvPatchField<tensor>& gradU =
        patch().lookupPatchField<volTensorField, tensor>("grad(U)");

    const fvPatchField<symmTensor>& sigmaPK2 =
        patch().lookupPatchField<volSymmTensorField, symmTensor>("sigmaPK2");
    const fvPatchField<symmTensor>& dEpsilonG =
        patch().lookupPatchField<volSymmTensorField, symmTensor>("dEpsilonG");

        tensorField F = I + gradU + graddU;// deformation gradient
        tensorField Finv = inv(F);// inverse of F
        scalarField J = det(F);
        vectorField nCurrent = Finv & n; // surface vector referring to deformed mesh
        nCurrent /= mag(nCurrent); // normal surface vector in deformed configuration

    vectorField tractionTrue = (traction_ - nCurrent*pressure_)*(this->db().time().value()); // constantly increased true traction from user input

    Traction = (mag(J * Finv & n) * tractionTrue & Finv) - (n & sigmaPK2); // traction in initial (referrence) configuration

     vectorField newGradient =
      Traction
      - (n & (mu.value()*graddU.T() - (mu + lambda).value()*graddU))
      - (n*lambda.value()*tr(dEpsilonG))
      - (n & (mu.value()*(graddU & gradU.T())))
      - (n & (mu.value()*(gradU & graddU.T())))
      - (n & (mu.value()*(graddU & graddU.T())));

    newGradient /= (2.0*mu + lambda).value();

    gradient() = newGradient;

    fixedGradientFvPatchVectorField::updateCoeffs();
}
```

Listing 5.4: `totalLagrangianTraction/totalLagrangianTractionFvPatchVectorField.C`

5.4 Application

The bending of a straight cantilever beam with transverse end point load is applied to test the implemented solver.

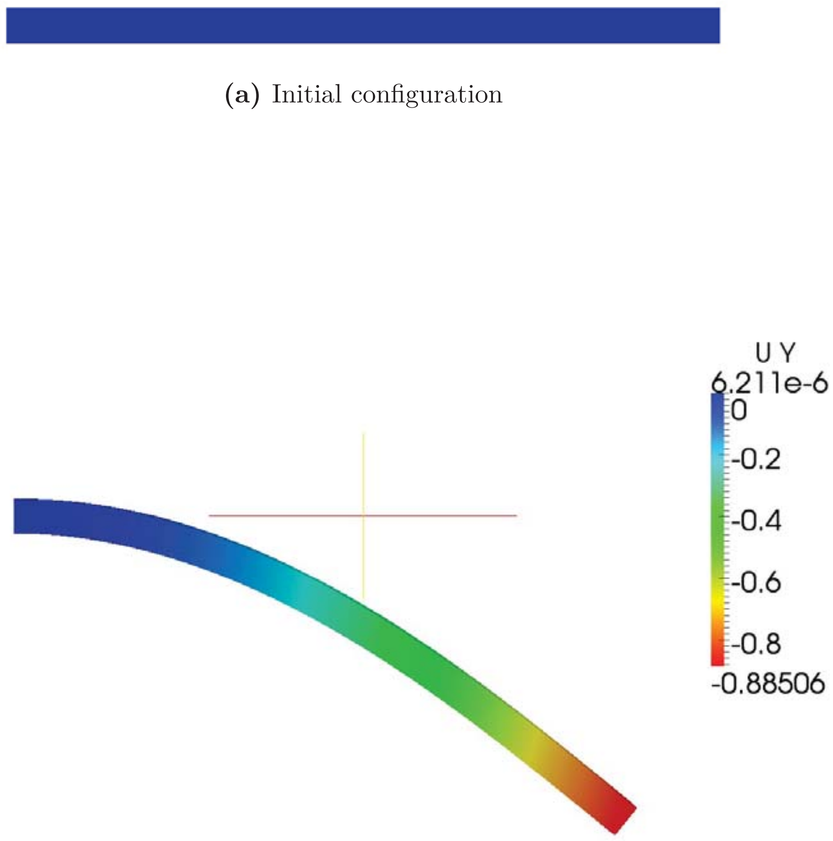
A square beam, of length $L = 2m$, height $h = 0.1m$ and cross sectional area $A = 0.01m^2$, is subjected to an end point load P (Figure 5.2 below). The material properties used in the simulation are $E = 200GPa$ and $\nu = 0.3$.



(a) geometry (b) load and boundary conditions

Figure 5.2: Beam bending case

The simulated initial and largely deformed beam configuration are both illustrated in Figure 5.3. It can be seen from Figure 5.4, good agreement is achieved bewteen the analytical vertical displacement V and horizontal displacement U at the end of beam [16] and the `totalLagrangianFoam` predictions.



(a) Initial configuration

(b) Deformed configuration

Figure 5.3: The cantilever beam test case.

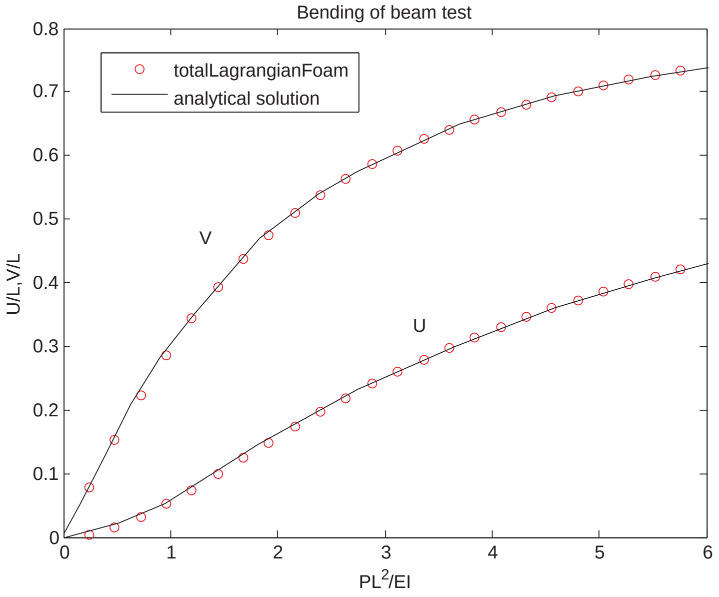


Figure 5.4: Comparison of displacement at the end of beam between analytical solution and `totalLagrangianFoam` results

6 The poro-elasticity solver

Due to my particular interest in soil mechanics, I will focus on developing a new poro-elasticity stress analysis solver tailored for porous soil in this chapter.

Soil is consisted of soil grain skeleton and the fluids (e.g. water and air) present in the pore structures. It is commonly acknowledged that the mechanical behavior of soils (and indeed other saturated geo-materials) is governed largely by the interaction of their solid skeleton with the pore fluid. Coupled analysis that accounts for both the solid skeleton response and the fluid flow dynamics is therefore very important for accurate soil modeling.

The new poro-elasticity solver is based on coupled formulation and uses the segregated solution procedure to handle the coupling. Details on the coupled theory, corresponding FVM numerical strategy, and implementations, will be described through the following sections.

6.1 Biot's consolidation model

The fundamental mathematical framework describing the coupled effects in porous soil was first established by Biot in 1941 [**?**], who - by assuming a linear elastic behavior of the soil skeleton and a Darcian fluid flow - formulated a coupled model with the soil skeleton displacements **u** and the pore fluid pressure p as the primary unknowns.

The so-called Biot's consolidation model contains two balance equations: the total momentum balance of the soil mixture and the conservation of the flow of water in the pores.

Firstly, the seepage of pore fluid flow which obeys Darcy's Law states,

$$\frac{k}{\gamma}\nabla^2 p - \frac{n}{K'}\frac{\partial p}{\partial t} = \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{\text{u-coupling}} \tag{6.1}$$

where
p, the pore water pressure;
k, the the permeability coefficient;
γ, the specific weight of water;
n, the porosity;
K', the effective bulk modulus of the pore fluid. It may be estimated assuming that air present in the pores remains close to atmospheric pressure:

$$\frac{1}{K'} = \frac{Sr}{K_w} + \frac{1 - Sr}{p_a} \tag{6.2}$$

with
Sr, the degree of saturation;
K_w, the pure water bulk modulus ($\approx 2 \times 10^9 Pa$);
p_a the atmospheric pressure ($\approx 10^5 Pa$).

Secondly, the total momentum balance equation for the soil mixture is expressed as:

$$\nabla \cdot \boldsymbol{\sigma} = \nabla \cdot (\boldsymbol{\sigma}' - p\mathbf{I}) = \mathbf{0} \tag{6.3}$$

where
$\boldsymbol{\sigma}$, the total stress tensor;
$\boldsymbol{\sigma}'$, the 'effective' soil stress tensor. For linear elastic soil skeleton, the stress-strain relation writes:
$\boldsymbol{\sigma}' = 2\mu\boldsymbol{\varepsilon} + \lambda \mathrm{tr}\boldsymbol{\varepsilon} = \mu\nabla\mathbf{u} + \mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u})$;
and $\mathbf{I}$, the identity tensor.

Substituting the linear elastic stress-strain relation into Eq. (6.3), it can be rewritten as,

$$\nabla \cdot [\mu\nabla\mathbf{u} + \underbrace{\mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u})}_{cross-componentcoupling}] - \underbrace{\nabla p}_{p-coupling} = \mathbf{0} \tag{6.4}$$

Hence, Biot's consolidation model is composed of Eq. (6.1) and Eq. (6.4), which shows strong two-way coupling since the displacement $\mathbf{u}$ present in the flow equation and the pore pressure p present in the momentum equation. Let me introduce the numerical strategy for dealing with this coupling below.

6.2 Numerical strategy

Similarly, the segregated discretization plus iterative solution procedure will be the key technique to solve the coupled consolidation model. The segregated implicit-explicit split strategy is performed as follows:

$$\underbrace{\frac{k}{\gamma}\nabla^2 p - \frac{n}{K'}\frac{\partial p}{\partial t}}_{implicit} = \underbrace{\frac{\partial}{\partial t}(\nabla \cdot \mathbf{u})}_{explicit} \tag{6.5}$$

$$\underbrace{\nabla \cdot [(2\mu + \lambda)\nabla\mathbf{u}]}_{implicit} = -\underbrace{\nabla \cdot \left\{\mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u}) - (\mu + \lambda)\nabla\mathbf{u}\right\}}_{explicit} + \underbrace{\nabla p}_{explicit} \tag{6.6}$$

The corresponding iterative solution procedure is illustrated in Figure 6.1. The pore pressure equation - Eq. (7.2) - will be solved first using the previous iteration value of displacement $\mathbf{u}$; thereafter the displacement equation - Eq. (6.6) - will be solved using the updated value of pore pressure. After solution of both the pore pressure and displacement vector is obtained, one iteration is completed and unless a converged solution is obtained, the two equations will be solved again using the latest available iterative values.

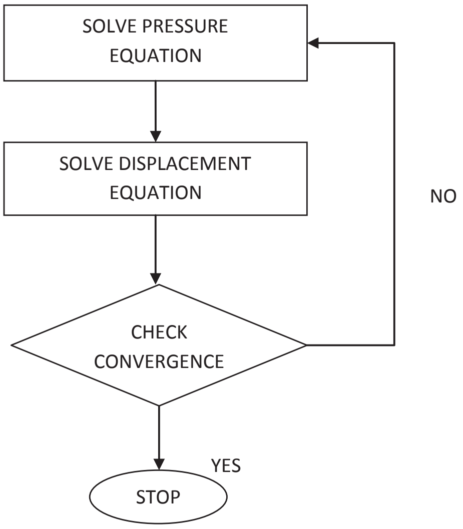


Figure 6.1: An outline of the iterative solution procedure employed in the poro-elasticity model

6.3 biotFoam implementation

Overall, the implementation work of `biotFoam` mainly contains the following two aspects:

- Solve a new scalar flow equation for pore fluid pressure
- Add a new explicit pressure coupling term in the momentum equation

I shall explain how we implement the above two features. Firstly, create the basic directory structure of t`biotFoam` using similar command lines as listed in Page. 6:

