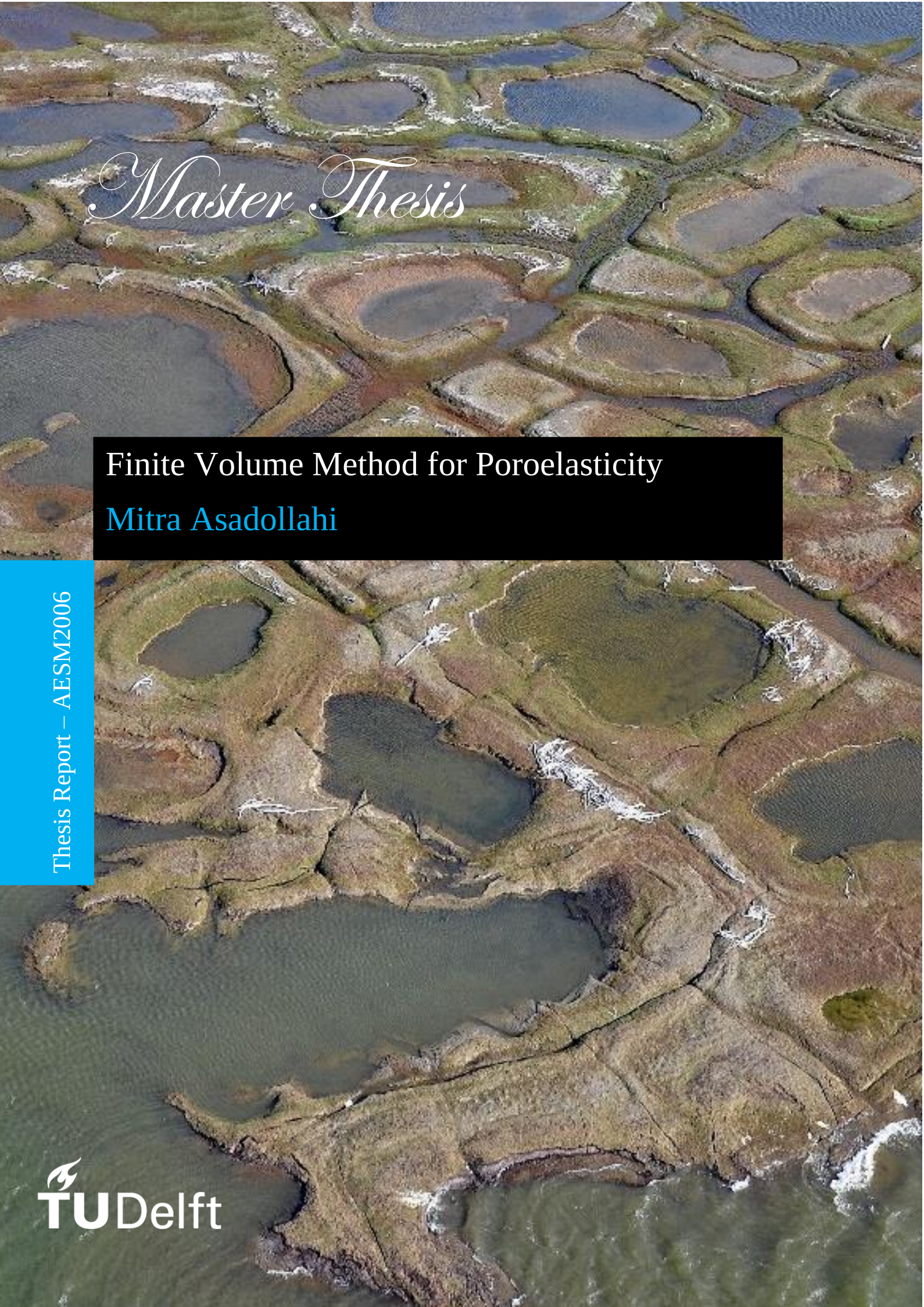


Finite Volume Method for Poroelasticity

Mitra Asadollahi

Thesis Report – AESM2006

TU Delft

Delft University of Technology
Faculty of Civil Engineering and Geosciences
Department of Geoscience & Engineering

Finite Volume Method for Poroelasticity

in partial fulfilment of the requirements for the degree of

Master of Science in Applied Earth Science
to be defended publicly on Thursday March 9, 2017 at 9:00 a.m.

Author

Mitra Asadollahi *Stdt. No.* 4412451

Supervisor

Dr. Auke Barnhoorn

Thesis Committee

Prof. Dr. Ir. Jan-Dirk Jansen
Prof. Dr. Ir. Bert Sluys
Dr. Denis Voskof
Dr. Phillip Vardon

Project Duration

9/11/2015-9/03/2017

February 24, 2017

I have no special talents.
I am just passionately curious.
-Albert Einstein

Contents

List of Figures

List of Tables

Abstract

When modelling fluid flow in subsurface, the impact of solid deformation on fluid flow is often oversimplified/neglected in reservoir simulators. It is assumed that solid volume/state of stress is a function of fluid pressure, while the opposite effect is not considered. This assumption is made mainly to reduce computational costs and complexity of fluid flow models. Nevertheless, this simplification is not valid in case of unconsolidated rocks. This oversimplification results in wrong estimation in prediction of surface subsidence, earthquakes and fault activation, fluid production and rock permeability values, etc. Numerous reports suggest that neglecting the two-way coupling (i.e. both fluid-to-solid and solid-to-fluid coupling) has led to disastrous events in many cases. This necessitates modified fluid models and simulators which take into account the two-way coupled nature of solid deformation and fluid flow in porous media.

Efforts have been made to model this two-way coupled nature properly which can be categorized as follows. There have been attempts to connect commercial softwares to model this problem. They fail due to differences in data structure, different underlying assumptions embedded and due to the increased computational costs. Therefore, it is essential to integrate fluid modelling and solid deformation into a single simulator. This requires developing new numerical models. Presuming an elastic nature for unconsolidated rocks, Biot's consolidation equations are employed to numerically model the two-way coupled solid deformation and fluid flow in porous media, so called poroelasticity.

During my master work, I developed 2D MATLAB codes based on two different finite volume discretization schemes (cell-centred and vertex-centred FVM). In the first stage, two cell-centred FVM 2D MATLAB codes were developed: One to model fluid flow, and one to model solid equations in poroelasticity. At the next stage, the two cell centred codes were iteratively coupled. Then, another 2D fully coupled model based on a vertex-centred finite volume discretization scheme was developed for poroelasticity. The fluid and solid data structure in both developed models are the same; in other words, unknowns are collocated. To verify and conduct error analysis on the developed finite volume based simulators, 2D MATLAB codes for one classic benchmark problem in poroelasticity, namely the Mandel problem, as well as 2D MATLAB codes for synthetic test cases were developed. At the next stage, the performance of the two models is compared. Though both methods illustrate adequate performance, fully coupled finite volume method is preferred. Finally, the reaction of the fully coupled finite volume model to different systems, its performance under stress and displacement boundary condition configuration, sensitivity of the model to uncertainty of input parameters, robustness of the model, and applications of this method are being analysed. Furthermore, thorough discussion on enhancement of the model is provided.

To our knowledge, it is for the first time that a cell-centred finite volume scheme is applied to poroelastic problems and is compared with the vertex-centred finite volume method. In addition, there is no prior investigation done of robustness for finite volume methods. The results showed that both cell-centred and vertex-centred FVM are computationally efficient in applications for poroelastic problems. However, the fully coupled vertex-centred finite volume model outperformed the sequentially coupled cell-centred model in terms of computational efficiency.

Acknowledgement

I would like to express my sincere gratitude to my supervisors, Dr. Auke Barnhoorn and Dr. Hadi Hajibeygi, and professors form Petroleum Engineering & Geoscience section for their support of my research. The partial financial support of my research by the DARSim group is also appreciated.
Besides my supervisors, I would like to thank my thesis committee for their insightful comments.
I also thank my fellow students for their stimulating supports. I would like to especially thank Menel Rahrah and Pieyao Luo, PhD candidates in Applied Mathematics Department, and Dr. Mehdi Musivand Arzanfudi, the postdoctoral employee in civil engineering faculty, for their collaborations and fruitful discussions.
I also appreciate the helpful responses I received through emails and Skype from Dr. Jan Martin Nordbotten and especially Dr. Mostafa Bakhoday Paskyabi from the University of Bergen.
Last but not the least, I would like to thank my family for supporting me throughout my studies for master of petroleum engineering at TU Delft.
I am so grateful for all I have learnt and experienced.

Yours sincerely,

Mitra Asadollahi

1 Introduction

In the field of petroleum engineering, when modelling fluid flow in the subsurface, the solid deformation impact on fluid flow is often neglected/oversimplified. Usually the solid deformation is post processed or analysed separately from the fluid flow(Kim et al., 2009.; Nordbotten, 2014a; Asadi and Ataie-Ashtiani, 2016). However, quantitative evidences in the petroleum engineering field clarify a demand for a numerical simulation tool which considers the interaction between the solid deformation and the fluid flow (Kim et al., 2011b; Nordbotten, 2014a). In this respect, two novel finite volume numerical models were developed using two different discretization and coupling schemes: One is a finite volume sequentially coupled model with cell-centred data, structured for both fluid and solid. The other model is a finite volume fully coupled vertex-centred data structured model. The models are able to predict horizontal displacement, subsidence, fluid flow and changes in the stress field due to fluid production/injection and appropriately account for solid deformation impact on fluid flow in homogeneous and heterogeneous reservoirs in an accurate and computationally efficient manner.
This chapter provides the reader with an introduction to poromechanics, the early efforts to model poromechanics in poroelastic materials and some quantified evidences clarifying the demand for accurate and efficient modelling. Also included are the objectives of this thesis and a summary.

1.1 Basic Concepts

A porous medium is a solid material with some void/pore space. The pore space is filled by one or several type(s) of fluid(s). Therefore, there are properties of the medium that are affected by both solid and fluid properties/conditions. For instance, how much a porous medium shrinks in volume under the pressure changes exerted uniformly on the outer perimeter of the medium at constant temperature, i.e. isothermal condition, which is known as matrix compressibility. Obviously, these properties are subject to change with changes in fluid/solid properties/conditions. In this respect, we define two limiting types of properties: *undrained* properties which are measured under a constant fluid content where no fluid goes out or comes into the domain and *drained* properties where the fluid is free to enter or exit the domain. Besides, two phenomena can be observed in a porous medium (Wang, 2000):

- ***Fluid-to-solid coupling*** where a change in fluid pressure or mass content results in a change in the solid volume/state of stress, for example the occurrence of the subsidence and the fault activation is an example of this phenomenon.
- ***Solid-to-fluid coupling*** where a change in the applied stress causes a change in the fluid pressure. Examples of this phenomenon are the hydrocarbon production due to the compaction or water level changes due to train/tidal movements. Strength of solid-to-fluid coupling is highly affected by the compressibility of fluid, solid and the porous medium as well as porosity values.

The two mentioned phenomena can occur simultaneously where a change in the fluid pressure/mass content causes a change in the solid volume/deformation and this change in solid volume/deformation influences the fluid model. This phenomenon is called *two-way coupling* or *coupled* solid deformation and fluid flow. Any study that considers the coupled nature of the solid and the fluid in the porous medium is called *poromechanics*.
In the following, the essence of poromechanics is explained and some of the early accomplishments in this field are mentioned. Finally, the objectives and challenges of this thesis, along with the thesis outline, are addressed.

1.2 Poromechanics

When modelling the fluid flow in a subsurface using a reservoir simulator, the impact of solid deformation on fluid flow is often neglected. It is assumed that solid volume/state of stress is a function of fluid pressure, while the opposite effect is not considered. The coupled nature of solid deformation and fluid flow is oversimplified into the compressibility of the medium. This assumption is made mainly to reduce computational costs and complexity of fluid flow models. In some cases considering two-way coupling is essential to properly estimate and predict occurrence of surface subsidence, earthquakes and fault activation, fluid production, rock permeability values, etc.
In a study carried out by Simpson (1976), over thirty cases were recorded which show solid particle movements are induced at a wide range of levels -micro-earthquakes to destructive earthquakes- by fluid movement in the subsurface due to fluid injection to/production from reservoirs. As Simpson mentions the changes in the total stress, the effective stress and the pore pressure leads to redistribution of the stress field which might result in micro- to marco-earthquakes. Moreover, as defined in Figure 1.1, there are three main stress regimes based on the ratio between principle/main stresses. Different types of faulting occur under different stress regimes. As is well known, most reservoirs are bounded by faults. As long as the stress regime stays the same. the fault does not activate. However, stress changes might lead to a change in the stress regime and consequently a fault activation. Thus, the poromechanical modelling of these phenomena are of essence.

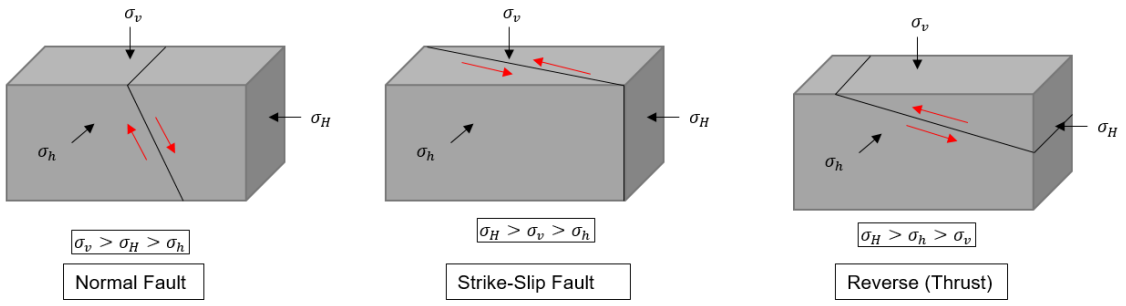


Figure 1.1: *Illustrates the main stress regimes.*

In other studies subsidence has been considered. The fluid extraction causes a shrinkage of the pore space and a subsidence. This subsidence is negligible in most cases; however, cases are reported in which a lack of an accurate two-way coupling model leads to wrong prediction of subsidence values and it becomes very costly. For instance, the Wilmington oil field in Long Beach, California (Allen, 1968.), the Groningen gas field in the Netherlands (Schoonbeek, 1976.), the Ekofisk oil field in the Norwegian sector of the North Sea (Hermansen et al., 1997.). Furthermore, the hydrocarbon production due to the compaction is not predicted accurately if the two-way coupling of the fluid and the solid is not considered. For instance, in the Ekofisk oil field, a high level of subsidence due to the pressure depletion led to a major production increase. Lastly, the prediction of the rock permeability in the low permeable rocks, the sand production and well-bore failure in the unconsolidated reservoir, and the fractures and rock permeability changes which occur due to fluid injection/production can be properly estimated if the coupled nature of the fluid and the solid in a porous medium is considered (Jacob, 1940; Hsieh et al., 1981; Hermansen et al., 1997.).

1.3 General Background

Poromechanics has a wide range of applicability in civil, environmental, bio and reservoir engineering fields. Early observations of the coupled nature of the fluid flow and the solid deformation can be traced back to the hydrology field when Darcy carried out experiments to quantify the well response to different loading pressures induced by train movements/tides.
Later, Karl Terzaghi (1923-1925) formulated the behaviour of the soil under the uniaxial strain. He formulated the results of the experiments as stated in Equation (1.1) which is known as the fluid diffusion equation (Terzaghi, 1943).

$$\frac{\partial p}{\partial t} = c\frac{\partial^2 p}{\partial x^2} \tag{1.1}$$

where p is the pressure $[Pa]$ and c is the consolidation coefficient $[m^2/s]$. Biot in 1941 extended Terzaghi's formulation of the linear poroelasticity to a three dimensional domain (Biot, 1941b,a). He proposed how his formulations can be extended to include the anisotropy, the thermoelasticity, the non-linear elasticity and how the equation coefficients can be quantified (Biot, 1955, 1956b,a; Biot and G., 1957; Biot, 1962).
McNamee and Gibson in 1960 used Biot's formulation via transform functions to axisymmetric cases, where the co-ordinates are expressed in terms of θ and r which is normally the case near the well-bore (McNamee and Gibson, 1960). Furthermore, there are numerous reports and cases in petroleum engineering and hydrocarbon fields on the wrong prediction of the reservoir behaviour in terms of hydrocarbon production forecasts, hydraulic fracturing effects, initial permeability estimations, subsidence, fault slip and earthquake predictions (Pratt and Janson, 1926; Greetsma, 1966; Haimson and Fairhurst, 1969; Greetsma, 1973a,b; Rice and Clearly, 1976; Neuzil et al., 1981; Hsieh et al., 1981; Segall, 1985; Rudnicki, 1986; Roeloffs, 1988; Segall, 1989; Segall et al., 1994; Roeloffs, 1995; Wang, 1997).
These reports confirm the demand for a poromechanic model to predict the behaviour of the

reservoir. The numerical models offer substantial advantages compared to experimental works. The most important advantages are (Jing and Hudson, 2002; Jing, 2003):

- The prediction of the sensitivity of results to the uncertainty of parameter variables.
- Full scale regional/reservoir simulations which are not feasible to do in the laboratory.
- The generality in terms of dealing with different configurations, for example in terms of boundary conditions, as compared to the lab experiments.

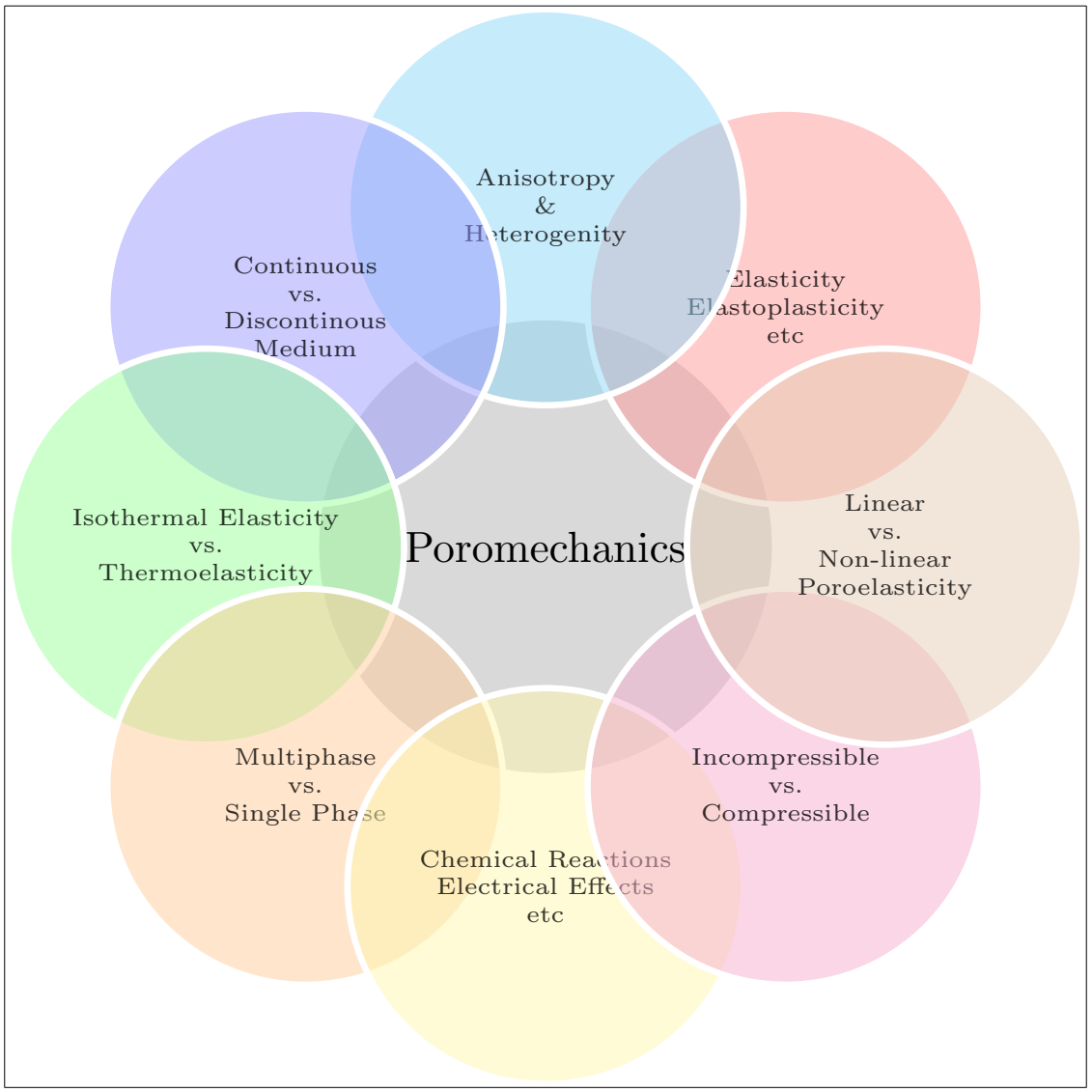


Figure 1.2: *Illustrates a general view of common topics addressed in poromechanics*

A poromechanic model can predict the behaviour of a porous medium under various conditions of the fluid and the solid: each of which has a wide range of applicability in the various fields; as shown in Figure 1.2. For instance, the rock is normally heterogeneous and its property might

differ with directions, i.e. heterogeneity and anisotropy, the stress and strain relationships in the porous material can be linear or non-linear which might demand poroelastic or elastoplastic modellings of the subsurface, the medium can be considered as continuous, i.e. without fractures, or the discontinuity may exist via fractures, the temperature gradient zero or non-zero which requires isothermal poromechanics or thermo-poromechanics modelling of the subsurface, respectively.

1.4 Objectives and Challenges

As mentioned, existing models that take into account the impact of coupled solid deformation and fluid flow are either very simple and skip mass conservation (Kim et al., 2011a) or use different data structures which bring complexity to the model (Nordbotten, 2014a). The objective of this work is to introduce an accurate method to account for the solid deformation and its effects on the fluid flow. The models are expected to predict horizontal displacement, subsidence, fluid flow and changes in the stress field due to fluid production/injection and appropriately account for solid deformation impact on fluid flow in homogeneous and heterogeneous reservoirs in an accurate and computationally efficient manner.
For this purpose, two poroelastic finite volume numerical simulators were developed: one fully coupled method and a sequentially coupled method, based on two different discretization techniques. The novelty of this work lies in the fact that it, for the first time, applies the cell-centred FVM method to poroelasticity. It compares the cell-centred and vertex-centred finite volume methods, compares iteratively coupled and fully coupled schemes for finite volume methods, and also investigates the vertex-centred finite volume method's performance in terms of reaction to different models. Sensitivity analysis is performed and robustness is also investigated.

1.5 Thesis Outline

The remainder of this report is structured as follows. In Section 2, a thorough literature review on numerical models in poromechanics, numerical poroelastic models, finite volume discretizations for poroelasticity and coupling schemes for poroelasticity are presented. In Section 3, governing equations, assumptions, the coupling scheme and the finite volume discretization techniques are presented. In Section 4, the developed models are verified, and compared. The fully coupled finite volume method is opted for further investigation and benchmarked. A parametric analysis is carried out on the preferred model to investigate the sensitivity of the model to input parameters in terms of uncertainty and robustness. Finally, an application of the model in terms of a practical problem is illustrated. In Section 5, the applications of the current model, its limitations and delimitations and how to overcome them are discussed. In Section 6, the report is concluded by some concluding remarks and recommendations. Appendices provide the reader with detailed explanations on governing equations, an analytical solution to the Mandel test case, the range of parameters in the poroelastic problems and some of the MATLAB codes developed through this master thesis.

2 Literature Review

A porous medium can be numerically modelled by subdividing the domain into a sufficient number of elements. Finite volume method is the most applicable method in modelling fluid flow in porous media due to its simple implementation, conformity with physics, i.e. mass/energy conservation and low computational costs (Nordbotten, 2014a). In this section, as illustrated in Figure 2.1, numerical methods to model poromechanics and poroelasticity with focus on a continuum medium are discussed. A short review on finite volume models is presented in this chapter and coupling methods connecting fluid flow and solid deformation are discussed.

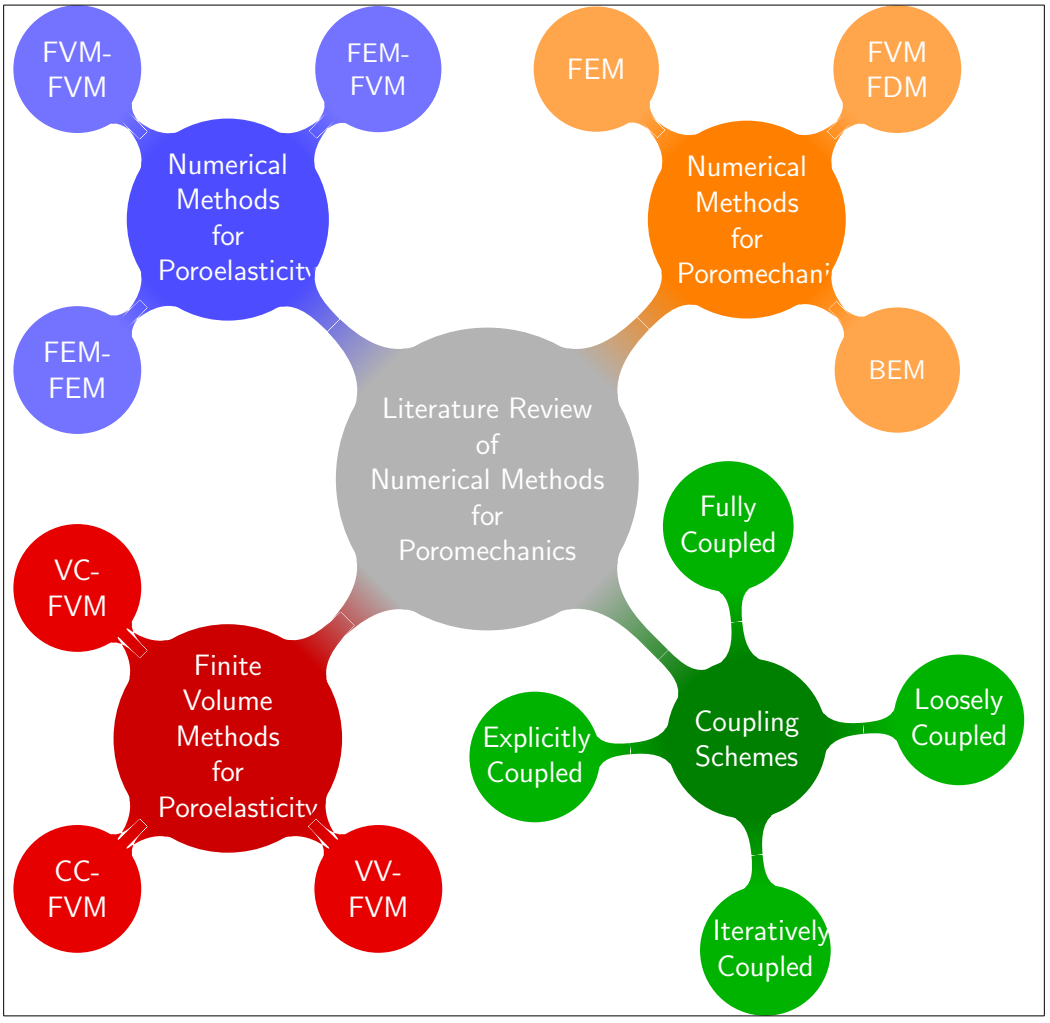


Figure 2.1: *Illustrates an overview of literature review section.*

2.1 A General Overview of Numerical Methods for Poromechanics

There are several numerical models in rock mechanics and each method has its specific advantages and disadvantages. Common methods to model continuum medium are: Finite Element Method (FEM), Boundary Element Method (BEM), Finite Volume/Difference Method (FVM/FDM), see Figure 2.2. It should be noted that there are a number of hybrid models in practice which subdivide the solid domain into near-field and far-field in which near-field is fracturing/(elasto-)plasctic and far-field goes through less deformation. In this approach, different sets of numerical methods are applied to model each sub-domain, and therefore continuity and compatibility of different methods should be ensured at interface of the near- and far- field sub-domains.

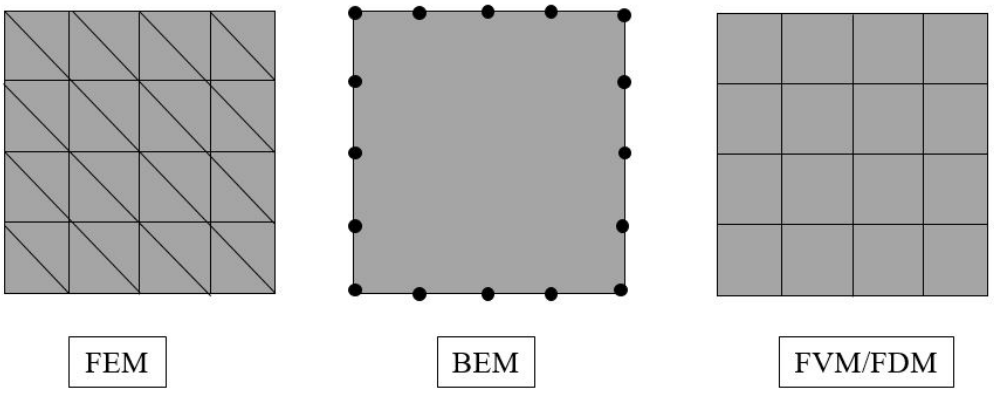


Figure 2.2: *Illustrates an overall view of the discretized domain in Finite Element Method (FEM) with structured triangular grids, Boundary Element Method (BEM), Finite Volume/Difference Method (FVM/FDM) with structured rectangular grids.*

2.1.1 Finite Element Method (FEM)

The finite element method subdivides the domain into sub-domains with certain shapes, for instance, triangular, and quadrilateral shapes, Figure 2.3; with a certain number of unknowns at nodes. The number of unknowns depends on the selected shape function. FEM is the most common numerical approach to model rock mechanics due to its high flexibility with respect to heterogeneity, anisotropy, fractures, none-linearity and complex boundary conditions. However, this method has problems such as locking. Locking effects are subdivided into two category:

- ***Numerical locking*** which refers to non-robustness of FEM near limiting values. *Shear locking* for thin shale plates, *membrane locking* can be caused by the interaction between bending and membrane energies and *volume locking* is observed in poroelasticity when modeling materials with Poisson ratio near 0.5, such as elastic incompressible materials (Wihler, 2006).
- ***Element locking*** which is numerical instability which can be caused by large aspect ratios, for example dx/dy, of elements.

Mesh generation in FEM is demanding because the number of elements should be large enough to capture the geometry and small enough to be time-wise cost effective. In addition, re-meshing and elements needed to model fracture growth in FEM makes this method to be difficult for fracture problems. However, efforts are made to remove the drawbacks by developing adaptive FEM to deliver locking-free FEM methods, generalized FEM (GFEM) to make FEM model complex geometry and extended FEM (X-FEM) to model fracture propagation (Ehlers, 1999; Jing, 2003; Lamb et al., 2010.; Moghaddam et al., 2016; Yang et al., 2016).

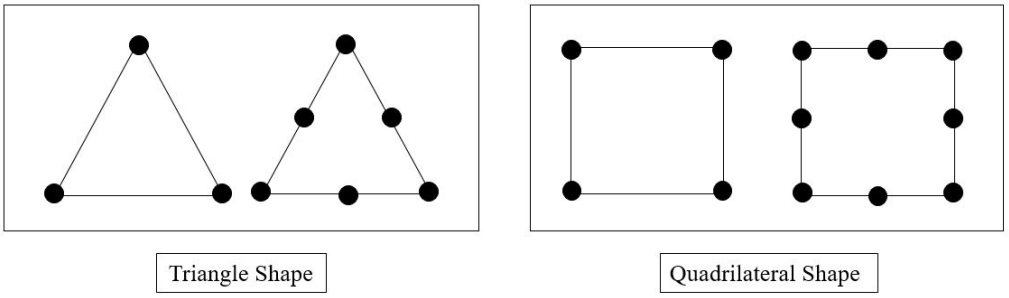


Figure 2.3: *Illustrates triangular and quadrilateral shapes for elements in FEM.*

2.1.2 Boundary Element Method (BEM)

For implementation of boundary element method discretization is necessary only at the domain boundary. The inner-domain solution is obtained by boundary integral equation. BEM comes with more accuracy and less computational costs compared to other methods due to its limited discretization. Nevertheless, due to its limited discretization, it has problems with heterogeneity, plasticity, non-linearity. Thus, BEM renders poor efficiency in the aforementioned cases. Overall, BEM is an efficient method to model large homogeneous linear elastic domains. It's worth mentioning that, in recent decades, enhanced BEM has been developed to improve efficiency of BEM and remove its shortcoming; examples are: BEM in Laplace domain, time-domain BEM and Convolution Quadrature BEM, i.e. CQ-BEM (Boyce and DiPrima, 1977; Banerjee and Butterfield, 1981; Crouch and Starfield, 1983; Cheng and Detournay, 1988; Jing and Hudson, 2002; Jing, 2003; Igumnov and Markov, 2012; Banjai and Schanz, 2015).

2.1.3 Finite Difference/Volume Method (FDM/FVM)

The finite difference/volume method translates the original differential equations into sets of algebraic equations. Finite difference method is one of the earliest method used to approximate solutions in engineering. Nonetheless, FDM is hardly applicable to rock mechanics since FDM suffers from inflexibility when it comes to complex boundary conditions and geometry, heterogeneity and fractures. On the other hand, FVM is flexible in terms of handling complex boundary conditions and geometry by proper grid generation. Besides, FVM assures local

conservation of mass/momentum/energy. Additionally, FVM uses less unknowns/nodes as compared with FEM; therefore, FVM requires less memory storage which results in lower computational cost. The main challenge in FVM is modeling fractures since interpolation functions used in FVM to model rock mechanics need continuity of the medium. It is worth mentioning that developments are being made regarding this issue (Vanselow, 1996; Caillabet et al., 2000; Granet et al., 2001; Jing and Hudson, 2002; Jing, 2003).

2.2 Numerical Methods for Poroelasticity

When solving solid and fluid equations, solid equation(s) can be solved with primary variable(s) such as stress, strain or displacements. In fluid equations the primary unknowns can be fluid mass content or pressure. Other variables can be calculated as secondary unknowns. For this reason there are different sets of equations used in modeling poroelastic problems, such as: Diffusive fluid, strain compressibility, equilibrium-storage equations, etc. Furthermore, other secondary variables can also be treated as primary unknowns and be included in the coefficient matrix and the unknown vector in order to increase accuracy. The compromise for having more variables in the unknown vector is, of course, higher computational costs (Korsawe et al., 2006). Most often, the numerical methods set pressure and displacement as primary unknowns while in analytical solutions volumetric strain and pressure are commonly chosen to be primary variables (Abousleiman et al., 1996).

There are several common numerical problems which make numerical modelling in poroelasticity quite a challenging task. Even though the first numerical models can be traced back a long time, the following issues are not yet fully resolved:

1. *Inf-sup issue* occurs when the medium approaches undrained condition/incompressible elasticity, non-physical pressure oscillations appear in the system. Ladyzenskaja-Babuska-Brezzi, i.e. LBB, is an example of inf-sup. This phenomenon was first observed by Babuska and Brezzi (Babuška, 1973; Brezzi and Bathe, 1990). LBB condition and patch tests(e.g., Zienkiewicz-Taylor patch test) were developed to test robustness of numerical schemes against inf-sup conditions (Chapelle and Bathe, 1993; Korsawe et al., 2006; Li, 2015; Guzmán and Olshanskii, 2016). Figure 2.4 represents some unknown placements that do (not) verify the inf-sup condition when applying FEM to modelling the fluid. Similar to Figure 2.4, there are unknown placements which do (not) verify the inf-sup condition when FEM is applied to poroelasticity. Therefore, positioning pressure and displacement unknowns faces limitations when applying FEM to poroelasticity.

2. *Ill-conditioning of algebraic systems* this may result in constrains on time-step in numerical models. The lower the time-step value is, the higher the probability of the problem being ill-conditioned and oscillations appearing in the system (Vermeer and Verruijt, 1981; Ferronato et al., 2001; Asadi and Ataie-Ashtiani, 2016).

3. *Discontinuities at interfaces* discontinuous faces at interfaces of the porous medium with different material properties might cause discontinuities in pressure gradients at these

interfaces. These discontinuities might lead to non-physical oscillations in results (Murad and Loula, 1992; Asadi and Ataie-Ashtiani, 2016).

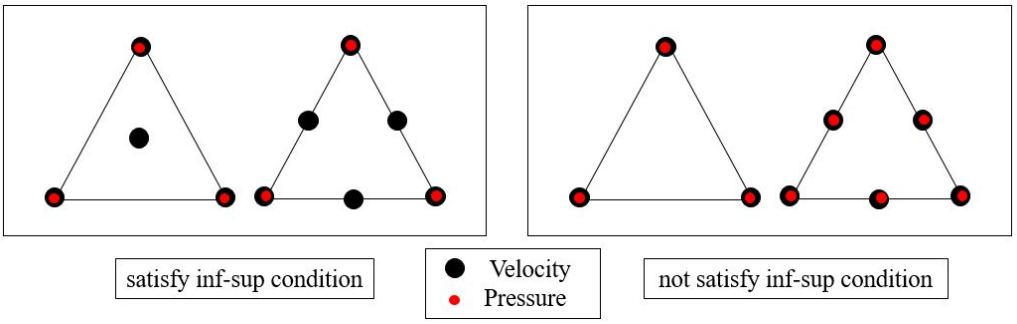


Figure 2.4: *Illustrates some unknown placements that do (left side) and do not (right side) satisfy the inf-sup condition when applying FEM to modelling the fluid.*

There are various ways to numerically model poroelasticity; however, the most popular numerical combinations are as follows:

2.2.1 Finite Element-Finite Element Numerical Method (FEM-FEM)

This numerical scheme encounters all aforementioned problems and is not mass conservative unless it is modified. Therefore, new FEM-FEM methods are developed, such as, General Finite Element Method (GFEM), (least square) mixed-FEMs (LS-MFEM), Virtual Finite Element Methods (VEM), and mesh-less schemes. Of all the methods mentioned, the mixed-FEM is the most used method where a quadrilateral shape function is used for displacement and a bi-linear shape function is employed for pressure, named as Q8P4. Attempts have been made to use low-ordered shape functions for both pressure and displacement, i.e. Q4P4; however, apparently there is a trade-off between robustness, i.e. successfully passing patch tests, and bending properties of the cell. In other words, rigid displacement mode goes along with stability and robustness. Besides, despite all the improvements in FEM, attempts are still being made to remove non-physical oscillations from present methods even in one dimensional problems (Korsawe et al., 2006; Wang et al., 2007; Li, 2015; Nilsen et al., 2016; Borregales et al., 2016; Rodrigo et al., 2016).

2.2.2 Finite Element-Finite Volume Numerical Method (FEM-FVM)

This technique is commonly used in modelling poroelasticity. Conventional FVM-FEM approaches suffer from poor efficiency in overcoming aforementioned problems. The most common approach in this category of work is employing Taylor-Hood FEM which is intrinsically LBB stable combined with Multi-point Flux Approximation, so called MPFA, which allows for accurate flux approximation at the cell face in unstructured grids. However, there are recent reports stating that Taylor-Hood FEM shows non-physical pressure oscillation in cases

of discontinuity of material property which of course dissipates through time and/or can be removed by constrained time-step/grid size (Castelletto et al., 2015; Li et al., 2015; Asadi and Ataie-Ashtiani, 2016; Garipov et al., 2016).

2.2.3 Finite Volume-Finite Volume Numerical Method (FVM-FVM)

This approach uses a lower number of degrees of freedom and is thus faster than the previous two methods. This method can be as accurate as FEM. This technique is among the most promising techniques that offer the same data structure for both solid and fluid equations. The same data structure brings the added value of easy combination with fast solvers like multigrid schemes, especially when used on structured grids. The aforementioned problems are naturally resolved when FVM [1] is used for modelling both fluid and solid due to intrinsic conformity of FVM with physics (Shin and Strikwerda, 1997; Gaspar et al., 2006a; Oosterlee and Gaspar, 2008; Asadi et al., 2014; Nordbotten, 2014a,b; Keilegavlen and Nordbotten, 2015; Luo et al., 2015).
Note that the problems/disadvantages mentioned in Section 2.1 might still be faced when employing FEM/FVM.

2.3 Discretization Schemes for Finite Volume Method

Finite Volume methods employed to solve poroelasticity can be cell-centred or vertex-centred. Vertex-centred methods have been used in modelling a solid. These methods allow for flexible treatment of boundary conditions and are proven to be stable and accurate. On the other hand, cell-centred FVM is commonly used in multi-phase and flow modelling. As a result, unknowns in poroelasticity can be placed in staggered grids, i.e. one primary unknown at cell centres (commonly pressure) and the other (commonly displacement) at cell-faces, or collocated grids, i.e. all unknowns are located at the same position. Accuracy of both methods are similar. Staggered grid shows high stability in poroelastic numerical modelling; however, applying fast solvers like multigrid to staggered grid and generalizing staggered grid to unstructured grid, curved domains and thermo-hydro-mechanical models are challenging tasks (Perić et al., 1988; Gaspar et al., 2006a, 2008b; Oosterlee and Gaspar, 2008). The limited applications for staggered grids creates a demand for collocated grids in poroelasticity. It is noted that when vertex-centred FVM for fluid flow modelling is applied, dual grids should be considered to check mass conservation; hence, this method is less popular compared to cell-centred FVM in fluid modelling. Generally, given the different finite volume methods available, the following choices can be made when employing FVM for poroelasticity[2].

[1] Since all FDM can be translated into FVM and FVM is more advanced and used compared to FDM in modelling rocks and poroelasticity; FVM is used to account for both terms

[2] The focus in this subsection is mainly on recent advances and methods developed for poroelastic finite volume methods

2.3.1 Vertex Centred-Vertex Centred FVM (VV-FVM)

This method is among poroelastic finite volume methods; however, when vertex-centred finite volume method is applied for fluid, the accuracy of pressure approximations would be higher than flux approximations, which might lead to issues regarding mass conservation in multi-phase and non-linear flow equations. Nonetheless, studies have been published which show robustness of vertex centred FVM for multiphase/fractured flow in porous medium (Reichenberger et al., 2006; Gaspar et al., 2007; Oosterlee and Gaspar, 2008; Gaspar et al., 2008a; Nordbotten, 2014a; Luo et al., 2015).

2.3.2 Vertex Centred-Cell Centred FVM (VC-FVM)

In this approach the choice of vertex centred FVM for solid and cell centr FVM for fluid seems to be an ideal combination [3] since each is proven to be highly efficient for solid and fluid computation, respectively. Despite this fact, unknowns in this method must be placed in a staggered grid. Consequently, this technique suffers from aforementioned disadvantages for staggered grid though offers stable numerical solutions for poroelasticity. VC-FVM models are being mainly and extensively studied by Gasper and Oosterlee (Gaspar et al., 2006a; Oosterlee and Gaspar, 2008). However, a one-dimensional VC-FVM was developed and tested in various coupling schemes (Asadi et al., 2014) in which this method showed acceptable results. Recently, a two-dimensional VC-FVM is developed by Deb and Jenny (Deb and Jenny, 2016); however, the robustness and accuracy of this method given in the presented results are being questioned.

2.3.3 Cell Centred-Cell Centred FVM (CC-FVM)

This technique is a recently developed scheme by Nordbotten (2014a). In this publication, it is suggested that low applicability of CC-FVM for solid modelling is due to its inaccurate stress calculation in former suggested numerical methods. Therefore, in this approach, he suggests Multi-point Stress Approximation (MPSA). MPSA is derived from MPFA to build-up stress values at cell-faces. Applicability of the so-called O-method in which points used to approximate flux look like an O, the so-called U-method in which points used to estimate flux form a U-shape line, and the so-called L-method in which points used to calculate flux form a shape similar to L, of multipoint approximation were tested. Figure 2.5 gives an illustration of multipoint approximation methods. Similar to their applications in fluid, these methods lose accuracy and obtain simplicity respectively.

In other works published by Nordbotten (2015b,a), extensive studies have been made on convergence and stability of this method. The major advantage of this method is that it uses the same data structure for solid and fluid, a low number of degrees of freedom which intrinsically

[3] There are different ways to locate displacement and pressure unknowns in staggered grids. There is a possible discretization scheme for this method such that no extrapolation function for displacement unknowns will be used.

can offer lower computational costs. Besides, the finite volume method has the flexibility in terms of generalization to unstructured grids.

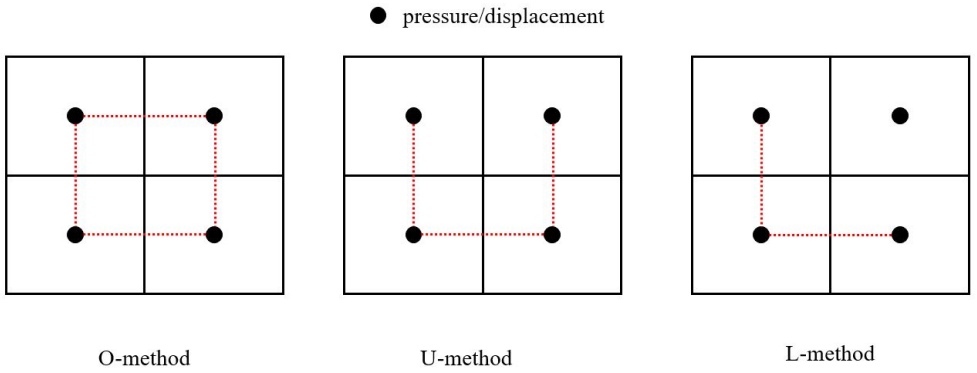


Figure 2.5: *Illustrates multipoint approximation methods utilized to estimate the fluid or solid fluxes based on primary unknowns, i.e. black solid circles.*

Additionally, J.M. Nordbotten extended his work to include fractures and two/multi-phase flow (Nordbotten, 2014b; Doster et al., 2015; Bjørnarå et al., 2016). In a recent study, some issues are addressed in symmetricity and stability of this method (Nilsen et al., 2016). A weakly enforced symmetry of the stress tensor is suggested to overcome this issue (Keilegavlen and Nordbotten, 2015). The shortcomings of this numerical method are still unidentified though this technique seems promising.
It is worth noting that further developments have been made to include more realistic assumptions in poroelasticity. Examples are poroelasticity in fractured media, non-linear poroelasticity, thermo-poroelasticity, etc, (Demirdžić and Martinović, 1993; Lei et al., 2014; Kim et al., 2015; Bryant et al., 2015; Luo et al., 2015; Deb and Jenny, 2016; Garipov et al., 2016). Attempts have also been made to deliver fast and flexible numerical schemes via multigrid modeling and/or unstructured grid, etc (Gaspar et al., 2006b, 2007; Oosterlee and Gaspar, 2008; Nordbotten, 2015b; Nilsen et al., 2016). Additional discussions on applicability of these methods are included in Section 5 (Wang, 2000; Gaspar et al., 2008a; Mikelić and Wheeler, 2013; Cardiff et al., 2016).