```
biotFoam
 |__Make
 |   |__files
 |   |__options
 |__poroTraction
 |   |__poroTractionFvPatchVectorField.C
 |   |__poroTractionFvPatchVectorField.H
 |__calculatePoroStress.H
 |__createFields.H
 |__readBiotFoamControls.H
 |__readSoilProperties.H
 |__biotFoam.C
```

Figure 6.2: The directory structure of `epStressedFoam`.

where
`createFields.H` initializes the pore fluid pressure, displacement, stress fields;
`readSoilProperties.H` reads the soil mechanical and hydraulic properties from user;
`readBiotFoamControls.H` reads convergence control parameters from user;
`biotFoam.C` solves the pore fluid flow equation and the momentum equation;
`calculatePoroStress.H` calculates the (effective and total) stress condition;
`poroTractionDisplacement` specifies the traction boundary condition for porous soil.

Let me then present the corresponding code representations of each file.

- `createFields.H`

```
Info<< "Reading displacement field U\n" << endl;
volVectorField U
(
    IOobject
    (
        "U",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

Info<< "Reading pore pressure field p\n" << endl;

volScalarField p
(
    IOobject
    (
        "p",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

Info<< "Calculating effective stress field sigma\n" << endl;
volSymmTensorField sigmaEff
(
    IOobject
    (
        "sigmaEff",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT, // read if initial stress condition specified
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedSymmTensor("sigmaEff",dimPressure,symmTensor::zero)
);

Info<< "Calculating total stress field sigma\n" << endl;
volSymmTensorField sigmaTot
(
    IOobject
    (
        "sigmaTot",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    sigmaEff-I*p
);
```

Listing 6.1: `createFields.H`

- `readSoilProperties.H`

```
    Info<< "Reading soil properties\n" << endl;

    IOdictionary soilProperties
    (
        IOobject
        (
            "soilProperties",
            runTime.constant(),
            mesh,
            IOobject::MUST_READ,
            IOobject::NO_WRITE
        )
    );

    dimensionedScalar E(soilProperties.lookup("E"));
    dimensionedScalar nu(soilProperties.lookup("nu"));
    dimensionedScalar k(soilProperties.lookup("k"));
    dimensionedScalar K(soilProperties.lookup("K"));
    dimensionedScalar Sr(soilProperties.lookup("Sr"));
    dimensionedScalar p0(soilProperties.lookup("p0"));
    dimensionedScalar gamma(soilProperties.lookup("gamma"));
    dimensionedScalar n(soilProperties.lookup("n"));

    dimensionedScalar Kprime = 1.0/(1.0/K + (1.0-Sr)/p0);