2.4 Coupling Schemes for Solid-Fluid Models

In poromechanics, based on strength of coupling between the solid deformation and fluid flow equations, different coupling schemes can be utilized to integrate solid deformation and a fluid model through time. The four main strategies to couple poromechanics are as follows:

2.4.1 Fully Coupled Solid-Fluid Modelling

In a fully coupled strategy, both the fluid flow problem and solid deformation are solved concurrently. The main advantage of this method is that convergence and stability are guaranteed. Nevertheless, when subsurface is being modelled, the domain size of solid should be considered

large enough to make zero displacement boundary condition assumptions realistic, similar to the illustration in Figure 2.6. However, in a fully coupled technique, the same domain should be used to solve the solid and fluid equations. Thus, there are two alternatives when modelling subsurface using the fully coupled scheme: Either to determine the exact values of stress boundary conditions at drainage boundaries of flow which is challenging to determine or enlarging the fluid domain to same size as solid domain which will increase the time- and memory- allocation. However, the results from such a method can be used as a reference to check the reliability of other methods.

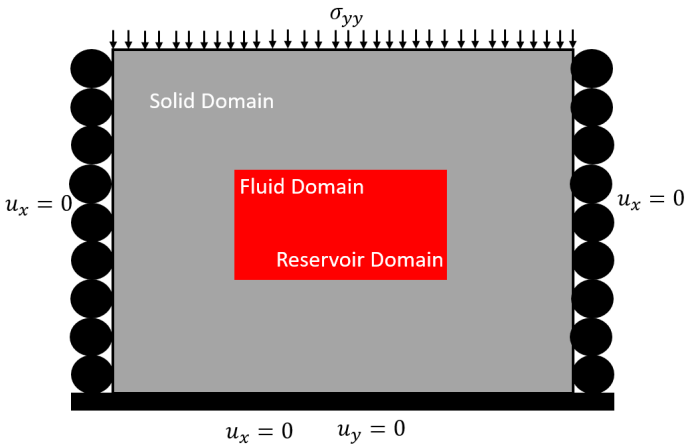


Figure 2.6: *Illustrates the scale of fluid and solid domain used in poromechanics schematically.*

2.4.2 Sequentially/Iteratively Coupled

A sequentially/iteratively coupled scheme considered the same time step for fluid and solid equations. This coupling scheme divides the problem domain into sub-problems of fluid and solid. Either the fluid problem is solved first, i.e. a fixed strain and fixed stress split, or the solid problem is solved first, i.e. drained and undrained split. Afterwards, iteration between the fluid and solid problem is done until converged solutions of solid and fluid unknowns are obtained, see Figure 2.7.
Sequentially/iteratively coupled schemes are subdivided into four common categories as follows (Kim et al., 2009.):

1. *Undrained split* this scheme freezes fluid mass content when solving the solid problem. This method is unconditionally stable though it requires an increased number of iterations as coupling strength between fluid and solid problem increases and is non-convergent in case of a fully incompressible system, i.e. when both solid particles and fluid are incompressible.

2. *Drained split* this scheme assumes that pressure is constant while solving the solid problem. This split is conditionally stable depending on coupling strength between fluid and solid problem and regardless of time step values. It is worth mentioning that this split

can be non-convergent even when it is stable.

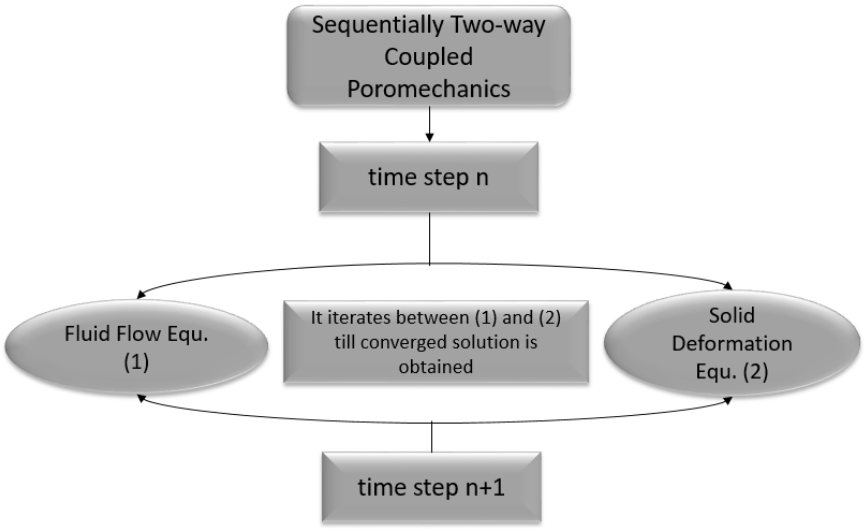


Figure 2.7: *Illustrates sequentially two-way coupled solid deformation and fluid flow in a general framework.*

3. *Fixed-strain split* this scheme assumes that the first derivative of strain with respect to time is constant while solving the fluid problem. This approach is conditionally stable. Similar to drained split, stability relies on coupling strength between fluid and solid problem and is independent of the time step value. This split can also be non-convergent, even when it is stable.

4. *Fixed-stress split* this scheme assumes no variation in total stress when solving the fluid problem, i.e. the first derivative of total stress with respect to time is zero. This technique is unconditionally stable regardless of coupling strength and compressibility of the medium. Thus, this split is as accurate and stable as fully coupled scheme.

The aforesaid techniques are called two-way coupling and Figures 2.8 and 2.9 present an illustration and summary of these iteratively coupling schemes. It is also worth mentioning that an optimum fixed stress scheme has been proposed by Trans et al. (2005) which is claiming to offer faster convergence while preserving same stability and accuracy as fixed-stress split. Besides, Kim et al. (2015) suggested two new splitting schemes, namely undrained-adiabatic and extended fixed-stress split for applications of sequential approach in thermoporoelasticity. The sequentially coupled method similar to fully coupled method fully couples the physics of the problem. In addition, this coupling scheme offers a lower time cost compared to fully coupled scheme when applied to a problem which is consisted of two differently sized sub-domains for fluid and solid sub-problems.

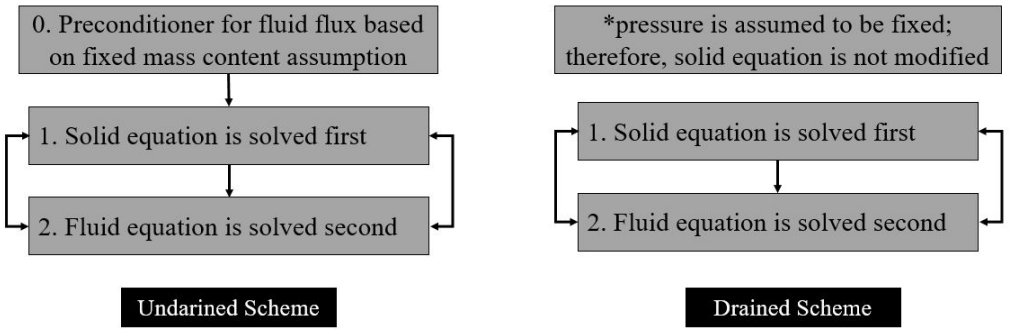


Figure 2.8: *Illustrates sequentially two-way coupled solid deformation and fluid flow in drained and undrained schemes.*

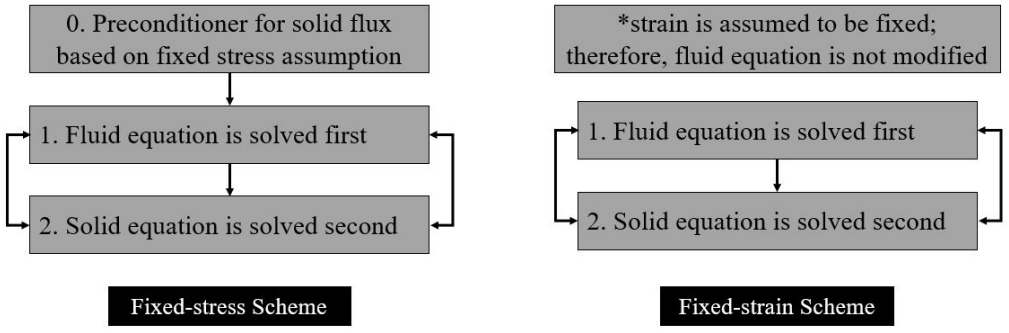


Figure 2.9: *Illustrates sequentially two-way coupled solid deformation and fluid flow in fixed stress and fixed strain schemes.*

2.4.3 Loosely Coupled

The loosely coupled scheme is staggered in time. This means after several time steps of solving fluid problem, coupled solid deformation and fluid flow equations are solved. Properties of both models are updated and again for a certain time interval only the fluid flow problem is solved. It is as if different time steps are chosen to solve the solid and fluid equations, as illustrated by Figure 2.10. This coupling scheme is known as a sub-cycling technique, as well.
Based on how and when to update the mechanical problem, a loosely coupled scheme can be subdivided into three sub-groups (Asadi et al., 2014):

1. *constant time method*
   In this method, regardless of the results from fluid/solid model, mechanical solutions are updated after a constant number of time-steps for fluid problem.

2. *pore pressure method*
   In this method, a fluid pressure, i.e. p_{t_1}, is compared constantly with the fluid pressure calculated at each time-step, i.e. p_{t_2}. The fluid pressure, i.e. p_{t_1}, is the fluid pressure

from the previous time step at which the mechanical model is updated. When the difference between the two pressure values exceeds a threshold value, mechanical variables are updated through coupled calculations.

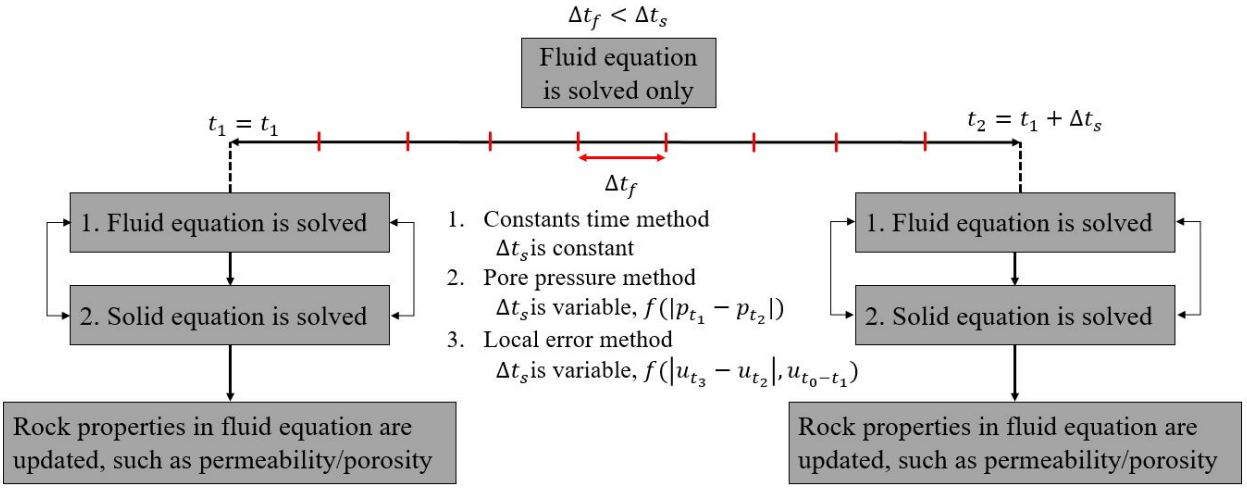


Figure 2.10: *Illustrates loosely coupled solid deformation and fluid flow schemes.*

3. *local error method*
   In this method, a local error is calculated based on displacement values obtained at $2\Delta t$ and Δt. In next step, this local error is compared against the global error, i.e. a physic-based threshold error. As a result of this comparison, time-step at which mechanical properties are updated can become smaller or larger or can be kept constant. This loosely coupled approach is more accurate compared to the other two techniques.

The loosely coupled scheme has the potential to deliver sufficiently accurate estimation of surface subsidence in case of low coupling strength (Asadi et al., 2014). Nonetheless, these techniques might encounter severe mass conservation problems due to large shrinkage in storage properties of fluid if the wrong time step value is chosen (Minkoff et al., 2003.).

2.4.4 Explicitly Coupled

The explicitly coupled scheme refers to a method in which, at each time step, the coupled solid deformation and fluid flow problem are solved in one iteration, i.e. the solution at the first iteration is assumed to be the converged solution. In some research work, the term "explicitly coupled" is defined as a method in which only the solid deformation is affected by the fluid flow and the solid deformation is not influencing the fluid equation, i.e. a one-way coupling of the solid-fluid problem.

It is shown that fixed-stress sequentially split offers exceptional advantages compared to other coupling methods as mentioned above (Minkoff et al., 2003.; Bagheri and Settari, 2005.; Trans

et al., 2005; Jha and Juanes, 2006.; Kim et al., 2009.; Lamb et al., 2010.; Kim et al., 2011b,a; Mikelić and Wheeler, 2013; Asadi et al., 2014).

Coupling Strength of Solid Deformation and Fluid Flow

The coupling strength of the fluid and solid problem in a porous medium is defined by the ratio of bulk stiffness of solid and fluid; see Equation (2.1). In other words, the coupling strength between solid and fluid is high when fluid stiffness is lower than solid stiffness.

$$\begin{cases} \text{low coupling strength} : \tau < 1 \equiv \dfrac{b^2}{K_{dr}} < S \\ \\ \text{high coupling strength} : \tau > 1 \equiv \dfrac{b^2}{K_{dr}} > S \end{cases} \tag{2.1a}$$

$$\tau = \frac{b^2}{K_{dr} S} \tag{2.1b}$$

where S is storativity $[1/Pa]$, b is biot coefficient $[-]$, K_{dr} is drained bulk modulus $[Pa]$, τ is coupling strength $[-]$. S represents the reciprocal of the stiffness in the fluid and b/K_{dr} represents the reciprocal of stiffness for the solid. In other words, the compressibility of the fluid system is determined via the term S while the compressibility of the solid system is determined via the term b/K_{dr}. When the compressibility of the solid system is higher than the compressibility of the fluid system, the coupling strength is high.

In general, in order to achieve the same data structure for fluid and solid unknowns, i.e. p, u_x, u_y, two finite volume methods are chosen to discretize both the fluid and solid equations over space: CC-FV and VV-FV discretization schemes. As mentioned, other methods normally use different data structure. In order to fully couple the physics of the problem and also compare the performance of different coupling schemes, fully coupled and sequentially coupled schemes are utilized. The fixed-strain split among all the other sequentially coupling schemes is chosen. The reason for this decision lies in the fact that it is the first time that the sequentially coupling scheme is applied to finite volume discretization scheme for two dimensional problems. Therefore, there is no quantitative evidence that the fixed-stress split is necessarily the optimised iteratively coupling method for finite volume method for poroelasticity in two dimensional problems. The main novelty in implementing the CC-FV method used in this work is that the solid and the fluid model are sequentially coupled rather than being fully coupled. The main difference between the VV-FV method in this work and other VV-FV methods applied to poroelasticity lies in the fact that the current VV-FV method uses total stress boundary condition which is a mixed-boundary condition of Neumann and Dirichlet boundary conditions. This makes the implementation of the VV-FV method to be a numerically challenging task due to over-determination of unknown systems. Comparing the VV-FV and CC-FV methods in applications for poroelasticity as well as the sequentially coupling and the fully coupling schemes in applications for FVM are other novelties lying in this work.

3 Methodology

This section describes the governing equations for poroelasticity and its limitations, common poroelastic boundary conditions, and sequentially and fully coupled work flows. Besides, discretization schemes used to solve the poroelastic problem are explained.

3.1 Governing Equation for Fluid

Fluid flow equations are derived from mass conservation laws. Conventional equations for the fluid flow in porous media are derived by writing mass conservation equation for fluid considering consolidated and rigid porous media. Nevertheless, another set of equations for fluid flow can be derived which takes into account the mass conservation for solid and fluid in the porous medium assuming porous rock is not consolidated yet (Verruijt, 2016). By writing the mass balance for fluid content over an elementary volume,i.e. V, Equation (3.1) is obtained. Figure 3.1 illustrates the control volume over which mass conservation is written. Equation (3.1) can be re-written as stated in Equation (3.2). Obviously, the rate of change in the mass is expressed in terms of $\partial(\phi\rho_f)/\partial t$.

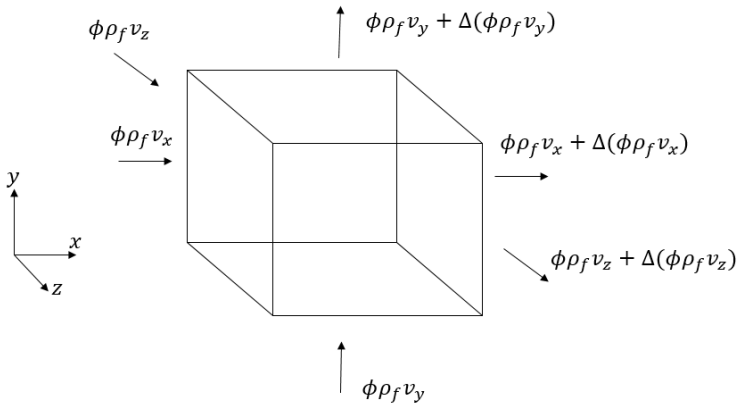


Figure 3.1: *Illustrates mass conservation for fluid over an elementary volume.*

$$\frac{\partial(\phi\rho_f)}{\partial t} + \frac{\partial(\phi\rho_f v_x)}{\partial x} + \frac{\partial(\phi\rho_f v_y)}{\partial y} + \frac{\partial(\phi\rho_f v_z)}{\partial z} = 0 \tag{3.1}$$

$$\frac{\partial(\phi\rho_f)}{\partial t} + \nabla.(\phi\rho_f v) = 0 \tag{3.2}$$

where ϕ is porosity $[-]$, ρ_f is fluid density $[kg/m^3]$ and v_i is fluid velocity in i direction $[m/s]$.

Over the same control volume as shown in Figure 3.1, a mass balance equation for solid can be written as stated in Equation (3.3).

$$\frac{\partial((1-\phi)\rho_s)}{\partial t} + \nabla \cdot ((1-\phi)\rho_s w) = 0 \tag{3.3}$$

where ρ_s is solid density $[kg/m^3]$ and w is solid particle velocity $[m/s]$. Interactions between fluid and solid particles and temperature variations are assumed to be negligible. Thus, by adding Equations (3.2) and (3.3), Equation (3.4), namely the storage equation, is obtained to model the fluid flow in porous media. In this derivation, linear compressibility for fluid and solid is assumed while products of small quantities are neglected. Moreover, Darcy's law is considered as the rheology law to relate the net fluid velocity to the fluid pressure; see Appendix A for details on assumptions and derivations.

$$b\frac{\partial \nabla \cdot u}{\partial t} + S\frac{\partial p}{\partial t} + \nabla \cdot (-\lambda_t \nabla p) = 0; \tag{3.4}$$

where $S = C_f\phi + (b-\phi)C_s$; is storativity $[1/Pa]$, C_f is the fluid compressibility $[1/Pa]$, C_s is the solid compressibility $[1/Pa]$, p is fluid pressure $[Pa]$, b is Biot coefficient $[-]$ which is the term describing the interaction between the fluid flow and the solid deformation, λ_t is fluid mobility $[m^2/(Pa.s)]$ and u is displacement $[m]$.

3.2 Governing Equation for Solid

The governing equation for the solid deformation is derived from Newton's second law, i.e. $\Sigma F = ma$. As illustrated in Figure 3.2, each object is exposed to a number of surface forces, i.e. forces acting on the surface of an object, and body forces, i.e. forces acting throughout the volume of an object, $\sigma.ndS$ and fdV terms in Equation (3.5a), respectively. The surface forces are the external tensile or compressive forces acting on the surface of the domain while the body forces are forces such as gravitational or electromagnetic forces which are directly related to the volume of an object in terms of magnitude.

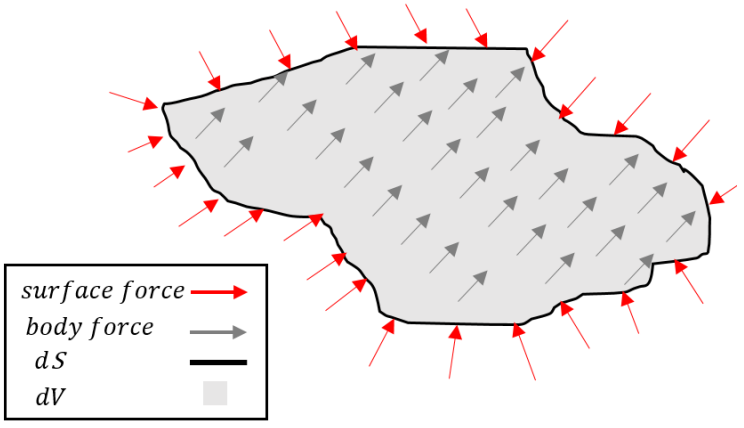


Figure 3.2: *Illustrates schematically surface forces around and body forces inside an arbitrary object.*

The vector sum of these forces may lead to the movement of the object as shown in Equation (3.5a) where a is the acceleration of an object with a density of ρ over the control volume dV. By applying the divergence theorem on Equation (3.5a) and assuming body forces to be merely caused by gravity, Equations (3.5b) and (3.5c) are obtained. Equation (3.5c) can be simplified as Equation (3.5d) considering the relative acceleration of subsurface rocks to be zero.

$$\oiint \sigma \cdot n dS - \iiint f dV = \iiint \rho a dV \tag{3.5a}$$

$$\iiint \nabla \cdot \sigma dV - \iiint f dV = \iiint \rho a dV \tag{3.5b}$$

$$\iiint (\nabla \cdot \sigma dV - f - \rho a) dV = 0 \rightarrow \nabla \cdot \sigma dV - f - \rho a = 0 \tag{3.5c}$$

$$\nabla \cdot \sigma - f = 0 \tag{3.5d}$$

where σ is the external stress $[Pa]$, f is the body force $[Pa]$, ρ is the density of the object $[kg/m^3]$, a is the acceleration of the medium $[m^2/s]$, V represents the volume $[m^3]$, and S represents the surface $[m^2]$. Stress is a second-order tensor; therefore, divergence of stress is a vector, as shown in Equation (3.6). Figure 3.3 illustrates the stress equilibrium in y direction over an elementary volume. In this report, the compressive stress is positively signed which is conforming with the soil mechanics convention and is in contrast to the solid mechanics convention.

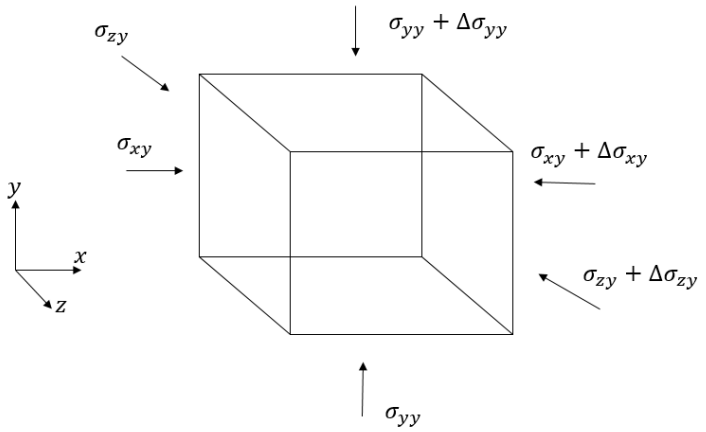


Figure 3.3: *Illustrates stresses on an elementary volume in y direction.*

$$\begin{cases} \nabla\cdot = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix}_{1*3} \\ \sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix}_{3*3} \end{cases} \Rightarrow \nabla \cdot \sigma - f = \begin{bmatrix} \frac{\partial \sigma_{xx}}{\partial x} + \frac{\sigma_{yx}}{\partial y} + \frac{\sigma_{zx}}{\partial z} - f_x \\ \frac{\partial \sigma_{xy}}{\partial x} + \frac{\sigma_{yy}}{\partial y} + \frac{\sigma_{zy}}{\partial z} - f_y \\ \frac{\partial \sigma_{xz}}{\partial x} + \frac{\sigma_{yz}}{\partial y} + \frac{\sigma_{zz}}{\partial z} - f_z \end{bmatrix}^T_{3*1} = 0 \tag{3.6}$$

where σ_{ij} is stress on i plane and in j direction $[Pa]$. The stress field is considered to be a symmetric field, i.e. $\sigma_{xy} = \sigma_{yx}$, $\sigma_{xz} = \sigma_{zx}$, $\sigma_{yz} = \sigma_{zy}$. In addition, total stresses are related to effective stresses by Biot coefficient (b) as stated in Equation (3.7). It is worth mentioning that shear stresses resulting in solid deformation are only shear stresses in the solid body and are not caused by fluid pressure. This fact seems in contrast with Darcy's law where viscous, i.e. shear, forces exist between fluid and solid particles. Polubarinova-Kochina (1977) showed that the shear forces resulted from the interaction between fluid and solid particales are quite negligible compared to the magnitude of shear stresses in the body of solid.

$$\sigma_{xx} = \sigma'_{xx} + bp \qquad \sigma_{xy} = \sigma'_{xy} \qquad \sigma_{xz} = \sigma'_{xz} \tag{3.7a}$$

$$\sigma_{yy} = \sigma'_{yy} + bp \qquad \sigma_{yx} = \sigma'_{yx} \qquad \sigma_{yz} = \sigma'_{yz} \tag{3.7b}$$

$$\sigma_{zz} = \sigma'_{zz} + bp \qquad \sigma_{zx} = \sigma'_{zx} \qquad \sigma_{zy} = \sigma'_{zy} \tag{3.7c}$$

where σ'_{ij} is stress on i plane and in j direction $[Pa]$. Equations (3.6) and (3.7) can be rewritten in two dimensions, as stated in Equations (3.8) and (3.9).

$$\begin{cases} \nabla. = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix}_{1*2} \\ \sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{bmatrix}_{2*2} \end{cases} \Rightarrow \nabla.\sigma = \begin{bmatrix} \frac{\partial \sigma_{xx}}{\partial x} + \frac{\sigma_{yx}}{\partial y} - f_x \\ \frac{\partial \sigma_{xy}}{\partial x} + \frac{\sigma_{yy}}{\partial y} - f_y \end{bmatrix}^T_{2*1} = 0 \tag{3.8}$$

$$\sigma_{xx} = \sigma'_{xx} + bp \qquad \sigma_{xy} = \sigma'_{xy} \tag{3.9a}$$

$$\sigma_{yy} = \sigma'_{yy} + bp \qquad \sigma_{yx} = \sigma'_{yx} \tag{3.9b}$$

The effective stresses are related to the solid deformation via Hook's law- the rheology law used for the solid- which assumes linear elasticity as stated in Equation (3.10), see Appendix A for detailed description of Hook's law. Minus signs in Equation (3.10) indicate an altered sign convention for strain, i.e. extensional strain to be positive, and stress, i.e. compressional stress to be positive. Figure 3.4 shows the sign convention for normal and shear stresses.

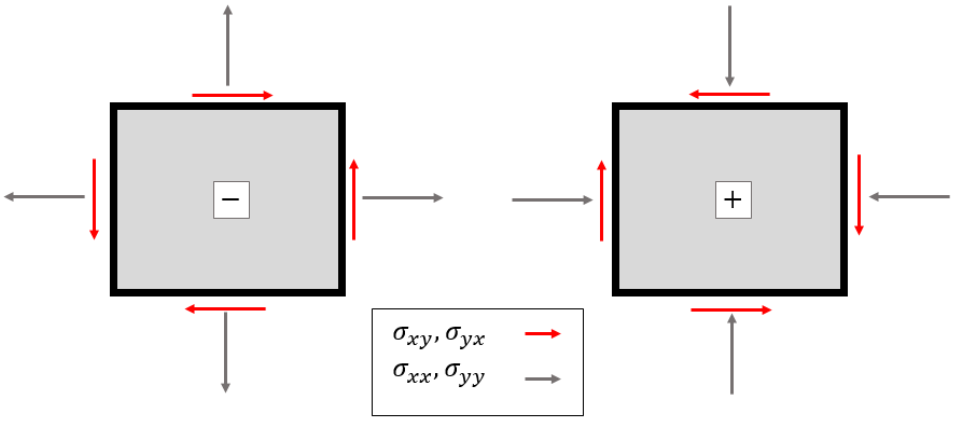


Figure 3.4: *Illustrates the sign convention for normal and shear stresses.*

Linear elasticity is the weakest part of the theory since for example the stress-strain relationship for most porous media under subsequent loading and unloading forces changes indicating non-linear elasticity. Besides, the behaviour of a solid medium becomes plastic if loading exceeds a threshold value. Nevertheless, for small strain/stress changes which is the case for most reservoirs in the large scale, linear-elasticity is a representative assumption.

$$\sigma'_{xx} = -\lambda\frac{\partial u_y}{\partial y} - (\lambda + 2\mu)\frac{\partial u_x}{\partial x} \tag{3.10a}$$

$$\sigma'_{yy} = -\lambda\frac{\partial u_x}{\partial x} - (\lambda + 2\mu)\frac{\partial u_y}{\partial y} \tag{3.10b}$$

$$\sigma'_{yx} = \sigma'_{xy} = -\mu\frac{\partial u_x}{\partial y} - \mu\frac{\partial u_y}{\partial x} \tag{3.10c}$$

where u_i is the displacement in i direction $[m]$, λ is Lame's first parameter showing incomprehensibility of the system $[Pa]$ and μ is Lame's second parameter or shear modulus showing rigidity of the system $[Pa]$. Elastic constants used in Equation (3.10) are constants and do not change with direction since isotropic medium is assumed. In Equation (3.10), Lame's parameters are used to relate effective stresses to displacement components; however, by utilizing the elasticity constant relations, as stated in Table 3.1, these equations can be expressed in terms of any other two elasticity constants. In the case of undrained/pure solid medium, the governing equations are obtained by setting pore pressure related terms to zero, i.e. $p, \frac{\partial p}{\partial x}, \frac{\partial p}{\partial y}$.

Table 3.1: *Illustrates elasticity constants relationships (Lay and Wallace, 1995).*

μ	K	λ	E	ν
$\frac{3(K-\lambda)}{2}$	$\lambda + \frac{2\mu}{3}$	$K - \frac{2\mu}{3}$	$\frac{9K\mu}{3K+\mu}$	$\frac{\lambda}{2(\lambda+\mu)}$
$\lambda\left(\frac{1-2\nu}{2\nu}\right)$	$\mu\left[\frac{2(1+\nu)}{3(1-2\nu)}\right]$	$\frac{2\mu\nu}{1-2\nu}$	$2\mu(1+\nu)$	$\frac{\lambda}{3K-\lambda}$
$3K\left(\frac{1-2\nu}{2+2\nu}\right)$	$\lambda\left(\frac{1+\nu}{3\nu}\right)$	$3K\left(\frac{\nu}{1+\nu}\right)$	$\mu\left(\frac{3\lambda+2\mu}{\lambda+\mu}\right)$	$\frac{3K-2\mu}{2(3K+\mu)}$
$\frac{E}{2(1+\nu)}$	$\frac{E}{3(1-2\nu)}$	$\frac{E\nu}{3(1+\nu)(1-2\nu)}$	$3K(1-2\nu)$	$\frac{3K-E}{6K}$

3.3 Initial and Boundary Conditions

Initial boundary conditions for the fluid are expressed in terms of pressure and for the solid in terms of displacements. Boundary conditions for the fluid are either constant pressure, for instance in a drainage boundary, or in terms of the first derivative of pressure, for example a non-permeable medium surrounding the domain. Figure 3.5 shows several common boundary conditions for a fluid medium and their schematic illustration.

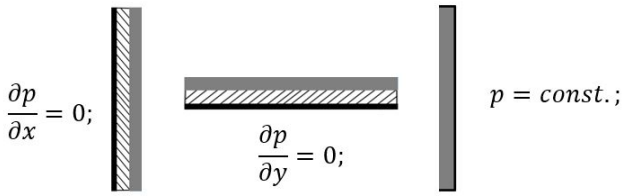


Figure 3.5: *Illustrates common boundary conditions for fluid.*

Boundary conditions for a solid, on the other hand, are either displacement boundary conditions, which is showing rigid rock in case of zero displacement, or stress boundaries or a combination of the two. Stress boundaries are common where the drainage boundary or source term for fluid exists since a change in pore pressure changes the stress conditions (Wang, 2000; Verruijt, 2016). Figure 3.6 summarizes the schematic presentation of several common boundary conditions for the solid.

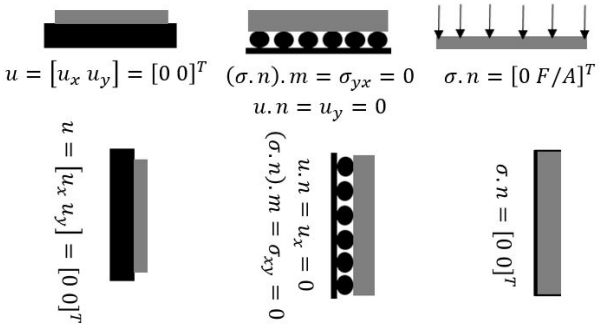


Figure 3.6: *Illustrates common boundary conditions for solid deformation where m is the vector normal to n.*

3.4 Time Integration

In this section, the two coupling techniques, i.e. the fixed strain sequentially coupled split and the fully coupled method, are explained in detail. These two coupling techniques are aimed to integrate the problem through time.

3.4.1 Sequentially Coupled Scheme

As discussed in Section 2.4.2, there are four types of sequentially coupled schemes: fixed strain, fixed stress, drained and undrained split. In all of the aforementioned splits, the iteration between fluid and solid equations is repeated unless a converged solution of displacements and pressure is obtained as shown by Figure 2.7. Among all these methods, fixed-stress is guaranteed to converge and stay stable under various conditions and irrespective of coupling strength while offering the additional advantage of scalable domains for fluid and solid- for the definition of coupling strength see Equation (2.1) (Kim et al., 2011a). As shown in Equation

(2.1), $\tau = b^2/(K_{dr} \times S)$, the coupling strength between solid and fluid equations, i.e. τ, is decreased when the compressibility of fluid medium, i.e. S, increases at a constant value for the compressibility of the solid domain, i.e. b/K_{dr}. However, the fixed strain sequentially coupled method is used in this work to build-up the numerical model. Figure 3.8 shows the work flow of the fixed strain sequentially coupled scheme in the discretized form at the time step $n+1$ for any spatial descritization. Evidently, the work flow, which is illustrated in Figure 3.8, is implemented for n being equal to two to its end value, i.e. $n = 2, 3, .., end$. Displacement and pressure for the first time step, i.e. $n = 1$, are provided by initial conditions. The initial guess for the value of "$\nabla.u$" at the time step $n+1$ is equivalent to its value at the time step n.

$$b\left[\frac{\partial \nabla \cdot u}{\partial t}\right]^n + S\left[\frac{\partial p}{\partial t}\right]^n + \nabla \cdot (-\lambda_t \nabla p^{n+1}) = q; \tag{3.11}$$

where q is a source term for fluid, $[1/s]$ and $\left[\frac{\partial \psi}{\partial t}\right]^{n+1}$ is the derivative of ψ in time, i.e. $\frac{\psi^{n+1} - \psi^n}{\Delta t}$. ψ can be any of the primary unknowns such as p, u_x, or u_y.

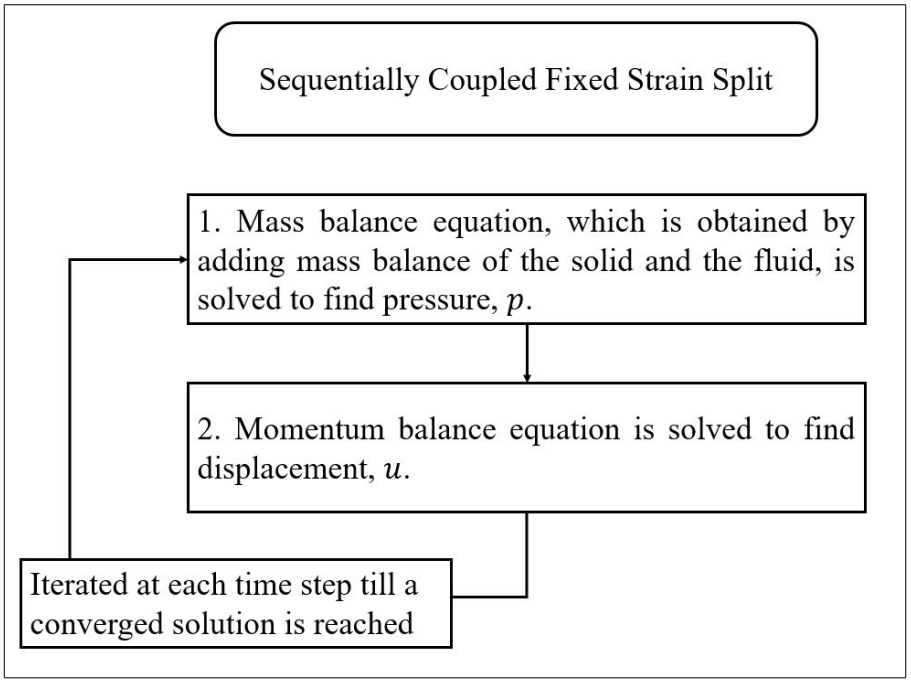


Figure 3.7: *Illustrates fixed strain sequentially two-way coupled scheme in general view.*

Convergence Criteria

Four criteria are used in the iterative sequential scheme to assess if the converged solution is obtained. As stated in Equation (3.12), the 2-norm/Euclidean norm, i.e. ℓ^2, of three primary

unknowns, i.e. u_x, u_y, and p, with respect to the previous iteration and normalized by corresponding values obtained at the first iteration are used as the first three convergence criteria, see Appendix A for definition of 2-norm/Euclidean norm. Normalization removes the impact of the scale on error calculation(s), for instance maximum values for u_x and u_y are normally smaller than 10^{-4} when expressed in *meter*. The last criterion is maximum allowable number of iterations. In Equation (3.12), a threshold value of 10^{-3} or smaller are set for ζ and φ stand for all primary variables in the equations, i.e. p, u_x, and u_y.

$$\frac{\ell^2(\varphi^{\nu} - \varphi^{\nu+1})}{\ell^2(\varphi^{\nu=2} - \varphi^{(\nu=1)\equiv(n)})} \leq \zeta \tag{3.12a}$$

$$\varphi = u_x, u_y, p \tag{3.12b}$$

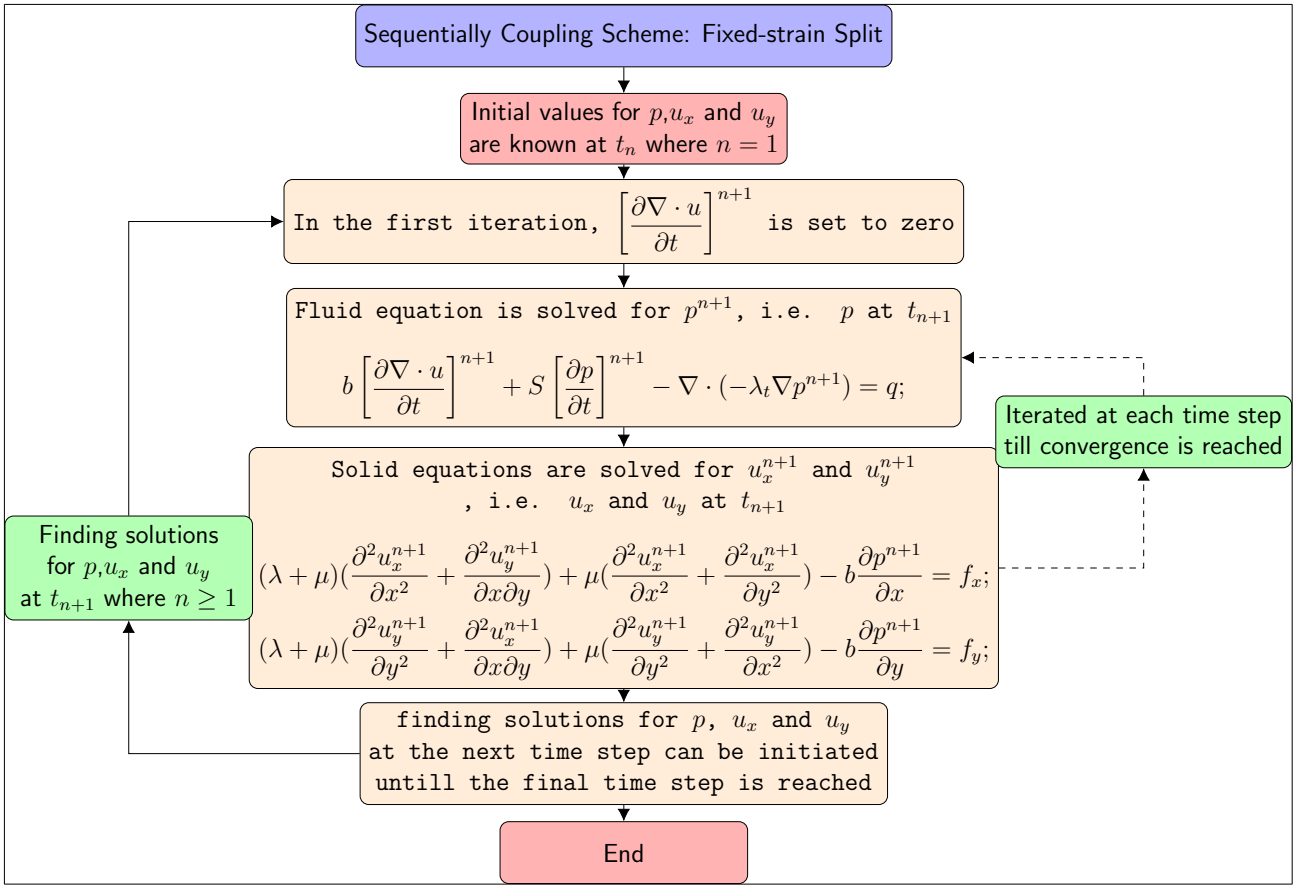


Figure 3.8: *Illustrates fixed strain work flow that is discretized at the time step $n+1$.*

3.4.2 Fully Coupled Scheme

The fully coupled scheme is similar to the sequentially coupled scheme with respect to the physical coupling of the problem. However, the fully coupled scheme finds the solution at

each time step through simultaneously solving the fluid and solid systems rather than iterating between the fluid and solid problems toward the solution at each time step. Therefore, the fully coupled scheme is expected to be a faster and a more accurate scheme compared to sequentially coupled method when problem conditions are the same. Figure 3.9 illustrates the time integration for the fully coupled method at the time step $n+1$.

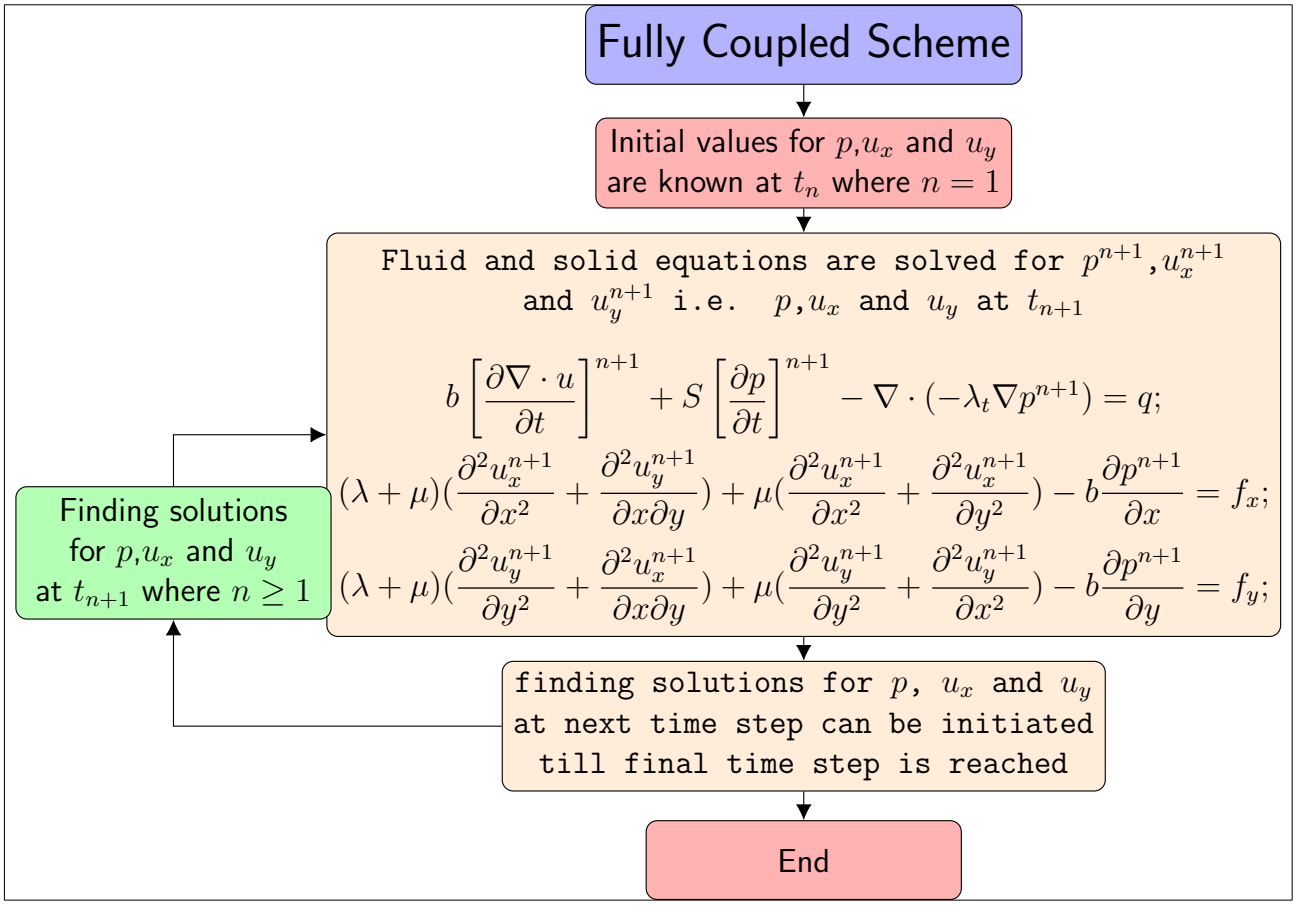


Figure 3.9: *Illustrates fully coupled work flow that is discretized at the time step $n+1$.*

3.5 Discretization Schemes

The finite volume method is used to model both the solid and the fluid domain, i.e. the spatial discretization of Equations (3.4) and (3.8). The finite volume method is the most prevalent method to model the fluid flow (Nordbotten, 2014a). On the other hand, the finite element method is widely used to model rock mechanics. A uniform structure for unknowns in coupled systems makes implementation of unstructured grids and fast solvers to be easier (Gaspar et al., 2008b). This can be achieved by the finite volume method. Besides, the finite volume method uses a lower number of degrees of freedom; therefore, it is less expensive computationally in terms of memory allocation and computational time in comparison with the finite element method while the maximum relative error observed in developed models was in order of 10^{-3}

on coarse grids which is sufficiently accurate. Structured rectangular grids are used to subdivide the problem domain into sub-domains as indicated in Figure 5.1. Equations (3.4) and (3.8) can be written as Equations (3.13) and (3.14) if integrated over the control volume $\Omega_{i,j}$, which is illustrated in Figure 5.1. The overline sign in these equations indicates average values of the overlined property in the control volume $\Omega_{i,j}$.

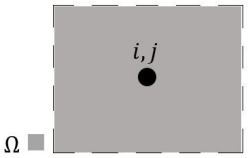


Figure 3.10: *Illustrates cell i, j with its control volume presented by dashed line and its domain illustrated by gray box named Ω.*

$$\int_{\Omega_{i,j}} b\frac{\partial \nabla \cdot u}{\partial t} dV + \int_{\Omega_{i,j}} S\frac{\partial p}{\partial t} dV + \int_{\Omega_{i,j}} \nabla.(-\lambda_t \nabla p) dV = \int_{\Omega_{i,j}} q dV \tag{3.13a}$$

$$b\frac{\partial \nabla \cdot \overline{\mathbf{u}}_{\mathbf{i,j}}}{\partial t} \Delta V_{\Omega_{i,j}} + S\frac{\partial \overline{\mathbf{p}}_{\mathbf{i,j}}}{\partial t} \Delta V_{\Omega_{i,j}} + \oint_{\partial\Omega_{i,j}} (-\lambda_t \nabla p) n.dS = \overline{\mathbf{q}}_{\mathbf{i,j}} \Delta V_{\Omega_{i,j}} \tag{3.13b}$$

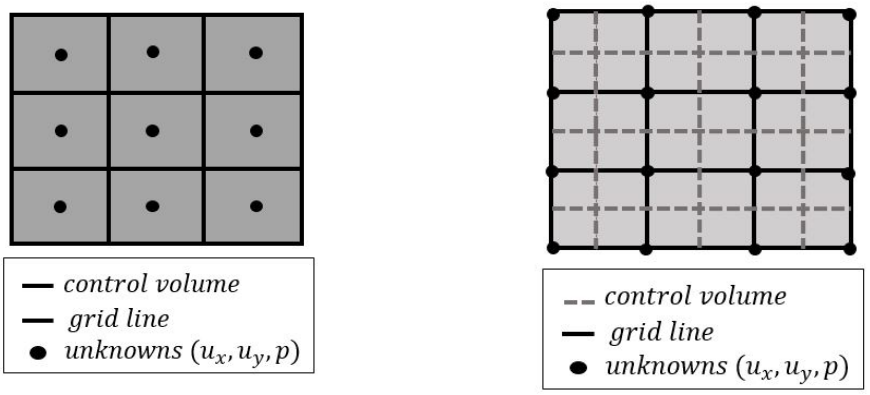
$$b\frac{\partial \nabla . \overline{\mathbf{u}}_{\mathbf{i,j}}}{\partial t} + S\frac{\partial \overline{\mathbf{p}}_{\mathbf{i,j}}}{\partial t} + \frac{1}{\Delta V_{\Omega_{i,j}}} \oint_{\partial\Omega_{i,j}} (-\lambda_t \nabla p) n.dS = \overline{\mathbf{q}}_{\mathbf{i,j}} \tag{3.13c}$$

$$\int_{\Omega_{i,j}} \nabla \cdot \sigma = \int_{\Omega_{i,j}} f dV \tag{3.14a}$$

$$\oint_{\partial\Omega_{i,j}} \sigma n.dS = \overline{\mathbf{f}}_{\mathbf{i,j}} \Delta V_{\Omega_{i,j}} \tag{3.14b}$$

$$\frac{1}{\Delta V_{\Omega_{i,j}}} \oint_{\partial\Omega_{i,j}} \sigma n.dS = \overline{\mathbf{f}}_{\mathbf{i,j}} \tag{3.14c}$$

Two discretization schemes of finite volume method are implemented to test sequentially coupled and fully coupled poroelasticity: cell centred-cell centred finite volume method (CC-FVM) and vertex-centred-vertex-centred finite volume method (VV-FVM). In both methods, primarily unknowns are u_x, u_y, and p. Additionally, unknowns are collocated in the centre of the control volumes, i.e. black dots in Figure 3.11, while secondary unknowns(i.e. stress, strain, velocity, flux) are at the face of control volumes, i.e. dashed lines in Figure 3.11b and solid black line in Figure 3.11a.



(a) *Illustrates CC-FVM.* **(b)** *Illustrates VV-FVM.*

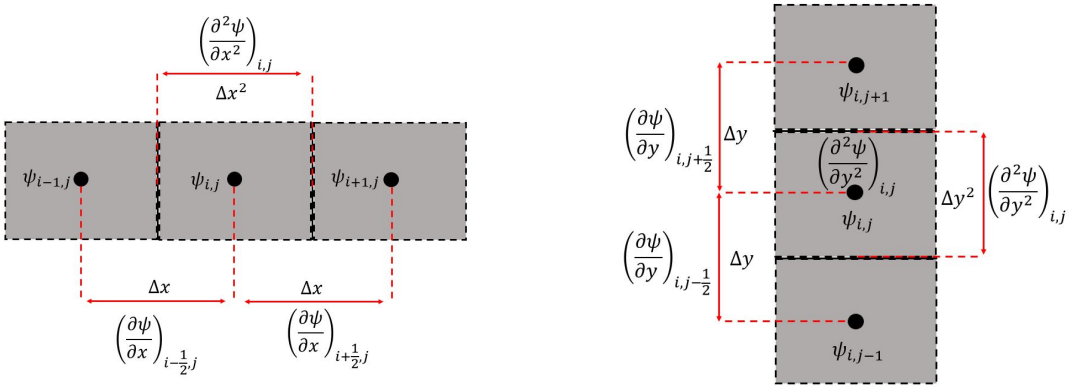
Figure 3.11: *Illustrates CC-finite volume method, (3.11a), versus VV-finite volume method, (3.11b).*

Figure 3.11 shows the difference between the control volume and unknown placements in the two aforementioned discretization scheme.

Central difference approximations with a second order accuracy, i.e. $O(h^2)$, are used to approximate pure derivatives and mix derivatives over space. Moreover, backward Euler method is used for the time derivative discretization. This approximation is first order accurate, i.e. $O(h)$. Equations (3.15)-(3.17) illustrate mathematical relations used in spatial numerical approximations. Figures 3.15-3.13 illustrate schematically how approximations in Equations (3.15)-(3.17) are derived and implemented.

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + O(\Delta x^2) \tag{3.15}$$

$$\frac{\partial^2 \psi}{\partial y^2} = \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} + O(\Delta y^2) \tag{3.16}$$



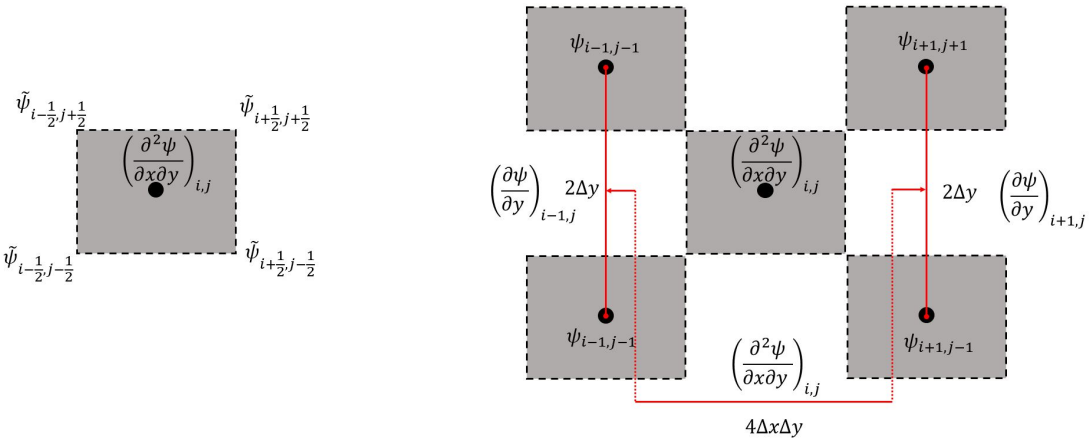
(a) *Illustrates the pure derivative in x-direction.* **(b)** *Illustrates the pure derivative in y-direction.*

Figure 3.12: *Illustrates approximation of the pure derivative in accord with Equations (3.15) and (3.16).*

Figure 3.13a and Equation (3.17a) show approximation of the mixed derivative in cell i, j based on imaginary corner points which can, by an appropriate extrapolation function, be translated to existing unknowns inside the domain. Utilizing a uniform averaging similar to multipoint o-method approximation to estimate the imaginary corner points results in an shown in Figure 3.13b and a relation as stated in Equation (3.17b).

$$\frac{\partial^2 \psi}{\partial y \partial x} = \frac{\overline{\psi}_{i+1/2,j+1/2} - \overline{\psi}_{i-1/2,j+1/2} - \overline{\psi}_{i+1/2,j-1/2} + \overline{\psi}_{i-1/2,j-1/2}}{\Delta y \Delta x} \tag{3.17a}$$

$$\frac{\partial^2 \psi}{\partial y \partial x} = \frac{\psi_{i+1,j+1} - \psi_{i-1,j+1} - \psi_{i+1,j-1} + \psi_{i-1,j-1}}{4\Delta y \Delta x} + O(\Delta x \Delta y) \tag{3.17b}$$



(a) *Illustrates the approximated mixed derivative based on imaginary corner nodes.*

(b) *Illustrates the approximated mixed derivative based on existing unknown nodes.*

Figure 3.13: *Illustrates approximation of the mixed derivative in accord with Equation (3.17).*

Moreover, backward Euler method is used for the time derivative discretization as stated Equation (3.18). This approximation is first order accurate, i.e. $O(h)$.

$$\left[\frac{\partial \psi}{\partial t}\right]^{n+1} = \frac{\psi^{n+1} - \psi^n}{\Delta t} + O(\Delta t) \tag{3.18}$$

Overall, a first order accuracy with respect to time is expected when space discretization is small enough, i.e. $(\Delta x/\Delta y) \to \epsilon$, and second order accurate solutions over space are expected when the time step is sufficiently small, i.e. $(\Delta t) \to \epsilon$, i.e. the time step error would become negligible, as indicated in Figure 3.14.

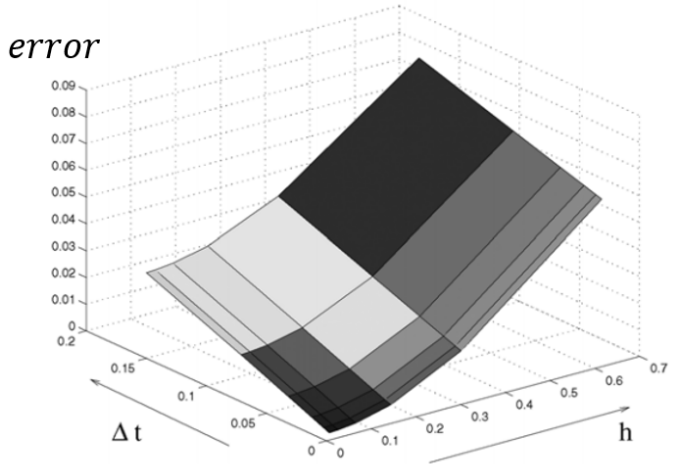


Figure 3.14: *Illustrates a schematic of error variation with respect to the time step and grid cell size when discretization over space and time are differently accurate (Formaggia et al., 2012).*

It is worth mentioning that numerical methods for fluid flow in porous media are abundant. Though there are works on numerical modelling of coupled solid and fluid modelling, there are a few works about pure/porous solid deformation in the petroleum engineering & geosciences section. One of the novelties in this work is to use FVM in modelling the solid while most numerical models use FEM to model the solid.

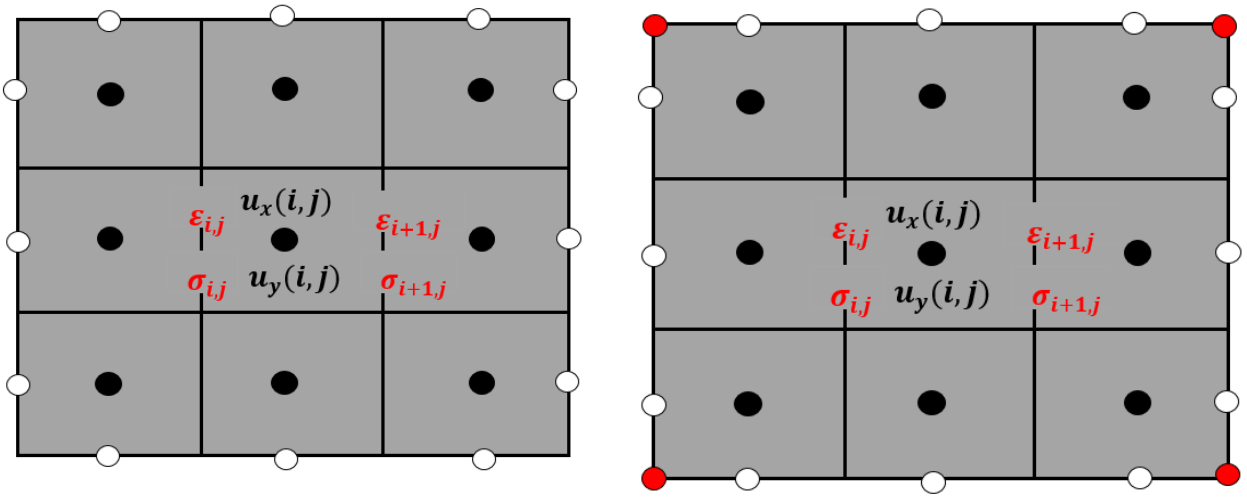
3.5.1 Cell Centred-Cell Centred FVM

The schematic of the cell-centred descretization scheme is as shown in Figure 3.11a, where primary unknowns, i.e. u_x, u_y, p, are cell centred. Nevertheless, to model the solid, a modified finite volume scheme is considered. A number of nodes at the boundaries of the problem domain, i.e. $\partial\Omega$, are added to remove one disadvantage of CC-FVM compared to FEM which is easy implementation of boundary conditions. The added nodes are the white nodes in Figure 3.15a. Equation (3.19) shows a generalized form of the coefficient matrix and unknown and solution vectors for CC-FVM developed.

$$\begin{pmatrix} A_{xx} & & A_{yx} \\ BC & \text{for} & (\nabla.\sigma)_x \\ A_{yy} & & A_{xy} \\ BC & \text{for} & (\nabla.\sigma)_y \end{pmatrix} \times \begin{pmatrix} u_x & \text{for black nodes} \\ u_x & \text{for white nodes} \\ u_y & \text{for black nodes} \\ u_y & \text{for white nodes} \end{pmatrix} = \begin{pmatrix} f_x \\ BCs \\ f_y \\ BCs \end{pmatrix} \tag{3.19}$$

In addition, an approximation of the mixed derivative is implemented as indicated by Figure 3.13a in which bi-linear extrapolation rules are used to estimate $\tilde{\psi}$ values. These extrapolation rules are similar to shape functions in FEM. Since linear elasticity is one of the basic assumptions in the derivation of constitutive equations, higher order extrapolation rules are not employed. This discretization scheme shows adequate efficiency in the absence of body forces or pressure

gradients. However, in the presence of body forces or pressure gradients, this technique might face numerical problems near boundaries. Three sources for this observation are speculated. One source is hypothesized to be the bi-linear extrapolation rule(s) used for imaginary corner points to approximate the mixed-derivative term, i.e. $\tilde{\psi}$ points in Equation (3.17a). To inspect the impact of the extrapolation rules, two different bi-linear extrapolation rules are applied in CC-FVM. The results show that CC-FVM is quite sensitive to the choice of extrapolation rule. Another source was guessed to be over-determined corner points; better said, at corner points, like any other point, there are only two primary unknowns, i.e. u_x, u_y, while two sets of boundary conditions are valid at this point, i.e. four equations which reduce to three independent equations through symmetric assumption for stress tensor- shear forces at both sides of a corner point are equal. To test this hypothesis, red nodes at corner points are added as illustrated in Figure 3.15b. Two unique and conclusive equations are set to be solved for each of these points. Equation (3.20) shows the general form of the coefficient matrix and unknowns and solution vectors for this modified CC-FVM. The added nodes, introduced more error into the system. Therefore, it was concluded that the combination of white and black nodes are the optimum number of nodes needed to model the solid boundary condition.



(a) *Illustrates white boundary nodes added for implementation of BCs.*

(b) *Illustrates red corner nodes added to remove over-determined effects.*

Figure 3.15: *Illustrates nodes added for flexible implementation of BCs and red nodes added to remove over-determine effects at the corner points.*

The last element introducing error to this system is the sequential method applied to couple fluid and solid equations. Estimated values at white nodes in Figure 3.15b are removed from

the solution matrix and only the black nodes are used in the coupled system.

$$\begin{pmatrix} A_{xx} & & A_{yx} \\ BC & \text{for} & (\nabla.\sigma)_x \\ BC & \text{for} & \nabla.\sigma \\ A_{yy} & & A_{xy} \\ BC & \text{for} & (\nabla.\sigma)_y \\ BC & \text{for} & \nabla.\sigma \end{pmatrix} \times \begin{pmatrix} u_x & \text{for black nodes} \\ u_x & \text{for white nodes} \\ u_x & \text{for red nodes} \\ u_y & \text{for black nodes} \\ u_y & \text{for white nodes} \\ u_y & \text{for red nodes} \end{pmatrix} = \begin{pmatrix} f_x \\ BCs \\ BCs \text{ in reduced form} \\ f_y \\ BCs \\ BCs \text{ in reduced form} \end{pmatrix} \tag{3.20}$$

3.5.2 Vertex Centred-Vertex Centred FVM

The discretization of the VV-FVM is as shown in Figure 3.11b. Since in this method all points needed to estimate the mixed-derivative are either fully-inside the domain or are immediate unknowns bi-linear extrapolation rules work efficiently. Besides, all boundary points are immediately solved without any approximations being made. Therefore, VV-FVM is expected to offer more accurate solutions with a higher flexibility in terms of implementing the boundary conditions. Besides, a smaller number of unknowns is used to solve the same problem. Therefore, this method is expected to be of more efficient in terms of computational time.

4 Numerical Results

The fully coupled and sequentially coupled finite volume models for poroelasticity are verified and compared in this section. After the primary investigation, the fully coupled method is preferred; therefore, it is used to be benchmarked versus the Mandel Test Case, to be tested for sensitivity analysis and to find the solutions for a practical problem.

4.1 Verification

In order to verify the model, synthetic test cases are utilized. Error and convergence study is applied to the models in order to determine the order of accuracy and convergence of the methods.
The term "synthetic test case" refers to the test case in which a set of exact solutions is considered for pressure and displacements. At the next stage, initial and boundary conditions as well as source terms for the problem are defined such that these solutions become the exact/analytical solutions of the Biot's consolidation equations. Thus, the numerical solutions are expected to be an approximation of this set of exact solutions when these set of initial and boundary conditions as well as source terms is applied to the Biot's consolidation equations. This method is adopted from the work carried out by Luo et al. (2015).
Consequently, the problem description for the synthetic test cases in this report is as follows: Exact solutions, initial and boundary conditions as well as source terms. The initial and boundary conditions as well as source terms are calculated via fundamental equations described in Section 3 as a function of exact solutions.
During the master work, different synthetic test cases are used to test the developed models. Results from two exemplary synthetic test cases are presented and described to verify and compare the sequentially coupled and the fully coupled model.

4.1.1 Synthetic Test Case I

The set of exact/analytical solutions for the synthetic test case I are firstly introduced. The assumptions and facts considered in order to define this set of solutions are discussed as well. At the next stage, the fundamental equations described in Section 3 are used to determine the initial and boundary conditions as well as the source terms as a function of these exact solutions. Thus, the synthetic test case description is ordered as follows: Exact solutions description, input parameters, initial and boundary conditions, source terms.

Exact Solution Description

Displacement functions, in the most complex case, mimic wave propagation functions while the pressure solution in the most nonlinear case, in the large scale, has a parabolic shape with a

downward concavity. The order of displacement values, expressed in meter, is approximately 10^{-5}. Bubble point pressure, for oil reservoirs, is normally around 10^5 to 10^7, $[Pa]$. Consequently, to make the single phase assumption be valid and appreciate the fact that reservoir fluid is volatile oil, the reservoir pressure is considered to be initially 10^6. Therefore, in the first step to build the synthetic test case, exact solution functions, which are equal to Table 4.1, are chosen to describe displacement and pressure as functions of time and spatial domains. The derivative of pressure function with respect to time is constant to validate the slightly compressible assumption.

Table 4.1: *Illustrates exact solutions for displacement and pressure.*

Variable	Unit	Exact Solution
$u_x(x,y,t)$	$[m]$	$10^{-5} \times \sin(\pi x/L_x) \times \sin(\pi y/L_y) \times \cos(\pi t\sqrt{1/L_x + 1/L_y})$
$u_y(x,y,t)$	$[m]$	$-10^{-5} \times \sinh(\pi(L_x - x)/L_x) \times \sin(\pi y/L_y) \times \cos(\pi t\sqrt{1/L_x + 1/L_y})$
$p(x,y,t)$	$[Pa]$	$10^6 \times (-(L_x - x)^2 + L_x^2)/L_x^2 \times (1 - t/t_c)$

Input Parameters

For each poroelastic problem, a characteristic time can be defined. The characteristic time indicates the time frame during which consolidation occurs (Verruijt, 2016). In other words, at times beyond the characteristic time, the fluid and solid equations can be solved in decoupled way; and therefore, there would be no need to solve the two equations simultaneously at each time step. The characteristic time is defined as shown in Equation (4.1a).

$$t_c = \frac{L_x L_y}{c_v} \tag{4.1a}$$

$$c_v = \frac{\lambda_t K_{1D}}{S K_{1D} + b^2} \tag{4.1b}$$

$$K_{1D} = \lambda + 2\mu \tag{4.1c}$$

where t_c is the characteristic time, $[s]$, c_v is the consolidation coefficient, $[m^2/s]$, and K_{1D} is the uniaxial bulk modulus, $[Pa]$. The values in Table 4.2 describe a slightly compressible system which is commonly the case in reservoir engineering fields. The system consists of a slightly compressible fluid, for example volatile oil, and a slightly compressible grain system, for example unconsolidated sandstone reservoir. However, in order to reduce the computational time, the reservoir size is chosen to be smaller than realistic values which are in order of hundreds to thousands of kilometre. More details on parameter selection can be found in Appendix C.

Table 4.2: *Illustrates values for model constants in reservoir layer.*

Input Parameter	Unit	Value
λ	$[Pa]$	3.05×10^7
μ	$[Pa]$	1.63×10^7
λ_t	$[m^2/(Pa.s)]$	2.43×10^{-5}
S	$[1/Pa]$	2.80×10^{-7}
c_v	$[m^2/s]$	≈ 86
$L_{x,y}$	$[m]$	100
t_c	$[s]$	≈ 115
b	$[-]$	1

Initial and Boundary Conditions

Table 4.4 and 4.3 represent the initial and boundary conditions for the synthetic test case I, respectively. The initial conditions reported in Table 4.4 are obtained by setting the time value to zero for the set of exact solutions reported in Table 4.1. As shown in Figure 4.1, it is assumed that the reservoir rock is bounded by rigid and non-permeable rocks at bottom and right side of the reservoir layer and a cap rock at the top. At the top and bottom left side of the problem domain a tensile and compressive normal stress, respectively, and upward shear stress are used. At the top boundary compressive normal stress is considered. For fluid flow, there is no flow at right, bottom and top sides. At the left side; however, zero pressure boundary condition is assumed. Dirichlet boundary conditions are determined such that these boundary conditions are valid, see Table 4.3.

Table 4.3: *Illustrates boundary conditions applied in the synthetic test case for definition of the boundaries see Figure 4.1.*

Boundary Side	Boundary Conditions
Γ_T	$u = \begin{bmatrix} c_1(x,t) & c_2(x,t) \end{bmatrix}$
Γ_R	$u = \begin{bmatrix} c_1'(y,t) & c_2'(y,t) \end{bmatrix}$
Γ_L	$u = \begin{bmatrix} 0 & 0 \end{bmatrix}$
Γ_B	$u = \begin{bmatrix} 0 & 0 \end{bmatrix}$

Table 4.4: *Illustrates initial conditions applied to the synthetic test case.*

Variable	Unit	Initial Condition
$u_x(x,y,0)$	$[m]$	$10^{-5} \times \sin(\pi x/L_x) \times \sin(\pi y/L_y)$
$u_y(x,y,0)$	$[m]$	$-10^{-5} \times \sinh(\pi(L_x - x)/L_x) \times \sin(\pi y/L_y)$
$p(x,y,0)$	$[Pa]$	$10^6 \times (-(L_x - x)^2 + L_x^2)/L_x^2$

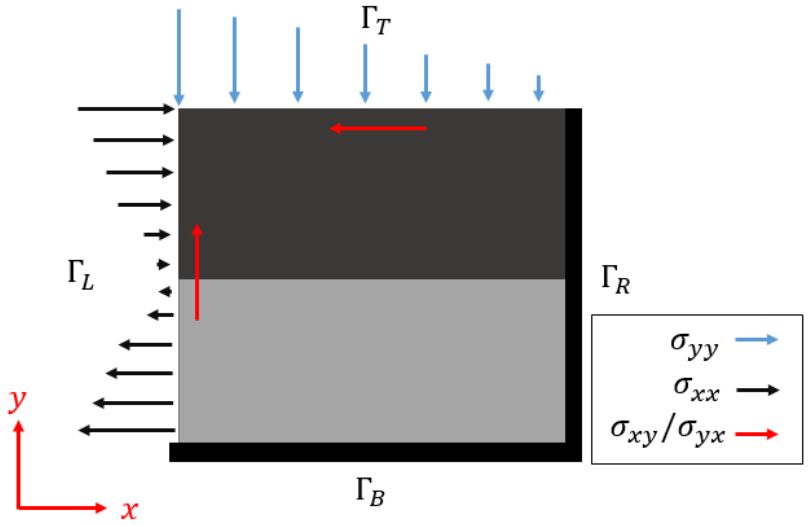


Figure 4.1: *Illustrates boundary conditions for the synthetic test case.*

Source Terms

Source terms inside the domain are obtained from the equilibrium and storage equations (Biot's consolidation equations) as a function of the set of exact solutions and input parameters reported in Tables 4.1 and 4.2, subsequently. The resulting source terms are as reported Table 4.5. The parameters A_1 to $A_4^{''}$ stand for constant values.

Table 4.5: *Illustrates source terms for equilibrium and continuity equations.*

Variable	Unit	Source Term
f_x	$[Pa/m]$	$-A_1 \sin(x)\sin(y)\cos(t) + A_2 \cos(y)\cosh(x)\cos(t) + b\frac{\partial p}{\partial x}(x,t)$
f_y	$[Pa/m]$	$-A_1^{'} \cos(\pi x/L_x)\cos(\pi y/L_y)$
q	$[1/s]$	$A_1^{''}xt + A_2^{''}t + A_3^{''}x + A_4^{''} + b\frac{\partial \nabla.u}{\partial t}$

Interpretation of the Synthetic Results

The exact and numerical solutions of pressure and displacement are presented on a 16×16 grid. Relative absolute error maps are generated by assuming Dirichlet boundary conditions for each property. The two fully coupled and sequentially coupled methods are run in two modes:

1. decoupled, i.e. $b = 0$: no effect from the solid equations goes into the fluid equation and the fluid equation has no impact on solid equation results.
2. coupled, i.e. $b = 1$: the solid equation influences the fluid equation in terms of $b\partial\nabla.u/\partial t$ and the fluid equation impacts the solid equation via pressure gradient terms, i.e. $\partial p/\partial x$

and $\partial p/\partial y$.

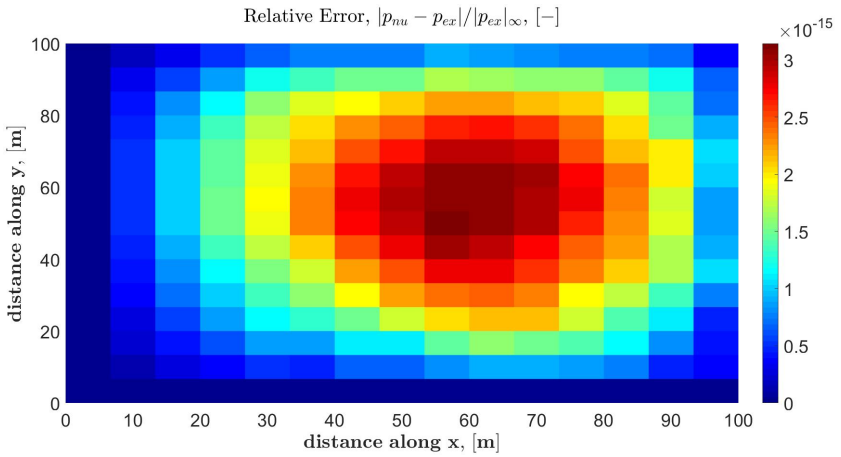
The exact and numerical solutions are obtained at time equal to $0.5t_c$. Therefore, the solutions can be simplified as stated in Table 4.6.

Table 4.6: *Illustrates exact solutions to displacement and pressure variables at time equal to* $0.5t_c$.

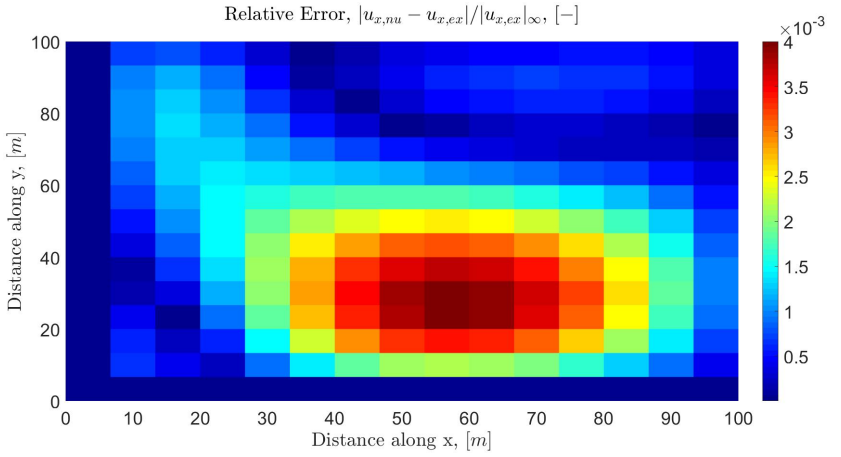
Variable	Unit	Exact Solution
$u_x(x,y,t)$	$[m]$	$0.5536 \times 10^{-5} \times \sin(\pi x/L_x) \times \sin(\pi y/L_y)$
$u_y(x,y,t)$	$[m]$	$0.5536 \times 10^{-5} \times \sinh(\pi(L_x - x)/L_x) \times \sin(\pi y/L_y)$
$p(x,y,t)$	$[Pa]$	$5 \times 10^5 \times (-(L_x - x)^2 + L_x^2)/L_x^2$

The error of the two methods in decoupled mode are presented in Figures 4.2 and 4.3. As observed in these figures, the order of accuracy in both methods is comparable. Since Dirichlet boundary conditions are assumed, it is expected that the error at the boundaries goes to zero and in the middle of the domain the maximum error would be observed, as it is to see in Figures 4.2 and 4.3a. However, for the solid cell-centred finite volume discretization, this is not the case, see Figures 4.3b and 4.3c. The reason for this observation is expected to be the linear extrapolation functions used to estimate the corner point unknowns based on inner cell-centred nodes. Since error accumulation in the boundary occurs only at the left side, where the u_y solution is highly nonlinear.

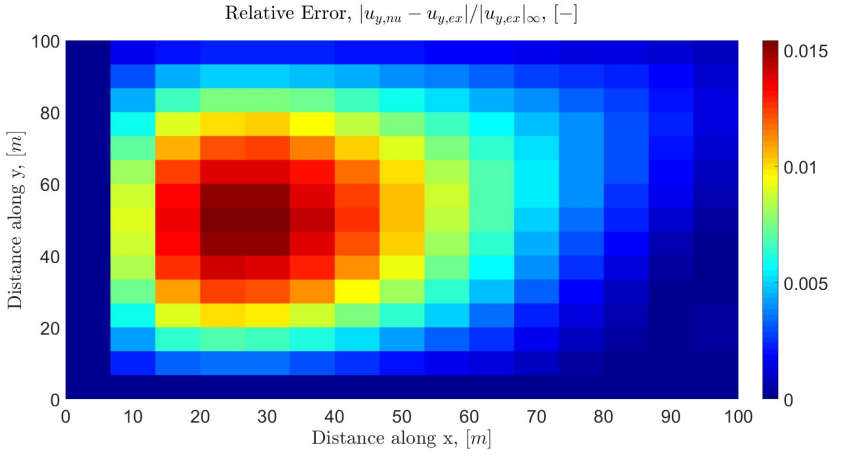
Error Maps in Decoupled Mode: Fully Coupled Method



(a) *Illustrates error values for pressure on* 16×16 *grid.*



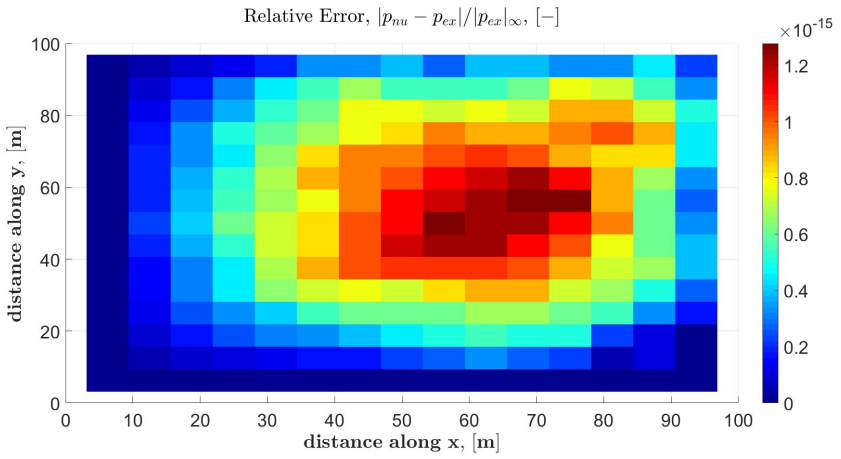
(b) *Illustrates error values for displacement in x-direction on* 16×16 *grid.*



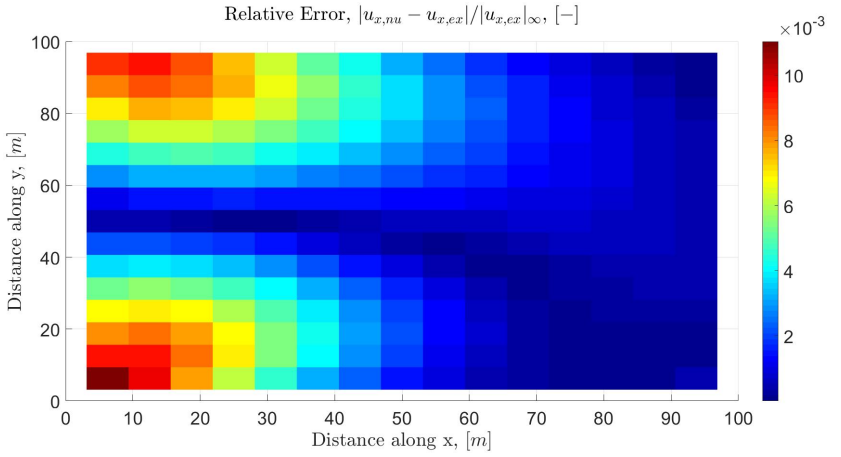
(c) *Illustrates error values for displacement in y-direction on* 16×16 *grid.*

Figure 4.2: *Illustrates relative error map values for pressure and displacements in the synthetic test case by fully coupled method run in decoupled mode.*

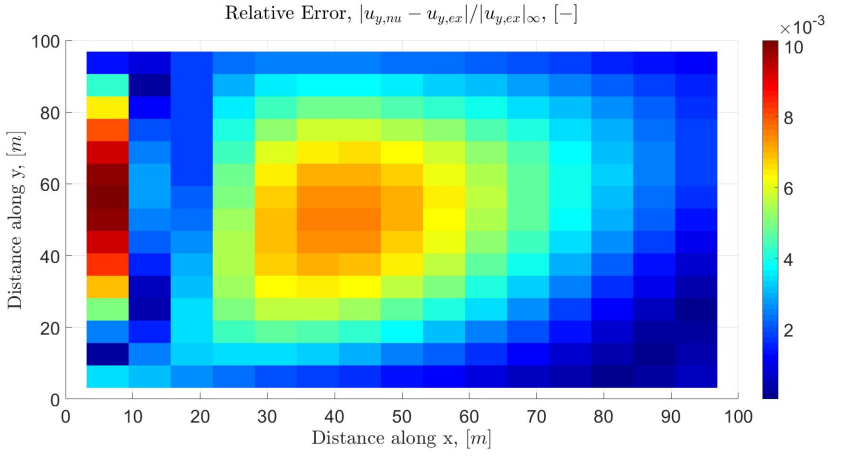
Error Maps in Decoupled Mode: Sequentially Coupled Method



(a) *Illustrates error values for pressure on* 16×16 *grid.*



(b) *Illustrates error values for displacement in x-direction on* 16×16 *grid.*



(c) *Illustrates error values for displacement in y-direction on* 16×16 *grid.*

Figure 4.3: *Illustrates relative error map values for pressure and displacements in the synthetic test case by the sequentially coupled method run in decoupled mode.*

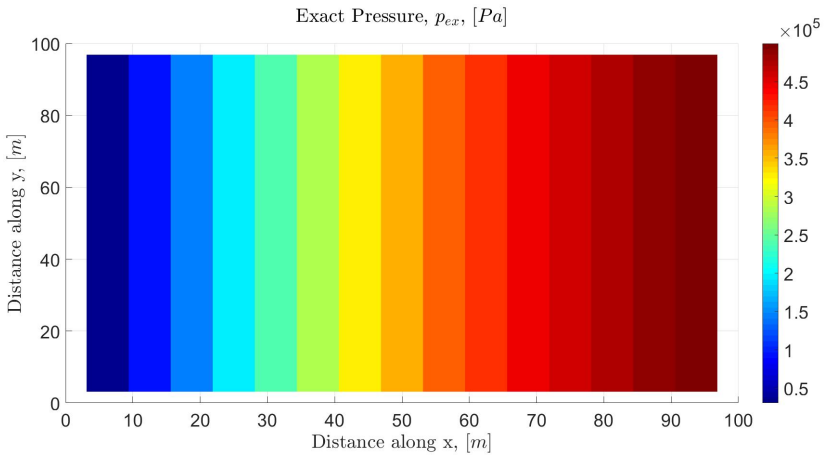
Sequentially Coupled Method

Figures 4.4, 4.5, and 4.6 illustrate coupled solutions of the sequentially coupled method. The exact and numerical values are accurate when the numerical solver is run in the uncoupled mode; however, when b is equal to one the error from the pressure solver offshoots the displacement solution, due to the difference in the scale of the solution. To prevent this, some parameters have been normalized. The solutions from this method are sufficiently accurate, i.e. the normalized relative error are in order of 10^{-3}, but further improvement of the method is expected through normalization of parameters.

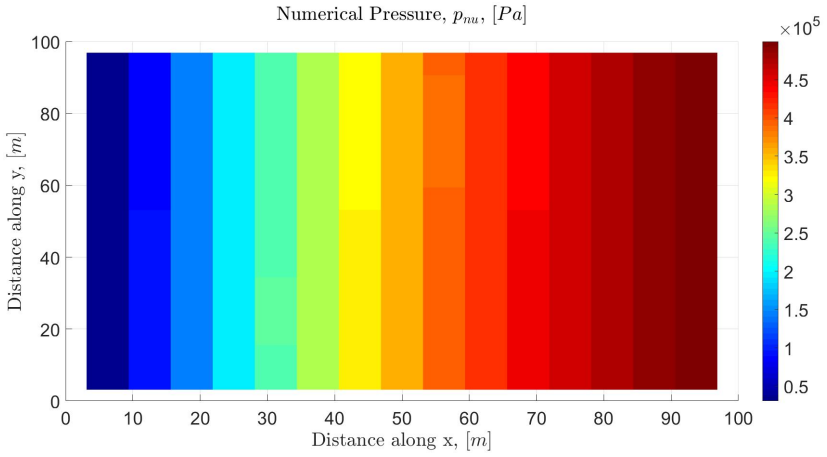
Fully Coupled Method

As illustrated by Figures 4.7c, 4.8c, and 4.9c, the error at the boundaries is zero since Dirichlet boundary conditions are applied. In the centre of the error maps and where the exact solution deviates most from the linear assumption, maximum values for the error are observed. In numerical approximations, linear extrapolation is used to approximate the derivatives between the grid points; therefore, the higher the non-linearity of a function, the higher the error in the approximation will be.