    Info<< "Calculating Lame's coefficients\n" << endl;

    dimensionedScalar mu = E/(2.0*(1.0 + nu));
    dimensionedScalar lambda = nu*E/((1.0 + nu)*(1.0 - 2.0*nu));

    Switch planeStress(soilProperties.lookup("planeStress"));

    if (planeStress)
    {
        Info<< "Plane Stress\n" << endl;

        lambda = nu*E/((1.0 + nu)*(1.0 - nu));
    }
    else
    {
        Info<< "Plane Strain\n" << endl;
    }

    Info<< "mu = " << mu.value() << " Pa\n";
    Info<< "lambda = " << lambda.value() << " Pa\n";

    dimensionedScalar Dp
    (
        "Dp",
        dimensionSet(0, 2, -1 , 0, 0),
        (k/gamma*Kprime/n).value()
    );

    dimensionedScalar Dp2
    (
        "Dp2",
        dimensionSet(1, -1, -2 , 0, 0),
        (Kprime/n).value()
    );

    Info<< "Dp = " << Dp.value() << " m^2/s \n";
    Info<< "Dp2 = " << Dp2.value() << " kg/m/s^2\n";
```

Listing 6.2: `readSoilProperties.H`

- `readBiotFoamControls.H`

```
const dictionary& stressControl = mesh.solutionDict().subDict("biotFoam");

int nCorr(readInt(stressControl.lookup("nCorrectors")));
scalar convergenceTolerance(readScalar(stressControl.lookup("
    residualDesired")));
```

Listing 6.3: `readBiotFoamControls.H`

- `biotFoam.C`

```
int main(int argc, char *argv[])
{
    #include "setRootCase.H"

    #include "createTime.H"
    #include "createMesh.H"
    #include "readSoilProperties.H"
    #include "createFields.H"

    // * * * * * * * * * * * * * * * * * * * * * * * * * * //

    Info<< "\nCalculating displacement field\n" << endl;

    while (runTime.loop())
    {
        Info<< "Iteration: " << runTime.value() << nl << endl;

        #include "readBiotFoamControls.H"

        int iCorr = 0;
        scalar residual = 1.0e10;

        do
        {
                p.storePrevIter();// for relaxation purpose

                fvScalarMatrix pEqn // create pore pressure equation matrix
                (
                 fvm::ddt(p) == fvm::laplacian(Dp, p) - fvc::div(fvc::ddt(Dp2,U))
                );

                residual = pEqn.solve().initialResidual();// solve pEqn
                p.relax();// under-relax pressure

           U.storePrevIter();
           volTensorField gradU = fvc::grad(U);

                fvVectorMatrix UEqn // create momentum equation matrix
                (
               fvm::laplacian(2*mu + lambda, U, "laplacian(DU,U)")
            + fvc::div
                    (
                      mu*gradU.T() + lambda*(I*tr(gradU)) - (mu + lambda)*gradU,
                      "div(sigmaEXP)"
                                           )
            == fvc::grad(p)// explicit p-coupling
                );

                if (residual > UEqn.solve().initialResidual())
             {
               residual = UEqn.solve().initialResidual();
             } // resdiual = max(UEqn.solve().initialResidual(), pEqn.solve().initialResidual())

                U.relax(); // under-relax displacement

        } while (residual > convergenceTolerance && ++iCorr < nCorr);

        #include "calculateStress.H"

        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
    }

    Info<< "End\n" << endl;

    return 0;
}
```

Listing 6.4: `biotoam.C`

- `calculatePoroStress.H`

```
        sigmaEff = mu*twoSymm(fvc::grad(U)) + (lambda*I)*tr(fvc::grad(U)); // update effective stress field

        sigmaTot = sigmaEff - I*p; // update total stress field

  if (runTime.outputTime())
    {
        volScalarField sigmaEffEq
        (
            IOobject
            (
                "sigmaEffEq",
                runTime.timeName(),
                mesh,
                IOobject::NO_READ,
                IOobject::AUTO_WRITE
            ),
            sqrt((3.0/2.0)*magSqr(dev(sigmaEff)))
        );

        Info<< "Max sigmaEffEq = " << max(sigmaEffEq).value()
            << endl;

        volVectorField V // Pore fluid flow velocity vector
        (
            IOobject
            (
                "V",
                runTime.timeName(),
                mesh,
                IOobject::NO_READ,
                IOobject::AUTO_WRITE
            ),
            -(k/gamma)*fvc::grad(p)
        );

        runTime.write();
    }
```

Listing 6.5: `calculatePoroStress.H`

- `poroTraction/poroTractionFvPatchVectorField.C`
  The porous traction boundary condition is based on the total force balance equation:

$$\begin{aligned}\mathbf{t} =& \mathbf{n}\cdot\boldsymbol{\sigma}\\ =& \mathbf{n}\cdot(\boldsymbol{\sigma}' - p\mathbf{I})\\ =& \mathbf{n}\cdot[\mu\nabla\mathbf{u} + \mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u})] - p\mathbf{n}\\ =& \mathbf{n}\cdot[\underbrace{(2\mu+\lambda)\nabla\mathbf{u}}_{implicit} + \underbrace{\mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u}) - (\mu+\lambda)\nabla\mathbf{u}}_{explicit}] - \underbrace{p\mathbf{n}}_{explicit}\end{aligned} \tag{6.7}$$

  Rearrange the terms to get the implicit normal displacement gradient:

$$n\cdot\nabla\mathbf{u} = \frac{\mathbf{t} - \mathbf{n}\cdot[\mu\nabla\mathbf{u}^T + \lambda\mathbf{I}tr(\nabla\mathbf{u}) - (\mu+\lambda)\nabla\mathbf{u}] + p\mathbf{n}}{2\mu+\lambda} \tag{6.8}$$

  The code snippet is shown in Listing 6.6.

```
void tractionDisplacementFvPatchVectorField::updateCoeffs()
```

```
{
    if (updated())
    {
        return;
    }

    const dictionary& materialProperties =
        db().lookupObject<IOdictionary>("soilProperties");

    dimensionedScalar E(materialProperties.lookup("E"));
    dimensionedScalar nu(materialProperties.lookup("nu"));

    dimensionedScalar mu = E/(2.0*(1.0 + nu));
    dimensionedScalar lambda = nu*E/((1.0 + nu)*(1.0 - 2.0*nu));

    Switch planeStress(materialProperties.lookup("planeStress"));

    if (planeStress)
    {
        lambda = nu*E/((1.0 + nu)*(1.0 - nu));
    }

    vectorField n = patch().nf(); //Surface normal

    scalar twoMuLamda =(2.0*mu + lambda).value();

    const fvPatchField<tensor>& gradU =
        patch().lookupPatchField<volTensorField, tensor>("grad(U)"
           );

    const fvPatchField<scalar>& p =
        patch().lookupPatchField<volScalarField, scalar>("p");

    gradient() =
    (
        (traction_ + pressure_*n)
      - (n & (mu.value()*gradU.T() - (mu + lambda).value()*gradU))
      - n*tr(graddU)*lambda.value()
      + n*p
    )/twoMuLambda;

    fixedGradientFvPatchVectorField::updateCoeffs();
```

Listing 6.6: `poroTraction/poroTractionFvPatchVectorField.C`

6.4 Application

The classic one-dimensional consolidation test is performed to validate the solver. Figure 6.3 outlines the case definition: a saturated soil column with height h is subjected to a surface step loading (T) of $1kN/m^2$ applied over a time of $0.1s$. The boundary conditions and soil material properties have also been shown in the figure itself.

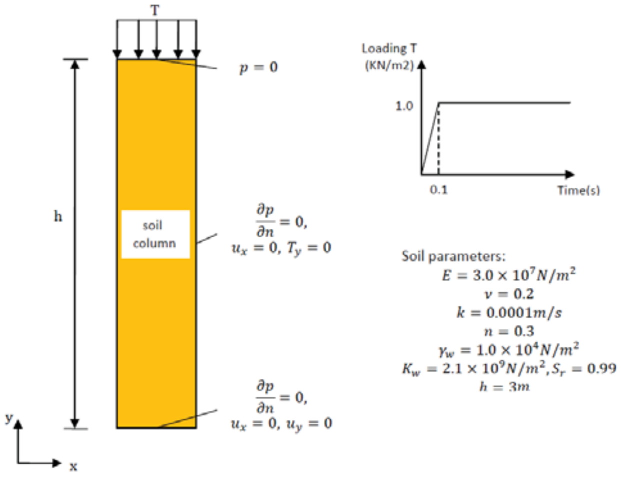

T
$p = 0$
h
soil column
$\frac{\partial p}{\partial n} = 0,$
$u_x = 0,\ T_y = 0$
$\frac{\partial p}{\partial n} = 0,$
$u_x = 0,\ u_y = 0$
y
x
Loading T (KN/m2)
1.0
0.1
Time(s)
Soil parameters:
$E = 3.0 \times 10^7 N/m^2$
$v = 0.2$
$k = 0.0001 m/s$
$n = 0.3$
$\gamma_w = 1.0 \times 10^4 N/m^2$
$K_w = 2.1 \times 10^9 N/m^2, S_r = 0.99$
$h = 3m$

Figure 6.3: A poro-elastic soil column subjected to a surface step loading

A comparison between the numerical prediction and the analytical solution [18] has been shown in Figure 6.4. The horizontal axis represents the pore pressure normalized by an initial pore pressure built up immediately after the loading, and the vertical axis stands for the depth of the soil column normalized by the total height. The five data series correspond to different dimensionless consolidation time, ($c_v t/h^2$), where c_v is the consolidation coefficient.

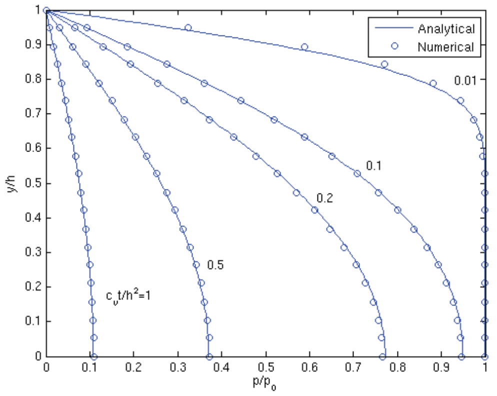
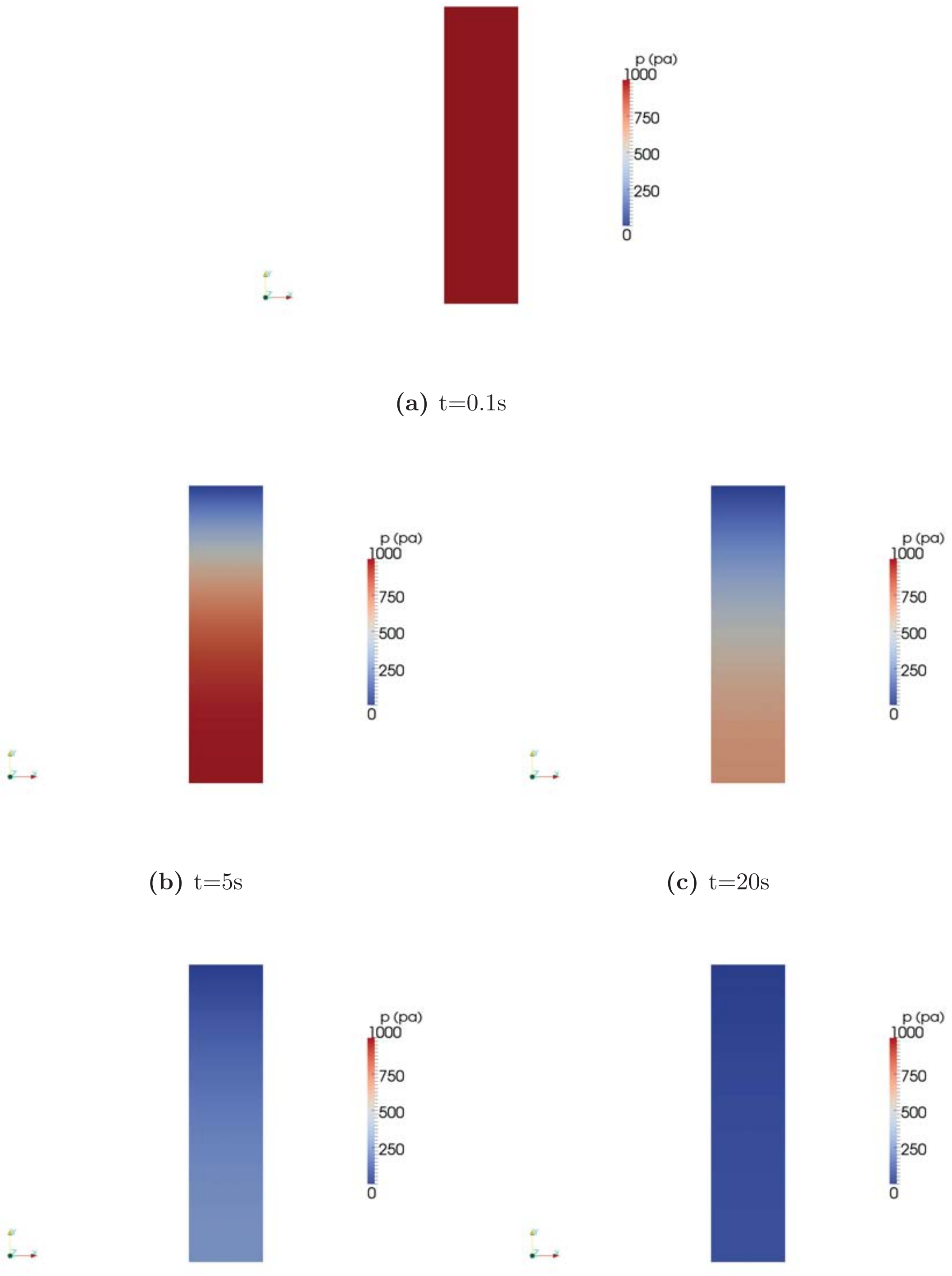


Figure 6.4: Numerical and analytical excess pore pressure dissipation isochrones (sampled along the center line of soil column)



(a) t=0.1s

(b) t=5s

(c) t=20s

(d) t=50s

(e) t=100s

Figure 6.5: Simulated excess pore pressure dissipation with time

7 More advanced developments

In the previous chapters, we have implemented the four new stress analysis solvers, each of whom covers a specific solid mechanic feature. In fact, we could easily move a step forward to develop more advanced solvers that handle multiple solid features, without additional effort. This is because in FVM, equations are solved in a segregated manner, where each component of a vector and/or any additional transport equations are solved sequentially and the coupling as well as the non-linearity are lagged. No matter how complex (e.g. multiple coupling and non-linear terms) the mathematical formulation is, always only one single fixed point iteration procedure iterating over all the explicit terms is needed.

In this chapter we will therefore attempt to implement three advanced stress analysis solvers, using the same segregated solution procedure. This time only key parts of the solver development will be described, for the implementation details readers are suggested to follow the guidance of previous solvers.

7.1 Poro-elasto-plasticity

The first advanced solver is the so-called poro-elasto-plasticity soil stress analysis solver, capable of accounting both the pore pressure coupling and the nonlinear plastic behavior of the soil skeleton.

The governing equation system for the porous elasto-plastic soil body states:

$$\begin{aligned}\underbrace{\nabla\cdot[(2\mu+\lambda)\nabla(\delta\mathbf{u})]}_{\text{implicit}} = &-\underbrace{\nabla\cdot\left\{\mu\nabla(\delta\mathbf{u})^{\mathrm{T}}+\lambda\mathbf{I}\mathrm{tr}[\nabla(\delta\mathbf{u})]-(\mu+\lambda)\nabla(\delta\mathbf{u})\right\}}_{\text{inter-component coupling, explicit}}\\ &+\underbrace{\nabla\cdot[2\mu(d\boldsymbol{\varepsilon}^p)+\lambda\mathbf{I}\mathrm{tr}\,(d\boldsymbol{\varepsilon}^p)]}_{\text{nonlinearity, explicit}}\\ &+\underbrace{\nabla(\delta p)}_{\text{pressure coupling, explicit}}\end{aligned} \tag{7.1}$$

$$\underbrace{\frac{k}{\gamma}\nabla^2 p-\frac{n}{K'}\frac{\partial p}{\partial t}}_{\text{implicit}} = \underbrace{\frac{\partial}{\partial t}(\nabla\cdot\mathbf{u})}_{\text{volumetric strain coupling, explicit}} \tag{7.2}$$

Iterative solution procedure shown in Figure 7.1 is applied to solve the above coupled nonlinear equation system.

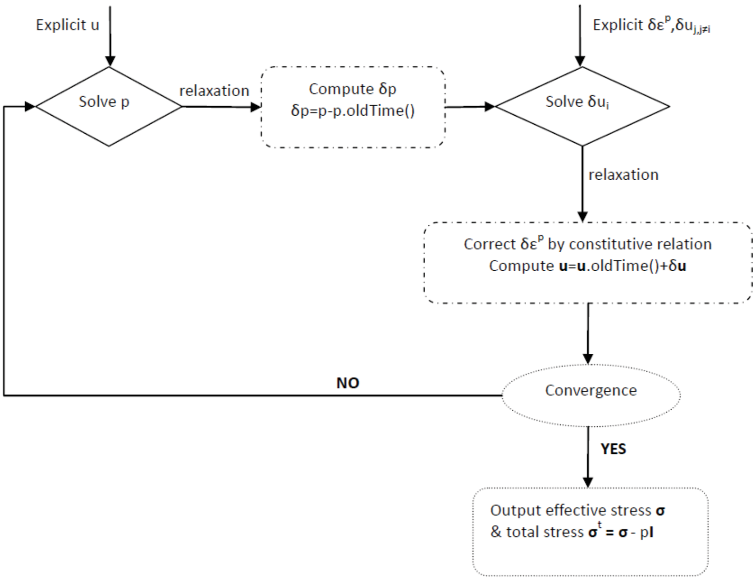

Explicit u
Solve p
relaxation
Compute δp
δp=p-p.oldTime()
Explicit δε^P,δu$_{j,j≠i}$
Solve δu$_i$
relaxation
Correct δε^P by constitutive relation
Compute **u**=**u**.oldTime()+δ**u**
NO
Convergence
YES
Output effective stress **σ**
& total stress **σ**^{t} = **σ** - p**I**

Figure 7.1: The iterative solution strategy of `biotEpFoam` in OpenFOAM

The corresponding implementation is present in Listing 7.1. Comments given describes different steps taken.