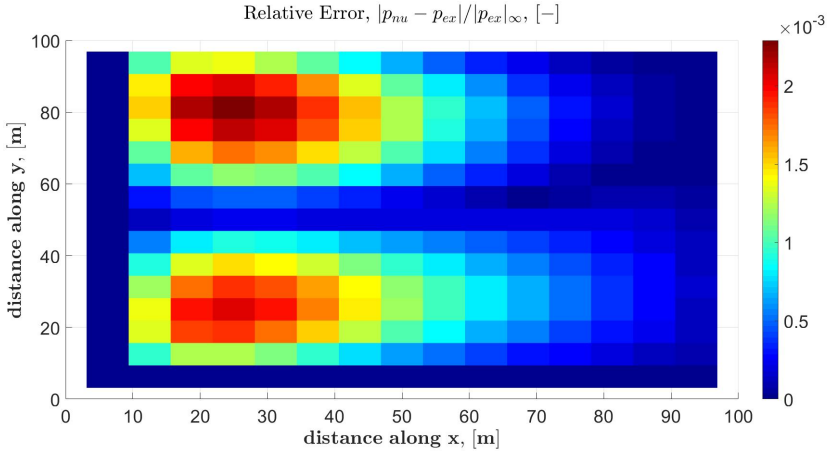
Pressure: Sequentially Coupled Method



(a) *Exact solution for the pressure on* 16×16 *grid.*



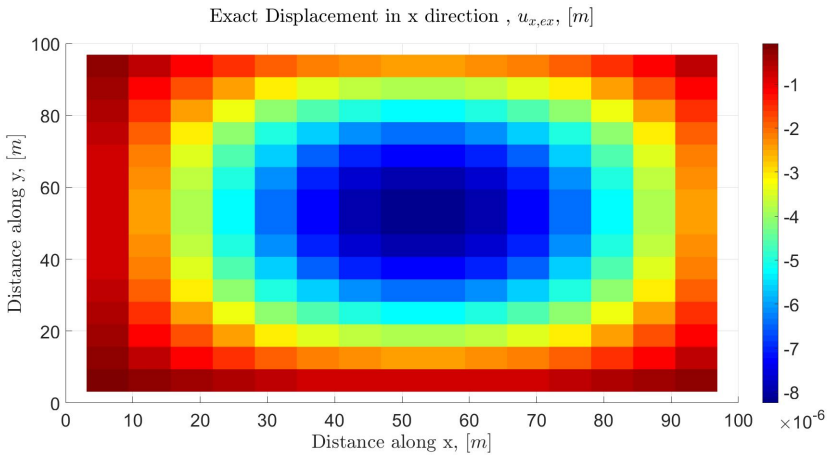
(b) *Numerical solution for the pressure on* 16×16 *grid.*



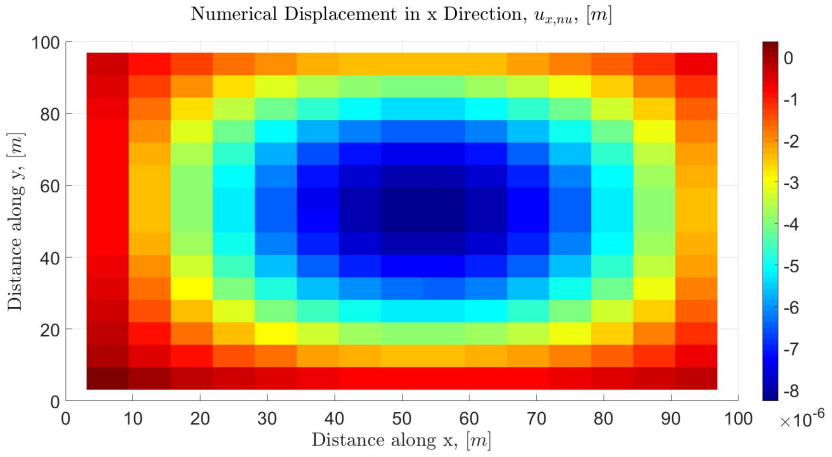
(c) *Normalized error values for the pressure on* 16×16 *grid.*

Figure 4.4: *Illustrates exact, numerical and error map values for the pressure in the synthetic test case by the sequentially coupled method.*

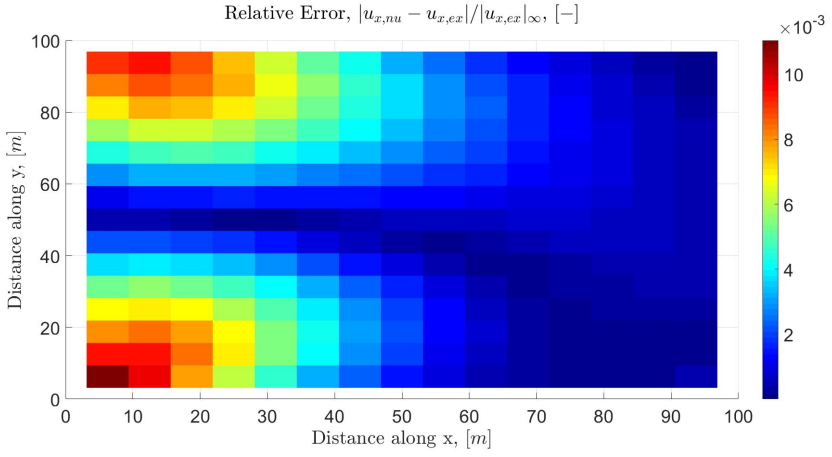
Displacement in x-direction: Sequentially Coupled Method



(a) *Exact solution for the displacement in the x-direction on* 16×16 *grid.*



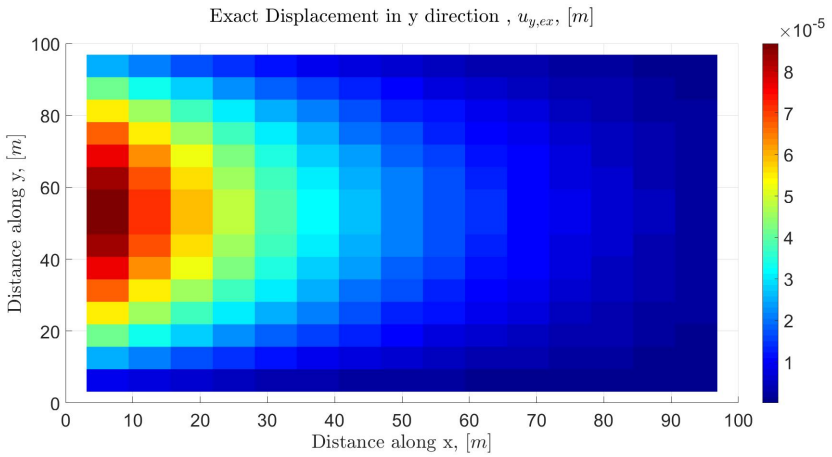
(b) *Numerical solution for the displacement in the x-direction on* 16×16 *grid.*



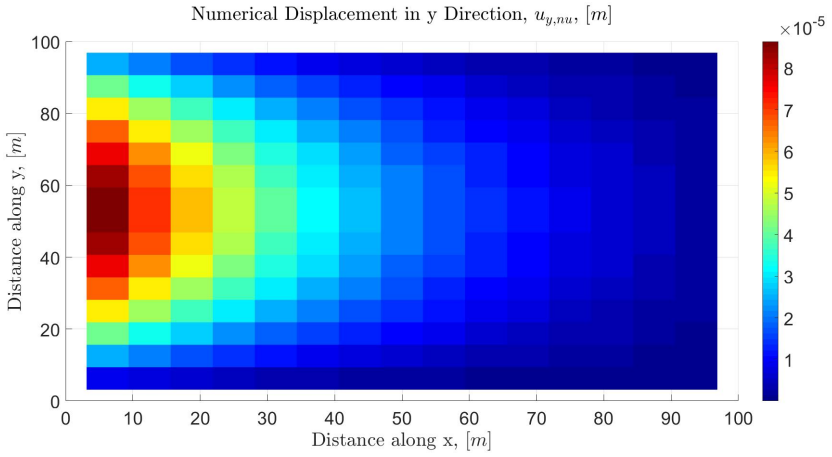
(c) *Normalized error values for the displacement in the x-direction on* 16×16 *grid.*

Figure 4.5: *Illustrates exact, numerical and error map values for the displacement in the x-direction in the synthetic test case by fully coupled method.*

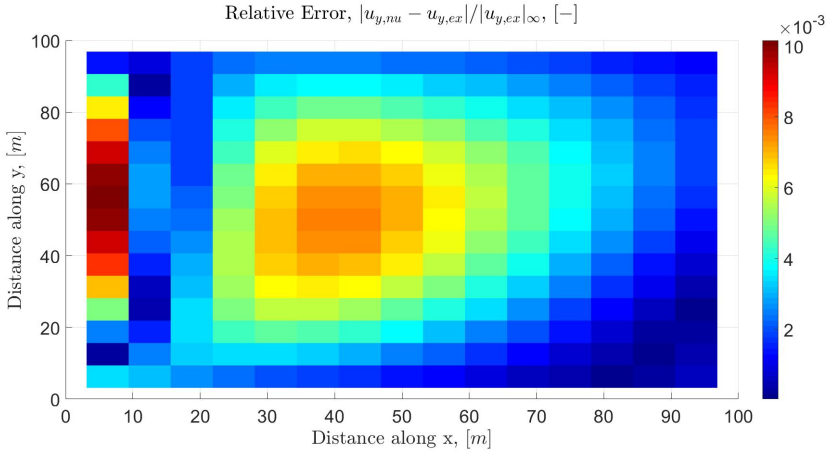
Displacement in y-direction: Sequentially Coupled Method



(a) *Exact solution for the displacement in the y-direction on* 16×16 *grid.*



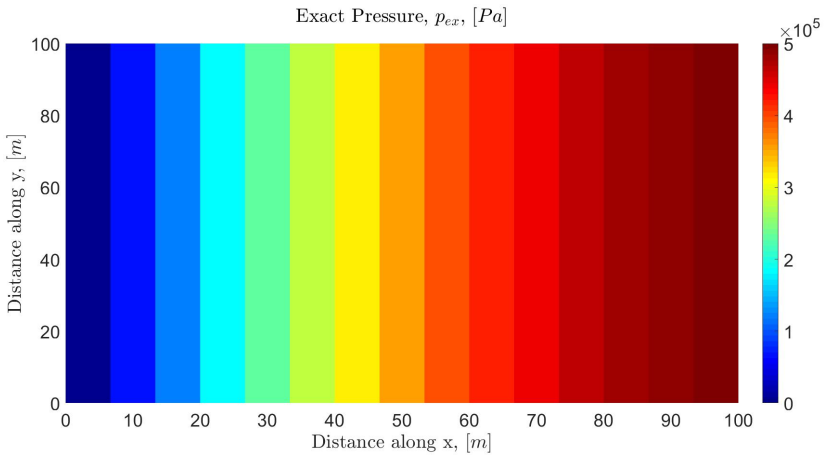
(b) *Numerical solution for the displacement in the y-direction on* 16×16 *grid.*



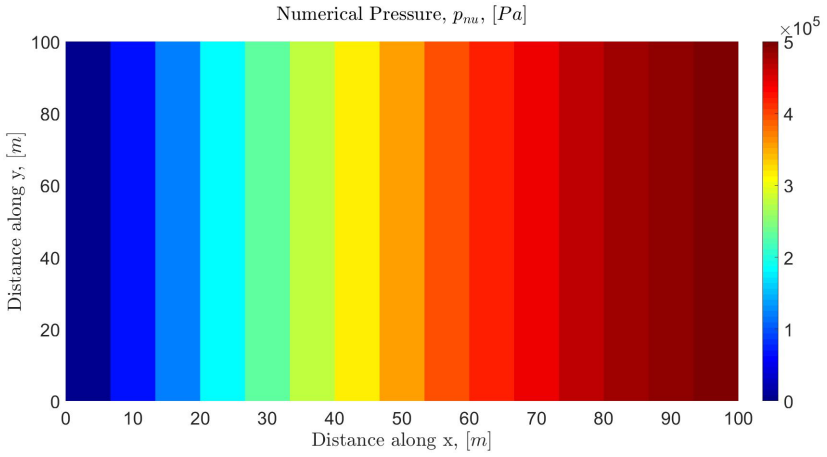
(c) *Error values for the displacement in y-direction on* 16×16 *grid.*

Figure 4.6: *Illustrates exact, numerical and error map values for the displacement in the y-direction in the synthetic test case by fully coupled method.*

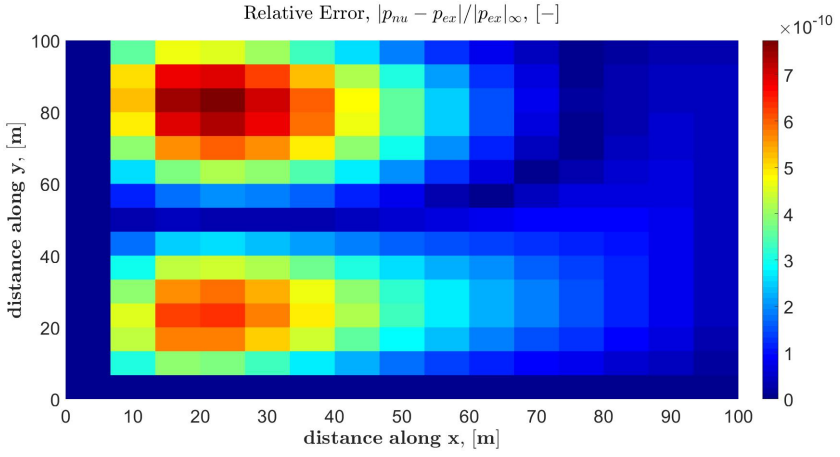
Pressure: Fully Coupled Method



(a) *Exact solution for the pressure on* 16×16 *grid.*



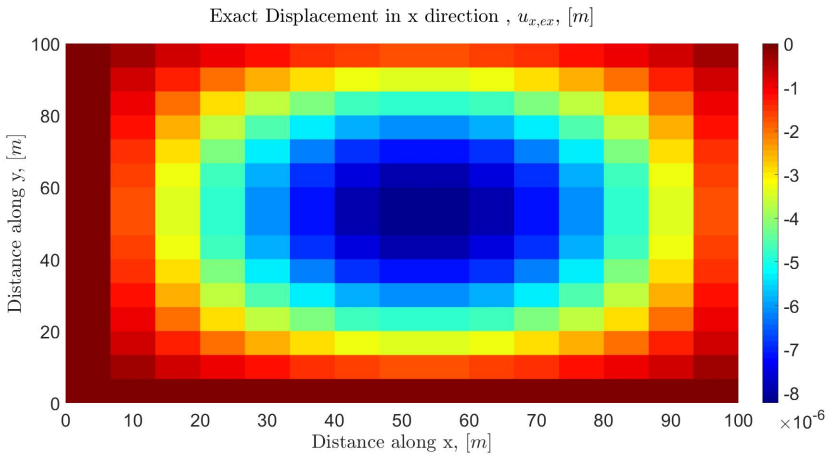
(b) *Numerical solution for the pressure on* 16×16 *grid.*



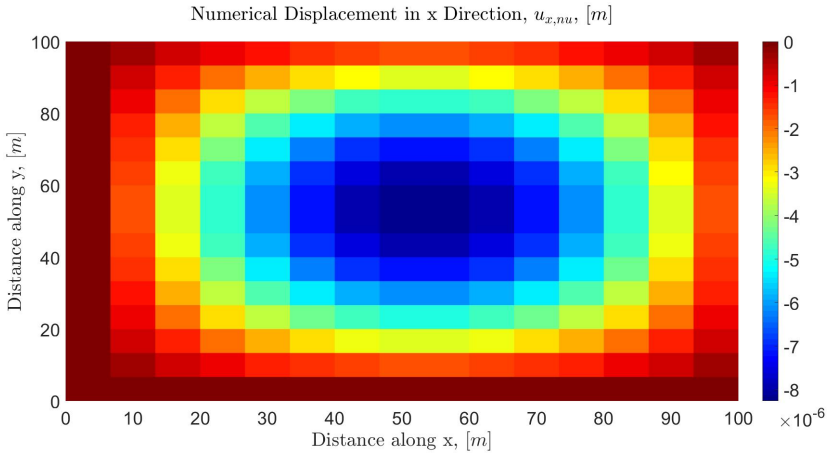
(c) *Error values for the pressure on* 16×16 *grid.*

Figure 4.7: *Illustrates exact, numerical and error map values for the pressure in the synthetic test case by fully coupled method.*

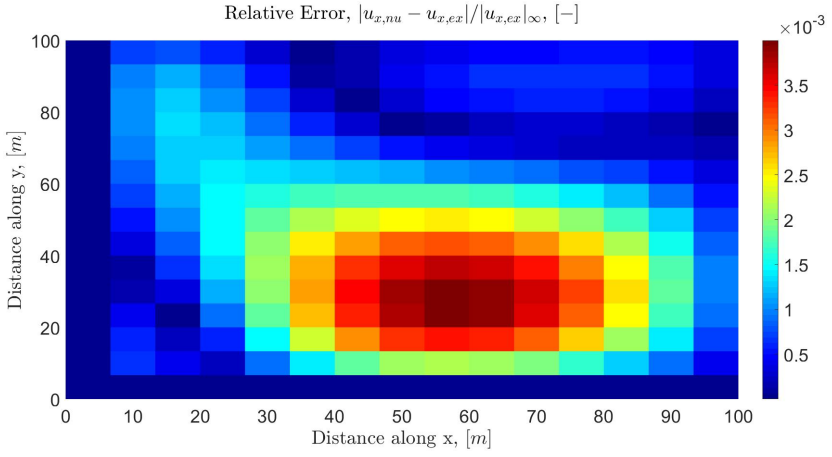
Displacement in x-direction: Fully Coupled Method



(a) *Exact solution for the displacement in the x-direction on* 16×16 *grid.*



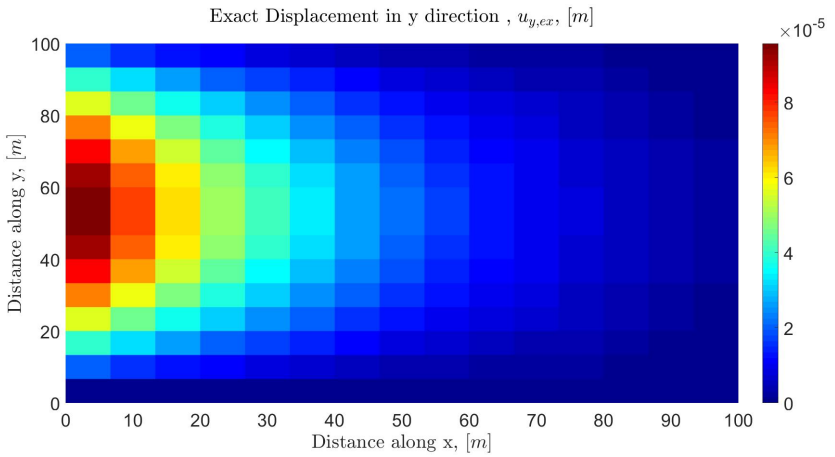
(b) *Numerical solution for the displacement in the x-direction on* 16×16 *grid.*



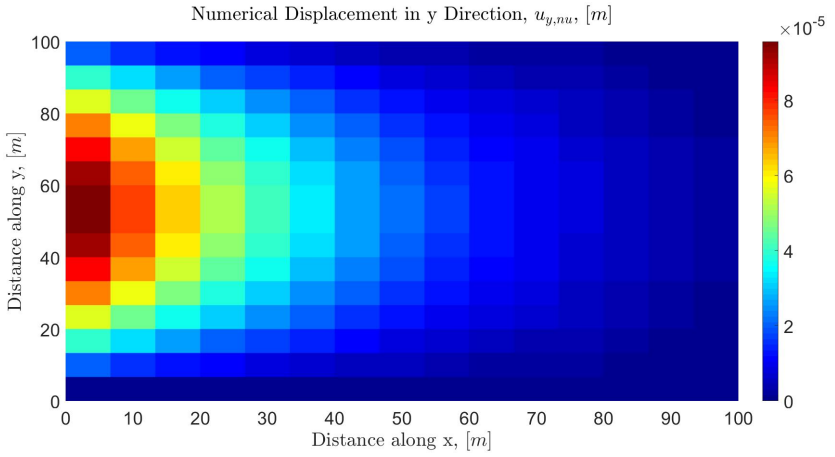
(c) *Error values for the displacement in the x-direction on* 16×16 *grid.*

Figure 4.8: *Illustrates exact, numerical and error map values for the displacement in the x-direction in the synthetic test case by fully coupled method.*

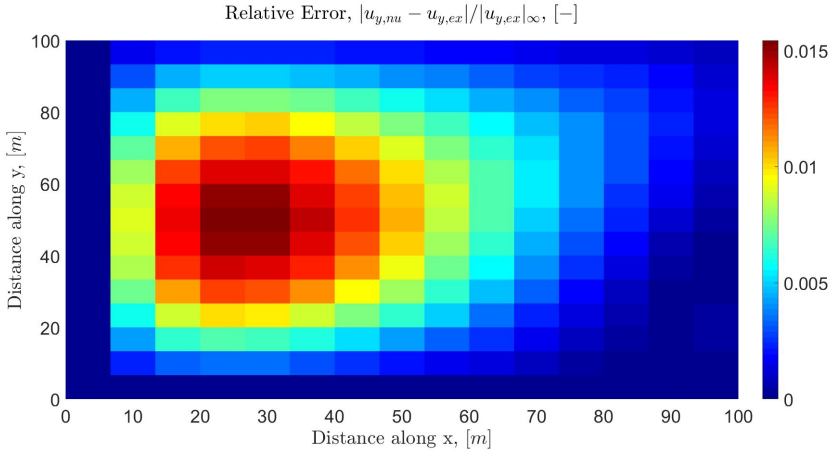
Displacement in y-direction: Fully Coupled Method



(a) *Exact solution for the displacement in the y-direction on* 16×16 *grid.*



(b) *Numerical solution for the displacement in the y-direction on* 16×16 *grid.*



(c) *Error values for the displacement in the y-direction on* 16×16 *grid.*

Figure 4.9: *Illustrates exact, numerical and error map values for the displacement in the y-direction in the synthetic test case by fully coupled method.*

4.1.2 Synthetic Test Case II: Error and Convergence Study

As shown in Section 3.5, the accuracy of the spatial discretization is supposed to be second-order while the time domain that is discretized by backward Euler method should be first order accurate. Therefore, second order accuracy is expected at relatively small values for the time-steps while at adequately small values for the grid, first order accuracy is observed. In other cases, the total error is a combination of the two level of accuracies in time and spatial domain, as illustrated in Figure 3.14.
Functions, parameters and the approach to study the convergence of the developed methods are the same as stated in the work carried out by Luo et al. (2015). Analytical solutions are stated in Table 4.7. The shear and bulk modulus and mobility values are 0.1 and 0.1 $[Pa]$, and 1 $[m.s]$, respectively. The final time is 0.5 $[s]$ and the domain size in meter is $(0,1)\times(0,1)$. The storativity in this model is assumed to be 0.01 $[Pa^{-1}]$. Moreover, Dirichlet boundary conditions are assumed at the boundaries for all the parameter variables. The relative maximum error in the last time-step, i.e. $t = 0.5$, is used to quantify the error values reported in Table 4.8. Solution functions are as stated in Table 4.7. Therefore, the error values for the displacements in both directions are the same and only one value is reported for the displacement error in Table 4.8.

Table 4.7: *Illustrates the analytical solutions and initial conditions used to study convergence of the solutions.*

Variable	Analytical Sol.	Initial Condition
p	$-2(\lambda+2\mu)\pi\sin(\pi x)\sin(\pi y)\sin(\pi t)$	0
u_x	$\cos(\pi x)\sin(\pi y)\sin(\pi t)$	0
u_y	$\sin(\pi x)\cos(\pi y)\sin(\pi t)$	0

The performance of the two numerical models is compared, the fully coupled method, i.e. FC, and the sequentially coupled method, i.e. SC, based on error and convergence study and time efficiency. It is worthy to note that there is an optimum value for the threshold error when defining the convergence criteria and too small values might result in increased calculated error[4]. In addition, the time values in Table 4.8 are relative and are used to compare the performance of the two methods. These values can be reduced with using sparse matrices and less for-loops in coding or implementing the codes in a faster programming language such as C++. For the sequentially coupled method, the computational time is dependent on more variables, such as the initial guess, the sequential coupling scheme, the convergence criteria, etc.

[4]The error in this context is defined as the deviation from the exact/analytical value rather than its value from the previous iteration

Table 4.8: *Illustrates the convergence of the numerical solutions for the fully coupled, i.e. FC, and the sequentially coupled, i.e. SC, methods.*

Grid	Method	RMSE(p)	RMSE(u)	$\frac{\lvert p-p_0\rvert_\infty}{\lvert p_0\rvert_\infty}$	$\frac{\lvert u-u_0\rvert_\infty}{\lvert u_0\rvert_\infty}$	Time $[s]$	$\frac{c_v * dt}{dx^2}$
$16 \times 16 \times 2$	FC	2.50×10^{-3}	8.47×10^{-4}	3.9×10^{-3}	1.6×10^{-3}	0.93	17
	SC	3.90×10^{-3}	2.40×10^{-3}	5.10×10^{-3}	9.90×10^{-3}	1.71	17
$32 \times 32 \times 4$	FC	3.45×10^{-4}	1.51×10^{-4}	4.56×10^{-4}	2.77×10^{-4}	5	36
	SC	5.48×10^{-4}	3.75×10^{-4}	1.10×10^{-3}	3.00×10^{-3}	9.6	36
$64 \times 64 \times 8$	FC	3.57×10^{-5}	2.57×10^{-5}	3.89×10^{-5}	4.58×10^{-5}	172.3	74
	SC	1.10×10^{-4}	4.43×10^{-5}	6.96×10^{-4}	8.70×10^{-4}	380.7	74

Table 4.9 illustrates the error values for the sequentially coupled method that run in decoupled mode. This table is provided to illustrate the performance of the developed cell-centred FVM codes run in isolated mode. The reduction in the maximum relative error is approximately quadratic for the pressure and the displacements. This error reduction might be accelerated or decelerated by using a different extrapolation function to define the values in the corner nodes in the displacement solver as mentioned in Section 3.5 for the CC-FV model. By comparing the time values of the sequentially coupled method from Tables 4.8 and 4.9, it is concluded that the convergence rate of the sequentially coupled method becomes faster, i.e. happens at a faster rate, with an optimized sequentially coupled scheme or by implementing this discretization method in a fully coupled mode.

Table 4.9: *Illustrates convergence of numerical solutions for solid and fluid cell-centred FVM codes used in the sequentially coupled method run in decoupled mode, i.e.* $b = 0$.

Grid	Code	RMSE(p)	RMSE(u)	$\frac{\lvert p-p_0\rvert_\infty}{\lvert p_0\rvert_\infty}$	$\frac{\lvert u-u_0\rvert_\infty}{\lvert u_0\rvert_\infty}$	Time $[s]$
$16 \times 16 \times 2$	Fluid	2.30×10^{-3}	–	3.40×10^{-3}	–	1.3
	Solid	–	1.80×10^{-3}	–	5.00×10^{-3}	1.3
$32 \times 32 \times 4$	Fluid	3.04×10^{-4}	–	8.28×10^{-4}	–	4.2
	Solid	–	2.50×10^{-4}	–	1.3×10^{-3}	4.2
$64 \times 64 \times 8$	Fluid	3.77×10^{-5}	–	2.03×10^{-4}	–	140
	Solid	–	3.22×10^{-5}	–	3.15×10^{-4}	140

Based on the values reported in Tables 4.8 and 4.9, the root mean square of pressure and displacements can be plotted against the multiplication of grid size in x- and y- directions, i.e. Δx and Δy , the in a logarithmic plot, as illustrated in Figure 4.10 and 4.11. The slope of these lines illustrate the accuracy of each method. Both the fully coupled vertex-centred model (VV-FC) and the sequentially coupled cell-centred model (CC-SC) render second order accuracy. The slope of the CC-unc shows a higher value compared to CC-SC which implies that a higher convergence rate can be obtained for CC-SC by improving the coupling scheme utilized for this method. In Figure 4.11, the slope of the VV-FC model is lower compared to CC-SC and CC-unc. However, the intercept of its trend-line is a larger negative value compared

to the intercept of the other two trend-lines drawn for CC-SC and CC-unc. This observation implies that this line is positioned below the other two lines; and therefore, the order of the error for this method is lower compared to the other two methods when keeping the grid size constant.

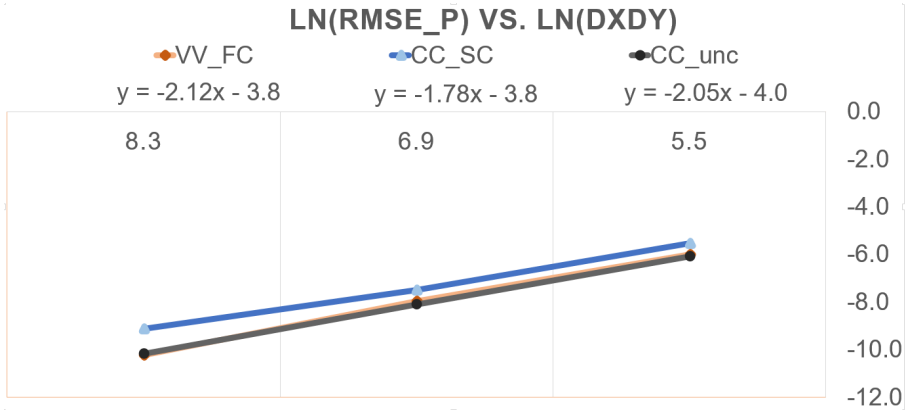


Figure 4.10: *Illustrates the convergence and error study plot for pressure solutions of VV-FC (coupled) and CC-SC (coupled and decoupled).*

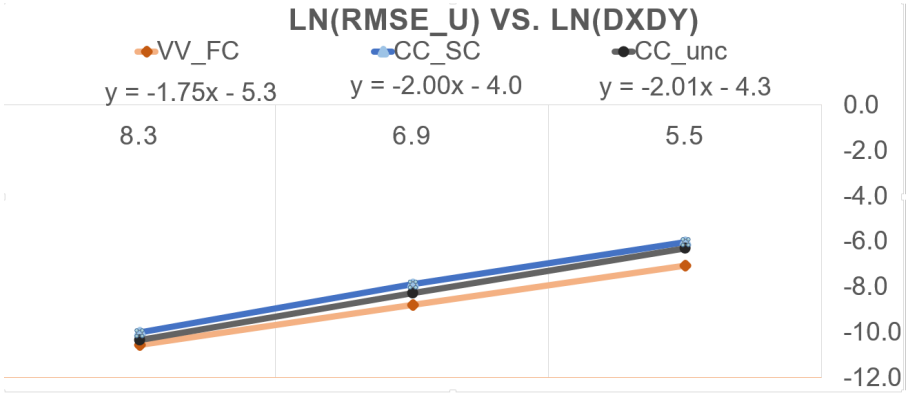


Figure 4.11: *Illustrates the convergence and error study plot for displacement solutions of VV-FC (coupled) and CC-SC (coupled and decoupled).*

Considering the aforementioned comparison, the fully coupled FVM method outperforms the sequentially coupled FVM method with respect to both error and time. Therefore, this method is used to be benchmarked against the Mandel test case, to observe the model reaction to different systems, to conduct the sensitivity analysis, and to solve the practical problem.

4.2 Benchmarking

The practice of comparing numerical solutions with analytical solutions are best called benchmarking (Oreskes et al., 1994). One well-known classical test case, i.e. Mandel test case, is

selected to benchmark the fully coupled method. The Mandel test case is a well-known plane strain test case in cartesian coordinates. The analytical solutions of this test case and the code for finding the analytical solutions are included in Appendix B and D.6, respectively.

Mandel Test Case

Mandel (1953) presented a benchmark plane strain consolidation test case in which non-monotonic pressure behaviour is observed. In this test case, the porous media is bounded by impermeable walls at left, top and bottom side-walls, see Figure 4.12. However, the fluid is free to drain from the medium at the right side. The model is fixed in place by rollers at the bottom and left sides while the medium is free to displace laterally at the right boundary. A piston is placed at the top of the medium to exert pressure at the top side.

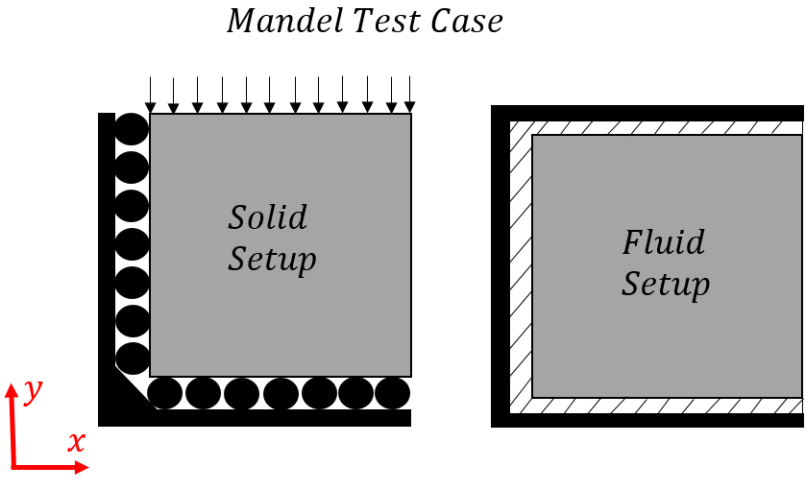


Figure 4.12: *Illustrates the setup and boundary conditions for the Mandel test case.*

As shown by Equation (4.2), which describes the initial values for the Mandel test case, initially, there is no force at the top of the medium and the medium is fully saturated with fluid. A sudden constant load is placed on the top of the medium. At this point, the fluid starts draining from the right side of the model. After a short time, an induced pressure build-up to values higher than its initial value at the left boundary due to pore space shrinkage at the drained edges. As time passes and the rate of fluid drainage and solid particle displacement reaches a balance, the induced pressure build-up disappears and, at the end of the consolidation process, the medium approaches drained porous medium behaviour. The numerical and analytical solutions for Mandel's problem in an incompressible system are compared. The input values

are as stated in Table 4.10[5].

$$p(x, y, 0) \approx 88.3[Pa] \tag{4.2a}$$

$$u_x(x, y, 0) \approx 1.18 \times 10^{-5} \frac{x}{L_x}[m] \tag{4.2b}$$

$$u_y(x, y, 0) \approx -1.32 \times 10^{-5} \frac{y}{L_y}[m] \tag{4.2c}$$

Table 4.10: *Illustrates the values used as input for Mandel's Problem.*

Domain Properties	Solid Properties	Fluid Properties	Fluid-Solid Properties
$L_x = 100\ [m]$	$\lambda = 4 \times 10^8\ [Pa]$	$\lambda_t = 1.01 \times 10^{-11}\ [m^2/(Pa.s)]$	$c_v = 1.01 \times 10^{-2}\ [m^2/s]$
$L_y = 100\ [m]$	$\mu = 4 \times 10^8\ [Pa]$	$S = 1.65 \times 10^{-10}\ [1/Pa]$	$t_c = 9.92 \times 10^5\ [s]$
$\sigma_{yy,t} = 200\ [Pa]$	-	-	$b = 1\ [-]$

The numerical solution for the pressure is compared with the analytical solution in one dimensional plots by comparing the values for the middle line which is equally distant from the bottom and top boundaries and at times equivalent to $\begin{bmatrix}0, & 0.01, & 0.1, & 0.5, & 1\end{bmatrix} \times t_c$. The fully coupled finite volume method is run with two configurations: the stress boundary conditions, as described in Figure 4.12, and dirichlet boundary conditions, also called displacement configuration. Figures 4.13,4.15, and 4.17 illusterate the response of the model with stress boundary conditions and Figures 4.14,4.16, and 4.18 illusterate the performance of the model with displacement boundary conditions. By comparision, a lateral variation is observed in the stress configuration which disappears in the displacement configuration and the analytical solution. The reason for this observation is postulated to be the total stress boundary condition, which is a term resulted from the summation of the pressure value, the first derivative of u_x and the first derivative of u_y, and the fact that with stress condition, at the corner points of the domain there are three boundary conditions including the values for u_x and u_y and two boundary conditions including the pressure condition. For the stress configuration, the minimum and maximum relative error for pressure and displacement at each time is reported in Table 4.11.

Table 4.11: *Illustrates the error values for FC numerical model that run for Mandel's problem with stress boundary conditions on a* 30×30 *grid,* $c_v dt/dx^2$ *is approximately 4.2.*

Time $[s]$	$\text{avg}(\frac{\lvert p - p_0 \rvert}{\lvert p_0 \rvert_\infty})$	$\text{avg}(\frac{\lvert u_x - u_{x,0} \rvert}{\lvert u_{x,0} \rvert_\infty})$	$\text{avg}(\frac{\lvert u_y - u_{y,0} \rvert}{\lvert u_{y,0} \rvert_\infty})$
0	–	–	–
$0.01 \times t_c$	0.017	0.023	0.018
$0.1 \times t_c$	0.032	0.031	0.027
$0.5 \times t_c$	0.014	0.028	0.018
t_c	0.032	0.013	0.008

The error values at the first time steps are small since the numerical solution is not distant from the initial condition. The error at the following time steps grows and then reduces due to decreased coupling between fluid and solid. However, the error reduction in the final time steps is not observed for the pressure solution. This observation is justifiable by the fact that analytical solutions for Mandel's problem are semi-analytical and are approximations of

[5]the model properties used for Mandel's test case are provided by Nicola Castelletto, 2016

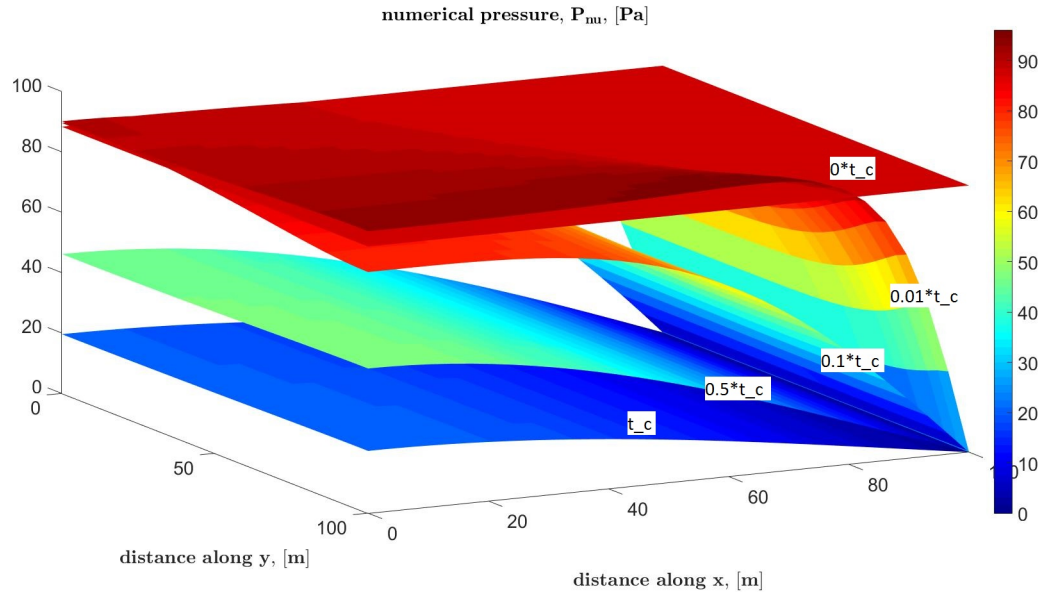
the exact solution. These approximated exact solutions might have deviated from the exact solutions in the last time steps. For stress configuration, the effect of grid refinement on relative error is illustrated in Table 4.12. To generate this table, a domain size of $(0,1) \times (0,1)$ and end time of 0.5 is assumed to reduce the computational costs. Table 4.12 represents normalized average error throughout the domain with stress boundary conditions.

Table 4.12: *Illustrates the error values for FC numerical model run for Mandel Problem with stress boundary conditions at the end time of* $0.5t$.

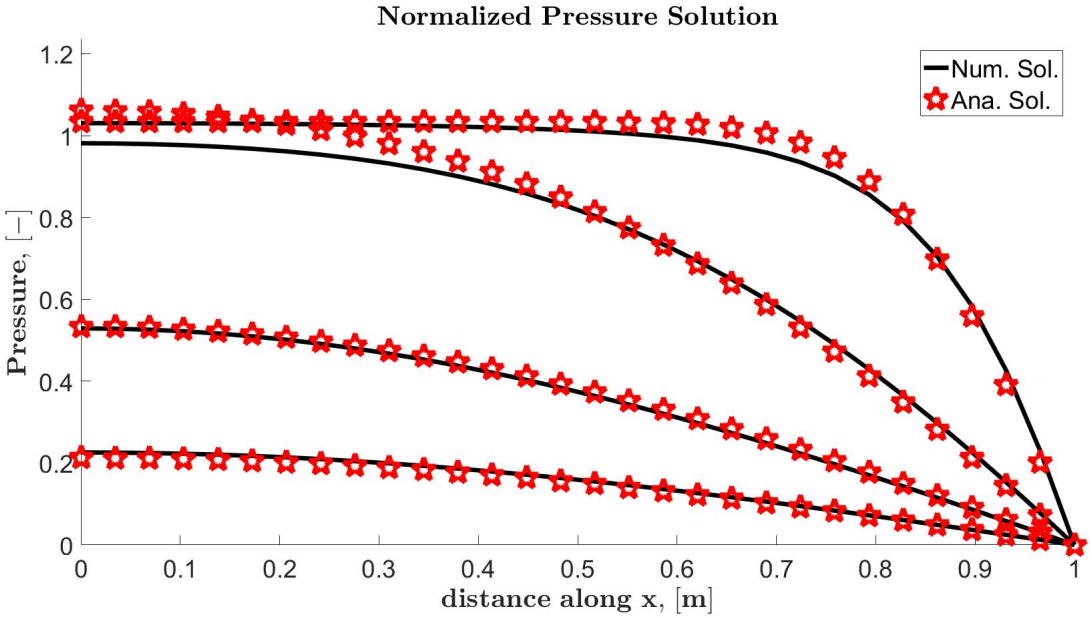
Grid	$\text{avg}(\frac{\lvert p-p_0\rvert}{\lvert p_0\rvert_\infty})$	$\text{avg}(\frac{\lvert u_x-u_{x,0}\rvert}{\lvert u_{x,0}\rvert_\infty})$	$\text{avg}(\frac{\lvert u_y-u_{y,0}\rvert}{\lvert u_{y,0\rvert_\infty}})$	$c_v dt/dx^2$
$16 \times 16 \times 2$	0.076	0.045	0.036	0.56
$32 \times 32 \times 4$	0.034	0.022	0.017	1.21
$64 \times 64 \times 8$	0.018	0.011	0.008	2.5

As observed in this table, the error decreases by refinement of grids through space and time. Based on the observations in figures and Table 4.12, it can be concluded that lateral variation observed in pressure and displacements solutions are merely an approximation and the discretzation error which would be reduced by an improved discretization and approximation of stress solutions at the boundaries.

Pressure Solutions: Stress Boundary Conditions



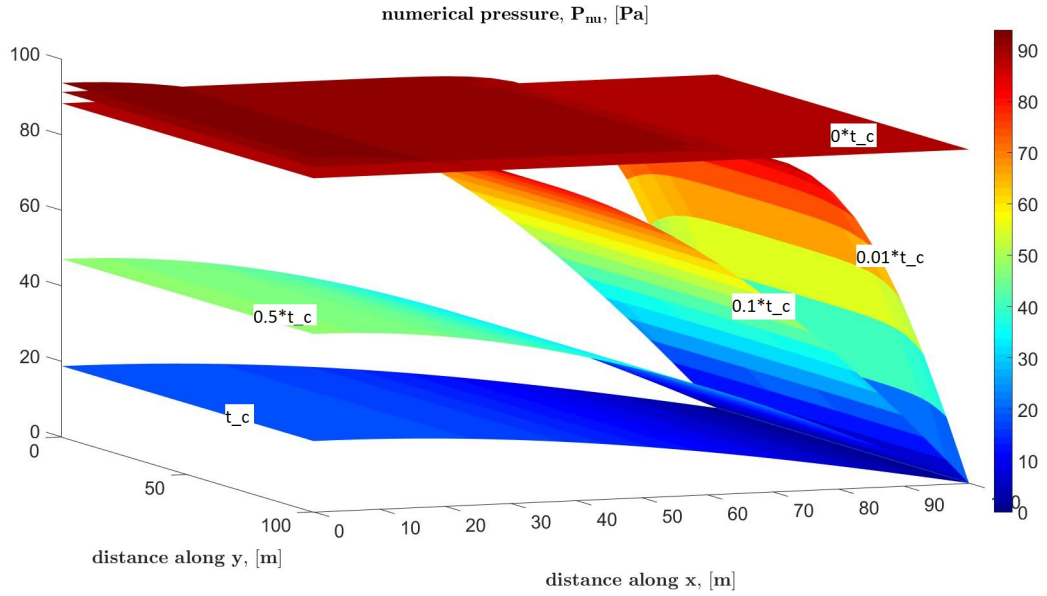
(a) *Numerical versus analytical solutions for the pressure with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



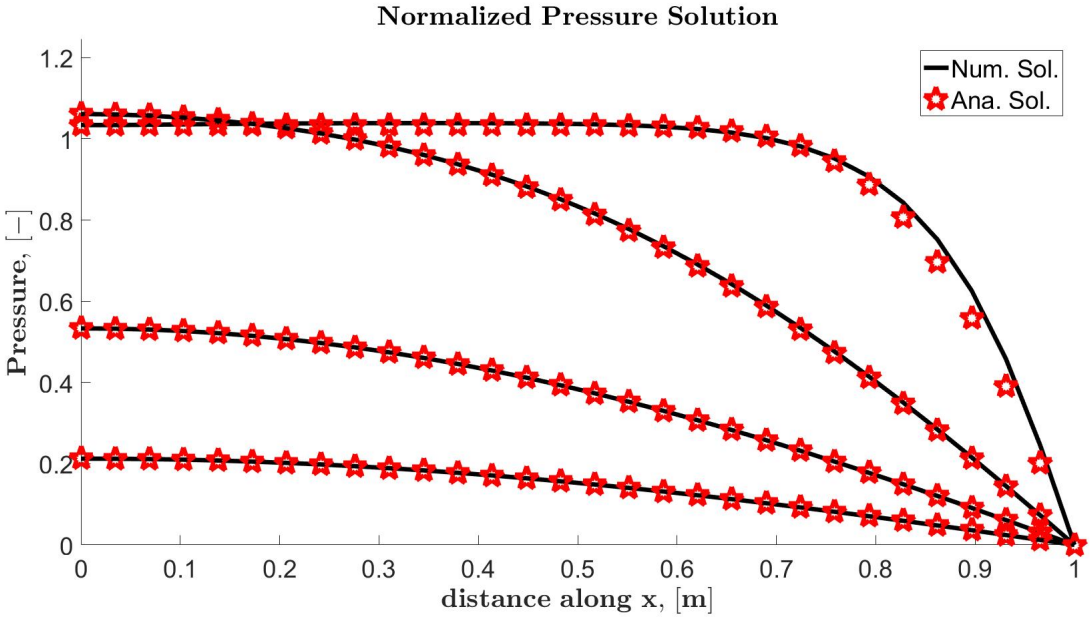
(b) *Numerical versus analytical solutions for the pressure with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.13: *Illustrates numerical and analytical pressure values with stress boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

Pressure Solutions: Dirichlet Boundary Condition



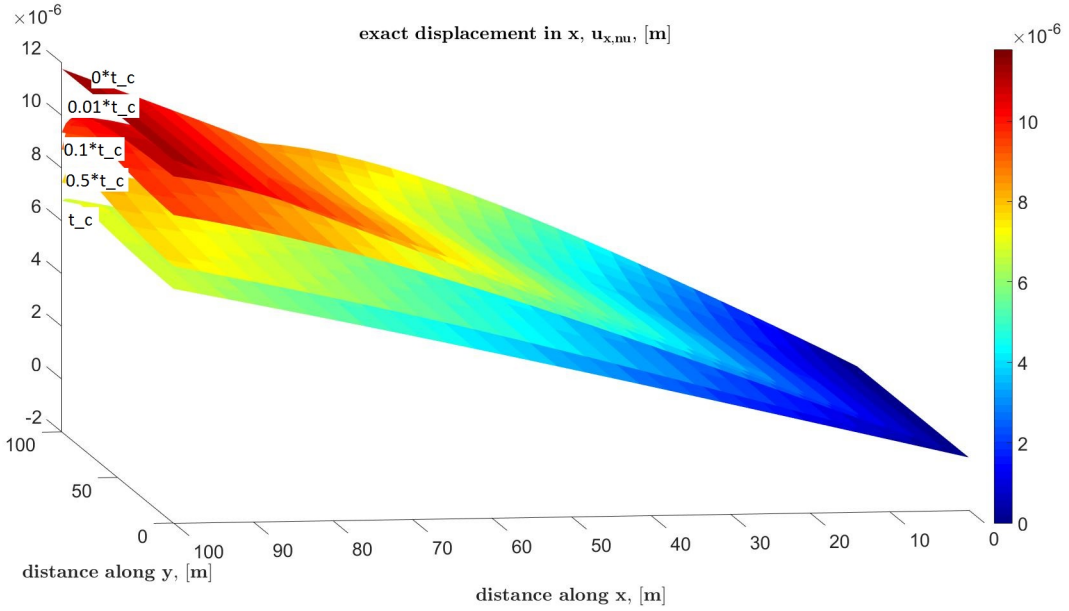
(a) *Numerical versus analytical solutions for the pressure with Dirichlet boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



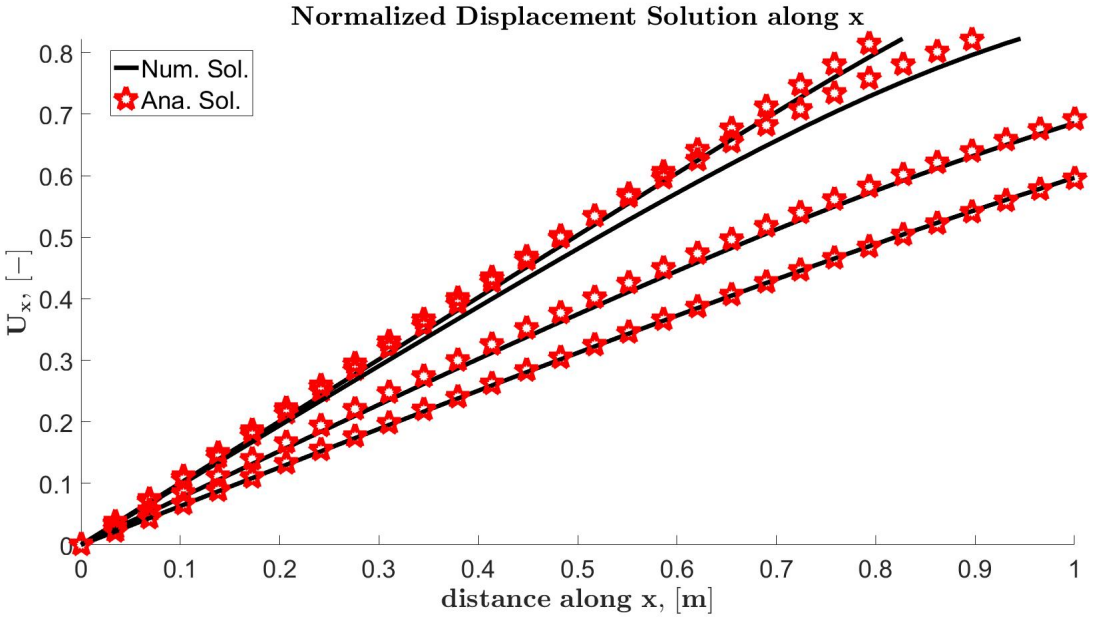
(b) *Numerical versus analytical solutions for the pressure with displacement boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.14: *Illustrates numerical and analytical pressure values with displacement boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

Displacement in x-direction Solutions: Stress Boundary Condition



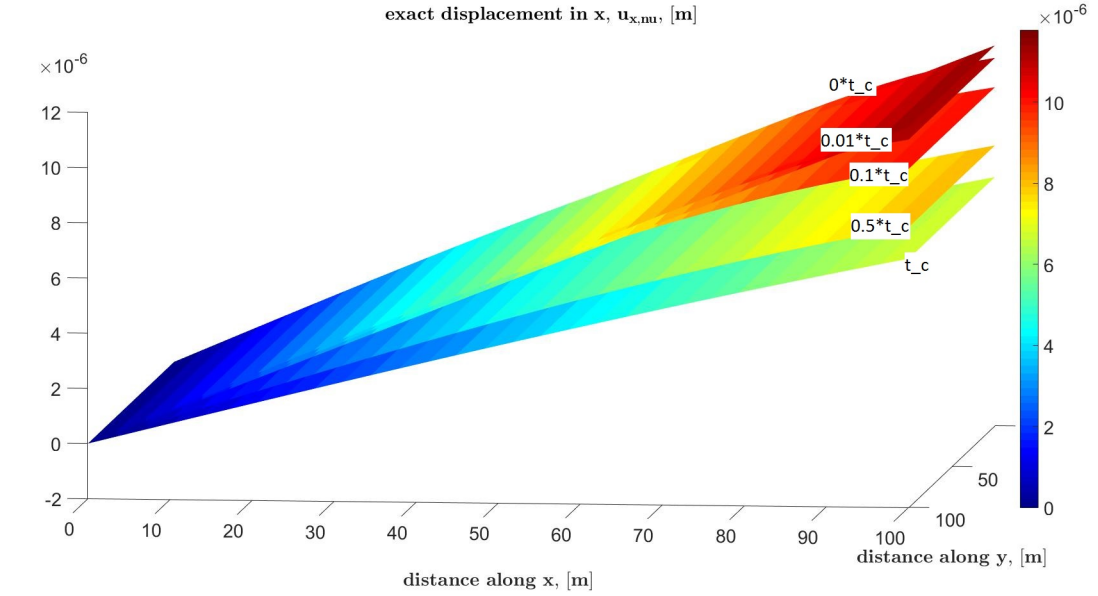
(a) *Numerical versus analytical solutions for the displacement in the x-direction with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