```
   int iCorr = 0;
   scalar pResidual = 0;
   scalar dUResidual = 0;

// Store the previous time step pressure and displacement field
   value, so as to calculate incremental values
   volScalarField p_old = p;
   volVectorField U_old = U;

   do
   {
       p.storePrevIter(); // store previous iteration for
          relaxation

       fvScalarMatrix pEqn // pressure equation matrix
       (
        fvm::ddt(p) == fvm::laplacian(Dp1, p, "laplacian(Dp1,p)")
                     - fvc::div(fvc::ddt(Dp2, U), "div(ddt(U))")
       );
       pResidual = pEqn.solve().initialResidual();
       p.relax(); // under-relax p

       volScalarField dp = p - p_old; // caculate incremental pore
           pressure which will be used in the incremental
          displacement equation
```

```
            dp.correctBoundaryConditions();

            dU.storePrevIter();
            volTensorField graddU = fvc::grad(dU);
            fvVectorMatrix dUEqn // displacement equation matrix
            (
                fvm::laplacian(2*mu + lambda, dU, "laplacian(DdU,dU)")
             ==
              - fvc::div
                (
                    mu*graddU.T() + lambda*(I*tr(graddU)) - (mu +
                       lambda)*graddU,
                    "div(sigmaExp)"
                )
              + fvc::div
                (
                    2.0*mu*dEpsP + lambda*I*tr(dEpsP),
                    "div(sigmaP)"
                )
              + fvc::grad(dp)
            );
            dUResidual = dUEqn.solve().initialResidual();
            dU.relax; // under-relax dU

            U = U_old + dU; // calculate the total displacement which
               will be used in the pore pressure equation
            U.correctBoundaryConditions();

        } while ((pResidual > pTolerance || dUResidual > dUTolerance)
           && ++iCorr < nCorr);

#       include "calculateStress.H"
```

Listing 7.1: `biotEpFoam` OpenFOAM solver code excerpt

7.2 Large deformation elasto-plasticity

Secondly, we could combine the geometrical nonlinearity and the material nonlinearity together, namely, to create a large deformation elasto-plasticity stress solver.

Under the small strain assumption, it is valid to do the following strain decomposition:

$$\delta \mathbf{E} = \delta \mathbf{E}^e + \delta \mathbf{E}^p \tag{7.3}$$

As a result, the elasto-plastic stress-strain relation can be written as:

$$\delta \mathbf{S} = 2\mu(\delta \mathbf{E} - \delta \mathbf{E}^p) + \lambda \mathrm{tr}(\delta \mathbf{E} - \delta \mathbf{E}^p)\mathbf{I} \tag{7.4}$$

The governing momentum equation contains several nonlinear terms and therefore is arranged in terms of different implicit and explicit discretizations, so as to allow the segregated solution procedure:

$$\underbrace{(2\mu+\lambda)\nabla^2(\delta\mathbf{u})}_{implicit} + \underbrace{\nabla\cdot\Big\{-(\mu+\lambda)\nabla(\delta\mathbf{u}) + \mu[\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla\mathbf{u}^{\mathrm{T}} + \nabla\mathbf{u}\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}} + \nabla(\delta\mathbf{u})\cdot\nabla(\delta\mathbf{u})^{\mathrm{T}}] + \lambda\mathrm{tr}(\delta\mathbf{E})\Big\}}_{explicit}$$

$$+\underbrace{\nabla\cdot[\delta\mathbf{S}\cdot\nabla\mathbf{u} + (\mathbf{S}+\delta\mathbf{S})\cdot\nabla(\delta\mathbf{u})]}_{explicit} - \underbrace{\nabla\cdot[(2\mu\delta\mathbf{E}^p + \lambda\mathrm{tr}(\delta\mathbf{E}^p)\mathbf{I})]}_{explicit} = \mathbf{0} \tag{7.5}$$

The global iterative solution procedure accordingly is implemented in Listing 7.2.