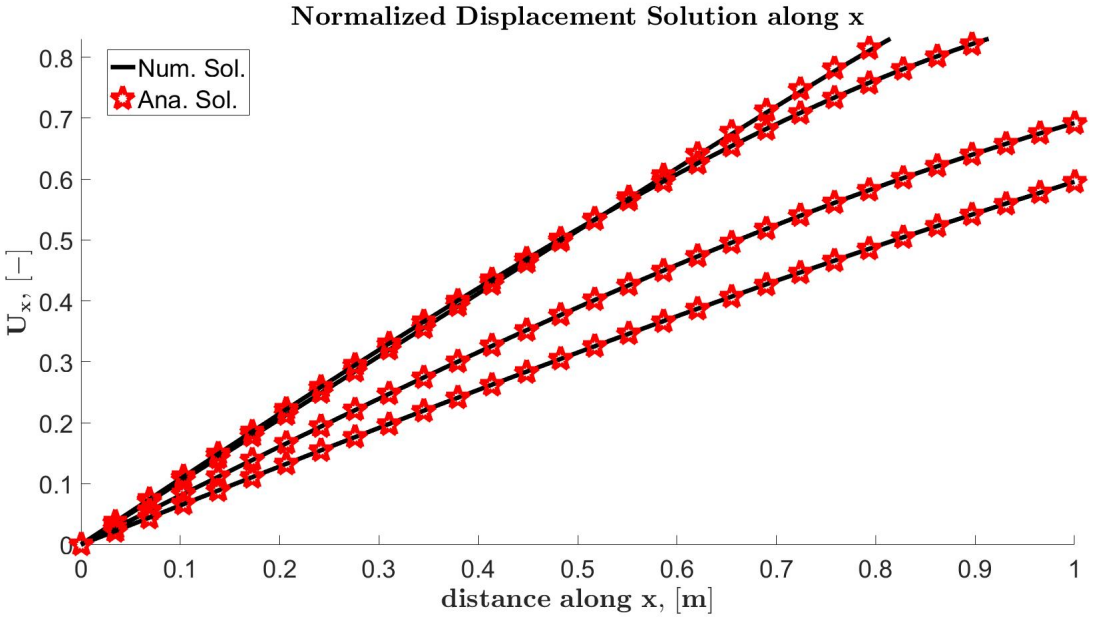
(b) *Numerical versus analytical solutions for the displacement in the x-direction with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.15: *Illustrates numerical and analytical displacement values with stress boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

Displacement in x-direction Solutions: Dirichlet Boundary Condition



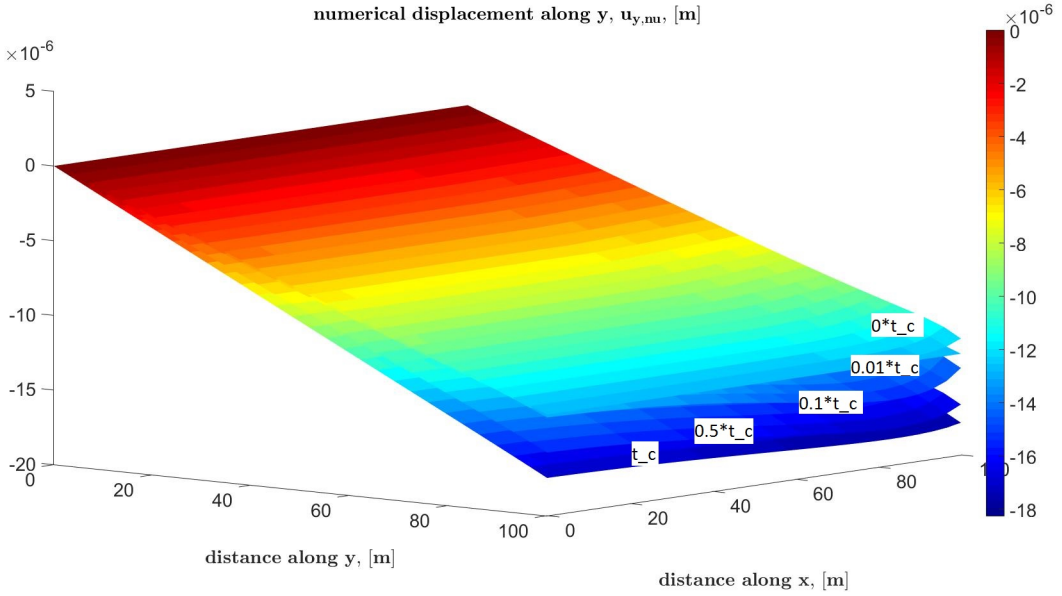
(a) *Numerical versus analytical solutions for the displacement in the x-direction with displacement boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



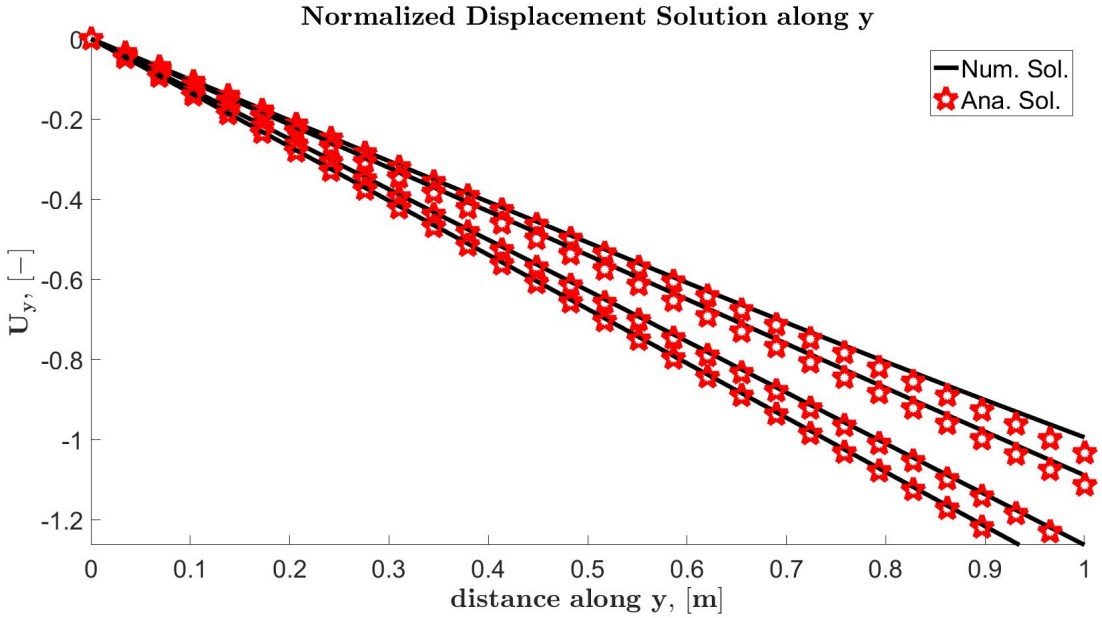
(b) *Numerical versus analytical solutions for the displacement in the x-direction with displacement boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.16: *Illustrates numerical and analytical displacement values with displacement boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

Displacement in y-direction Solutions: Stress Boundary Condition



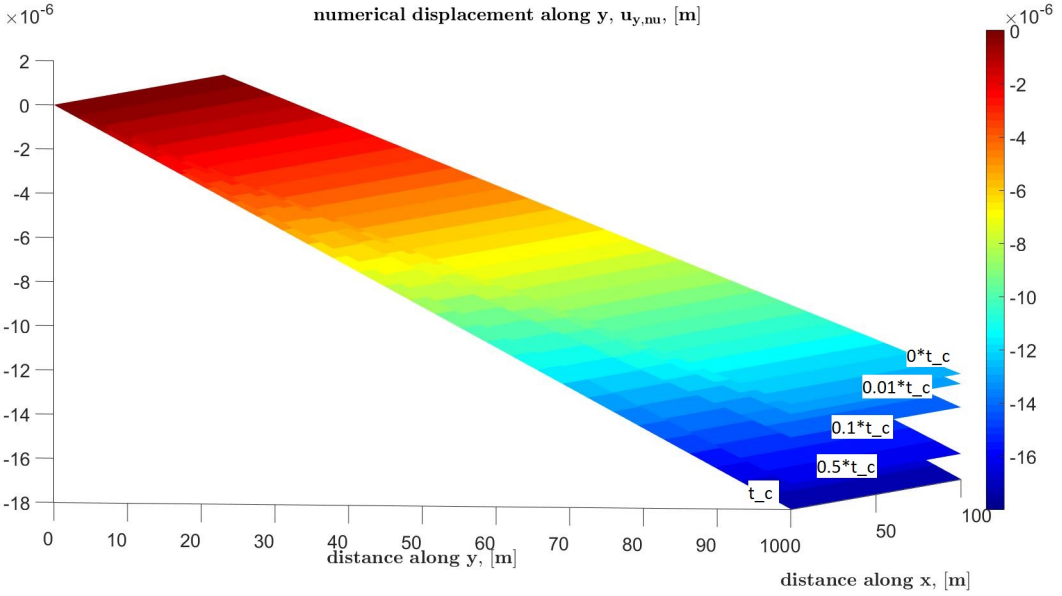
(a) *Numerical versus analytical solutions for the displacement in the y-direction with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



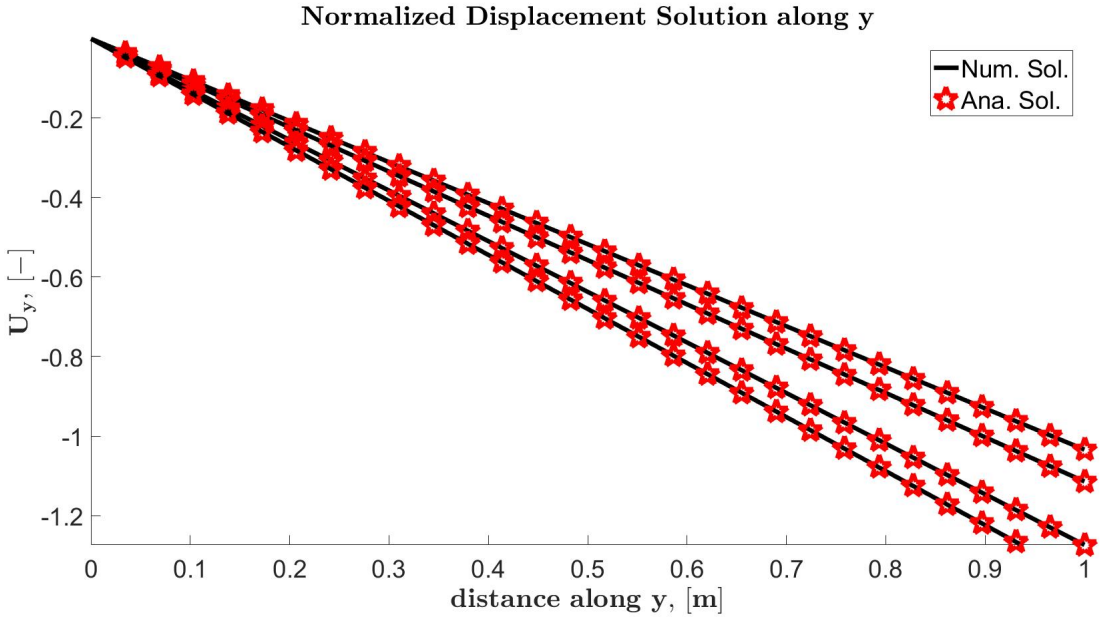
(b) *Numerical versus analytical solutions for the displacement in the y-direction with stress boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.17: *Illustrates numerical and analytical displacement values in the y-direction with stress boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

Displacement in y-direction Solutions: Dirichlet Boundary Condition



(a) *Numerical versus analytical solutions for the displacement in the y-direction with displacement boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0, & 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



(b) *Numerical versus analytical solutions for the displacement in the y-direction with displacement boundary conditions on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.18: *Illustrates numerical and analytical displacement values in the y-direction with displacement boundary conditions on Mandel setup by fully coupled FVM model on a* 30×30 *grid.*

4.3 Model Reaction to Different Systems

The reaction of the fully coupled model to fully-, semi- and in-compressible systems[6] in consolidated rocks is investigated in terms of pressure solutions with both displacement and stress (boundary conditions) configuration. Table 4.13 shows the properties of each system in consolidated rocks, for more details see the Appendix C. The Table 4.14 shows the response of the model to each system in terms of measured error in estimating pressure and displacement values. Figures 4.19a and 4.19b show the numerical pressure solutions of these systems for stress and displacement (boundary conditions) configurations, respectively. The model is tested on a $(x \times y \times t)$=$30 \times 30 \times 200$, and a $(0,1) \times (0,1)$ domain size. The end time is t_c and the pressure solutions are compared at times equivalent to $\begin{bmatrix}0.01, & 0.1, & 0.5, & 1\end{bmatrix} \times t_c$.

Table 4.13: *Illustrates values for storativity and mobility for consolidated oil reservoir rock with fluid type of varying compressibility.*

Type of System	λ_t $[m^2/(Pa.s)]$	S $[1/Pa]$	c_v $[m^2/s]$	λ $[Pa]$	μ $[Pa]$
Incompressible Fluid	1.00×10^{-11}	1.73×10^{-9}	4.5×10^{-3}	3.45×10^{8}	8.39×10^{8}
Slightly Compressible Fluid	7.14×10^{-8}	1.29×10^{-8}	5.3	3.45×10^{8}	8.39×10^{8}
Fully Compressible Fluid	5.00×10^{-7}	1.74×10^{-8}	28	3.45×10^{8}	8.39×10^{8}

The values in Table 4.14 is aimed to show how the model responses to different systems and to make a comparison of stress and displacement (boundary condition) configurations. In this table, the first three rows report the normalized averaged value of the error for each primary unknown in the system. In both configurations, the computational error decreases as the fluid in the system becomes more compressible, i.e. S increases. This is expected to be due to the fact that there is a lower value for coupling strength in the system, i.e. $\tau = b^2/(S \times K_{dr})$. The fourth column reports the normalized maximum deviation from the initial pressure in each system. The value decreases with an increase in the fluid compressibility in the system since the fluid is produced faster and less non-monotonic pressure behaviour is observed in the porous media.

Table 4.14: *Illustrates effect of fluid compressibility on poroelastic effects in consolidated rocks and impact of the boundary condition on error values.*

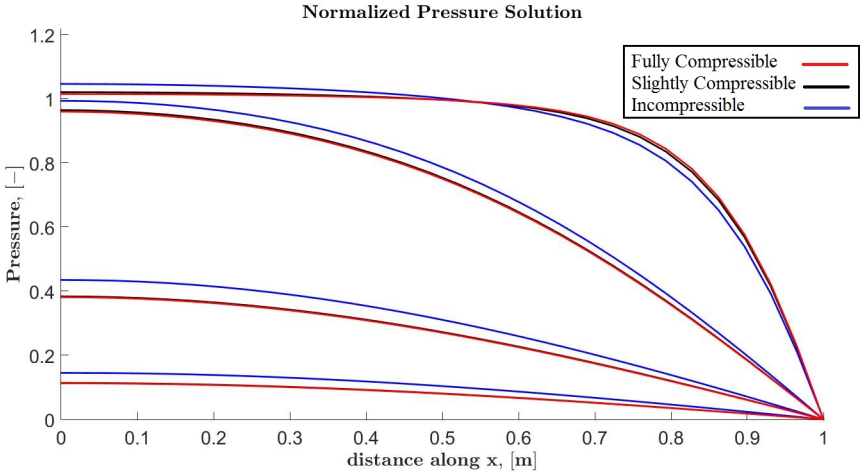
Type of System	$\frac{\lvert p-p_0\rvert_{avg}}{\lvert p_0\rvert_\infty}$	$\frac{\lvert u_x-u_{x,0}\rvert_{avg}}{\lvert u_{x,0}\rvert_\infty}$	$\frac{\lvert u_y-u_{y,0}\rvert_{avg}}{\lvert u_{y,0}\rvert_\infty}$	$\frac{\Delta p_{max}}{p_0}$	$\frac{\Delta u_{max}}{u_0}$	$\frac{\lvert\overline{\Delta p}\rvert}{p_0}$
Incompressible-S-BC	0.010	0.009	0.002	0.16	0.13	0.070
Slightly Compressible-S-BC	0.005	0.007	0.001	0.06	0.02	0.081
Fully Compressible-S-BC	0.004	0.005	8×10^{-4}	0.05	0.012	0.083
Incompressible-D-BC	0.007	9×10^{-4}	1.2×10^{-4}	0.03	0.12	0.070
Slightly Compressible-D-BC	0.009	2.3×10^{-5}	8.2×10^{-7}	0.004	0.022	0.085
Fully Compressible-D-BC	0.009	1.4×10^{-5}	5×10^{-7}	0.003	0.016	0.085

By comparing the response of the system in the two configurations for this parameter, i.e. $\Delta p_{max}/p_0$, it is illustrated that the values reported in the stress configuration render an overestimation to an extent. However, this observation is a boundary effect and can be removed by either considering a domain larger than the domain of interest or averaging pressure over the domain of interest, as it is observable by the values reported in the last column of Table 4.14.

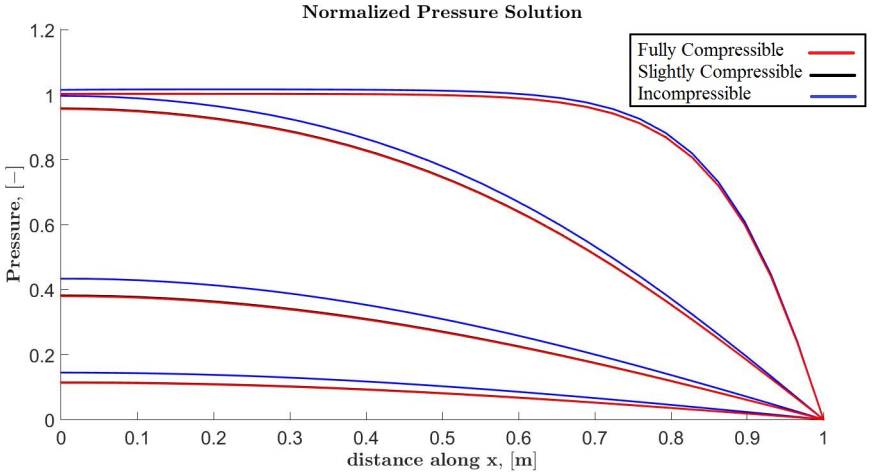
[6]For modelling a fully coupled system diagonalizing the coefficient matrices is necessary

On the contrary, comparison of normalized maximum displacement deviation from its initial values, i.e. $\Delta u_{max}/u_0$, which is equivalent to prediction of the model for maximum subsidence and horizontal displacement values, render accurate solutions for both configurations. Thus, the error in the prediction of pressure values is expected to be affected by the range of parameters, as well. In other words, the error accumulation in the pressure equation is higher due to the difference in range of input parameters in solid and fluid equations.

Pressure Solutions for Different Fluid Compressibilites



(a) *Numerical pressure solution with stress (boundary condition) configuration on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.



(b) *Numerical pressure solution with displacement (boundary condition) configuration on a* 30×30 *grid at times:* $\begin{bmatrix} 0.01, & 0.1, & 0.5, & 1 \end{bmatrix} \times t_c$.

Figure 4.19: *Illustrates a comparison between the reaction of the stress and displacement boundary condition configurations of fully coupled FVM method in terms of numerical pressure values for fluids with different compressibilities in consolidated rockes on Mandel setup in a* 30×30 *grid domain.*

To further improve the model in this aspect, investigation on an optimal normalizing of input parameters is encouraged. In Figure 4.19, the model response in terms of numerical pressure solution through time is reported for the point equally distant from the domain boundaries. It is illustrated that the higher the compressibility of the fluid is, the faster the pressure drops in the system. As expected the pressure is predicted with an acceptable level of accuracy in both configurations.

4.4 Sensitivity Analysis

Two aims are established in the sensitivity analysis: one is to measure the sensitivity of the results generated by the developed model to the uncertainty in input parameter variables; the second aim is to illustrate the robustness of the results. The input variables used to conduct the sensitivity analysis are those properties that are commonly directly measured and are the only input variables needed to describe the system in poroelasticity: porosity ϕ, Elastic modulus E, Poisson ratio ν, fluid permeability k_{fl}, fluid viscosity μ_{fl}, domain size L_x and L_y, solid bulk modulus K_s, fluid compressibility c_f, and the pressure forced on top of the domain F.

4.4.1 Uncertainty to Input Parameters

In order to test the model sensitivity to measurement or uncertainty in input properties, and in order to investigate which parameters are the most important properties of the model, same percentage of deviation is considered for all properties, i.e. $\pm 20\%$ (see Table 4.15). Thus, the input parameters would vary one by one between the upper bound, base, and the lower bound values to see how much the error value changes. The error in this section is estimated in the last time step and in terms of its deviation from the results of base input variables since it aims to illustrate the result deviations from the true values in case of wrong estimation of the input parameters.

Table 4.15: *Illustrates values for storativity and mobility for consolidated oil reservoir rock with fluid type of varying compressibility.*

-	$\phi[-]$	$E[Pa]$	$\nu[-]$	$k_{fl}[m^2]$	$\mu_{fl}[Pa.s]$
Lower Bound Values	0.16	8.00×10^{9}	0.20	7.90×10^{-11}	7.84×10^{-4}
Base Case Values	0.20	1.00×10^{10}	0.25	9.87×10^{-11}	9.81×10^{-4}
Upper Bound Values	0.24	1.20×10^{10}	0.30	1.18×10^{-10}	1.2×10^{-3}
-	$L_x[m]$	$L_y[m]$	$K_s[Pa]$	$c_f[Pa^{-1}]$	$F[Pa]$
Lower Bound Values	80	80	8×10^{79}	3.5×10^{-8}	1.6×10^{6}
Base Case Values	100	100	1×10^{80}	4.4×10^{-8}	2×10^{6}
Upper Bound Values	120	120	1.2×10^{80}	5.2×10^{-8}	2.4×10^{6}

The results of this part is illustrated in terms of tornado plots for the pressure and the displacement in x- and y- directions in Figures 4.20-4.22. As expected, the solutions are directly and mostly affected by the parameters determining the initial values of the problem, and those parameters that affect the solutions significantly such as fluid compressibility in fluid equation in Figure 4.20 and elastic modulus for solid equations in Figures 4.21 and 4.22.

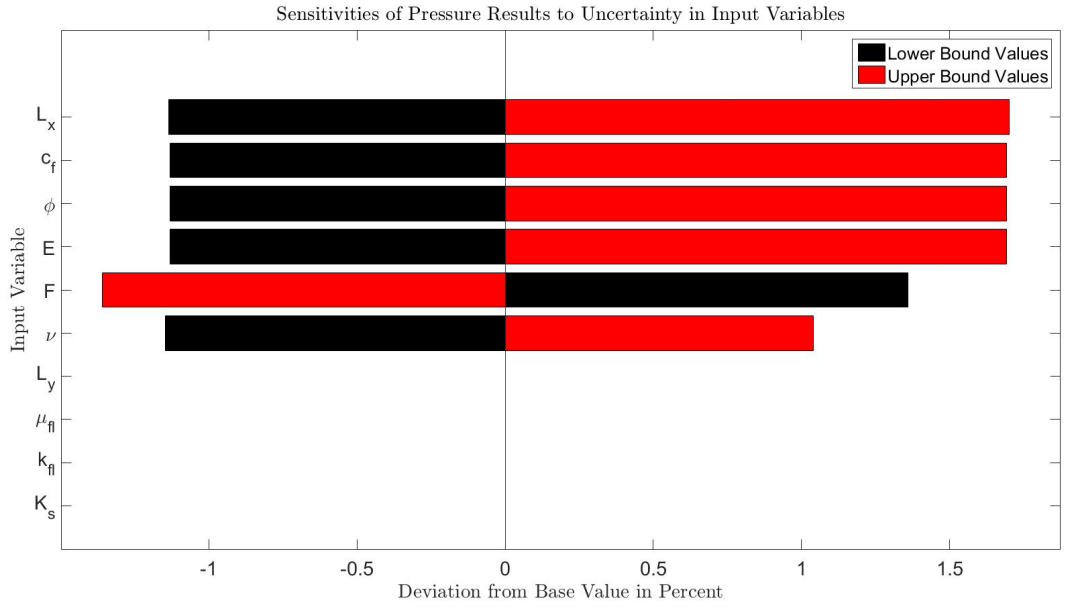


Figure 4.20: *Illustrates the sensitivity of the pressure solution to the accuracy of the measurement/uncertainty of input variables.*

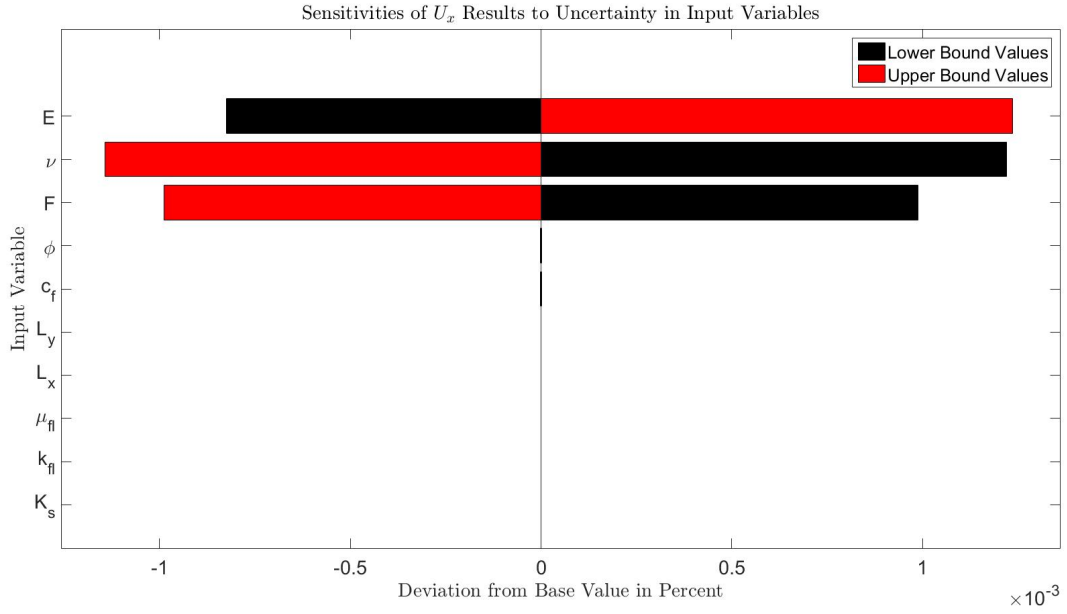


Figure 4.21: *Illustrates the sensitivity of the displacement solution in x-direction to to the accuracy of the input variables.*

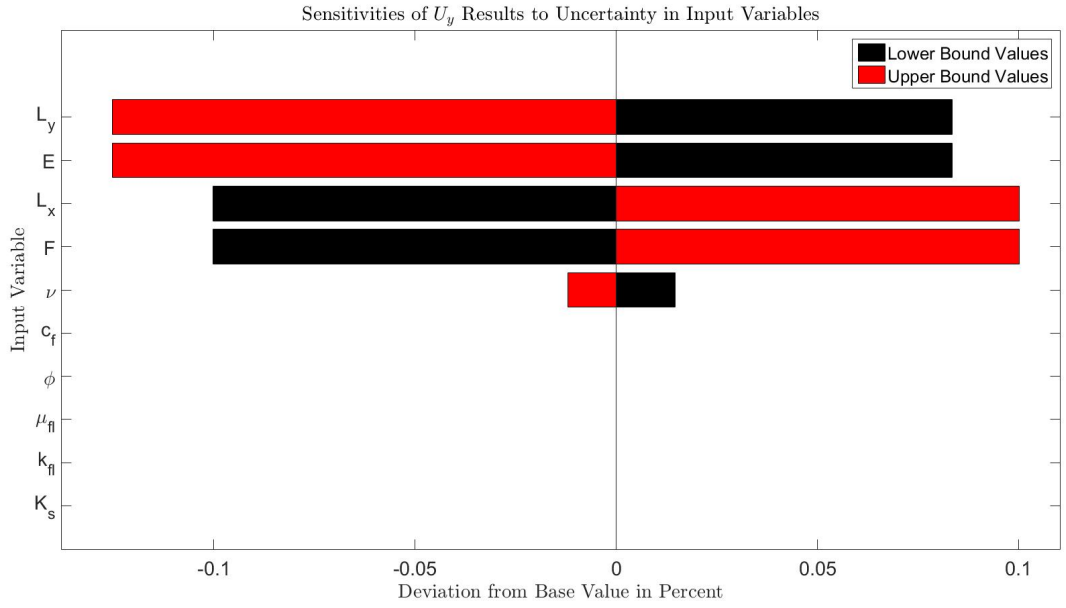


Figure 4.22: *Illustrates the sensitivity of the displacement solution in y-direction to the accuracy of the input variables.*

4.4.2 Robustness of the Model

The error in this section is estimated in terms of normalized root mean square error obtained by comparing the numerical solution to the analytical solution at the end time. Values stated in row number 3 of Table 4.16 are the base values. The input parameters change one at a time, from the smallest values that are sensible to the largest values that are possible. Figures 4.23, 4.24, and 4.25 illustrate the sensitivity of the accuracy of the model to values of input variables.

Table 4.16: *Illustrates values for storativity and mobility for consolidated oil reservoir rock with fluid type of varying compressibility.*

-	$\phi[-]$	$E[Pa]$	$\nu[-]$	$k_{fl}[m^2]$	$\mu_{fl}[Pa.s]$
1	0.05	1.00×10^{6}	0	9.87×10^{-16}	9.81×10^{-7}
2	0.10	1.00×10^{8}	0.15	9.87×10^{-14}	9.81×10^{-6}
3	0.20	1.00×10^{10}	0.25	9.87×10^{-12}	9.81×10^{-5}
4	0.30	1.00×10^{11}	0.35	9.87×10^{-10}	9.81×10^{-4}
5	0.40	1.00×10^{12}	0.45	9.87×10^{-8}	9.81×10^{-3}
-	$L_x[m]$	$L_y[m]$	$K_s[Pa]$	$c_f[Pa^{-1}]$	$F[Pa]$
1	50	50	0.51×10^{11}	4.4×10^{-14}	2×10^{1}
2	100	100	1×10^{60}	4.4×10^{-12}	2×10^{3}
3	200	200	1×10^{100}	4.4×10^{-10}	2×10^{6}
4	300	300	1×10^{200}	4.4×10^{-8}	2×10^{8}
5	400	400	1×10^{400}	4.4×10^{-6}	2×10^{14}

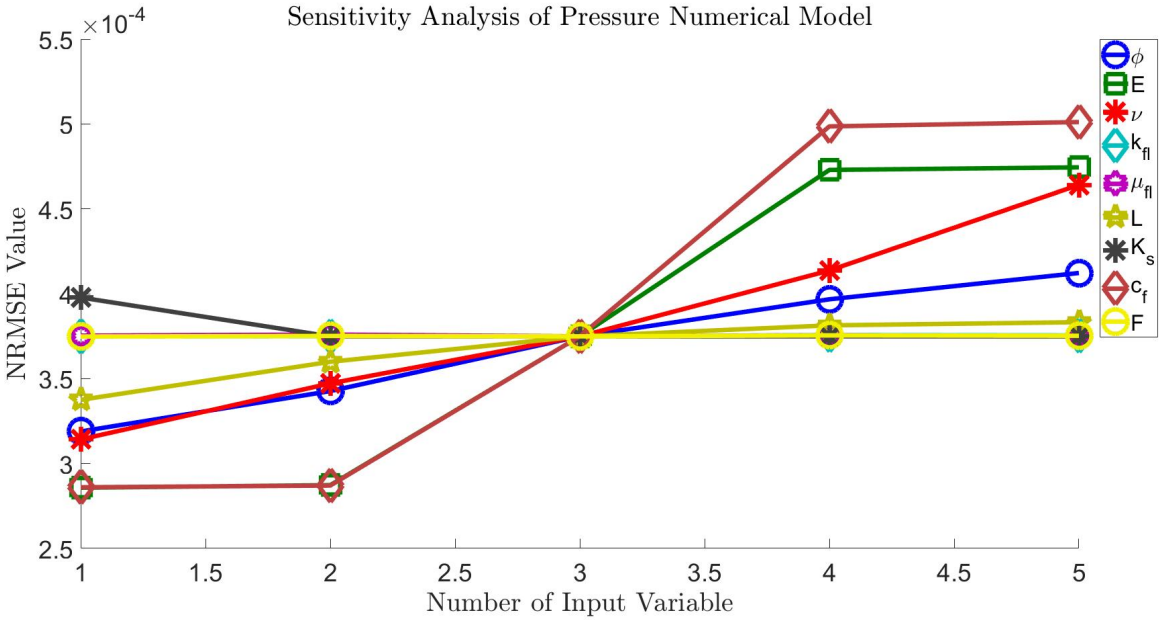


Figure 4.23: *Illustrates the sensitivity of model accuracy in predicting pressure values to range of input variables.*

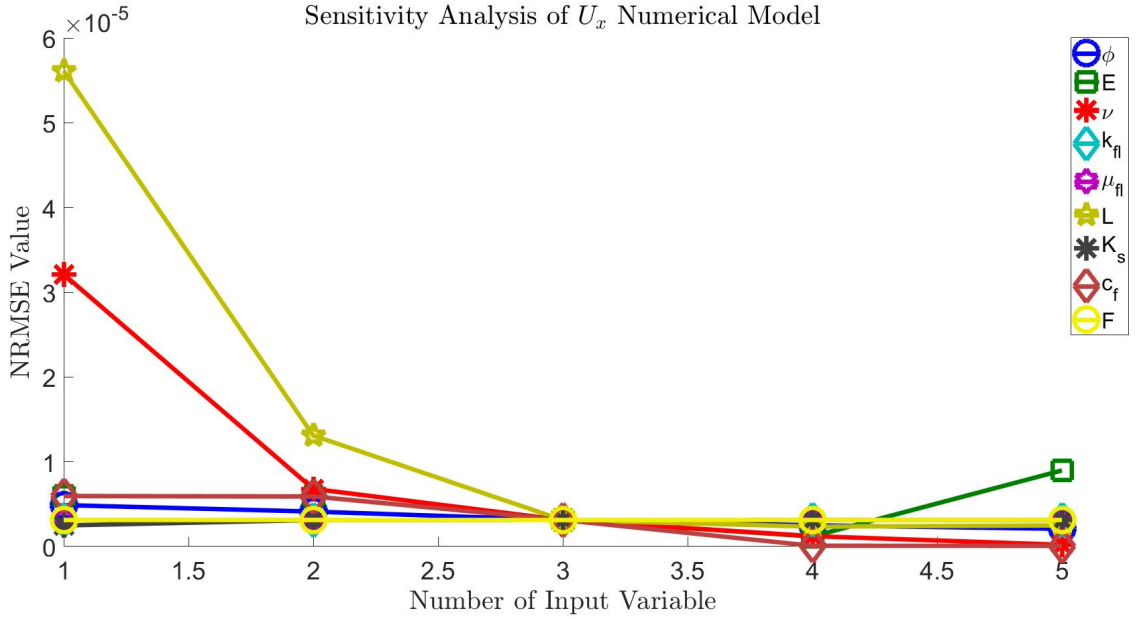


Figure 4.24: *Illustrates the sensitivity of model accuracy in predicting displacement solution in x-direction to range of input variables.*

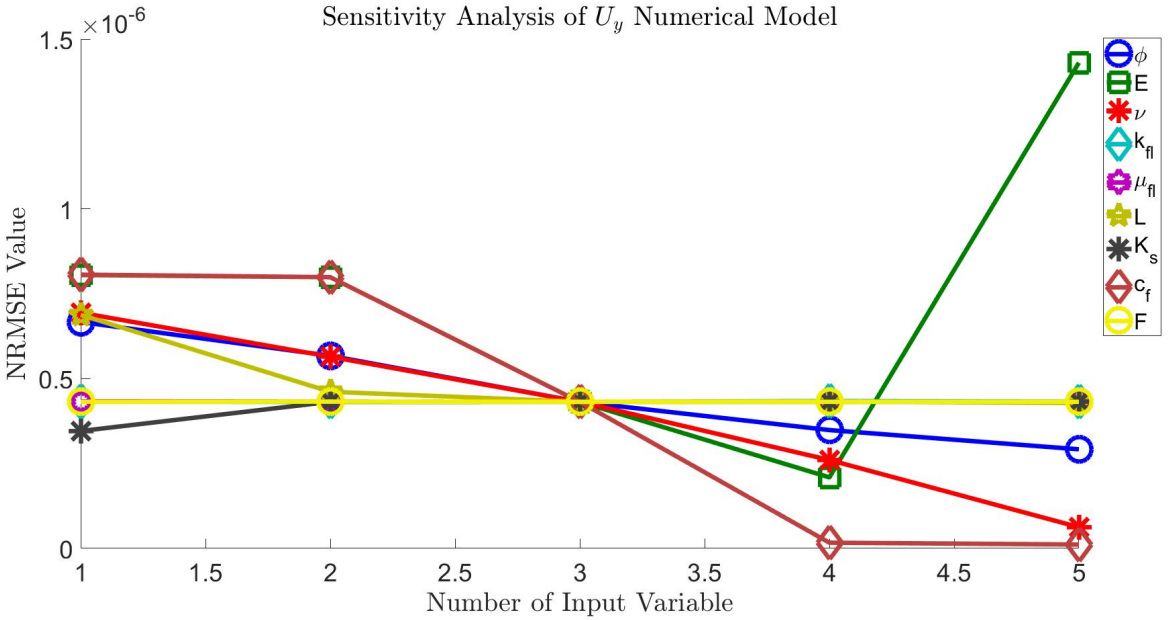


Figure 4.25: *Illustrates the sensitivity of model accuracy in predicting displacement solution in y-direction to range of input variables.*

Figures 4.23, 4.24, and 4.25, illustrate that the developed model is robust in terms of sensitivity to range of input parameters and give an indication of optimizing the model so that by choosing the optimal input parameters the most accurate solutions would be achieved. Dissimilar to Section 4.4.1, the parameters related to model initializations, such as the force on top of the domain, have negligible impact on the error values. In order to investigate the effect of domain size, the error refinement impact should be removed; therefore, the number of grids was chosen so that the grid is constantly equal to 10 $[m]$. The parameters that have the main impact on accuracy of the model are elastic modulus and compressibility of the fluid. After these two parameters, porosity and passion ratio have a linear impact on the error values. These four parameters are the basic parameters to build up the values for storativity coefficient, Lame's constant, and shear modulus, i.e. S, λ, and μ. At high values of elastic modulus error increases since fully incompressible systems in poroelasticity become badly scaled. At high values of fluid compressibility, pressure solution faces error since the model assumption which is a linearly compressible system is violated. Values lower than 0.5×10^{11} is not considered for the solid bulk modulus since no positive roots for the characteristic equation can be found; therefore, no analytical solution for Mandel tests case is found.

Considering the figures and tables in the sensitivity analysis, it has been shown that the fully coupled model renders negligible computational errors in response to the uncertainty in input parameters or the different range of input parameters.

4.5 Practical Problem

The behaviour of a practical problem is numerically predicted. The problem set-up is as illustrated in Figure 4.26 in which a reservoir layer is bounded by a base impermeable layer from the bottom side and covered by a cap rock from the top side. The reservoir layer is producing under open-hole conditions at left and right sides. A no flow boundary condition is assumed for flow at these interfaces since these rocks are impermeable and a drainage boundary condition is assumed at the left and right hand sides of the reservoir model. The base rock allows no displacement of the reservoir rock in the vertical or horizontal directions while the reservoir rock is free to deform in any direction at the other three sides.

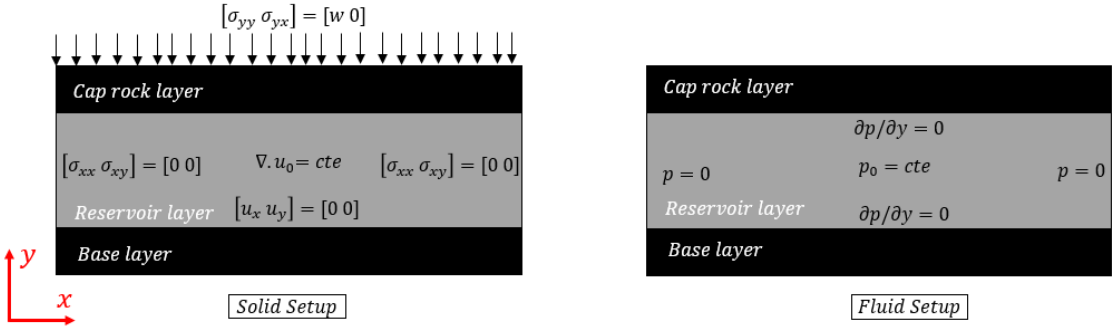


Figure 4.26: *Illustrates the initial and boundary conditions for the problem set-up.*

Model Description

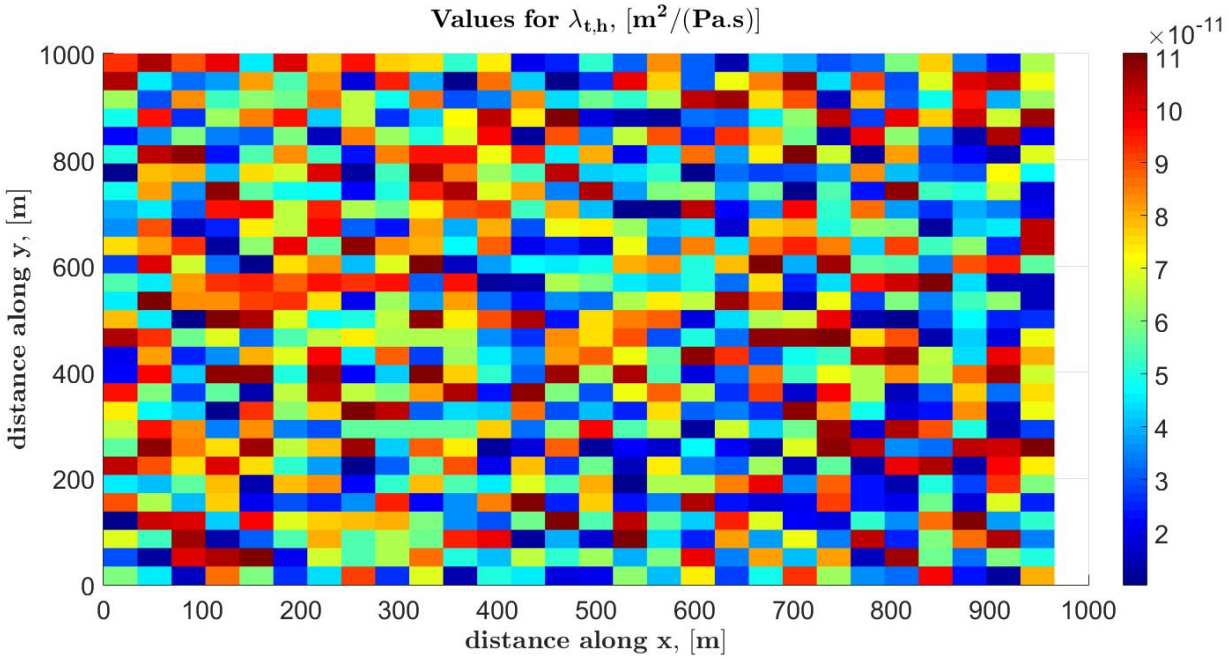
Two homogeneous and heterogeneous test cases are considered both describing an incompressible system of fluid and reservoir rock. The system properties for homogeneous test case are illustrated by Table 4.17. For the heterogeneous test case, the mean values are the same as homogeneous test case with $\pm 10\%$ variation from the mean values- uniformly distributed probability function is used to generate the heterogeneous properties presented in Figures 4.27 and 4.28. For both cases, the problem is descritized on a $(x \times y \times t)$=30 $\times$ 30 $\times$ 300, i.e. dx/dy is 34.48 $[m]$ and dt is 5.7427×10^4 $[s]$. The results are obtained at times equivalent to $[0.003,\ \ 0.033,\ \ 0.333,\ \ 0.666] \times t_c$ equivalent to $\sim [1,\ \ 7,\ \ 66,\ \ 133]$ days.

Table 4.17: *Illustrates values for the problem set-up in a homogeneous reservoir.*

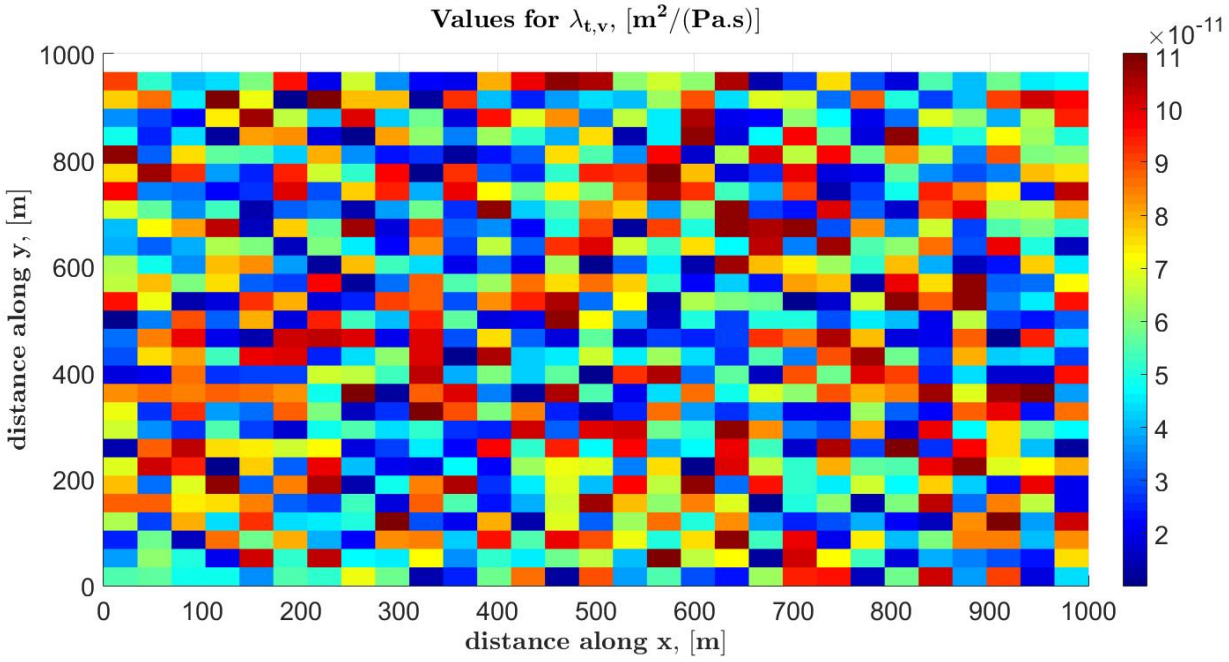
Input Parameter	Unit	Value	Input Parameter	Unit	Value
λ	$[Pa]$	4×10^{10}	b	$[-]$	1
μ	$[Pa]$	4×10^{10}	$L_{x,y}$	$[m]$	1000
λ_t	$[m^2/(Pa.s)]$	1.01×10^{-11}	S	$[1/Pa]$	1.65×10^{-10}
c_v	$[m^2/s]$	0.0580	t_c*	$[s]$	1.73×10^7
$\sigma_{yy,t}/w$	$[Pa]$	2×10^7			

* equivalent to approximately 200 days.

Heterogeneous Reservoir: Fluid Properties



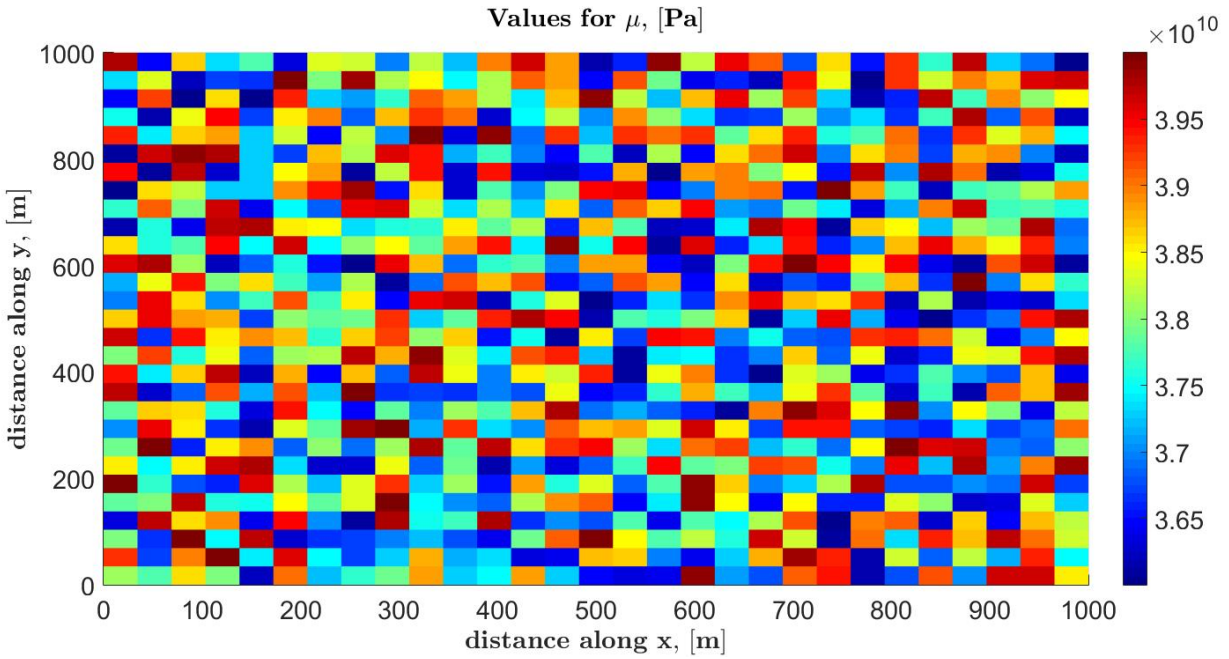
(a) *Illustrates values for vertical fluid mobilities with a mean value of* 1.0061×10^{-11} *and a* ± 10 *variation through the reservoir domain.*



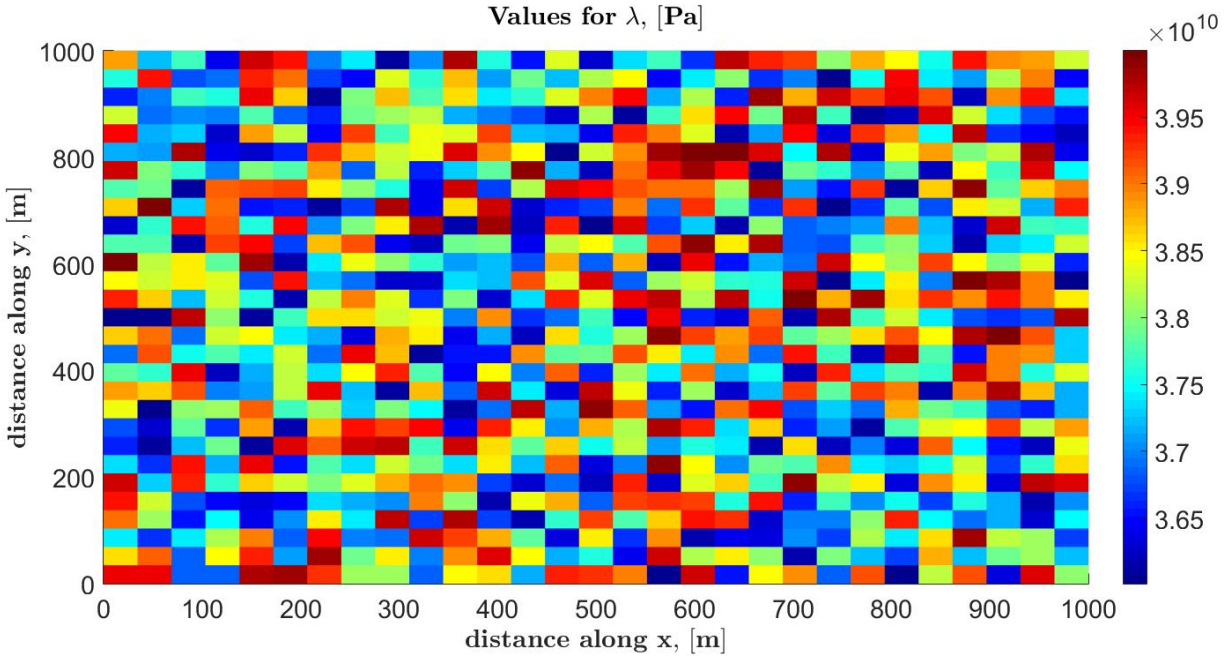
(b) *Illustrates values for horizontal fluid mobilities with a mean value of* 1.0061×10^{-11} *and a* ± 10 *variation through the reservoir domain.*

Figure 4.27: *Illustrates input variables for fluid properties for the heterogeneous test case.*

Heterogeneous Reservoir: Solid Properties



(a) *Illustrates values for shear modulus with a mean value of* 4×10^{10} *and a* $\pm 10\%$ *variation through the reservoir domain.*



(b) *Illustrates values for Lame's constants with a mean value of* 4×10^{10} *and a* $\pm 10\%$ *variation through the reservoir domain..*

Figure 4.28: *Illustrates input variable of solid properties for the heterogeneous test case.*

Initial Conditions for Pressure and Displacements

Initial state of pressure and displacement in y-direction are illustrated in Figures 4.29 and 4.30. Initial displacement in x-direction is considered to be zero [7].

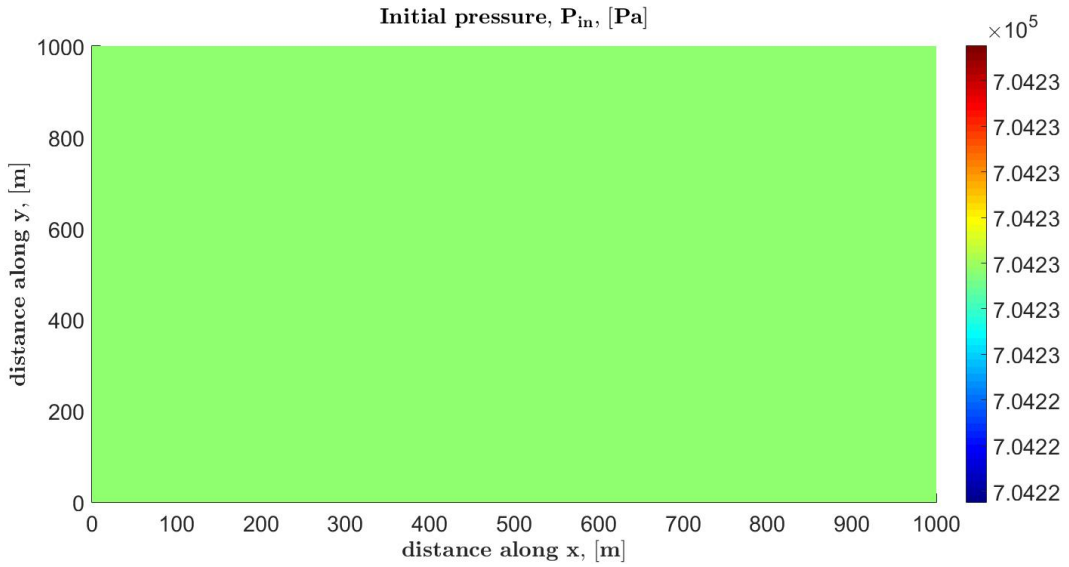


Figure 4.29: *Illustrates the initial condition for pressure throughout the reservoir domain.*

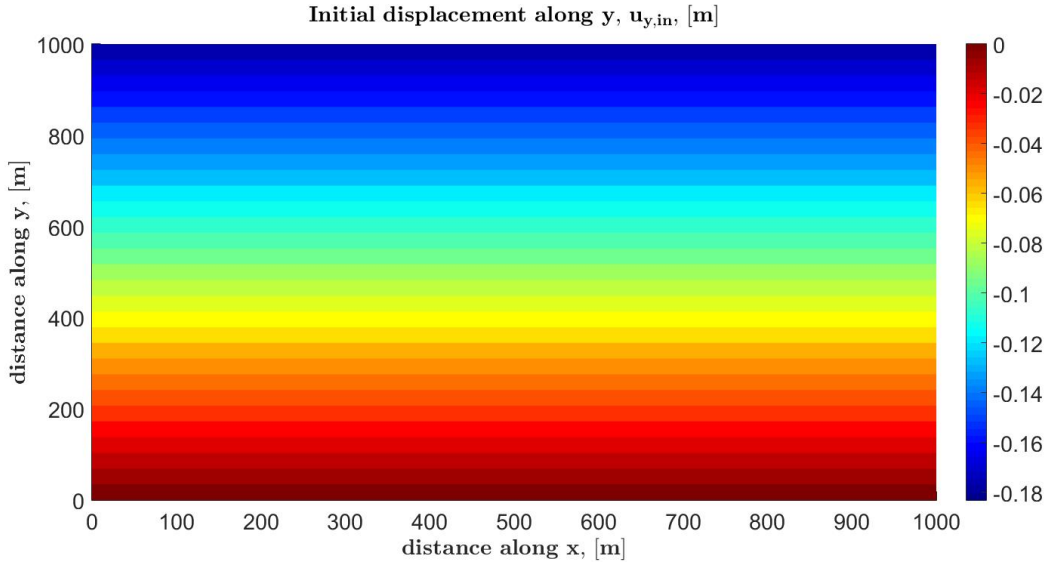
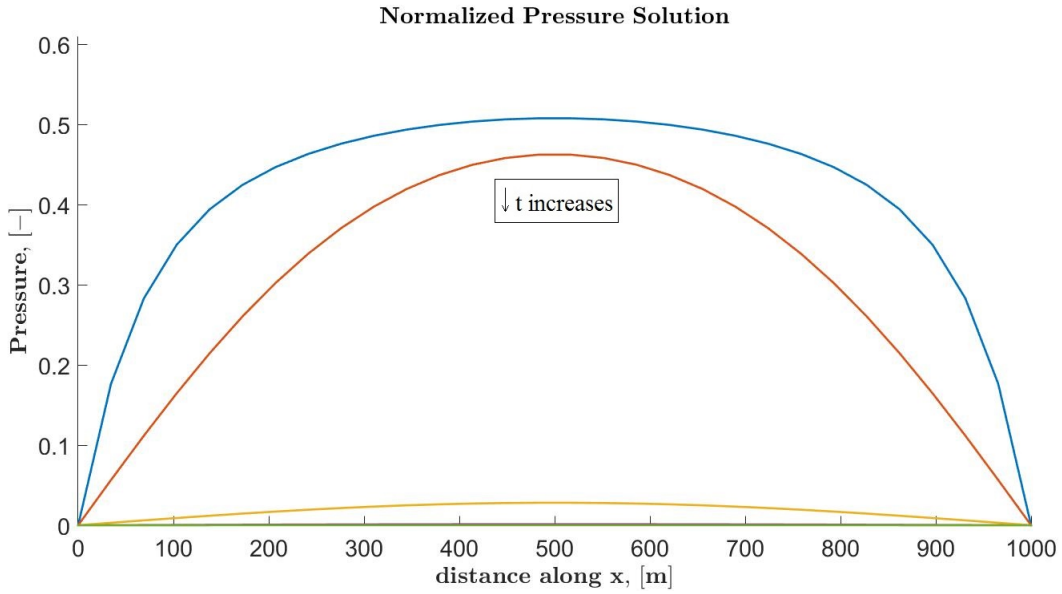


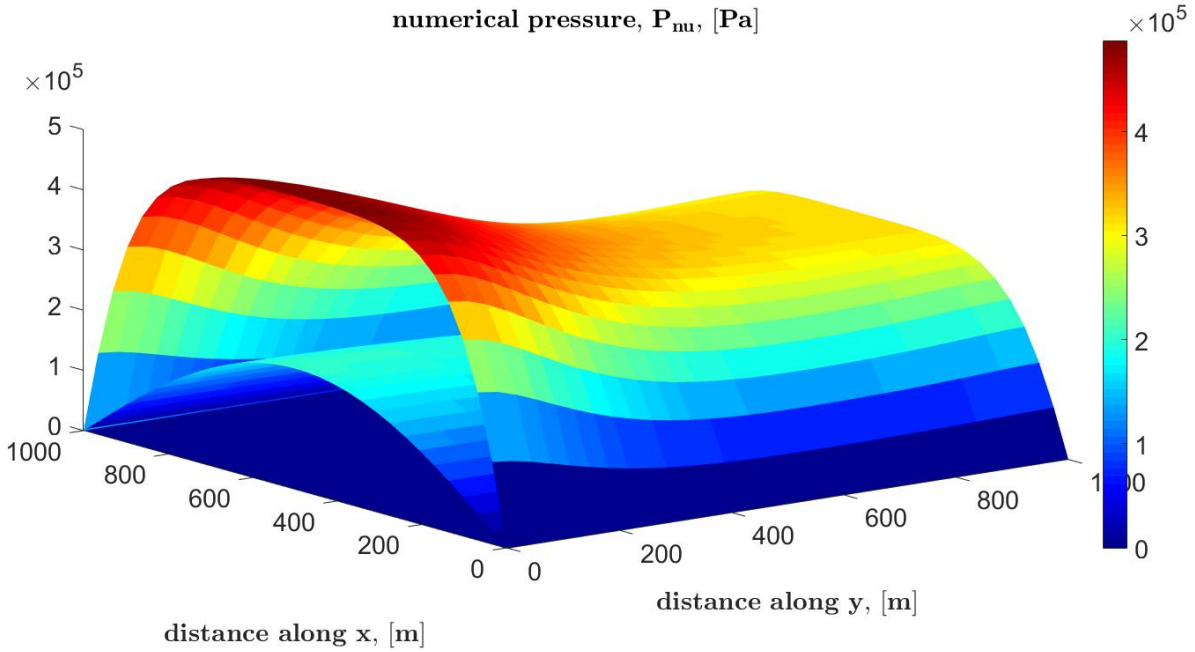
Figure 4.30: *Illustrates the initial condition for displacement in y-direction throughout the reservoir domain.*

[7] the relation used to build the initial conditions for displacement in y-direction and pressure is the same as stated in Appendix : Equations (B.1a) and (B.1c).

Homogeneous Reservoir: Pressure Solution



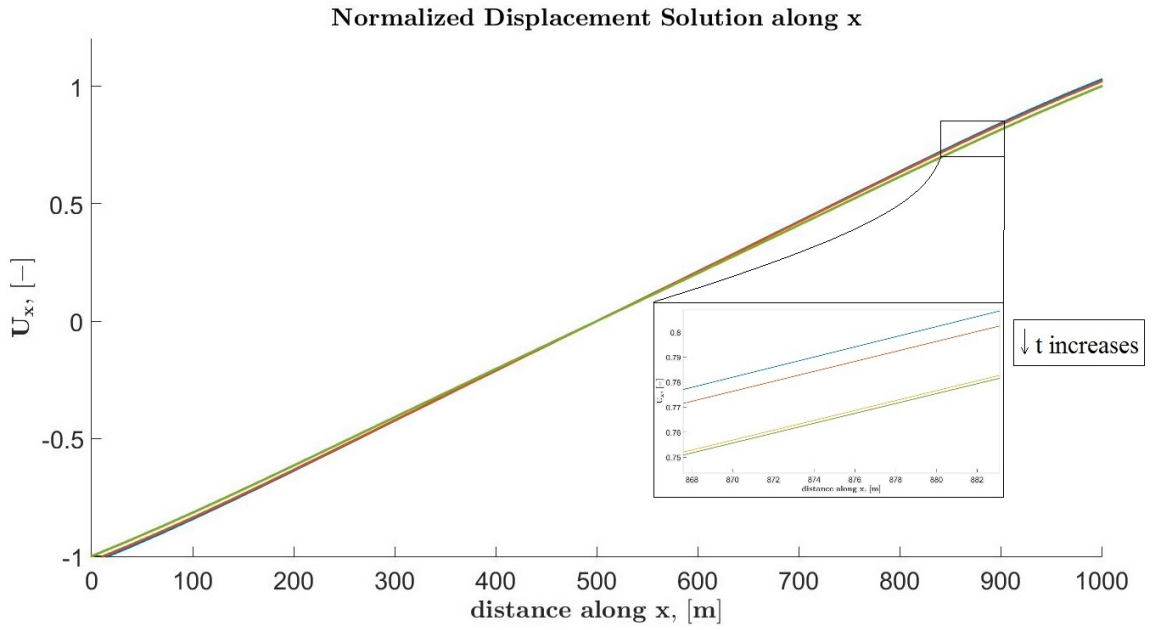
(a) *Illustrates pressure solution for homogeneous reservoir in one dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1, \quad 7, \quad 66, \quad 133]$ *days.*



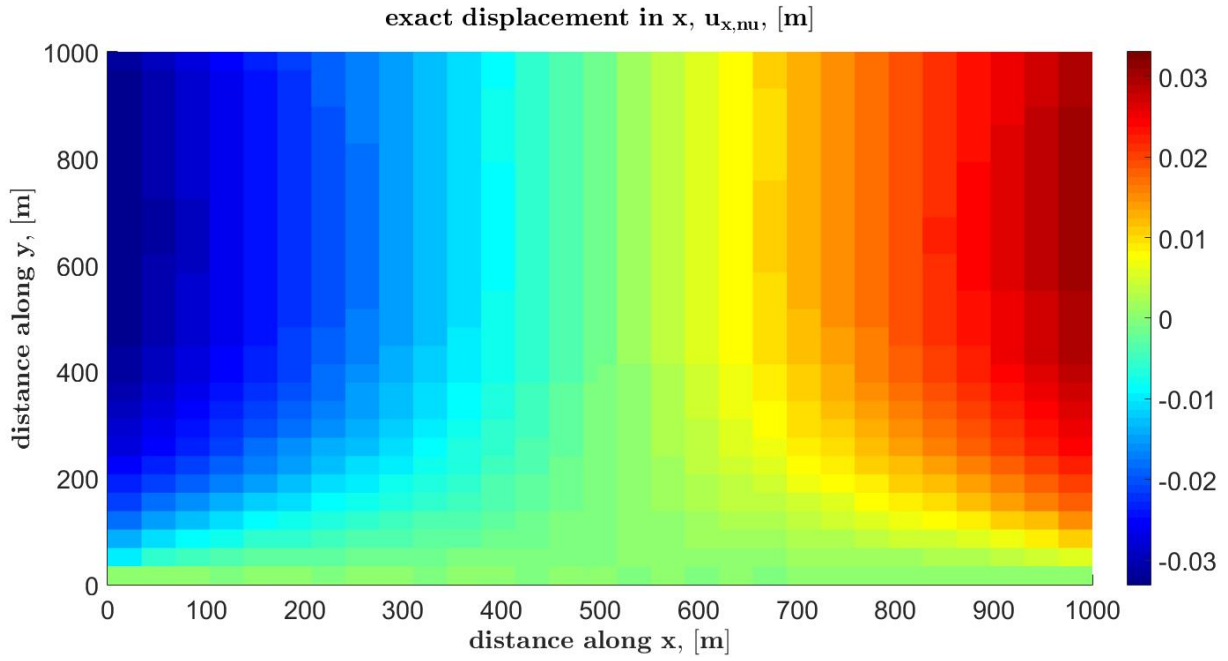
(b) *Illustrates pressure solution for homogeneous reservoir in two dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1, \quad 7, \quad 66, \quad 133]$ *days.*

Figure 4.31: *Illustrates pressure solution in one dimension and two dimension for the homogeneous test case.*

Homogeneous Reservoir: Displacement in x-direction



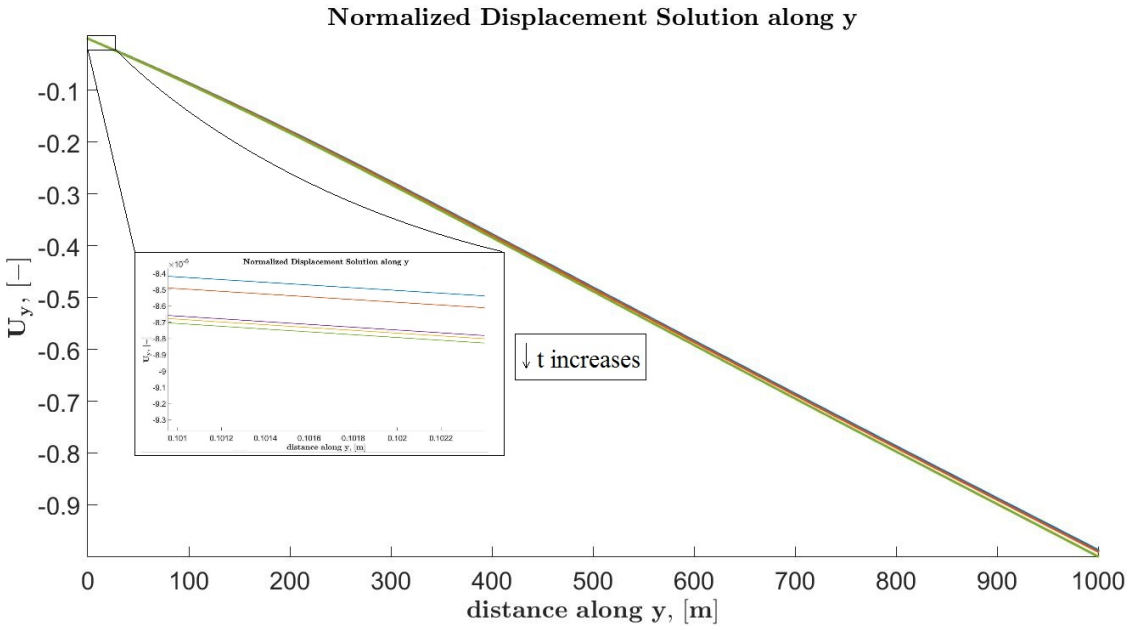
(a) *Illustrates displacement solution in x-direction for homogeneous reservoir in one dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1, \quad 7, \quad 66, \quad 133]$ *days.*



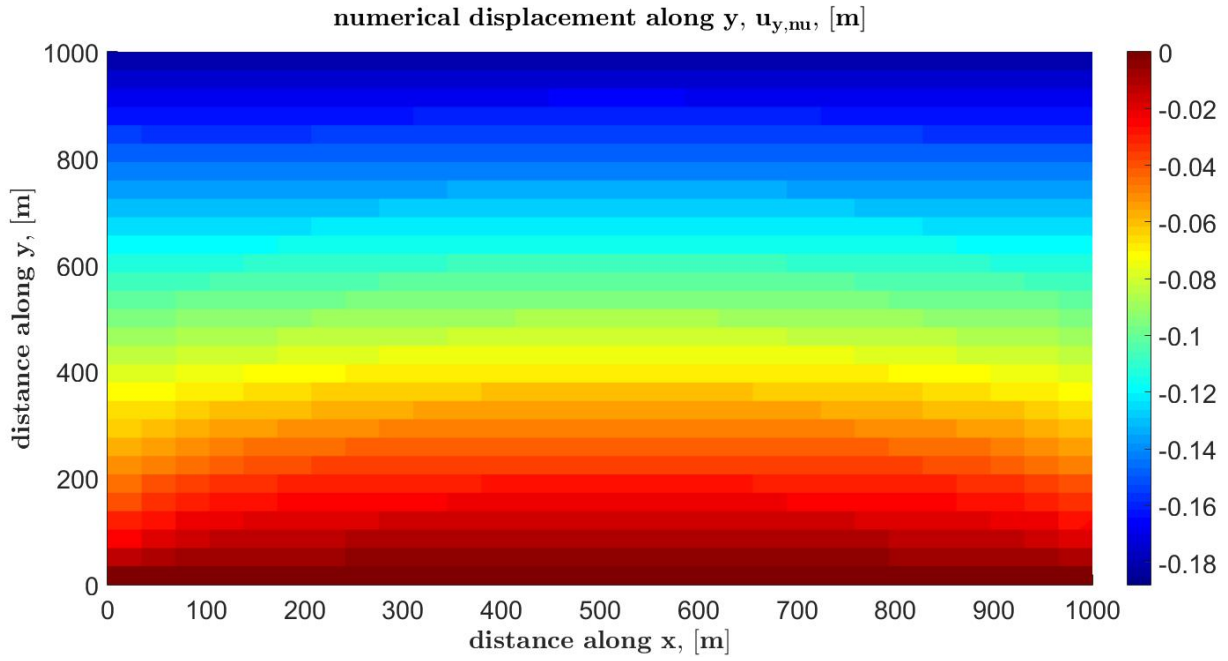
(b) *Illustrates displacement solution in x-direction for homogeneous reservoir in two dimension at the first time step.*

Figure 4.32: *Illustrates displacement solution in x in one dimension and two dimension for the homogeneous test case.*

Homogeneous Reservoir: Displacement in y-direction



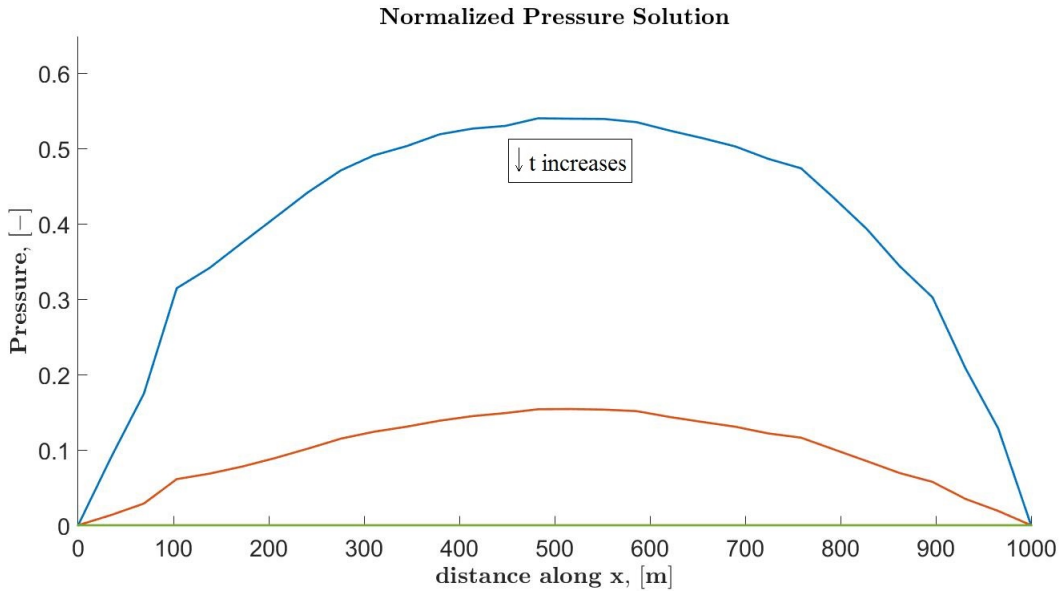
(a) *Illustrates displacement solution in y-direction for homogeneous reservoir in one dimension at times equivalent to* $[0.003,\ \ 0.033,\ \ 0.333,\ \ 0.666] \times t_c$*, i.e.* $\sim [1,\ \ 7,\ \ 66,\ \ 133]$ *days.*



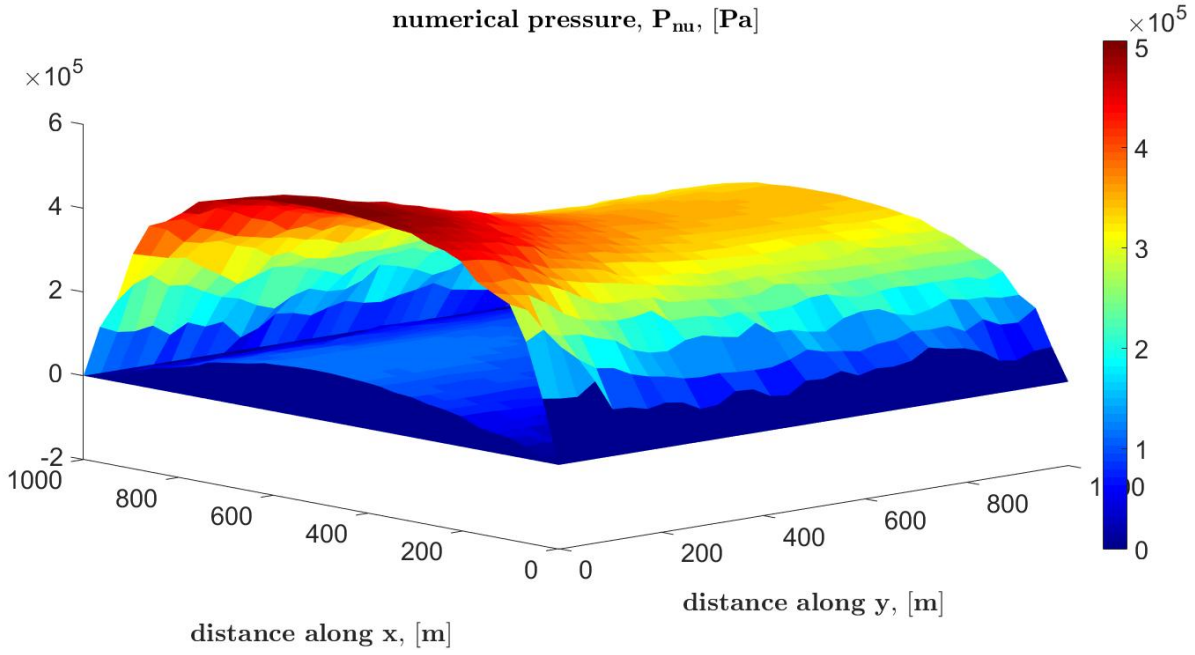
(b) *Illustrates displacement solution in y-direction for homogeneous reservoir in two dimension at the first time step.*

Figure 4.33: *Illustrates displacement solution in y in one dimension and two dimension for the homogeneous test case.*

Heterogeneous Reservoir: Pressure Solution



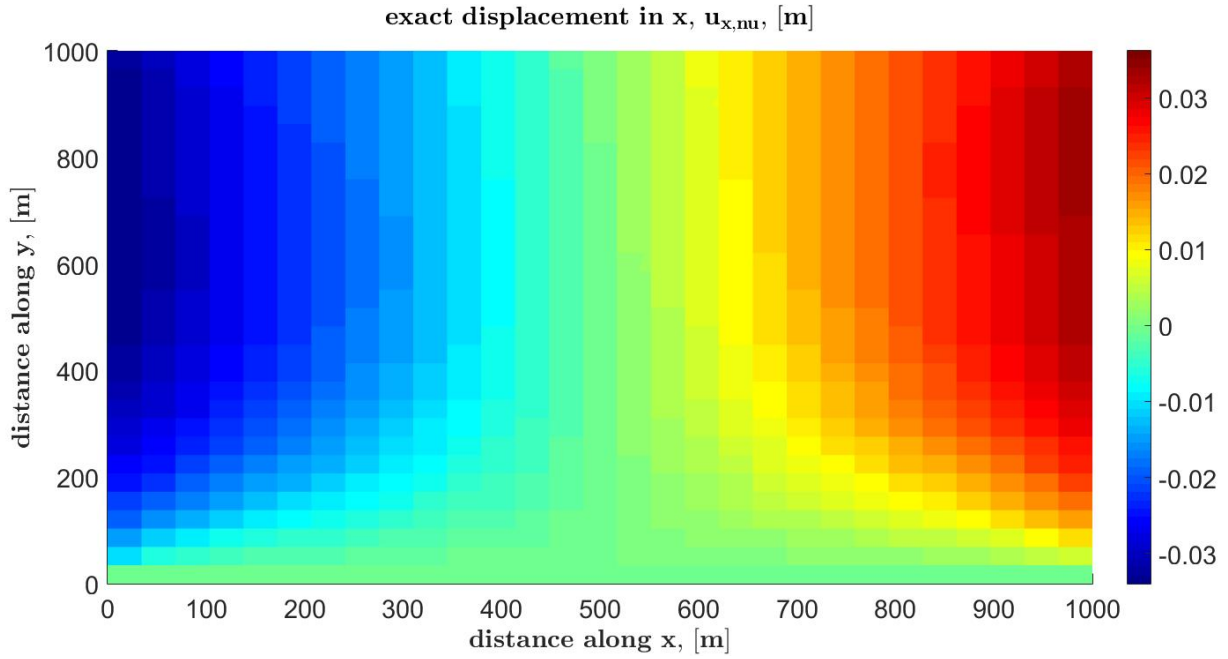
(a) *Illustrates pressure solution for heterogeneous reservoir in one dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1 \quad 7 \quad 66 \quad 133]$ *days.*



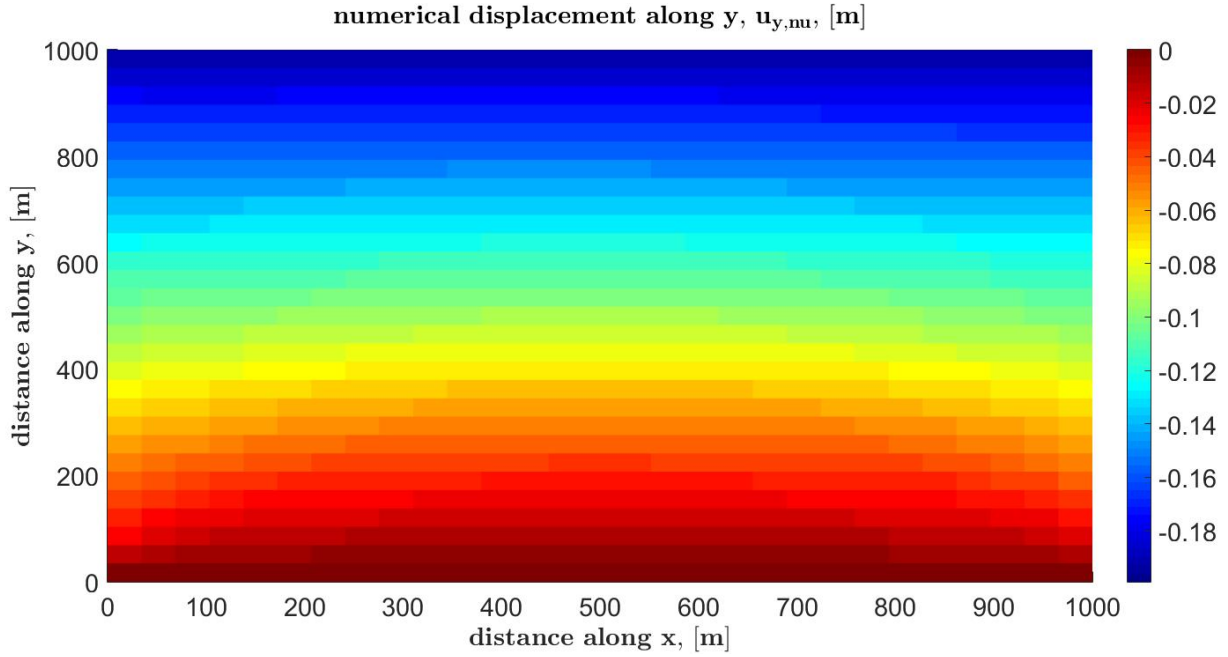
(b) *Illustrates pressure solution for heterogeneous reservoir in two dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1 \quad 7 \quad 66 \quad 133]$ *days.*

Figure 4.34: *Illustrates pressure solution in one dimension and two dimension for the heterogeneous test case.*

Heterogeneous Reservoir: Displacement Solutions



(a) *Illustrates displacement solution in x-direction for heterogeneous test case throughout the domain at the first time step.*



(b) *Illustrates displacement solution in y-direction for heterogeneous test case throughout the domain at the first time step.*

Figure 4.35: *Illustrates displacement solutions for heterogeneous reservoir in two dimension at the first time step.*

Interpretation of Results

In this example, a highly consolidated reservoir is considered. Therefore, displacement solutions renders low level of changes through time as it is expected and shown by Figures 4.32a and 4.33a. By comparing the pressure solutions in Figures 4.31a, 4.34a, and 4.36, it is evident that not considering displacement effects in the production might result in an underestimation of the reservoir pressure drop. As by comparing the values for the second time step, a conventional reservoir model anticipates a pressure value of $\sim 0.8p_0$ while the poroelastic homogeneous model predicts a reservoir pressure of $\sim 0.4p_0$ which is half of the value predicted by the conventional fluid flow equations. This value is even smaller in presence of heterogeneity which is expected to occur since in the heterogeneous case higher mobility is available to the fluid and the fluid would find the easiest path to be produced. Moreover, the porous rock will compress faster in case of heterogeneity. Therefore, the fluid will be produced at a faster rate resulting in a reservoir pressure decline in a faster rate and this effect is observed with a higher intensity at the sides of the domain where higher solid deformation occurs.

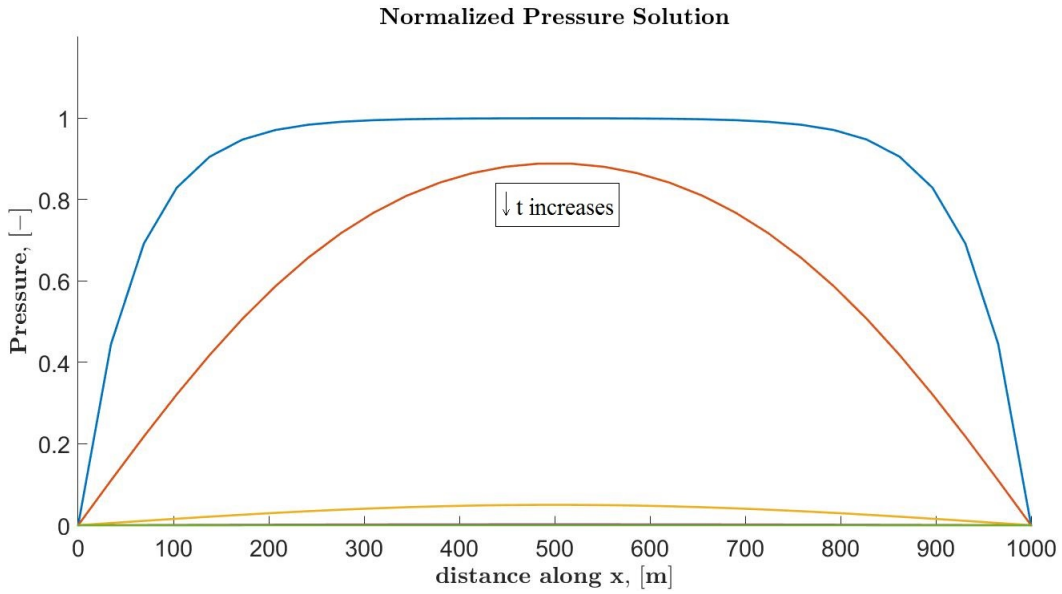


Figure 4.36: *Illustrates uncoupled pressure solution for homogeneous reservoir reservoir in one dimension.*

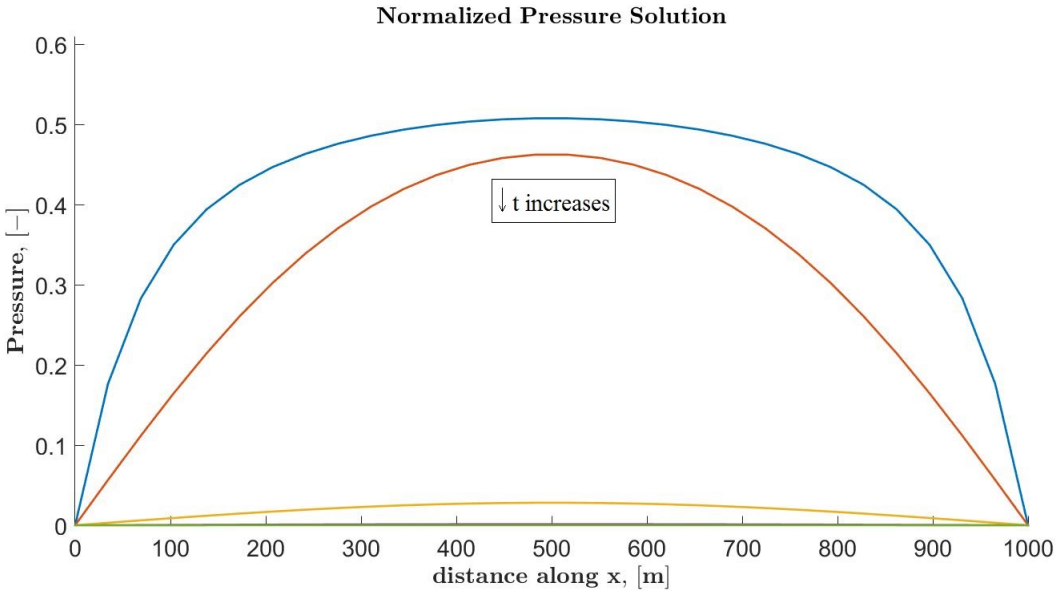


Figure 4.31a: *Illustrates pressure solution for homogeneous reservoir in one dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1, \quad 7, \quad 66, \quad 133]$ *days. (repeated from page 75)*

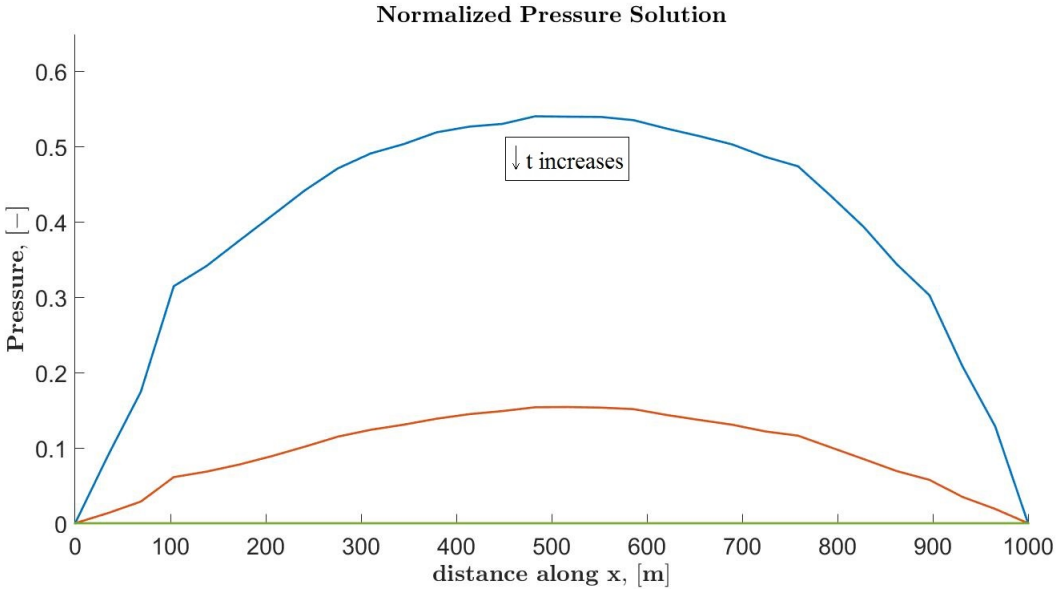


Figure 4.34a: *Illustrates pressure solution for heterogeneous reservoir in one dimension at times equivalent to* $[0.003, \quad 0.033, \quad 0.333, \quad 0.666] \times t_c$, *i.e.* $\sim [1 \quad 7 \quad 66 \quad 133]$ *days. (repeated from page 78)*

5 Discussion

In this section, firstly, the applications of the proposed models are discussed. Next, some hints are discussed on how to improve the developed model to be suitable for other configurations and to be functional under more complex assumptions. Finally, the applicability of each possible extension is described.
The proposed sequentially cell-centred and fully coupled vertex-centred finite volume developed models can be applied to predict (oil/water) reservoir behaviour in the subsurface under two-dimensional plane strain condition in Cartesian coordinates. In such reservoirs, the isothermally poroelastic assumption is a valid assumption. It is common practice in the field of petroleum engineering to consider all oil properties solely dependent on reservoir pressure while assuming constant temperature. Temperature variation is only considered in pipes and surface facilities. Hence, isothermal condition is a valid assumption. Besides, such reservoirs are consolidated enough that they rarely undergo large strains. Therefore, elastic behaviour for the solid will be a valid assumption.
By comparison of the two models in Section 4.1, the fully coupled model outperforms the sequential model in terms of computational time and error. This comparison was carried out based on an adopted method from the work carried out by Luo et al. (2015). On the other hand, the sequentially coupled model provides lower computational costs in terms of coupling domains with different size. For instance, a solid domain much larger than the fluid domain increases the computational time of the fully coupled model in comparison with the sequentially coupled model. Integration of thermal effects is expected to be easier with the sequentially coupled method, as well. The cell centred and vertex centred discretization schemes treat the boundaries of the domain differently. Capturing the total stress boundary conditions,which is a sum up of three terms $\partial u_x/x$, $\partial u_y/y$, and p, with the finite volume method is a challenging task. The cell centred method requires additional unknown nodes and extrapolation functions throughout the solid domain in order to properly capture the boundary conditions. It was found through numerical investigations that the number of unknown nodes in the proposed finite volume method is the optimum number of unknowns; however, improved extrapolation functions or coupling scheme would improve further the efficiency of the current cell centred finite volume model. In general, both methods perform well and adequately; however, the fully coupled vertex centred model is preferred.
The fully coupled model is studied under two different configurations: displacement (boundary condition) and stress (boundary condition). These two configurations have been studied since it is either displacement or total stress that can be physically measured. It is shown that the proposed model gives sufficiently accurate (the order of normalized error for displacement and total stress configuration is 10^{-3} and 10^{-4}, subsequently, with relatively coarse grid size) results under both configurations. However, there are slight pressure deviations from the exact result if a stress boundary condition is applied to more than one boundary throughout the domain. However, these errors are negligible and are natural since an over-determination in boundaries occurs for both fluid and solid equations. The stress (boundary condition) configuration can be further improved by applying numerical stabilizers to the schemes. With the proposed model, the prediction errors are low and acceptable. By considering a problem domain, larger than the

domain of interest, the boundary condition errors can be discarded; and therefore, the current model can adopt to the level of accuracy that the user expects. Overall, the fully coupled vertex centred finite volume model works efficiently under both configurations and renders low error with various input parameters from the lowest physically possible values to the highest. The proposed vertex centred finite volume method can be used as a benchmark simulator for other finite volume methods developed for poroelasticity.