```
do
  {
    iCorr++;
    dU.storePrevIter();

    // store previous iteration, since later dU will be under-
       relaxed
    dU.storePrevIter();

    // solve the system with 'lagged' plasticity and cross-
       component coupling
    fvVectorMatrix dUEqn
    (
        fvm::d2dt2(rho, dU)
     ==
        fvm::laplacian(2*mu + lambda, dU, "laplacian(DU,dU)")

      + fvc::div
        (
            - ((mu + lambda)*gradDU)
            + (mu*gradDU.T())
            + (mu*(gradDU & gradU.T()))
            + (mu*(gradU & gradDU.T()))
            + (mu*(gradDU & gradDU.T()))
            + (lambda * tr(dEpsG) * I )
            + (dSigma & gradU)
            + ((sigma+dSigma) & gradDU),
            "div(sigma)"
        )

      - fvc::div(2.0*mu*(mesh.Sf() & fvc::interpolate(dEpsP)))

      - fvc::div(lambda*(mesh.Sf() & I*fvc::interpolate(tr(
          dEpsP))))
    );
    solverPerf = dUEqn.solve();

    dU.relax();
    gradDU = fvc::grad(dU);
    DF = gradDU.T();

    // update stress and plastic strains
     # include "correctPlasticity.H"

    dEpsG = symm(gradDU)
              +0.5*symm(gradDU & gradU.T())
              +0.5*symm(gradU & gradDU.T())
              +0.5*symm(gradDU & gradDU.T());

    dSigma = 2.0*mu*(dEpsG-dEpsP) + lambda*tr(dEpsG-dEpsP)*I;

  } while
(
 solverPerf.initialResidual() > convergenceTolerance
 && iCorr < nCorr
 );
```

Listing 7.2: `totalLaEpFoam` OpenFOAM solver code excerpt

7.3 Anisotropic poro-elasticity

The third advanced stress solver attempts to capture general anisotropic soil behaviors, including both mechanical(elastic properties) and hydraulic(permeability) anisotropy.

The anisotropic Biot's consolidation equations are expressed as:

$$\underbrace{\frac{n}{K'}\frac{\partial p}{\partial t} - \frac{1}{\gamma_w}\nabla\cdot(\mathbf{k}\cdot\nabla p)}_{implicit} + \underbrace{\frac{\partial}{\partial t}(\nabla\cdot\mathbf{u})}_{explicit} = 0 \tag{7.6}$$

$$\underbrace{\frac{\partial^2(\rho\mathbf{u})}{\partial t^2}}_{implicit} = \underbrace{\nabla\cdot(\mathbf{K}\cdot\nabla\mathbf{u})}_{implicit} + \underbrace{\nabla\cdot(\mathbf{C}:\boldsymbol{\varepsilon} - \mathbf{K}\cdot\nabla\mathbf{u})}_{explicit} - \underbrace{\nabla p}_{explicit} \tag{7.7}$$

where

$\mathbf{k}$ is the diagonal permeability tensor: $\mathbf{k} = \begin{pmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{pmatrix}$ The solution procedure of `anisoBiotFoam` is similar to that of the isotropic `biotFoam` solver: namely, the pressure equation is solved first, followed by the anisotropic momentum equation; after solution of both the pore pressure and displacement vector is obtained, one iteration is completed and unless a converged solution is obtained, the two equations will be solved again using the latest available iterative values. The corresponding implementation codes is present in Listing 7.3.

```
        do
        {
            p.storePrevIter();

            fvScalarMatrix pEqn
            (
                fvm::ddt(p) == fvm::laplacian(Dp, p) - fvc::div(fvc::
                   ddt(Dp2,U))
            );

            pResidual = pEqn.solve().initialResidual();
            p.relax();

            U.storePrevIter();

            fvVectorMatrix UEqn
            (
                fvm::d2dt2(rho, U)
             ==
                fvm::laplacian(K, U, "laplacian(DU,U)")

              + fvc::div(C && epsilonG)

              - fvc::div(K & gradU)

              - fvc::grad(p)
            );

            UResidual = UEqn.solve().initialResidual();
            U.relax();
            gradU = fvc::grad(U);

#       include "calculateSigmaEpsilonOrtho.H"

            initialResidual = max(pResidual,UResidual);

        } while (initialResidual > convergenceTolerance && ++iCorr <
           nCorr);
```

Listing 7.3: `anisoBiotFoam` OpenFOAM solver code excerpt

8 Convergence improvements

The fixed point iteration procedure associated with FVM is very simple, as there is no need to form and update the large Jacobian matrix like in FEM. Convergence can usually be achieved within certain number of iterations. However, there are occasions when very slow convergence or no convergence occurs due to strong coupling and/or high non-linearity. In those cases, some convergence improvement methods are necessary. In this chapter, we will explore the two different approaches that can help for convergence.

8.1 Explicit approach: relaxation methods

The first option is the explicit approach, namely, using the relaxation methods. Relaxation can be explained briefly by the formulation below:

$$\tilde{x}^i = (1-\theta)x^{i-1} + \theta x^i \tag{8.1}$$

where
$\tilde{x}^i$, stands for the i_{th} under-relaxed iterative value of a generalized unknown variable x;
θ, the under-relaxation factor with a value between 0 to 1. The value 0 means no update of the variable, while value 1 means full-update. In general, the smaller the relaxation factor, the stabler the solution procedure, but the more iteration numbers needed.

The relaxation factor θ can be specified either as constant value (fixed under-relaxation) or more efficiently as adaptive value based on the iteration sequences (adaptive under-relaxation, e.g. Aitken's method, Anderson acceleration, etc.).

Relaxation methods are easy to implement in OpenFOAM, since they do not alter the segregated solution procedure, instead they simply slow down the change of successive iterations and thus promote the stabilization of the solution method. In the following, I shall explain how to implement the fixed under-relaxation and one of the adaptive under-relaxation methods - Aitken's method - in OpenFOAM.

8.1.1 Fixed under-relaxation

Fixed under-relaxation method is very simple to use. You might have already noticed that, we applied this method in some of our previous solvers (e.g. Listing 5.2 and 6.4) by two steps. Firstly, we call a member function of a variable (scalar,vector,tensor fields) class, e.g.:

```
dU.storePrevIter();
```

which stores the previous iteration value.

And then after we solve the mentum equation, we call another member function of the variable:

```
dU.relax();
```

which simply under-relax the solved $d\mathbf{U}$. The relaxation factor is read from user specified sub-dictionary `relaxationFactors` in the `system/fvSolution` file. Let me show you an example here:

```
relaxationFactors
{
 dU   0.5;
 p    0.75;
}
```

which specifies that the increment of displacement vector shall be under-relaxed by a factor of 0.5 while the pore fluid pressure shall only be under-relaxed by 0.75.

Fixed under-relaxation might not be a good choice when for example strong plasticity occurs, since we have to use very small relaxation factor to promote the convergence. The small relaxation factor will slow down the whole simulation, even for the elastic range. In this case, we shall turn to more efficient relaxation methods, which allow us to dynamically adjust the relaxation factor accordingly.

8.1.2 Adaptive under-relaxation

Aitken's method is one of such efficient adaptive under-relaxation methods. It is most powerful for accelerating a linear convergent sequence, by adaptively computing the relaxation factors based on the errors in two successive approximations as follows:

$$r^i = \tilde{x}^{i+1} - x^i \tag{8.2}$$

$$\theta^{i+1} = -\theta^i \frac{r^{i-1}\left(r^i - r^{i-1}\right)}{\left(r^i - r^{i-1}\right)\left(r^i - r^{i-1}\right)} \tag{8.3}$$

where
x, the solving variable;
$\tilde{x}$, denotes the solved value before under-relaxation;
r, the error between two successive iterations;
θ, dynamic relaxation factor. θ^0 can be read from user input.

Though the implementation of Aitken's method involves more work than the simple fixed under-relaxation, the effort is worthwhile since we can save considerable iterations. As an example I will describe the implementation of Aitken's method for under-relaxing the increment of displacement. First, we need to create an error field and a scalar under-relaxation factor in the `createFields.H` file:

```
  volVectorField err
    (
        IOobject
        (
            "err",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        mesh,
        dimensionedVector("err", dimLength, vector::zero)
    );

    scalar theta(0);
```

Then we shall modify the main `.C` file to incorporate the computation of relaxation factor. Inside the momentum loop, we firstly store the previous computing variable as well as the previous error:

```
dU.storePrevIter();
volVectorField dUPrevIter = dU.prevIter();
err.storePrevIter();
volVectorField errPrevIter = err.prevIter();
const vectorField& errPrevIterI = errPrevIter.internalField();
```

After we solve the momentum equation, we update the error field, calculate the dynamic relaxation factor, and then perform the under-relaxation to obtain a better approximation value:

```
err = dU - dUPrevIter;
const vectorField& errI = err.internalField();
if (iCorr == 1)
 {
 theta = theta0;
 }
else
 {
 theta *= -(
           gSum(errPrevIterI & (errI - errPrevIterI))
           /(
            gSum(mag(errI - errPrevIterI))+SMALL
            )
           );
 }
dU.relax(theta);
```

The above Aitken's procedure can be performed on any computation variable of interest. Moreover, there are some other advanced adaptive relaxation methods, which use the whole sequence and distribution of the errors. Those methods can speed up the iterations significantly, however, not as trivial as the Aitken's method to use. Readers of interest are invited to have a look at the reference [19].

8.2 Implicit approach: block matrix solver

The second option, quite different from those relaxation methods mentioned before, is the implicit approach using block matrix solver. The so-called block matrix solver algorithm is intended for use in implicit solutions of strongly coupled variables sharing a common mesh [20, 21], where most of the explicit under-relaxation methods fail to give convergence or produce very slow convergence.

8.2.1 Theory of the block matrix solver

Let us first have a brief understanding on the theory of the new *block matrix solver* method.

Given a resulting finite volume discretization of a coupled equation set:

$$a_P \mathbf{x}_P + \sum_N a_N \mathbf{x}_N = \mathbf{b} \tag{8.4}$$

where, $\mathbf{x}$ is a vector of m arbitrary variables that we would like to solve from the equation set, and a is the coefficient matrix of dimension $m \times m$. The lower index P stands for the current computing cell and N is the neighbor of cell P.

The *block matrix solver* algorithm will differs from the conventional *segregated* algorithm as follows:

- The *segregated* approach - no coupling between variables, namely all the off-diagonal coefficients are zero:

$$a_P \mathbf{x}_P + \sum_N a_N \mathbf{x}_N = \begin{bmatrix} a_{11} & & \\ & \ddots & \\ & & a_{mm} \end{bmatrix}_P \mathbf{x}_P + \sum_N \begin{bmatrix} a_{11} & & \\ & \ddots & \\ & & a_{mm} \end{bmatrix}_N \mathbf{x}_N \tag{8.5}$$

- The *block matrix solver* approach - coupling between variables in owner and neighbor cells, namely the off-diagonals are non-zeros:

$$a_P\mathbf{x}_P + \sum_N a_N\mathbf{x}_N = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mm} \end{bmatrix}_P \mathbf{x}_P + \sum_N \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mm} \end{bmatrix}_N \mathbf{x}_N \quad (8.6)$$

We then write out the assembled sparse linear system as:

$$[A][\mathbf{X}] = [B] \quad (8.7)$$

- In *segregated* approach, we iteratively solve m small sparse linear systems of $[A]$, each is:

  $$[A_{small}] = [\underline{1 \cdot n} \times \underline{1 \cdot n}], [\mathbf{X}] = [\underline{1 \cdot n} \times 1] \quad (8.8)$$

  where, $n =$ number of cells.

- In *block matrix solver* approach, we solve once the large sparse linear system $[A]$:

  $$[A] = [\underline{m \cdot n} \times \underline{m \cdot n}], [\mathbf{X}] = [\underline{m \cdot n} \times 1] \quad (8.9)$$

Favorably, the sparseness pattern of block matrix $[A]$ is unchanged from the segregated small scalar matrix $[A_{small}]$, due to the same mesh set-up.

8.2.2 Implementation of the block matrix solver algorithm

The implementation of block matrix solver is not as trivial as the previous relaxation methods, as we no longer use the default segregated solution procedure which has been used for years in OpenFOAM. Development of new block matrix structure in OpenFOAM is needed. We can firstly gain some insight from the `blockCoupledScalarTransportFoam` solver, available in the OpenFOAM-1.6-ext path of `$Foam_APP/solvers/coupled`.

The original `blockCoupledScalarTransportFoam` solver solves a coupled two-phase fluid/solid heat transfer problem:

$$\nabla \cdot \phi T_f - \nabla \cdot D_{T_f} \nabla T_f = \alpha(T_s - T_f) \quad (8.10)$$

$$-\nabla \cdot D_{T_s} \nabla T_s = \alpha(T_f - T_s) \quad (8.11)$$

where
T_s, T_f, the temperatures of solid and fluid, respectively;
$\phi, D_{T_f}, D_{T_s}, \alpha$, material properties.

In this heat transfer problem, the two scalar equations are coupled simply through linear source terms, i.e. T_s present in Eq. (8.10) and T_f in Eq. (8.11). We might have two options to deal with the coupling:

- The conventional segregated implementation
  For the conventional segregated solution, relaxation and several iterations may be required. The code excerpt of such implementation is present below.

```
  do
  {
    solve
      (
       fvm::div(phi, Tf) - fvm::laplacian(DTf, Tf)
        == alpha*Ts - fvm::Sp(alpha, Tf)
       );
    Tf.relax();

    solve
      (
       - fvm::laplacian(DTs, Ts) == alpha*Tf - fvm::Sp(alpha, Ts)
       );
    Ts.relax();

   } while (initialResidual > convergenceTolerance && ++iCorr <
nCorr);
```

- The block matrix solver implementation
  For the block matrix implementation, all the terms in the equation system will be treated implicitly, thus no iteration is needed. A new `blockMatrixTools` class is defined to help for the construction of the large block matrix, preserving the same sparseness as the above small matrix system. The following codes describe the key parts of block matrix solver implementation.

```
 fvScalarMatrix TfEqn
 (
  fvm::div(phi, Tf)-fvm::laplacian(DT, Tf)==alpha*Ts-fvm::Sp(alpha, Tf)
  );

 fvScalarMatrix TsEqn
 (
  -fvm::laplacian(DTs, Ts)==alpha*Tf-fvm::Sp(alpha, Ts)
  );

  // Prepare block system
  BlockLduMatrix<vector2> blockM(mesh);

  //- Transfer the coupled interface list for processor/cyclic/etc. boundaries
  blockM.interfaces() = blockT.boundaryField().blockInterfaces();
```

```
// Grab block diagonal and set it to zero
Field<tensor2>& d = blockM.diag().asSquare();
d = tensor2::zero;

// Grab linear off-diagonal and set it to zero
Field<vector2>& l = blockM.lower().asLinear();
Field<vector2>& u = blockM.upper().asLinear();
u = vector2::zero;
l = vector2::zero;

vector2Field& blockX = blockT.internalField();
vector2Field blockB(mesh.nCells(), vector2::zero);

//- Inset equations into block Matrix
blockMatrixTools::insertEquation(0, TfEqn, blockM, blockX, blockB);
blockMatrixTools::insertEquation(1, TsEqn, blockM, blockX, blockB);

//- Add off-diagonal terms and remove from block source
forAll(d, i)
{
 d[i](0, 1) = -alpha.value()*mesh.V()[i];
 d[i](1, 0) = -alpha.value()*mesh.V()[i];

 blockB[i][0] -= alpha.value()*blockX[i][1]*mesh.V()[i];
 blockB[i][1] -= alpha.value()*blockX[i][0]*mesh.V()[i];
 }

 //- Block coupled solver call
 BlockSolverPerformance<vector2> solverPerf =
 BlockLduSolver<vector2>::New
 (
   blockT.name(),
   blockM,
   mesh.solutionDict().solver(blockT.name())
 )->solve(blockX, blockB);

 solverPerf.print();

 // Retrieve solution
 blockMatrixTools::blockRetrieve(0, Tf.internalField(), blockX);
 blockMatrixTools::blockRetrieve(1, Ts.internalField(), blockX);

 Tf.correctBoundaryConditions();
```

```
        Ts.correctBoundaryConditions();
```

It is important to note that for this very simple heat transfer problem, the two scalar equations are coupled through linear source terms. While, for our stress analysis problems, even the most simple linear elasticity case, the equations are coupled via differential operators. This difference complicates the implementation significantly. The currently available `blockMatrixTools` class only contains member functions that help to fill in the block diagonals, source terms and variables with the information from the conventional `fvScalarMatrix` and `fvVectorMatrix`. We have to extend the functions to fill in the block off-diagonals (namely coupling under differential operator) as well. Readers are invited to have a look at my previous work [5], where the more complicated Biot's poro-elastic consolidation equations have been implemented using block matrix solver procedure.

In general, the implicit block matrix solver gives good convergence for strong coupling and/or high nonlinear problems. However, the situation is not as clear-cut as it might seem: for example, we cannot tell in advance whether the block solution gives an a priori advantage over the segregated solution even for a simple linear elastic stress analysis problem: this is a question of the trade-off between the high expense of constructing and solving a large matrix and the simple iterative solvers with necessary iterations over the explicit coupling and non-linearity.

Bibliography

[1] OpenCFD. OpenFOAM - *The Open Source CFD Toolbox - User's Guide*. OpenCFD Ltd., United Kingdom, 1.4 edition, 11 April 2007. 1, 18

[2] OpenCFD. *OpenFOAM - The Open Source CFD Toolbox - Programmer's Guide*. OpenCFD Ltd., United Kingdom, 1.4 edition, 11 April 2007. 1

[3] Weller H.G., Tabor G., Jasak H. and Fureby C., A tensorial approach to computational continuum mechanics using object-oriented techniques. American Institute of Physicals 1998; Vol.12, No.6, 620-631 1

[4] Gradinaru DM. *Iterative Methods for Non-Linear Systems of Equations.* Lecture Notes. Online Website: http://www.math.ethz.ch/education/bachelor/lectures/fs2012/other/nm_pc/NPch1.pdf 3

[5] Tang, T., 2012, *Implementation of solid body stress analysis in OpenFOAM*, Project work from CFD with OpenSource software: A course at Chalmers University of Technology, 64p 3, 76

[6] Tang, T., 2014, *The CoCoons Project: The stressedFoam solver*, CoCoons Solvers Guide to OpenFOAM® Technology. 3

[7] De Souza Neto E. A., Peric D. and Owen D. R. J., 2008, Computational Methods for Plasticity: Theory and Applications. 6

[8] Demirdžić, I., Martinović, D., 1993, *Finite Volume Method for Thermo-Elasto-Plastic Stress Analysis*. Computer Methods in Applied Mechanics and Engineering, 109, 331-349 6

[9] Demirdzic I, Martinovic D. Finite volume method for stress analysis in complex domains. *International Journal for Numerical Methods in Engineering* 1994; **37**:3751-3766. 6

[10] Jasak, H., Weller, H.G., 2000, *Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity*. International Journal for Numerical Methods in Engineering, Vol. 48, Issue 2, 267-287 6

[11] Tutorial On J2 Plasticity, http://csm.mech.utah.edu/content/wp-content/uploads/2011/10/9tutorialOnJ2Plasticity.pdf. 13

[12] Clausen J, Damkilde L, Andersen L. An efficient return algorithm for non-associated plasticity with linear yield criteria in principal stress space. *Computers and Structures* 2007; **85(23-24)**:1795-1807. 18

[13] Demirdzic I, Horman I, Martinovic D. Finite volume analysis of stress and deformation in hygro-thermo-elastic orthotropic body. *Computer Methods in Applied Mechanics and Engineering 2000*, 190:1221-1232. 21

[14] Cardiff P, Karac A, Ivankovic A, A large strain finite volume method for orthotropic bodies with general material orientations, *Computer Methods in Applied Mechanics and Engineering* 2013, Vol. 268, 318-335. 21

[15] Lekhnitskii S. *Theory of Elasticity of an Aniostropic Body.* Mir Publishers, 1981. 37

[16] Mattiasson K, Numerical results from large deformation beam and frame problems analysed by means of elliptic integrals. *International Journal for Numerical Methods in Engineering* 1981. Vol. 17, p.145-153. 49

[17] Biot MA. General theory of three-dimensional consolidation. *Journal of Applied Physics* 1941; **12(2)**: 155-164.

[18] Verruijt A. *Soil Mechanics.* Delft University of Technology, 2001. 61

[19] Michiel W., Efficiency Improvement of Strongly Coupled Fluid-Structure Interaction Simulations, Master thesis 2010, Delft Univeristy of Technology. 72

[20] Clifford I., Block-Coupled Simulations Using OpenFOAM, 6th OpenFOAM Workshop 2011. 72

[21] Kissling, K., Springer, J., Jasak, H., Schutz, S., Urban, K., and Piesche, M., A coupled pressure based solution algorithm based on the volume-of-fluid approach for two or more immiscible fluids, *V European Conference on Computational Fluid Dynamics* 2010, Lisbon. 72