An application example of this model has been shown in an oil field with two wells producing from a very consolidated sandstone under open hole production conditions in homogeneous and heterogeneous reservoirs, with 20% of heterogeneity being included throughout the reservoir layer. It has been shown that production prediction can be highly underestimated if the solid deformations effects are not been included. This underestimation increases in case of heterogeneous reservoir. In the following sections titled as "Uniaxial Strain" and "Plane Strain in Cartesian Coordinates", the applications of the current model are further discussed. In Section 5.1, the reader is provided with hints on how to further improve the current model.

Uniaxial Strain

Uniaxial strain is the case when only displacement in one direction is allowed. In other words, the poroelastic problem can be considered as one dimensional. This behaviour is observed in many laboratory tests where a rock sample,i.e. rock plug, is surrounded by the plug holder which is a rigid body. Therefore, this model can be used to validate such laboratory tests or to predict the results of such tests. Moreover, initiation of natural hydraulic fractures can be predicted by the Terzaghi test case, (Engelder and Lacazette, 1990). In Terzaghi test case, a vertical natural hydraulic fracture will be created as the negation of horizaontal compressive stress and pore-pressure approaches zero, i.e. "$\sigma_{xx} - p \sim 0$". The same principle can be applied to predict the cap rock integrity when injecting fluids in the subsurface, (Mourgues et al., 2011). Additionally, sedimentation and erosion can be considered as compressive and tensile stresses, respectively, applied on deeper buried layers. The current model can be used to predict the changes in the pressure and displacements of layers, which are buried at the appropriate depth, through sedimentation and erosion processes, (D Bredehoeft and Hanshaw, 1968; Neuzil and Pollock, 1983). Lastly, a reservoir's behavior can be predicted under any periodic or continuous and cyclic or constant unloading/loading if the extend of the reservoir in the two horizontal direction can be considered as infinite compared to it's vertical height.

Plane Strain in Cartesian Coordinate

A plane strain condition in Cartesian coordinate happens when one horizontal displacement is zero. Clearly, it is also assumed that the body forces in this direction are also zero. This assumption is valid when a reservoir is under production/injection with a long line of wells. This model can be applied to predict the subsidence due to oil extraction or elevation due to fluid injections in the subsurface (Segall, 1985). In addition, fault slip is measure in the fault

plane (Rudnicki, 1986, 1987; Rudnicki and Hsu, 1988; Rudnicki and Roeloffs, 1988; Wang, 1997); therefore, plane strain assumption in Cartesian coordinate is a common assumption for this case. Prediction of (initiation of) sand production/well-bore failure/appropriate gravel pack properties is another application of the current model (Yi et al., 2004).

5.1 Limitations and Delimitations

The word limitation refers to potential weaknesses in a study which exists in every work. Besides, the word delimitation implies the boundaries of a study which are defined by the objectives and research questions. Knowing limitations and delimitations enables an individual to properly apply and further extend the developed model when it is necessary. Therefore, this section is evolved around discussions on limitations and delimitations of the developed model.

5.1.1 Gravity Effects

Gravity force is always present and acts as a body force in the equilibrium equation and as a contributing or resisting drive force for fluid to flow; however, the gravity effect is negligible in the consolidation equations if the medium is adequately thin or hard. In case of highly unconsolidated materials or in then unconsolidated reservoirs gravity effects might play a significant role. A dimensionless gravity parameter is defined to determine whether gravity effect is negligible or not as shown in Equation (5.1), (Mei, 1985; Tsai et al., 2006; Tseng et al., 2008; Musivand Arzanfudi, 2016). For the gravity parameter,i.e. M, values in range of zero to unity are realistic and values below 0.05 can be considered as negligible (Tseng et al., 2008).

$$M = \frac{(1-\phi)g\Delta\rho h}{D} \tag{5.1}$$

where M is gravity parameter $[-]$, ϕ is porosity $[-]$, g is gravitational constant $[m/s^2]$, $\Delta\rho$ is difference between density of solid and fluid $[kg/m^3]$, h is layer thickness $[m]$ and D is p wave modulus $[Pa]$. If the gravity parameter is not small, gravity effect can be considered by modifying the formerly stated equations to Equation (5.2).

$$\nabla.\sigma = \begin{bmatrix} \frac{\partial\sigma_{xx}}{\partial x} + \frac{\sigma_{yx}}{\partial y} - \rho_b g_x - f_x \\ \frac{\partial\sigma_{xy}}{\partial x} + \frac{\sigma_{yy}}{\partial y} - \rho_b g_y - f_y \end{bmatrix}^T_{2*1} = 0 \tag{5.2a}$$

$$b\frac{\partial\nabla.u}{\partial t} + S\frac{\partial p}{\partial t} + \nabla.(-\lambda_t\nabla p + \rho_f\mathbf{g}) = 0; \tag{5.2b}$$

where g_x/g_y is gravitational field components $[m/s^2]$ and ρ_b is bulk density $[kg/m^3]$.

5.1.2 Non-linear Poroelasticity

In production from tight reservoirs, for example coal, peat or shale, non-linear behavior between the fluid pressure and the fluid flux will be observed. In these cases, the fluid is normally an incompressible fluid. However, in these cases since the strain values are small, linear strain stress can be considered. In other words, non linearity in the poroelastic problem occurs when it can be assumed that the stress-strain relationship is linear though changes in stress and strain cause local changes in material properties. This nonlinearity can be applied to the current model by considering the porosity and the permeability of the medium as a function of the pressure and displacements in the medium. This functionality has a general form as stated in Equation (5.3) (Palmer et al., 1996; Berger et al.; Bemer et al., 2001; Tavakoli and Ferronato, 2013; Luo et al., 2015; Gaspar et al., 2016).

$$k = Ak_0 e^{-(B\nabla.u - bp)} \tag{5.3a}$$

$$\phi = A^{'}\phi_0 e^{-(B^{'}\nabla.u - bp)} \tag{5.3b}$$

where A, $A^{'}$, $B^{'}$ and B are positive constants and k_0 and ϕ_0 are the initial porosity and permeability values. To add this non linearity to the current model Equation (5.3) will be added to the set of equations solved at each timestep. This non linearity is usually taken into account for highly unconsolidated solid media such as coal.

5.1.3 Poroelastoplasticity

At relatively larger strain values, a linear relationship between strain and stress is no longer a valid assumption. This will be the case when modeling highly unconsolidated materials, such as soil. Besides, for modeling near well-bore behavior a poroelastoplastic model will render more accurate results. To model poroelastoplasticity, the strain-stress relationship will be non linear; however, the relationship between strain and stress derivatives' will remain linear. Therefore, this behavior can be formulated for moderate strain values as stated in Equation (5.4) (Ma et al., 2016; Tang et al., 2015). The fluid flow equation does not change

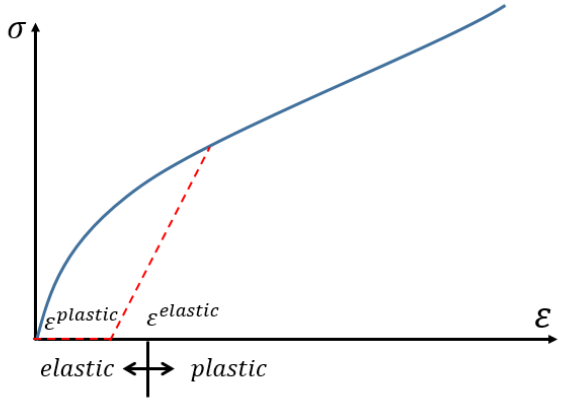


Figure 5.1: *Illustrates Mohr-Coulomb diagram and the fact that elastoplastic stress can be predicted by elastic and plastic strain behaviours.*

$$d\boldsymbol{\sigma}^{'} = -2\mu(d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}^{p}) - \lambda tr(d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}^{p}) \tag{5.4}$$

where $d\boldsymbol{\varepsilon}^{p}$ is incremental plastic strain tensor on the plane $[-]$. Therefore, equilibrium and fluid equations can be rewritten as in Equation (5.5). Plastic strain is obtained based on constitutive relationships in which stress and displacement values are function variables and dilation and friction angles and probably other material properties are function constants.

$$\nabla.[\mu\nabla d\mathbf{u} + \mu\nabla^{T} d\mathbf{u} + Itr(\nabla d\mathbf{u})] = \nabla.[2\mu(d\boldsymbol{\varepsilon}^{p}) + \lambda Itr(d\boldsymbol{\varepsilon}^{p})] + b\nabla p \tag{5.5a}$$

$$S\frac{\partial p}{\partial t} + \nabla.(-\lambda_{t}\nabla p) = -b\frac{\partial \nabla.\mathbf{u}}{\partial t}; \tag{5.5b}$$

where $tr(m)$ is the trace of the m matrix, I is the identity matrix.

5.1.4 Thermoporoelasticity

In this report, one of the fundamental assumptions is the isothermal condition; however, if the area is geothermally active, the thermo-hydrodynamic coupling could not be ignored. In these cases, if it is assumed that elastic behaviour remains valid, thermoelastic behaviour of the medium can be modelled as described in Equation (5.6) (Demirdžić and Martinović, 1993;

Bai and Abousleiman, 1997; Coussy, 2007; Wang et al., 2009; Li et al., 2016).

$$(2\mu+\lambda)\frac{\partial^2 u_x}{\partial x^2}+(\mu+\lambda)\frac{\partial^2 u_y}{\partial x\partial y}+\mu\frac{\partial^2 u_x}{\partial y^2}=b\frac{\partial p}{\partial x}+\beta\frac{\partial T}{\partial x}+f_x \tag{5.6a}$$

$$(2\mu+\lambda)\frac{\partial^2 u_y}{\partial y^2}+(\mu+\lambda)\frac{\partial^2 u_x}{\partial x\partial y}+\mu\frac{\partial^2 u_y}{\partial x^2}=b\frac{\partial p}{\partial y}+\beta\frac{\partial T}{\partial y}+f_y \tag{5.6b}$$

$$S\frac{\partial p}{\partial t}+\nabla.(-\lambda_t\nabla p)=-b\frac{\partial \nabla.u}{\partial t}+\alpha_t\frac{\partial T}{\partial t}+q \tag{5.6c}$$

$$\underbrace{-K^*\nabla^2 T}_{\text{conductive term}}+\underbrace{s(v.\nabla)T}_{\text{convective term}}+\beta T_0 b\frac{\partial \nabla.u}{\partial t}+s^*\frac{\partial T}{\partial t}=q_T \tag{5.6d}$$

where α_t is the total thermal expansion coefficient $[1/K]$; defined as $\phi\alpha_f+(1-\phi)\alpha_s$ by fluid and solid thermal expansion factors, i.e. α_f and α_s, respectively, β is thermal expansion factor $[Pa/K]$; defined as $\alpha_h(3\lambda+2\mu)$, K^* is thermal conductivity $[N/(m.K)]$; defined as $\phi*{K_f}^*+(1-\phi)*{K_s}^*$ by fluid and solid thermal conductivity shown respectively by ${K_f}^*$ and $*{K_s}^*$, s is intrinsic heat capacity $[(J.Kg)/(m^3.K)]$; defined as $\rho_f C_f\phi$ with C_f being fluid heat capacity $[J/K]$, T_0 is the initial temperature $[K]$ and s^* is total intrinsic heat capacity $[(J.Kg)/(m^3.K)]$; defined as $s+\rho_s C_s(1-\phi)$ with C_s being solid heat capacity $[J/K]$ and q_T is heat source term $[W]$. A simplified one dimensional specification of Equation (5.6) can be obtained as stated in Equations (5.7) considering heat transfer via convection is negligible and source terms are zero.

$$(\lambda+2\mu)\frac{\partial^2 u_x}{\partial x^2}=b\frac{\partial p}{\partial x}+\beta\frac{\partial T}{\partial x} \tag{5.7a}$$

$$S\frac{\partial p}{\partial t}-\lambda_t\frac{\partial^2 p}{\partial x^2}=-b\frac{\partial^2 u_x}{\partial t\partial x}+\alpha_t\frac{\partial T}{\partial t} \tag{5.7b}$$

$$s^*\frac{\partial T}{\partial t}-K^*\frac{\partial^2 T}{\partial x^2}=-\beta T_0\frac{\partial^2 u}{\partial t\partial x} \tag{5.7c}$$

Under thermo-poroelastic condition, an initial condition for temperature will be added. Besides, boundary conditions for temperature will be either simple Dirichlet or Neumann; however, note should be taken that stress terms will be dependent on temperature, as well. Additionally, changes in fluid properties due to changes in temperature should be considered.

5.1.5 Compressiblity and Incompressiblity

For fluids that are highly compressible, especially if fluid type is gas, the term "$b\frac{\partial \nabla.u}{\partial t}$" becomes negligible compared to the term "$S\frac{\partial p}{\partial t}$". In other words, coupling strength, see Equation (2.1), becomes so low that the problem becomes decoupled, (Verruijt, 2016). Therefore, the governing equation for flow can be modified as stated in Equation (5.8).

$$S\frac{\partial p}{\partial t}+\nabla.(-\lambda_t\nabla p)=q \tag{5.8}$$

This equation can be solved independently of equilibrium condition and then solid equation variables, i.e. displacements and stresses, can be found via correlations or by solving equilibrium equations as stated before.
On the other hand, for the system of an incompressible fluid and an unconsolidated rock, coupling strength becomes higher since the value for the S term becomes lower. As a result, the term "$S\frac{\partial p}{\partial t}$" can be omitted in the fluid equation, see Equation (5.9).

$$\nabla.(-\lambda_t \nabla p) = -b\frac{\partial \nabla.u}{\partial t} + q \tag{5.9}$$

Finally, in fully incomprehensible systems, the governing equation reduces to the conventional fluid equation for incompressible systems with no coupling or time-dependent terms, see Equation (5.10).

$$\nabla.(-\lambda_t \nabla p) = q \tag{5.10}$$

5.1.6 Multi-phase Fluids

In order to model the multi-phase behaviour of fluids in poroelasticity, a fluid equation will be added to the system to account for saturation changes in each phase see Equation (5.11) (Lewis et al., 1998; Bataee et al., 2016; Doster et al., 2015; Bjørnarå et al., 2016; Musivand Arzanfudi, 2016).

$$\phi\frac{\partial S_\alpha}{\partial t} + \frac{1}{\rho_\alpha}\nabla.\left[\rho_\alpha \frac{k_\alpha}{\mu_\alpha}(-\nabla p_\alpha)\right] = q_\alpha \tag{5.11a}$$

$$\phi\frac{\partial S_\beta}{\partial t} + \frac{1}{\rho_\beta}\nabla.\left[\rho_\beta \frac{k_\beta}{\mu_\beta}(-\nabla p_\beta)\right] = q_\beta \tag{5.11b}$$

where S_α is the saturation of the phase α $[-]$, k_α is the effective permeability of the phase α $[m^2]$, p_α is the pressure of the phase α $[Pa]$ and q_α is the source term of the phase α $[1/s]$ .The pressure term in the equilibrium equation will be calculated as stated in Equation (5.12b).

$$\nabla.[\mu\nabla d\mathbf{u} + \mu\nabla^T d\mathbf{u} + Itr(\nabla d\mathbf{u})] = b\nabla p \tag{5.12a}$$

$$p = S_\alpha p_\alpha + S_\beta p_\beta \tag{5.12b}$$

5.1.7 Fractures

Fractures are common phenomena in the subsurface. For different fracture densities, different approaches are employed to model them. Here, two popular methods to model fractures are discussed:

1. Dual porosity/permeability
2. Discrete fracture network

The former approach is applicable to models with high fracture density and where fractures follow a known pattern. In this case, the flow and solid equations are modified and a third equation is added to account for mass transfer in fractures. In other words, fractures can be considered as a new medium. However, in a discrete fracture network where the fracture density is lower and fractures do not have a specific pattern, the modeling fractures would be more complicated and iterative loops and implementations of appropriate convergence criteria would be needed. The reason is that fractures would be an inseparable part of the solid and fluid medium. (Caillabet et al., 2000; Granet et al., 2001; Lei et al., 2014; Detournay and Peirce, 2014; Bryant et al., 2015; Talebian, 2015; Musivand Arzanfudi, 2016)

5.1.8 Axisymmetry/Plane Strain in Polar Domain

The plane strain condition in a polar domain refers to the case in which strain occurs in a polar domain and strain in the direction perpendicular to this surface is zero. On the other hand, the axially symmetric refers to the model in polar domain in which angular movement is prohibited, see Figure 5.2. Some applications for this method are to model the stresses near the borehole or around a pipe and to predict permeability in low permeability reservoirs (Brace et al., 1968; Hsieh et al., 1981; Verruijt, 2016). It should be noted that permeability values can be predicted by implementing the pulse-decay tests. In pulse-decay test, pressure at one end of a core plug is increased and the pressure build-up at the other end of the core plug is measured. By using the conventional fluid equations, which also take into account for diffusion, the permeability of the medium is estimated from such a test. However, this method fails to render sufficiently accurate results in the case of low permeability values. The reason is that the well-test results are analysed in one dimension; however, the pressure field would be axisymmetric due to coupled nature of the solid and fluid behaviour, and the induced poroelastic strains affect the permeability values (Walder and Nur, 1986). In plane strain in polar domain, the problem is two dimensional; however, since the coordinates are different and the control volumes are radial, the displacement, strain, stress relationships and, consequently, governing equations could be different. The relations for stress strain are as stated in Equation (5.13), (Wang, 2000). Hook's law is used to relate the stress and the strain. For axially symmetric similar procedure would be followed. Additionally, it should be taken into account that the fluid equation should also be obtained by writing mass balance in the polar coordinate.

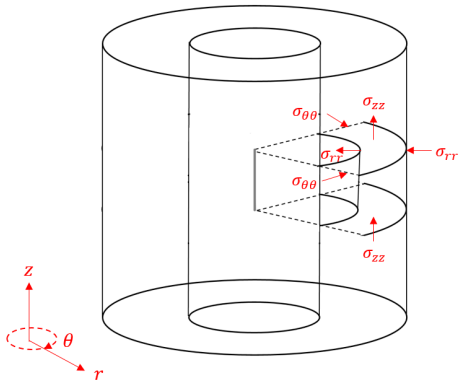


Figure 5.2: *Illustrates stresses in a polar domain.*

$$\varepsilon_{\theta\theta} = \frac{u_r}{r} \tag{5.13a}$$

$$\varepsilon_{rr} = \frac{\partial u_r}{\partial r} \tag{5.13b}$$

$$\varepsilon_{r\theta/\theta r} = \frac{1}{2}\left(\frac{\partial u_\theta}{\partial r} + \frac{u_\theta}{r}\right) \tag{5.13c}$$

$$\sigma_{rr} = -(2\mu + \lambda)\varepsilon_{rr} - \lambda\varepsilon_{\theta\theta} + bp \tag{5.13d}$$

$$\sigma_{\theta\theta} = -(2\mu + \lambda)\varepsilon_{\theta\theta} - \lambda\varepsilon_{rr} + bp \tag{5.13e}$$

$$\frac{\partial \sigma_{rr}}{\partial r} + \frac{\sigma_{rr} - \sigma_{\theta\theta}}{r} = 0 \tag{5.13f}$$

$$\tag{5.13g}$$

where σ_{rr} is normal radial stress $[Pa]$, $\sigma_{\theta\theta}$ is normal angular stress $[Pa]$, ε_{rr} is pure radial strain $[-]$, $\varepsilon_{\theta\theta}$ is pure angular strain $[-]$, u_r is radial displacement $[m]$, and u_θ is angular displacement $[m]$.

5.2 Implementation and Application

In the underground, the coupled behaviour between the solid and the fluid would be observed. In shallow depth, the solid medium is soil in which fractures' effect can be neglected; however, on the other hand, the medium response to stresses would not be elastic but rather elastoplastic. Moreover, other phenomena can yet be observed, such as thermo-hydro-mechanical behaviour or presence of different phases of fluids or gravity effects. For instance in shallow geothermal modelling, thermo-elastoplastic modelling should be taken into account. Additionally, in order to model the behaviour of the medium around a pipe, in some cases, plane strain in polar domain can be applied.

In deeper depth, the poroelastic behaviour would be observed as the porous medium would go through several cycles of stresses. Different phenomena are observed in the subsurface, such as faults, fractures and thermo-hydro-dynamic responses. In order to model each of these phenomena, a different combination of aforementioned models should be employed. For example, to model near fault zones, both elastoplastic and elastic modellings are common practices, in order to predict and prevent near well-bore failure in reservoirs, the combination of plane strain in polar domain and elastoplastic modelling would render sufficiently accurate results, in order to enhance accuracy of modelling of poroelasticity in tight reservoirs, non-linear poroelastic behaviour is advised to be considered. In modelling for deep geothermal purposes, both thermal effects and fractures should be added to the current models.

6 Conclusions and Recommendations

6.1 Conclusions

Two finite volume discretization schemes with collocated unknowns were developed: the cell centred-cell centred discretization scheme (CC-FV) and vertex centred-vertex centred discretization scheme (VV-FC). Collocation of unknowns in the proposed models enables easy integration with multi-grid/multi-scale solvers. Due to same data structure for pressure and displacement unknowns, inclusion of unstructured grids is facilitated. The numerical examples showed that the proposed models can be used to efficiently and accurately model horizontal displacement, i.e. u_x, vertical displacement/subsidence, i.e. u_y, and pressure, i.e. p, for applications in poroelasticity. However, the CC-FV method utilizes the extrapolation/shaped functions in order to approximate the mixed derivatives. Numerical experiments showed that the performance of the CC-FV method is highly dependent on the choice of the extrapolation/shaped functions. Besides, the number of unknown nodes in CC-FV discretization scheme was tested. It is believed that the current CC-FV uses the optimum number of unknown grid points.
The time integration of the fully coupling scheme (FC) and the (fixed-strain) sequentially coupling scheme (SC) are compared in terms of computational efficiency and accuracy. The computational cost of the SC scheme is two times larger compared to the FC scheme. The performance of the SC scheme is dependent on different contributing factors such as the convergence criteria and the scale of the problem. As it was shown in the synthetic test case I, to find the accurate solutions for pressure and displacements making the system dimensionless is required when the pressure is in order of 10^5 $[Pa]$ and displacements in order of 10^{-5} $[m]$.
The VV-FC model is tested in two different configurations: Stress and displacement (boundary condition) configuration, i.e. (S-BC) and (D-BC), respectively. The model is tested against the Mandel test case and for systems with different fluid compressibilities. The performance of the proposed model under both configurations is accurate. However, in case of the S-BC, the system becomes overdetermined and intrinsically lateral variations in solutions appear. To our knowledge, none of the current finite volume methods use the S-BC as defined in this work.
The performance of the VV-FC model is further investigated for different range of input parameters. The VV-FC model is able to predict displacements and pressure values with high accuracy under various ranges of input parameters (Poisson ratios of 0.5 being included).
Last but not least, an application of the VV-FC model through a practical problem is illustrated. By comparing the pressure solutions, it is evident that not considering displacement effects in the production might result in an underestimation of the reservoir pressure drop. As by comparing the values for the second time step, a conventional reservoir model anticipates a pressure value of $\sim 0.8p_0$ while the poroelastic homogeneous model predicts a reservoir pressure of $0.4p_0$ which is half of the value predicted by the conventional fluid flow equations. This value is even smaller in presence of heterogeneity which is expected to occur since in the heterogeneous case higher mobility is available to the fluid and the fluid would find the easiest path to be produced. Moreover, the porous rock will compress faster in case of heterogeneity. Therefore, the fluid will be produced at a faster rate resulting in a reservoir pressure decline in

a faster rate and this effect is observed with a higher intensity at the sides of the domain where higher solid deformation occurs.

6.2 Recommendations

As mentioned, by enhancing the extrapolation functions in CC-FV method, the efficiency of the CC-FV method will be improved. The efficiency of SC scheme can be improved in terms of accuracy and computational time. In order to achieve this purpose, the stability analysis and one dimensional coupling of the solid and the fluid domain is encouraged to be carried out. The performance of SC model is dependent on the scale of the two sub-problems; therefore, it is encouraged for the SC scheme to model the poroelastic problems in field unit/dimensionless system units. To our knowledge, there are no introduced dimensionless system units for poroelastic problems. The performance of the VV-FC model under S-BC configuration can further be improved by either using stabilizers to remove spatial oscillation of solutions as a result of an overdetermined system or finding an alternative definition for the total stress boundary condition. The latter is not encouraged to be carried out. In order to model the fully incompressible systems in poroelastic problems with finite volume method, stabilizers are required.
For applications of poroelasticity in geothermally active reservoirs, effects of temperature and fractures should also be included. The axisymetric modelling of the poroelasticity would widen its applicability to near well-bore failure modelling or sand production modelling. For applications in hydraulic fracturing, effects of fractures and non-linear elasticity should also be included. For applications in shallow geothermal systems, effects of nonlinear elasticity, temperature, and elastoplasticity should be added to the current model. For modelling tight reservoirs such as shale, non-linear poroelasticity is a suitable assumption. For reservoirs with high gravity parameter, gravity effects should be included such as very thick reservoirs layers. In order to model undersaturated reservoirs with high pressure drops, volatile oil reservoirs, CO_2 sequestration, and immiscible injection scenarios multiphase fluid should be considered.

Appendix A. Definition and Derivation of Equations

In this appendix derivation of equations and definition of some constitutive equations and mathematical operators are described.

A.1 Storage Equation

Equation (3.2) by considering negligible values for multiplication of fluid velocity and gradient of pressure, negligible variation of fluid density,i.e. ρ_f, over space and considering fluid compressibility as described in Equation (A.1) will reduce to Equation (A.2).

$$\frac{\partial \rho_f}{\partial p} = \rho_f C_f \tag{A.1}$$

$$\frac{\partial \phi}{\partial t} + \phi C_f \frac{\partial p}{\partial t} + \nabla.(\phi v) = 0 \tag{A.2}$$

Equation (A.3) describes variation in solid particle volumes when the porous medium is under total isotropic stress σ and undrained condition, i.e. no flow occurring into or out of the porous medium. Considering Equation (A.3), density of solid can be assumed to be proportional to isotropic total stress and pore pressure, as expressed in Equation (A.4).

$$\Delta V_s = -(1-\phi) C_s \Delta p V - C_s(\Delta\sigma - \Delta p)V \tag{A.3}$$

$$\frac{\partial \rho_s}{\partial t} = \frac{\rho_s C_s}{1-\phi}\left(\frac{\partial \sigma}{\partial t} - \phi \frac{\partial p}{\partial t}\right) \tag{A.4}$$

Assuming negligible variation of solid density, i.e. ρ_s over space and negligible values for multiplication of particle velocity and gradient of stress and pressure, Equation (A.5) is obtained. Equation (A.6) is obtained by adding Equation (A.5) and Equation (A.2).

$$-\frac{\phi}{\partial t} + C_s\left(\frac{\partial \sigma}{\partial t} - \phi \frac{\partial p}{\partial t}\right) + \nabla.([1-\phi]w) = 0 \tag{A.5}$$

$$\phi(C_f - C_s)\frac{\partial p}{\partial t} + C_s \frac{\partial \sigma}{\partial t} + \nabla.w + \nabla.(\phi[v-w]) = 0 \tag{A.6}$$

In Equation (A.6), the term "$v - w$" is the vector summation of velocities of solid and fluid which is expected to be proportional to influx fluid into/out of the medium, which is known as Darcy velocity, i.e. "$-\lambda_t \nabla p$". The term "$\nabla.w$" which is the divergence of solid velocity can be re-written as derivative of divergence of displacement with respect to time, i.e."$\frac{\partial \nabla.u}{\partial t}$". Furthermore, total stress is proportional to displacement and pore pressure as expressed in Equation (A.7). Consequently, Equation (A.6) can be simply written as expressed Equation (3.4), (Verruijt, 2016).

$$\sigma = bP + K_{dr} \nabla \cdot u \tag{A.7}$$

A.2 Hooke's Law

Hooke's law relates the strain (deformation) of an elastic object to the stress applied to it by first-order linear approximation. Mathematical formulation of Hooke's law is as stated in Equation (A.8), (Sokolnikoff et al., 1956).

$$\sigma_{ij} = -\lambda\delta_{ij}\varepsilon_{kk} - 2\mu\varepsilon_{ij} \tag{A.8a}$$

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{A.8b}$$

$$\varepsilon_{kk} = \varepsilon_{xx} + \varepsilon_{yy} \tag{A.8c}$$

$$\varepsilon_{ij} = \frac{1}{2}(\frac{\partial u_i}{j} + \frac{\partial u_j}{i}) \tag{A.8d}$$

A.3 2-norm/Euclidean norm

The 2-norm/Euclidean norm of a vector element, named x, is as stated in the Equation (A.9).

$$\ell^2(x) = \sqrt[2]{\sum_{i=1}^{n} x_i^2} \tag{A.9}$$

A.4 Normalized Root Mean Square Error: NRMSE

The normalized root mean square error of an array, named x, with exact solutions of the same array, named x_{ex}, is as stated in the Equation (A.10).

$$NRMSE_x = \frac{\sqrt[2]{\frac{1}{n}\sum_{i=1}^{n}(x_i - x_{ex,i})^2}}{|max(x_{ex}) - min(x_{ex})|} \tag{A.10}$$

Appendix B. Analytical Solution of Classical Test Cases

Analytical solution of the Mandel's problem is described. Mandel test case is a benchmark poroelastic problem (Verruijt, 2016; Cheng et al., 2016). Mandel problem illustrates a two-dimensional poroelastoc problem in which none monotonic pressure behavior of the fluid is observed due to the two-way coupled nature of the solid and fluid in the porous medium.

B.1 Mandel Test Case

The Mandel problem is one of the most challenging problems for two dimensional poroelasticity. The setup and boundary conditions for the Mandel problem is mentioned before in Chapter 4, i.e. the results section. The initial and analytic solution of this problem is as stated in equations (B.2) and (B.1).

$$p(x,y,0) = \frac{1}{3L_x} B(1+\nu_u)F \tag{B.1a}$$

$$u_x(x,y,0) = \frac{F\nu_u x}{2\mu L_x} \tag{B.1b}$$

$$u_y(x,y,0) = -\frac{F(1-\nu_u)y}{2\mu L_y} \tag{B.1c}$$

$$p(x,y,t) = 2p_0 \sum_{m=1}^{\infty} \frac{\sin\alpha_m}{\alpha_m - \sin\alpha_m \cos\alpha_m} \times \left(\cos[\frac{\alpha_m x}{L_x}] - \cos\alpha_m\right) \exp[-\frac{\alpha_m^2 c_v t}{L_x L_y}] \tag{B.2a}$$

$$u_x(x,y,t) = \left(\frac{F\nu}{2\mu L_x} - \frac{F\nu_u}{\mu L_x} \sum_{m=1}^{\infty} \frac{\sin\alpha_m \cos\alpha_m}{\alpha_m - \sin\alpha_m \cos\alpha_m} \times \exp[-\frac{\alpha_m^2 c_v t}{L_x^2}]\right) x + \frac{F}{\mu} \sum_{m=1}^{\infty} \frac{\cos\alpha_m}{\alpha_m - \sin\alpha_m \cos\alpha_m} \sin[\frac{\alpha_m x}{L_x}] \exp[-\frac{\alpha_m^2 c_v t}{L_x L_y}] \tag{B.2b}$$

$$u_y(x,y,t) = \left(-\frac{F(1-\nu)}{2\mu L_x} + \frac{F(1-\nu_u)}{\mu L_x} \times \sum_{m=1}^{\infty} \frac{\sin\alpha_m \cos\alpha_m}{\alpha_m - \sin\alpha_m \cos\alpha_m} \exp[-\frac{\alpha_m^2 c_v t}{L_x L_y}]\right) y \tag{B.2c}$$

where B is Skempton's coefficient $[-]$, F is the force applied to the top of the medium $[N]$, ν is drained possion's ratio $[-]$, ν_u is undrained possion's ratio $[-]$ and α_n is summation of positive roots, i.e. α, of the characteristic equation, stated as in Equation (B.3).

$$\tan(\alpha) = \frac{1-\nu}{\nu_u - \nu}\alpha \tag{B.3}$$

Appendix C. Range of Parameters

The order of each parameter, like S, k_{dr}, etc, is described in detail in this subsection, (McCain, 1973; Bai and Abousleiman, 1997; Punmia and Jain, 2005; Kelkar, 2008; Berkowitz, 2012; Bear, 2013; Koliji, 2016). The Table C.1 indicates the viscosity values for incompressible to highly compressible fluids. In this table, water and black oil properties are used to represent

incompressible fluids, volatile oil properties are used to represent slightly compressible fluids and gas represents highly compressible fluids in the subsurface. The Table C.2 is presenting several rock properties for unconsolidated rocks. It should be noted that precise values for rock properties and especially fluid properties are case dependent and only obtainable after lab experiments. It's good to mention that, typical values for a consolidated and incompressible oil reservoir rock type is reported in the last column of Tables C.3 and C.4. Rocks with compressibility values of $10^{-8} - 10^{-7}$ $[Pa]$ are assumed to be slightly compressible and with values lower and upper this range are considered in- and fully- compressible rocks, respectively. Additionally, values in Tables C.1, C.2, C.3 and C.4 are obtained by averaging the value at upper and lower bounds of the range of each property.

Table C.1: *Illustrates viscosity and compressibility values for different types of fluids.*

Fluid Type	μ $[Pa.s]$	c_f $[1/Pa]$
Incompressible Fluid[(1)]	5.5×10^{-4}	1.5×10^{-9}
Slightly Compressible Fluid [(2)]	2.0×10^{-4}	5.8×10^{-8}
Fully Compressible Fluid [(3)]	2.4×10^{-5}	8.0×10^{-8}
Water[(4)]	8.9×10^{-4}	5.1×10^{-10}

(1) (PetroWiki, 2016), (McCain, 1973), p 246.
(2) (PetroWiki, 2016), (McCain, 1973), p 288-290.
(3) (McCain, 1973), p 170-187.
(4) (McCain, 1973), p 451-460.

Table C.2: *Illustrates absolute permeability, porosity, compressibility and elasticity constant values for unconsolidated rocks. Typical values for properties of an oil reservoir hard rock is stated in the last column.*

Rock Type	k $[m^2]$ [(1)]	ϕ $[-]$ [(2)]	c_s $[1/Pa]$ [(3)]	λ $[Pa]$ [(4)]	μ $[Pa]$ [(4)]
Well Sorted Gravel	5.5×10^{-8}	0.27	2.25×10^{-8}	1.40×10^{8}	9.32×10^{7}
Well Sorted Gravel and/or Sand	5.5×10^{-9}	0.27	1.09×10^{-7}	5.88×10^{7}	3.33×10^{7}
Very Fine Sand, Silt, etc	5.0×10^{-13}	0.32	2.03×10^{-7}	9.34×10^{6}	9.49×10^{6}
Peat/Coal	5.5×10^{-12}	0.42	4.63×10^{-6}	6.61×10^{5}	3.64×10^{5}
Clay	5.0×10^{-14}	0.35	6.10×10^{-7}	2.34×10^{7}	6.08×10^{6}
Consolidated Reservoir Rock	5.0×10^{-12}	0.20	1.80×10^{-9}	3.45×10^{8}	8.39×10^{8}

(1) (Punmia and Jain, 2005) p 178, (Bear, 2013), p 136.
(2) (Punmia and Jain, 2005) p 16, (Bear, 2013), p 46.
(3) (Kelkar, 2008), p 109-113.
(4) (Koliji, 2016), (Berkowitz, 2012) p 86-90.

Table C.3: *Illustrates values for storativity and mobility for consolidated oil reservoir rock with fluid type of varying compressibility.*

Type of System	λ_t $[m^2/(Pa.s)]$	S $[1/Pa]$	c_v $[m^2/s]$	λ $[Pa]$	μ $[Pa]$
Incompressible Fluid	1.00×10^{-11}	1.73×10^{-9}	0.0045	3.45×10^{8}	8.39×10^{8}
Slightly Compressible Fluid	7.14×10^{-8}	1.29×10^{-8}	5.3	3.45×10^{8}	8.39×10^{8}
Fully Compressible Fluid	5.00×10^{-7}	1.74×10^{-8}	28	3.45×10^{8}	8.39×10^{8}

Table C.4: *Illustrates values for storativity and mobility with respect to compressibility of the system.*

Type of System	λ_t $[m^2/(Pa.s)]$	S $[1/Pa]$	c_v $[m^2/s]$	λ $[Pa]$	μ $[Pa]$
Incompressible System	1.65×10^{-4}	1.79×10^{-8}	7.8×10^{3}	1.48×10^{8}	9.32×10^{7}
Slightly Compressible System	2.43×10^{-5}	2.80×10^{-7}	86	3.05×10^{7}	1.63×10^{7}
Fully Compressible System	5.14×10^{-7}	3.57×10^{-6}	1.20×10^{-1}	6.61×10^{5}	3.64×10^{5}

Appendix D. MATLAB Codes

To make codes understandable, they are written in user friendly way and according to concepts and solution work-flows presented in the report since the primarily objective of this master thesis is developing algorithm rather than coding. All the codes used to generate the results are developed by me from scratch; however, some of the are not presented in this report. For the spatial discretization of the fully coupled vertex-centered model, is adopted from the work carried out by Luo et al. (2015).

D.1 Codes for Solid Solver

```
global BC_b BC_t BC_l BC_r N_x N_y L_x L_y lambda miu
% Author:                    Mitra Asadollahi #4412451
% Description:
    % This script numerically estimates the pure solid deformation solutions
    % also called undrained poroelasticity solid deformation solutions with
    % cell-centered finite volume method in both homogeneous and
    % heterogeneous porous media.
    % governing equations are back calculated to check the correct
    % implementation of the code.
    % stress, strain and displacement values are calculated at cell
    % interfaces and grid cells.
% setting boundary conditions
    % 1: [ux uy]=[c1 c2];
    % 2: u.n=c1; (sigma.n).m=c2;
    % 3: sigma.n=[c1 c2]
    BC_b=2;              %  boundary condition at the bottom interface
    BC_t=3;              %  boundary condition at the top interface
    BC_l=2;              %  boundary condition at the left interface
    BC_r=3;              %  boundary condition at the right interface
% Initialization
    L_x=10;              % length of the domain in x direction [m]
    L_y=10;              % length of the domain in y direction [m]
    N_x=20;               % number of grid cels in x direction [m]
    N_y=20;               % number of grid cels in y direction [m]
    dx=L_x/N_x;          % grid length in x direction [m]
    dy=L_y/N_y;          % grid length in y direction [m]
    x_c=dx/2:dx:L_x;     % position of grid cells in x direction [m]
    y_c=dy/2:dy:L_y;     % position of grid cells in y direction [m]
    x_i=0:dx:L_x;        % position of grid interface in x direction [m]
    y_i=0:dy:L_y;        % position of grid interface in y direction [m]
    HT=0; %heterogeinity exists if HT is unity
    if HT==1
            miu=10*rand(N_x,N_y).*ones(N_x,N_y);     %shear modulus [Pa]
            lambda=10*rand(N_x,N_y).*ones(N_x,N_y); %lame's parameter [Pa]
    else
            miu=ones(N_x,N_y);                       %shear modulus [Pa]
            lambda=ones(N_x,N_y);                    %lame's parameter [Pa]
    end
    % body forces [Pa/m]
    f_x=zeros(N_x,N_y);
    f_y=zeros(N_x,N_y);
    % stress values at the interfaces
    % x=0; sigma_xx sigma_xy
        str_xx_l=zeros(1,N_y);
        str_xy_l=zeros(1,N_y);
        u_x_l=zeros(1,N_y);
        u_y_l=zeros(1,N_y);
    % x=L_x; sigma_xx sigma_xy
```

```
        str_xx_r=zeros(1,N_y);
        str_xy_r=zeros(1,N_y);
        u_x_r=zeros(1,N_y);
        u_y_r=zeros(1,N_y);
    % y=0; sigma_yy sigma_yx
        str_yy_b=zeros(N_x,1);
        str_yx_b=zeros(N_x,1);
        u_x_b=zeros(N_x,1);
        u_y_b=zeros(N_x,1);
    % y=L_y; sigma_yy sigma_yx
       str_yy_t=200*ones(N_x,1);
       str_yx_t=zeros(N_x,1);
        u_x_t=zeros(N_x,1);
        u_y_t=zeros(N_x,1);
%numerical solution
     [Ux_nu,Uy_nu,div_x,div_y,divU,stress_yy,stress_xx,stress_yx,stress_xy]...
         =DispSolver_verification(str_xx_l,str_xy_l,u_y_r,u_x_r,...
    u_y_l,u_x_l,u_y_t,u_x_t,u_y_b,u_x_b,f_x,f_y,str_xx_r,str_xy_r,...
    str_yx_t,str_yy_t,str_yx_b,str_yy_b);
%net displacement
net_u=sqrt(Ux_nu.^2+Uy_nu.^2);
%Plots
%plot(Ux);
figure;
surf(y_c,x_c,Ux_nu);
title('$$\bf{U_x} $$ $$ Numerical $$ $$Solution, $$ $$\Gamma_{t} $$',...
    'interpreter','latex','FontSize', 18);
ylabel('$$\bf{x}, $$ $$on$$ $$\Gamma_{b} $$','interpreter','latex','FontSize', 18);
xlabel('$$\bf{y},$$ $$on$$ $$\Gamma_{r} $$','interpreter','latex','FontSize', 18);
colorbar('fontsize',12);
set(gca,'fontsize',12);
 az = 270;
el = 90;
view(az, el);
figure;
%plot(Uy);
surf(y_c,x_c,Uy_nu);
title('$$\bf{U_y}$$ $$ Numerical $$ $$Solution, $$ $$\Gamma_{t} $$',...
    'interpreter','latex','FontSize', 18);
ylabel('$$\bf{x}, $$ $$on$$ $$\Gamma_{b} $$','interpreter','latex','FontSize', 18);
xlabel('$$\bf{y},$$ $$on$$ $$\Gamma_{r} $$','interpreter','latex','FontSize', 18);
colorbar('fontsize',12);
set(gca,'fontsize',12);
 az = 270;
el = 90;
view(az, el);
figure;
surf(y_c,x_c,div_x);
title('$$\bf{(\nabla \cdot \sigma)_x} Numerical Solution, \Gamma_{t} $$'...
    ,'interpreter','latex','FontSize', 18);
ylabel('$$\bf{x}, $$ $$on$$ $$\Gamma_{b} $$','interpreter','latex','FontSize', 18);
xlabel('$$\bf{y},$$ $$on$$ $$\Gamma_{r} $$','interpreter','latex','FontSize', 18);
colorbar('fontsize',12);
set(gca,'fontsize',12);
 az = 270;
el = 90;
view(az, el);
figure;
surf(y_c,x_c,div_y);
title('$$\bf{(\nabla \cdot \sigma)_y} Numerical Solution, \Gamma_{t} $$'...
    ,'interpreter','latex','FontSize', 18);
ylabel('$$\bf{x}, $$ $$on$$ $$\Gamma_{b} $$','interpreter','latex','FontSize', 18);
xlabel('$$\bf{y},$$ $$on$$ $$\Gamma_{r} $$','interpreter','latex','FontSize', 18);
colorbar('fontsize',12);
set(gca,'fontsize',12);
 az = 270;
```

```
el = 90;
view(az, el);
figure;
surf(y_c,x_c,net_u);
title('$$\bf{\bar U} $$ $$ Numerical $$ $$Solution, $$ $$\Gamma_{t} $$',...
    'interpreter','latex','FontSize', 18);
ylabel('$$\bf{x}, $$ $$on$$ $$\Gamma_{b} $$','interpreter','latex','FontSize', 18);
xlabel('$$\bf{y},$$ $$on$$ $$\Gamma_{r} $$','interpreter','latex','FontSize', 18);
colorbar('fontsize',12);
set(gca,'fontsize',12);
 az = 270;
el = 90;
view(az, el);
%% displacement solver function
for j=1:(N_y+2)
     for i=1:(N_x+2)
        if (i==(N_x+1)&&j==N_y+2)||(i==(N_x+2)&&j==N_y+2)||(i==(2)&&j==1)||(i==1&&j==1)
            % do nothing : no equations corresponding to these points
            % number of points are ranged from 1 to 18
        elseif (i==(N_x+2)&& j==1) %Point 3: bottom interface fixed corner
            if BC_b==1
                % displacement uy=0;
                    I=Index(N_x,i,j);%uy: i,j-1/2
                    A(I,I)=-1;
                    f(I)=f(I)+u_y_b(i-2);
                % dismpalcement ux=0;
                    I=N/2+Index(N_x,i,j);%ux: i,j-1/2
                    A(I,I)=-1;
                    f(I)=f(I)+u_x_b(i-2);
            elseif BC_b==2
                % displacement uy=0;
                    I=Index(N_x,i,j);%uy: i,j-1/2
                    A(I,I)=-1;
                    f(I)=f(I)+u_y_b(i-2);
                % sigma_yx=0
                    I=N/2+Index(N_x,i,j);%ux: i,j-1/2
                    A(I,I)=-2*(miu(i-2,j))/dy;
                    J=N/2+Index(N_x,i-1,j+1);%ux: i,j
                    A(I,J)=2*(miu(i-2,j))/dy;
                    J=Index(N_x,i-1,j+1);%uy: i,j
                    A(I,J)=(miu(i-2,j))/dx;
                    J=Index(N_x,i-2,j+1);%uy: i-1,j
                    A(I,J)=-(miu(i-2,j))/dx;
                    f(I)=f(I)+sigma_yx_b(i-2);
            else
                % sigma_yx=0
                    I=N/2+Index(N_x,i,j);%ux: i,j-1/2
                    A(I,I)=-2*(miu(i-2,j))/dy;
                    J=N/2+Index(N_x,i-1,j+1);%ux: i,j
                    A(I,J)=2*(miu(i-2,j))/dy;
                    J=Index(N_x,i-1,j+1);%uy: i,j
                    A(I,J)=(miu(i-2,j))/dx;
                    J=Index(N_x,i-2,j+1);%uy: i-1,j
                    A(I,J)=-(miu(i-2,j))/dx;
                    f(I)=f(I)+sigma_yx_b(i-2);
                % sigma_yy=F/A
                    I=Index(N_x,i,j);%uy: i,j-1/2
                    A(I,I)=-2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
                    J=Index(N_x,i-1,j+1);%uy: i,j
                    A(I,J)=2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
                    J=N/2+Index(N_x,i-1,j+1);%ux: i,j
                    A(I,J)=(lambda(i-2,j))/dx;
                    J=N/2+Index(N_x,i-2,j+1);%ux: i-1,j
                    A(I,J)=-(lambda(i-2,j))/dx;
                    f(I)=f(I)+sigma_yy_b(i-2);

```

```
181             end
182
183         elseif ((3<i)&&(i<(N_x+2))&& j==1)%Point 2: bottom interface fixed corner
184             if BC_b==1
185                 % displacement ux=0
186                     I=N/2+Index(N_x,i,j);
187                     A(I,I)=-1;
188                     f(I)=f(I)+u_x_b(i-2);
189                 % displacement uy=0
190                     I=Index(N_x,i,j);
191                     A(I,I)=-1;
192                     f(I)=f(I)+u_y_b(i-2);
193             elseif BC_b==2
194                 % displacement uy=0
195                     I=Index(N_x,i,j);%uy: i,j-1/2
196                     A(I,I)=-1;
197                     f(I)=f(I)+u_y_b(i-2);
198                 %sigma_yx=0
199                     I=N/2+Index(N_x,i,j);%ux: i,j-1/2
200                     A(I,I)=-2*(miu(i-2,j))/dy;
201                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j
202                     A(I,J)=2*(miu(i-2,j))/dy;
203                     J=Index(N_x,i-2,j+1);%uy: i-1,j
204                     A(I,J)=-(3/4)*(miu(i-2,j))/dx;
205                     J=Index(N_x,i,j+1);%uy: i+1,j
206                     A(I,J)=(3/4)*(miu(i-2,j))/dx;
207                     J=Index(N_x,i-2,j+2);%uy: i-1,j+1
208                     A(I,J)=(1/4)*(miu(i-2,j))/dx;
209                     J=Index(N_x,i,j+2);%uy: i+1,j+1
210                     A(I,J)=-(1/4)*(miu(i-2,j))/dx;
211                     f(I)=f(I)+sigma_yx_b(i-2);
212             else
213                 %sigma_yx=0
214                     I=N/2+Index(N_x,i,j);%ux: i,j-1/2
215                     A(I,I)=-2*(miu(i-2,j))/dy;
216                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j
217                     A(I,J)=2*(miu(i-2,j))/dy;
218                     J=Index(N_x,i-2,j+1);%uy: i-1,j
219                     A(I,J)=-(3/4)*(miu(i-2,j))/dx;
220                     J=Index(N_x,i,j+1);%uy: i+1,j
221                     A(I,J)=(3/4)*(miu(i-2,j))/dx;
222                     J=Index(N_x,i-2,j+2);%uy: i-1,j+1
223                     A(I,J)=(1/4)*(miu(i-2,j))/dx;
224                     J=Index(N_x,i,j+2);%uy: i+1,j+1
225                     A(I,J)=-(1/4)*(miu(i-2,j))/dx;
226                     f(I)=f(I)+sigma_yx_b(i-2);
227                 %sigma_yy=F/A
228                     I=Index(N_x,i,j);%uy: i,j-1/2
229                     A(I,I)=-2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
230                     J=Index(N_x,i-1,j+1);%uy: i,j
231                     A(I,J)=2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
232                     J=N/2+Index(N_x,i-2,j+1);%ux: i-1,j
233                     A(I,J)=-(3/4)*(lambda(i-2,j))/dx;
234                     J=N/2+Index(N_x,i,j+1);%ux: i+1,j
235                     A(I,J)=(3/4)*(lambda(i-2,j))/dx;
236                     J=N/2+Index(N_x,i-2,j+2);%ux: i-1,j+1
237                     A(I,J)=(1/4)*(lambda(i-2,j))/dx;
238                     J=N/2+Index(N_x,i,j+2);%ux: i+1,j+1
239                     A(I,J)=-(1/4)*(lambda(i-2,j))/dx;
240                     f(I)=f(I)+sigma_yy_b(i-2);
241             end
242         elseif (i==(3)&& j==1) %Point 1: bottom interface fixed corner point
243             if BC_b==1
244                 % displacement uy=0
245                     I=Index(N_x,i,j);
246                     A(I,I)=-1;
```

```
247                     f(I)=f(I)+u_y_b(i-2);
248                 % displacement ux=0
249                     I=N/2+Index(N_x,i,j);
250                     A(I,I)=-1;
251                     f(I)=f(I)+u_x_b(i-2);
252             elseif BC_b==2
253                 % displacement uy=0
254                     I=Index(N_x,i,j);
255                     A(I,I)=-1;
256                     f(I)=f(I)+u_y_b(i-2);
257                 % sigma_yx=0 ux i,j-1/2
258                     I=N/2+Index(N_x,i,j);%ux: i,j-1/2
259                     A(I,I)=-2*(miu(i-2,j))/dy;
260                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j
261                     A(I,J)=2*(miu(i-2,j))/dy;
262                     J=Index(N_x,i-1,j+1);%uy: i,j
263                     A(I,J)=-(miu(i-2,j))/dx;
264                     J=Index(N_x,i,j+1);%uy: i+1,j
265                     A(I,J)=(miu(i-2,j))/dx;
266                     f(I)=f(I)+sigma_yx_b(i-2);
267
268             else
269                 % sigma_yx=0 ux i,j-1/2
270                     I=N/2+Index(N_x,i,j);%ux: i,j-1/2
271                     A(I,I)=-2*(miu(i-2,j))/dy;
272                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j
273                     A(I,J)=2*(miu(i-2,j))/dy;
274                     J=Index(N_x,i-1,j+1);%uy: i,j
275                     A(I,J)=-(miu(i-2,j))/dx;
276                     J=Index(N_x,i,j+1);%uy: i+1,j
277                     A(I,J)=(miu(i-2,j))/dx;
278                     f(I)=f(I)+sigma_yx_b(i-2);
279                 % sigma_yy=F/A uy i,j-1/2
280                     I=Index(N_x,i,j);%uy: i,j-1/2
281                     A(I,I)=-2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
282                     J=Index(N_x,i-1,j+1);%uy: i,j
283                     A(I,J)=2*(2*miu(i-2,j)+lambda(i-2,j))/dy;
284                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j
285                     A(I,J)=-(lambda(i-2,j))/dx;
286                     J=N/2+Index(N_x,i,j+1);%ux: i+1,j
287                     A(I,J)=(lambda(i-2,j))/dx;
288                     f(I)=f(I)+sigma_yy_b(i-2);
289
290             end
291
292         elseif (i==(N_x+2)&& j==2) %Point 8: left interface fixed corner
293             if BC_r==1
294                  %ux: i+1/2,j=0
295                     I=N/2+Index(N_x,i,j);%ux: i+1/2,j
296                     A(I,I)=-1;
297                     f(I)=f(I)+u_x_r(j-1);
298                  %uy: i+1/2,j=0
299                     I=Index(N_x,i,j);%uy: i+1/2,j
300                     A(I,I)=-1;
301                     f(I)=f(I)+u_y_r(j-1);
302             elseif BC_r==2
303                 %ux: i+1/2,j=0
304                     I=N/2+Index(N_x,i,j);%ux: i+1/2,j
305                     A(I,I)=-1;
306                     f(I)=f(I)+u_x_r(j-1);
307                 %sigma_xy=0 for uy: i+1/2,j
308                     I=Index(N_x,i,j);%uy: i+1/2,j
309                     A(I,I)=(2*miu(i-2,j-1)/dx);
310                     J=Index(N_x,i-1,j);%uy: i,j
311                     A(I,J)=-(2*miu(i-2,j-1)/dx);
312                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1
```

```
313                     A(I,J)=(miu(i-2,j-1)/(dy));
314                     J=N/2+Index(N_x,i-1,j);%ux: i,j
315                     A(I,J)=(-miu(i-2,j-1)/(dy));
316                     f(I)=f(I)+sigma_xy_r(j-1);
317                 else
318                       %sigma_xy=0 for uy: i+1/2,j
319                     I=Index(N_x,i,j);%uy: i+1/2,j
320                     A(I,I)=(2*miu(i-2,j-1)/dx);
321                     J=Index(N_x,i-1,j);%uy: i,j
322                     A(I,J)=-(2*miu(i-2,j-1)/dx);
323                     J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1
324                     A(I,J)=(miu(i-2,j-1)/(dy));
325                     J=N/2+Index(N_x,i-1,j);%ux: i,j
326                     A(I,J)=(-miu(i-2,j-1)/(dy));
327                     f(I)=f(I)+sigma_xy_r(j-1);
328                   %sigma_xx=0 for ux: i+1/2,j
329                     I=N/2+Index(N_x,i,j);%ux: i+1/2,j: Index 5+18
330                     A(I,I)=(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
331                     J=N/2+Index(N_x,i-1,j);%ux: i,j: Index 4+18
332                     A(I,J)=-(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
333                     J=Index(N_x,i-1,j+1);%uy: i,j+1: Index 9
334                     A(I,J)=+lambda(i-2,j-1)/dy;
335                     J=Index(N_x,i-1,j);%uy: i,j: Index 4
336                     A(I,J)=-lambda(i-2,j-1)/dy;
337                     f(I)=f(I)+sigma_xx_r(j-1);
338
339                 end
340
341
342
343             elseif (i==(N_x+2)&& j==(N_y+1))% Point 18
344                 if BC_r==1                          %top interface free corner
345                     %ux: i+1/2,j=0
346                         I=N/2+Index(N_x,i,j);%ux: i+1/2,j
347                         A(I,I)=-1;
348                         f(I)=f(I)+u_x_r(j-1);
349                     %uy: i+1/2,j=0
350                         I=Index(N_x,i,j);%uy: i+1/2,j
351                         A(I,I)=-1;
352                         f(I)=f(I)+u_y_r(j-1);
353                 elseif BC_r==2
354                     %ux: i+1/2,j=0
355                         I=N/2+Index(N_x,i,j);%ux: i+1/2,j
356                         A(I,I)=-1;
357                         f(I)=f(I)+u_x_r(j-1);
358                     %sigma_xy=0 for uy: i+1/2,j
359                         I=Index(N_x,i,j);%uy: i+1/2,j
360                         A(I,I)=(2*miu(i-2,j-1)/dx);
361                         J=Index(N_x,i-1,j);%uy: i,j
362                         A(I,J)=-(2*miu(i-2,j-1)/dx);
363                         J=N/2+Index(N_x,i-1,j-1);%ux: i,j-1
364                         A(I,J)=-(miu(i-2,j-1)/(dy));
365                         J=N/2+Index(N_x,i-1,j);%ux: i,j
366                         A(I,J)=+(miu(i-2,j-1)/(dy));
367                         f(I)=f(I)+sigma_xy_r(j-1);
368                 else
369                    %sigma_xy=0 for uy: i+1/2,j
370                         I=Index(N_x,i,j);%uy: i+1/2,j
371                         A(I,I)=(2*miu(i-2,j-1)/dx);
372                         J=Index(N_x,i-1,j);%uy: i,j
373                         A(I,J)=-(2*miu(i-2,j-1)/dx);
374                         J=N/2+Index(N_x,i-1,j-1);%ux: i,j-1
375                         A(I,J)=-(miu(i-2,j-1)/(dy));
376                         J=N/2+Index(N_x,i-1,j);%ux: i,j
377                         A(I,J)=+(miu(i-2,j-1)/(dy));
378                         f(I)=f(I)+sigma_xy_r(j-1);
```

```
379                 %sigma_xx=0 for ux: i+1/2,j
380                       I=N/2+Index(N_x,i,j);%ux: i+1/2,j
381                       A(I,I)=(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
382                       J=N/2+Index(N_x,i-1,j);%ux: i,j
383                       A(I,J)=-(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
384                       J=Index(N_x,i-1,j-1);%uy: i,j-1
385                       A(I,J)=-lambda(i-2,j-1)/dy;
386                       J=Index(N_x,i-1,j);%uy: i,j
387                       A(I,J)=lambda(i-2,j-1)/dy;
388                       f(I)=f(I)+sigma_xx_r(j-1);
389
390                  end
391
392             elseif (j==(2)&& i==1) %Point 4: Right hand side fixed corner
393                 if BC_l==1
394                      %ux: i+1/2,j=0
395                           I=N/2+Index(N_x,i,j);%ux: i+1/2,j
396                           A(I,I)=-1;
397                           f(I)=f(I)+u_x_l(j-1);
398                      %uy: i+1/2,j=0
399                           I=Index(N_x,i,j);%ux: i+1/2,j
400                           A(I,I)=-1;
401                            f(I)=f(I)+u_y_l(j-1);
402                 elseif BC_l==2
403                      %ux: i+1/2,j=0
404                           I=N/2+Index(N_x,i,j);%ux: i+1/2,j
405                           A(I,I)=-1;
406                            f(I)=f(I)+u_x_l(j-1);
407                          %sigma_xy=0 for uy: i-1/2,j
408                               I=Index(N_x,i,j);%uy: i-1/2,j
409                               A(I,I)=-(2*miu(i,j-1)/dx);
410                               J=Index(N_x,i+1,j);%uy: i,j
411                               A(I,J)=(2*miu(i,j-1)/dx);
412                               J=N/2+Index(N_x,i+1,j+1);%ux: i,j+1
413                               A(I,J)=(miu(i,j-1)/(dy));
414                               J=N/2+Index(N_x,i+1,j);%ux: i,j
415                               A(I,J)=-(miu(i,j-1)/(dy));
416                               f(I)=f(I)+sigma_xy_l(j-1);
417
418                 else
419                      %sigma_xy=0 for uy: i-1/2,j
420                           I=Index(N_x,i,j);%uy: i-1/2,j
421                           A(I,I)=-(2*miu(i,j-1)/dx);
422                           J=Index(N_x,i+1,j);%uy: i,j
423                           A(I,J)=(2*miu(i,j-1)/dx);
424                           J=N/2+Index(N_x,i+1,j+1);%ux: i,j+1
425                           A(I,J)=(miu(i,j-1)/(dy));
426                           J=N/2+Index(N_x,i+1,j);%ux: i,j
427                           A(I,J)=-(miu(i,j-1)/(dy));
428                           f(I)=f(I)+sigma_xy_l(j-1);
429                      %sigma_xx=0 for ux: i-1/2,j
430                           I=N/2+Index(N_x,i,j);%ux: i-1/2,j
431                           A(I,I)=-(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
432                           J=N/2+Index(N_y,i+1,j);%ux: i,j
433                           A(I,J)=(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
434                           J=Index(N_x,i+1,j+1);%uy: i,j+1
435                           A(I,J)=+(lambda(i,j-1))/dy;
436                           J=Index(N_x,i+1,j);%uy: i,j
437                           A(I,J)=-(lambda(i,j-1))/dy;
438                           f(I)=f(I)+sigma_xx_l(j-1);
439                 end
440
441               elseif(i==(2)&&j==2)%Point 5: Right hand side cell at fixed end
442                  %1st equ : div(sigma)y=0
443                   I=Index(N_x,i,j);%uy: i,j
444                   A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
```

```
445                 3*miu(i-1,j-1)/dx^2;
446             J=Index(N_x,i,j+1);%uy: i,j+1
447             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
448             J=Index(N_x,i+1,j-1);%uy: i,j-1/2
449             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
450             J=Index(N_x,i+1,j);%uy: i+1,j
451             A(I,J)=(miu(i-1,j-1)/dx^2);
452             J=Index(N_x,i-1,j);%uy: i-1/2,j
453             A(I,J)=2*(miu(i-1,j-1)/dx^2);
454             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
455             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
456             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
457             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
458             J=N/2+Index(N_x,i,j);%ux: i,j
459             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
460             J=N/2+Index(N_x,i+1,j+1);%ux: i+1,j+1
461             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
462             f(I)=f(I)+f_y(i-1,j-1);
463             %2nd equ : div(sigma)x=0
464             I=N/2+Index(N_x,i,j);%ux: i,j
465             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
466                 3*miu(i-1,j-1)/dy^2;
467             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
468             A(I,J)=miu(i-1,j-1)/dy^2;
469             J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1/2
470             A(I,J)=2*miu(i-1,j-1)/dy^2;
471             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
472             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
473             J=N/2+Index(N_x,i-1,j);%ux: i-1/2,j
474             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
475             J=Index(N_x,i+1,j);%uy: i+1,j
476             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
477             J=Index(N_x,i,j+1);%uy: i,j+1
478             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
479             J=Index(N_x,i,j);%uy: i,j
480             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
481             J=Index(N_x,i+1,j+1);%uy: i+1,j+1
482             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
483             f(I)=f(I)+f_x(i-1,j-1);
484         elseif(i==(N_x+1)&&j==2)%Point 7: Left hand side cell at fixed end
485             %1st equ: div.(sigma)y=0
486             I=Index(N_x,i,j);%uy: i,j
487             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
488                 3*miu(i-1,j-1)/dx^2;
489             J=Index(N_x,i,j+1);%uy: i,j+1
490             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
491             J=Index(N_x,i+1,j-1);%uy: i,j-1/2
492             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
493             J=Index(N_x,i+1,j);%uy: i+1/2,j
494             A(I,J)=2*(miu(i-1,j-1)/dx^2);
495             J=Index(N_x,i-1,j);%uy: i-1,j
496             A(I,J)=(miu(i-1,j-1)/dx^2);
497             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
498             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
499             J=N/2+Index(N_x,i,j);%ux: i,j
500             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
501             J=N/2+Index(N_x,i-1,j+1);%ux: i-1,j+1
502             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
503             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
504             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
505             f(I)=f(I)+f_y(i-1,j-1);
506             %2nd equ: div.(sigma)x=0
507             I=N/2+Index(N_x,i,j);%ux: i,j
508             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
509                 3*miu(i-1,j-1)/dy^2;
510             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
```

```
511             A(I,J)=(miu(i-1,j-1))/(dy^2);
512             J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1/2
513             A(I,J)=2*(miu(i-1,j-1))/(dy^2);
514             J=N/2+Index(N_x,i+1,j);%ux: i+1/2,j
515             A(I,J)=2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/dx^2);
516             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
517             A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/dx^2);
518             J=Index(N_x,i,j+1);%uy: i,j+1
519             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
520             J=Index(N_x,i,j);%uy: i,j
521             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
522             J=Index(N_x,i-1,j+1);%uy: i-1,j+1
523             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
524             J=Index(N_x,i-1,j);%ux: i-1,j
525             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
526             f(I)=f(I)+f_x(i-1,j-1);
527         elseif (i==(1)&& j==(N_y+1))%Point 14
528             if BC_l==1                  % top right hand side at free corner
529                 %ux: i+1/2,j=0
530                     I=N/2+Index(N_x,i,j);
531                     A(I,I)=-1;
532                     f(I)=f(I)+u_x_l(j-1);
533                 %uy: i+1/2,j=0
534                     I=Index(N_x,i,j);
535                     A(I,I)=-1;
536                     f(I)=f(I)+u_y_l(j-1);
537             elseif BC_l==2
538                 %ux: i+1/2,j=0
539                         I=N/2+Index(N_x,i,j);
540                         A(I,I)=-1;
541                          f(I)=f(I)+u_x_l(j-1);
542                 %sigma_xy=0 for uy: i-1/2,j
543                         I=Index(N_x,i,j);%uy: i-1/2,j
544                         A(I,I)=-(2*miu(i,j-1)/dx);
545                         J=Index(N_x,i+1,j);%uy: i,j
546                         A(I,J)=(2*miu(i,j-1)/dx);
547                         J=N/2+Index(N_x,i+1,j);%ux: i,j
548                         A(I,J)=(miu(i,j-1)/(dy));
549                         J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1
550                         A(I,J)=-(miu(i,j-1)/(dy));
551                         f(I)=f(I)+sigma_xy_l(j-1);
552             else
553                 %sigma_xy=0 for uy: i-1/2,j
554                         I=Index(N_x,i,j);%uy: i-1/2,j
555                         A(I,I)=-(2*miu(i,j-1)/dx);
556                         J=Index(N_x,i+1,j);%uy: i,j
557                         A(I,J)=(2*miu(i,j-1)/dx);
558                         J=N/2+Index(N_x,i+1,j);%ux: i,j
559                         A(I,J)=(miu(i,j-1)/(dy));
560                         J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1
561                         A(I,J)=-(miu(i,j-1)/(dy));
562                         f(I)=f(I)+sigma_xy_r(j-1);
563                 %sigma_xx=0 for ux: i-1/2,j
564                         I=N/2+Index(N_x,i,j);%ux: i-1/2,j
565                         A(I,I)=-(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
566                         J=N/2+Index(N_x,i+1,j);%ux: i,j
567                         A(I,J)=(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
568                         J=Index(N_x,i+1,j-1);%uy: i,j-1
569                         A(I,J)=-lambda(i,j-1)/dy;
570                         J=Index(N_x,i+1,j);%uy: i,j
571                         A(I,J)=(lambda(i,j-1))/dy;
572                         f(I)=f(I)+sigma_xx_l(j-1);
573             end
574
575
576          elseif(i==(2)&&j==(N_y+1))%Point 15: right hand side top cell at free corner
```

```
577             %1st equ: div.(sigma)y=0
578             I=Index(N_x,i,j);%uy: i,j
579             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
580                 3*miu(i-1,j-1)/dx^2;
581             J=Index(N_x,i-1,j+1);%uy: i,j+1/2
582             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
583             J=Index(N_x,i,j-1);%uy: i,j-1
584             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
585             J=Index(N_x,i+1,j);%uy: i+1,j
586             A(I,J)=(miu(i-1,j-1)/dx^2);
587             J=Index(N_x,i-1,j);%uy: i-1/2,j
588             A(I,J)=2*(miu(i-1,j-1)/dx^2);
589             J=N/2+Index(N_x,i,j-1);%ux: i,j-1
590             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
591             J=N/2+Index(N_x,i,j);%ux: i,j
592             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
593             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
594             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
595             J=N/2+Index(N_x,i+1,j-1);%ux: i+1,j-1
596             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
597             f(I)=f(I)+f_y(i-1,j-1);
598             %2nd equ div.(sigma)x=0
599             I=N/2+Index(N_x,i,j);%ux: i,j
600             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
601                 3*miu(i-1,j-1)/dy^2;
602             J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1/2
603             A(I,J)=2*miu(i-1,j-1)/dy^2;
604             J=N/2+Index(N_x,i,j-1);%ux: i,j-1
605             A(I,J)=miu(i-1,j-1)/dy^2;
606             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
607             A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
608             J=N/2+Index(N_x,i-1,j);%ux: i-1/2,j
609             A(I,J)=2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
610             J=Index(N_x,i,j-1);%uy: i,j-1
611             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
612             J=Index(N_x,i,j);%uy: i,j
613             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
614             J=Index(N_x,i+1,j);%uy: i+1,j
615             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
616             J=Index(N_x,i+1,j-1);%uy: i+1,j-1
617             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
618             f(I)=f(I)+f_x(i-1,j-1);
619         elseif(i==(N_x+1)&&j==(N_y+1))%Point 17: Left hand side top cell at free end
620             %1st equ: div.(sigma)y=0
621             I=Index(N_x,i,j);%uy: i,j
622             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
623                 3*miu(i-1,j-1)/dx^2;
624             J=Index(N_x,i-1,j+1);%uy: i,j+1/2
625             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
626             J=Index(N_x,i,j-1);%uy: i,j-1
627             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
628             J=Index(N_x,i+1,j);%uy: i+1/2,j
629             A(I,J)=2*(miu(i-1,j-1)/dx^2);
630             J=Index(N_x,i-1,j);%uy: i-1,j
631             A(I,J)=(miu(i-1,j-1)/dx^2);
632             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
633             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
634             J=N/2+Index(N_x,i,j);%ux: i,j
635             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
636             J=N/2+Index(N_x,i,j-1);%ux: i,j-1
637             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
638             J=N/2+Index(N_x,i-1,j-1);%ux: i-1,j-1
639             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
640             f(I)=f(I)+f_y(i-1,j-1);
641             %2nd equ: div.(sigma)x=0
642             I=N/2+Index(N_x,i,j);%ux: i,j
```

```
643             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
644                  3*miu(i-1,j-1)/dy^2;
645             J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1/2
646             A(I,J)=2*miu(i-1,j-1)/dy^2;
647             J=N/2+Index(N_x,i,j-1);%ux: i,j-1
648             A(I,J)=miu(i-1,j-1)/dy^2;
649             J=N/2+Index(N_x,i+1,j);%ux: i+1/2,j
650             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
651             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
652             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
653             J=Index(N_x,i,j-1);%uy: i,j-1
654             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
655             J=Index(N_x,i,j);%uy: i,j
656             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
657             J=Index(N_x,i-1,j);%uy: i,j
658             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
659             J=Index(N_x,i-1,j-1);%uy: i-1,j-1
660             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
661             f(I)=f(I)+f_x(i-1,j-1);
662
663         elseif (j==(N_y+2)&&i==1) %Point 19 BC at free interface
664             if BC_t==1
665                 %ux: i,j+1/2=0
666                     I=N/2+Index(N_x,i,j);
667                     A(I,I)=-1;
668                     f(I)=f(I)+u_x_t(i);
669                 %uy: i,j+1/2=0
670                     I=Index(N_x,i,j);
671                     A(I,I)=-1;
672                     f(I)=f(I)+u_y_t(i);
673             elseif BC_t==2
674                %uy: i,j+1/2=0
675                     I=Index(N_x,i,j);%ux: i,j+1/2: Index
676                     A(I,I)=-1;
677                     f(I)=f(I)+u_y_t(i);
678                %sigma_yx=0 : ux: i,j+1/2
679                     I=N/2+Index(N_x,i,j);%ux: i,j+1/2: Index
680                     A(I,I)=2*(miu(i,j-2))/(dy);
681                     J=N/2+Index(N_x,i+1,j-1);%ux: i,j: Index
682                     A(I,J)=-2*(miu(i,j-2))/(dy);
683                     J=Index(N_x,i+2,j-1);%uy: i+1,j: Index
684                     A(I,J)=(miu(i,j-2))/(dx);
685                     J=Index(N_x,i+1,j-1);%uy: i,j: Index
686                     A(I,J)=-(miu(i,j-2))/(dx);
687                     f(I)=f(I)+sigma_yx_t(i);
688             else
689                 %sigma_yx=0 : ux: i,j+1/2
690                     I=N/2+Index(N_x,i,j);%ux: i,j+1/2: Index
691                     A(I,I)=2*(miu(i,j-2))/(dy);
692                     J=N/2+Index(N_x,i+1,j-1);%ux: i,j: Index
693                     A(I,J)=-2*(miu(i,j-2))/(dy);
694                     J=Index(N_x,i+2,j-1);%uy: i+1,j: Index
695                     A(I,J)=(miu(i,j-2))/(dx);
696                     J=Index(N_x,i+1,j-1);%uy: i,j: Index
697                     A(I,J)=-(miu(i,j-2))/(dx);
698                     f(I)=f(I)+sigma_yx_t(i);
699                 %sigma_yy=0 :uy: i,j+1/2
700                     I=Index(N_x,i,j);%uy: i,j+1/2
701                     A(I,I)=2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
702                     J=Index(N_x,i+1,j-1);%uy: i,j
703                     A(I,J)=-2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
704                     J=N/2+Index(N_x,i+2,j-1);%ux: i+1,j
705                     A(I,J)=(lambda(i,j-2))/(dx);
706                     J=N/2+Index(N_x,i+1,j-1);%ux: i,j
707                     A(I,J)=-(lambda(i,j-2))/(dx);
708                     f(I)=f(I)+sigma_yy_t(i);
```

```
709             end
710
711         elseif (j==(N_y+2)&&i==N_x) %Point 21 : BC at top free interface
712             if BC_t==1                          %left most interface
713               %ux: i,j+1/2=0
714                 I=N/2+Index(N_x,i,j);
715                 A(I,I)=-1;
716                 f(I)=f(I)+u_x_t(i);
717               %uy: i,j+1/2=0
718                 I=Index(N_x,i,j);
719                 A(I,I)=-1;
720                 f(I)=f(I)+u_y_t(i);
721             elseif BC_t==2
722                %uy: i,j+1/2=0
723                 I=Index(N_x,i,j);
724                 A(I,I)=-1;
725                 f(I)=f(I)+u_y_t(i);
726               %sigma_yx=0 : ux: i,j+1/2
727                 I=N/2+Index(N_x,i,j);%ux: i+1/2,j
728                 A(I,I)=2*(miu(i,j-2))/(dy);
729                 J=N/2+Index(N_x,i+1,j-1);%ux: i,j
730                 A(I,J)=-2*(miu(i,j-2))/(dy);
731                 J=Index(N_x,i,j-1);%uy: i,j-1
732                 A(I,J)=-(miu(i,j-2))/(dx);
733                 J=Index(N_x,i+1,j-1);%uy: i,j
734                 A(I,J)=(miu(i,j-2))/(dx);
735                 f(I)=f(I)+sigma_yx_t(i);
736             else
737              %sigma_yy=0 :uy: i,j+1/2
738                 I=Index(N_x,i,j);%uy: i,j+1/2
739                 A(I,I)=2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
740                 J=Index(N_x,i+1,j-1);%uy: i,j
741                 A(I,J)=-2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
742                 J=N/2+Index(N_x,i,j-1);%ux: i-1,j
743                 A(I,J)=-(lambda(i,j-2))/(dx);
744                 J=N/2+Index(N_x,i+1,j-1);%ux: i,j
745                 A(I,J)=(lambda(i,j-2))/(dx);
746                  f(I)=f(I)+sigma_yy_t(i);
747             %sigma_yx=0 : ux: i,j+1/2
748                 I=N/2+Index(N_x,i,j);%ux: i+1/2,j
749                 A(I,I)=2*(miu(i,j-2))/(dy);
750                 J=N/2+Index(N_x,i+1,j-1);%ux: i,j
751                 A(I,J)=-2*(miu(i,j-2))/(dy);
752                 J=Index(N_x,i,j-1);%uy: i,j-1
753                 A(I,J)=-(miu(i,j-2))/(dx);
754                 J=Index(N_x,i+1,j-1);%uy: i,j
755                 A(I,J)=(miu(i,j-2))/(dx);
756                 f(I)=f(I)+sigma_yx_t(i);
757             end
758
759
760         elseif ((2<i)&&(i<(N_x+1))&& j==2) % Point 6: middle cells at fixed end
761             % 1st equ: div.(sigma)y=0;
762             I=Index(N_x,i,j);%uy: i,j
763             A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
764                 2*miu(i-1,j-1)/dx^2;
765             J=Index(N_x,i,j+1);%uy: i,j+1
766             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
767             J=Index(N_x,i+1,j-1);%uy: i,j-1
768             A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
769             J=Index(N_x,i+1,j);%uy: i+1,j
770             A(I,J)=(miu(i-1,j-1)/dx^2);
771             J=Index(N_x,i-1,j);%uy: i-1,j
772             A(I,J)=(miu(i-1,j-1)/dx^2);
773             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
774             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
```

```
775             J=N/2+Index(N_x,i+1,j+1);%ux: i+1,j+1
776             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
777             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
778             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
779             J=N/2+Index(N_x,i-1,j+1);%ux: i-1,j+1
780             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
781             f(I)=f(I)+f_y(i-1,j-1);
782             % 2nd equ: div.(sigma)x=0;
783             I=N/2+Index(N_x,i,j);%ux: i,j
784             A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
785                 3*miu(i-1,j-1)/dy^2;
786             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
787             A(I,J)=miu(i-1,j-1)/dy^2;
788             J=N/2+Index(N_x,i+1,j-1);%ux: i,j+1
789             A(I,J)=2*miu(i-1,j-1)/dy^2;
790             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
791             A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
792             J=N/2+Index(N_x,i-1,j);%ux: i-1,j
793             A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
794             J=Index(N_x,i+1,j);%uy: i+1,j
795             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
796             J=Index(N_x,i+1,j+1);%uy: i+1,j+1
797             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
798             J=Index(N_x,i-1,j);%uy: i-1,j
799             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
800             J=Index(N_x,i-1,j+1);%uy: i-1,j+1
801             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
802             f(I)=f(I)+f_x(i-1,j-1);
803         elseif ((2<j)&&(j<(N_y+1))&& i==(1))% Point 9
804             if BC_l==1                      %Right hand side none corner point
805               %ux: i+1/2,j=0
806                 I=N/2+Index(N_x,i,j);%ux: i+1/2,j
807                 A(I,I)=-1;
808                 f(I)=f(I)+u_x_l(j-1);
809               %uy: i+1/2,j=0
810                 I=Index(N_x,i,j);%uy: i+1/2,j
811                 A(I,I)=-1;
812                 f(I)=f(I)+u_y_l(j-1);
813             elseif BC_l==2
814               %ux: i+1/2,j=0
815                 I=N/2+Index(N_x,i,j);%ux: i+1/2,j
816                 A(I,I)=-1;
817                 f(I)=f(I)+u_x_l(j-1);
818               %sigma_xy=0 for uy: i-1/2,j
819                 I=Index(N_x,i,j);%uy: i-1/2,j
820                 A(I,I)=-(2*miu(i,j-1)/dx);
821                 J=Index(N_x,i+1,j);%uy: i,j
822                 A(I,J)=(2*miu(i,j-1)/dx);
823                 J=N/2+Index(N_x,i+1,j+1);%ux: i,j+1
824                 A(I,J)=(3*miu(i,j-1)/(4*dy));
825                 J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1
826                 A(I,J)=-(3*miu(i,j-1)/(4*dy));
827                 J=N/2+Index(N_x,i+2,j+1);%ux: i+1,j+1
828                 A(I,J)=-(miu(i,j-1)/(4*dy));
829                 J=N/2+Index(N_x,i+2,j-1);%ux: i+1,j-1
830                 A(I,J)=(miu(i,j-1)/(4*dy));
831                 f(I)=f(I)+sigma_xy_l(j-1);
832             else
833               %sigma_xy=0 for uy: i-1/2,j
834                 I=Index(N_x,i,j);%uy: i-1/2,j
835                 A(I,I)=-(2*miu(i,j-1)/dx);
836                 J=Index(N_x,i+1,j);%uy: i,j
837                 A(I,J)=(2*miu(i,j-1)/dx);
838                 J=N/2+Index(N_x,i+1,j+1);%ux: i,j+1
839                 A(I,J)=(3*miu(i,j-1)/(4*dy));
840                 J=N/2+Index(N_x,i+1,j-1);%ux: i,j-1
```

```
841                 A(I,J)=-(3*miu(i,j-1)/(4*dy));
842                 J=N/2+Index(N_x,i+2,j+1);%ux: i+1,j+1
843                 A(I,J)=-(miu(i,j-1)/(4*dy));
844                 J=N/2+Index(N_x,i+2,j-1);%ux: i+1,j-1
845                 A(I,J)=(miu(i,j-1)/(4*dy));
846                 f(I)=f(I)+sigma_xy_l(j-1);
847              %sigma_xx=0 for ux: i-1/2,j
848                 I=N/2+Index(N_x,i,j);%ux: i-1/2,j
849                 A(I,I)=-(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
850                 J=N/2+Index(N_x,i+1,j);%ux: i,j
851                 A(I,J)=(2*(2*miu(i,j-1)+lambda(i,j-1))/dx);
852                 J=Index(N_x,i+1,j+1);%uy: i,j+1
853                 A(I,J)=(3/4)*lambda(i,j-1)/dy;
854                 J=Index(N_x,i+1,j-1);%uy: i,j-1
855                 A(I,J)=-(3/4)*(lambda(i,j-1))/dy;
856                 J=Index(N_x,i+2,j+1);%uy: i+1,j+1
857                 A(I,J)=(-1/4)*(lambda(i,j-1))/dy;
858                 J=Index(N_x,i+2,j-1);%uy: i+1,j-1
859                 A(I,J)=(+1/4)*(lambda(i,j-1))/dy;
860                 f(I)=f(I)+sigma_xx_l(j-1);
861              end
862
863
864
865           elseif((2<i)&&(i<(N_x+1))&&j==(N_y+1))%Point 16: middle top cells at free end
866              %1st equ: div.(sigma)y=0
867              I=Index(N_x,i,j);%uy: i,j
868              A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
869                   2*miu(i-1,j-1)/dx^2;
870              J=Index(N_x,i-1,j+1);%uy: i,j+1/2
871              A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
872              J=Index(N_x,i,j-1);%uy: i,j-1
873              A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
874              J=Index(N_x,i+1,j);%uy: i+1,j
875              A(I,J)=(miu(i-1,j-1)/dx^2);
876              J=Index(N_x,i-1,j);%uy: i-1,j
877              A(I,J)=(miu(i-1,j-1)/dx^2);
878              J=N/2+Index(N_x,i+1,j);%ux: i+1,j
879              A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
880              J=N/2+Index(N_x,i+1,j-1);%ux: i+1,j-1
881              A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
882              J=N/2+Index(N_x,i-1,j);%ux: i-1,j
883              A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
884              J=N/2+Index(N_x,i-1,j-1);%ux: i-1,j-1
885              A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
886              f(I)=f(I)+f_y(i-1,j-1);
887              %2nd equ: div.(sigma)x=0
888              I=N/2+Index(N_x,i,j);%ux: i,j
889              A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
890                   3*miu(i-1,j-1)/dy^2;
891              J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1/2
892              A(I,J)=2*miu(i-1,j-1)/dy^2;
893              J=N/2+Index(N_x,i,j-1);%ux: i,j-1
894              A(I,J)=miu(i-1,j-1)/dy^2;
895              J=N/2+Index(N_x,i+1,j);%ux: i+1,j
896              A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
897              J=N/2+Index(N_x,i-1,j);%ux: i-1,j
898              A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
899              J=Index(N_x,i+1,j);%uy: i+1,j
900              A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
901              J=Index(N_x,i+1,j-1);%uy: i+1,j-1
902              A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
903              J=Index(N_x,i-1,j);%uy: i-1,j
904              A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
905              J=Index(N_x,i-1,j-1);%uy: i-1,j-1
906              A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
```

```
907             f(I)=f(I)+f_x(i-1,j-1);
908         elseif ((1<i)&&(i<(N_x))&&j==(N_y+2)) %Point 20: BC at free interface
909             if BC_t==1
910                 %uy: i,j+1/2=0
911                     I=Index(N_x,i,j);
912                     A(I,I)=-1;
913                     f(I)=f(I)+u_y_t(i);
914                %ux: i,j+1/2=0
915                     I=N/2+Index(N_x,i,j);
916                     A(I,I)=-1;
917                     f(I)=f(I)+u_x_t(i);
918             elseif BC_t==2
919                 %uy: i,j+1/2=0
920                     I=Index(N_x,i,j);%uy: i,j+1/2
921                     A(I,I)=-1;
922                     f(I)=f(I)+u_y_t(i);
923                 %sigma_xy=0 : ux: i,j+1/2
924                     I=N/2+Index(N_x,i,j);%ux: i,j+1/2
925                     A(I,I)=2*(miu(i,j-2))/(dy);
926                     J=N/2+Index(N_x,i+1,j-1);%ux: i,j
927                     A(I,J)=-2*(miu(i,j-2))/(dy);
928                     J=Index(N_x,i+2,j-1);%uy: i+1,j
929                     A(I,J)=(3/4)*(miu(i,j-2))/(dx);
930                     J=Index(N_x,i,j-1);%uy: i-1,j
931                     A(I,J)=-(3/4)*(miu(i,j-2))/(dx);
932                     J=Index(N_x,i+2,j-2);%uy: i+1,j-1
933                     A(I,J)=-(1/4)*(miu(i,j-2))/(dx);
934                     J=Index(N_x,i,j-2);%uy: i-1,j-1
935                     A(I,J)=(1/4)*(miu(i,j-2))/(dx);
936                     f(I)=f(I)+sigma_yx_t(i);
937             else
938                %sigma_xy=0 : ux: i,j+1/2
939                     I=N/2+Index(N_x,i,j);%ux: i,j+1/2
940                     A(I,I)=2*(miu(i,j-2))/(dy);
941                     J=N/2+Index(N_x,i+1,j-1);%ux: i,j
942                     A(I,J)=-2*(miu(i,j-2))/(dy);
943                     J=Index(N_x,i+2,j-1);%uy: i+1,j
944                     A(I,J)=(3/4)*(miu(i,j-2))/(dx);
945                     J=Index(N_x,i,j-1);%uy: i-1,j
946                     A(I,J)=-(3/4)*(miu(i,j-2))/(dx);
947                     J=Index(N_x,i+2,j-2);%uy: i+1,j-1
948                     A(I,J)=-(1/4)*(miu(i,j-2))/(dx);
949                     J=Index(N_x,i,j-2);%uy: i-1,j-1
950                     A(I,J)=(1/4)*(miu(i,j-2))/(dx);
951                     f(I)=f(I)+sigma_yx_t(i);
952                 %sigma_yy=F/A :uy: i,j+1/2
953                     I=Index(N_x,i,j);%uy: i,j+1/2
954                     A(I,I)=2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
955                     J=Index(N_x,i+1,j-1);%uy: i,j
956                     A(I,J)=-2*(lambda(i,j-2)+2*miu(i,j-2))/(dy);
957                     J=N/2+Index(N_x,i+2,j-1);%ux: i+1,j
958                     A(I,J)=(3/4)*(lambda(i,j-2))/(dx);
959                     J=N/2+Index(N_x,i,j-1);%ux: i-1,j
960                     A(I,J)=-(3/4)*(lambda(i,j-2))/(dx);
961                     J=N/2+Index(N_x,i+2,j-2);%ux: i+1,j-1
962                     A(I,J)=-(1/4)*(lambda(i,j-2))/(dx);
963                     J=N/2+Index(N_x,i,j-2);%ux: i-1,j-1
964                     A(I,J)=(1/4)*(lambda(i,j-2))/(dx);
965                     f(I)=f(I)+sigma_yy_t(i);
966             end
967
968
969          elseif((2<j)&&(j<(N_y+1))&&((2<i)&&(i<(N_x+1))))%Point 11
970             %1st equ: div.(sigma)y=0   %middle cell at none corner points
971             I=Index(N_x,i,j);%uy: i,j
972             A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-...
```

```
973                 2*miu(i-1,j-1)/dx^2;
974             J=Index(N_x,i,j+1);%uy: i,j+1
975             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
976             J=Index(N_x,i,j-1);%uy: i,j-1
977             A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
978             J=Index(N_x,i+1,j);%uy: i+1,j
979             A(I,J)=(miu(i-1,j-1)/dx^2);
980             J=Index(N_x,i-1,j);%uy: i-1,j
981             A(I,J)=(miu(i-1,j-1)/dx^2);
982             J=N/2+Index(N_x,i+1,j+1);%ux: i+1,j+1
983             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
984             J=N/2+Index(N_x,i-1,j-1);%ux: i-1,j-1
985             A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
986             J=N/2+Index(N_x,i-1,j+1);%ux: i-1,j+1
987             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
988             J=N/2+Index(N_x,i+1,j-1);%ux: i+1,j-1
989             A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
990             f(I)=f(I)+f_y(i-1,j-1);
991             %2nd equ: div.(sigma)x=0
992             I=N/2+Index(N_x,i,j);%ux: i,j
993             A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
994                 2*miu(i-1,j-1)/dy^2;
995             J=N/2+Index(N_x,i,j+1);%ux: i,j+1
996             A(I,J)=miu(i-1,j-1)/dy^2;
997             J=N/2+Index(N_x,i,j-1);%ux: i,j-1
998             A(I,J)=miu(i-1,j-1)/dy^2;
999             J=N/2+Index(N_x,i+1,j);%ux: i+1,j
1000            A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
1001            J=N/2+Index(N_x,i-1,j);%ux: i-1,j
1002            A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
1003            J=Index(N_x,i+1,j+1);%uy: i+1,j+1
1004            A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
1005            J=Index(N_x,i-1,j-1);%uy: i-1,j-1
1006            A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
1007            J=Index(N_x,i-1,j+1);%uy: i-1,j+1
1008            A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
1009            J=Index(N_x,i+1,j-1);%uy: i+1,j-1
1010            A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(4*dy*dx);
1011            f(I)=f(I)+f_x(i-1,j-1);
1012            elseif((2<j)&&(j<(N_y+1))&&i==(N_x+1))%Point 12
1013            %1st equ: div.(sigma)y=0 %left hand side cell at none corner
1014            I=Index(N_x,i,j);%uy: i,j
1015            A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-3*miu(i-1,j-1)/dx^2;
1016            J=Index(N_x,i,j+1);%uy: i,j+1
1017            A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
1018            J=Index(N_x,i,j-1);%uy: i,j-1
1019            A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
1020            J=Index(N_x,i-1,j);%uy: i-1,j
1021            A(I,J)=(miu(i-1,j-1)/dx^2);
1022            J=Index(N_x,i+1,j);%uy: i+1/2,j
1023            A(I,J)=2*(miu(i-1,j-1)/dx^2);
1024            J=N/2+Index(N_x,i,j+1);%ux: i,j+1
1025            A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1026            J=N/2+Index(N_x,i-1,j-1);%ux: i-1,j-1
1027            A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1028            J=N/2+Index(N_x,i,j-1);%ux: i,j-1
1029            A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1030            J=N/2+Index(N_x,i-1,j+1);%ux: i-1,j+1
1031            A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1032            f(I)=f(I)+f_y(i-1,j-1);
1033            %2nd equ: div.(sigma)x=0;
1034            I=N/2+Index(N_x,i,j);%ux: i,j
1035            A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
1036                2*miu(i-1,j-1)/dy^2;
1037            J=N/2+Index(N_x,i,j+1);%ux: i,j+1
1038            A(I,J)=miu(i-1,j-1)/dy^2;
```

```
1039                J=N/2+Index(N_x,i,j-1);%ux: i,j-1
1040                A(I,J)=miu(i-1,j-1)/dy^2;
1041                J=N/2+Index(N_x,i+1,j);%ux: i+1/2,j
1042                A(I,J)=2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
1043                J=N/2+Index(N_x,i-1,j);%ux: i-1,j
1044                A(I,J)=((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2));
1045                J=Index(N_x,i,j+1);%uy: i,j+1
1046                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1047                J=Index(N_x,i-1,j-1);%uy: i-1,j-1
1048                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1049                J=Index(N_x,i,j-1);%uy: i,j-1
1050                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1051                J=Index(N_x,i-1,j+1);%uy: i-1,j+1
1052                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
1053                f(I)=f(I)+f_x(i-1,j-1);
1054             elseif ((2<j)&&(j<(N_y+1))&&i==(N_x+2))%Point 13
1055                if BC_r==1                  %on left hand side none corner point
1056                   %ux: i+1/2,j=0
1057                         I=N/2+Index(N_x,i,j);%ux: i+1/2,j
1058                         A(I,I)=-1;
1059                         f(I)=f(I)+u_x_r(j-1);
1060                   %uy: i+1/2,j=0
1061                         I=Index(N_x,i,j);%uy: i+1/2,j
1062                         A(I,I)=-1;
1063                         f(I)=f(I)+u_y_r(j-1);
1064                elseif BC_r==2
1065                   %ux: i+1/2,j=0
1066                         I=N/2+Index(N_x,i,j);%ux: i+1/2,j
1067                         A(I,I)=-1;
1068                         f(I)=f(I)+u_x_r(j-1);
1069                   %sigma_xy=0 for uy: i+1/2,j
1070                         I=Index(N_x,i,j);%uy: i+1/2,j
1071                         A(I,I)=(2*miu(i-2,j-1)/dx);
1072                         J=Index(N_x,i-1,j);%uy: i,j
1073                         A(I,J)=-(2*miu(i-2,j-1)/dx);
1074                         J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1
1075                         A(I,J)=(3*miu(i-2,j-1)/(4*dy));
1076                         J=N/2+Index(N_x,i-1,j-1);%ux: i,j-1
1077                         A(I,J)=-(3*miu(i-2,j-1)/(4*dy));
1078                         J=N/2+Index(N_x,i-2,j-1);%ux: i-1,j-1
1079                         A(I,J)=(miu(i-2,j-1)/(4*dy));
1080                         J=N/2+Index(N_x,i-2,j+1);%ux: i-1,j+1
1081                         A(I,J)=-(miu(i-2,j-1)/(4*dy));
1082                         f(I)=f(I)+sigma_xy_r(j-1);
1083                else
1084                   %sigma_xx=0 for ux: i+1/2,j
1085                         I=N/2+Index(N_x,i,j);%ux: i+1/2,j
1086                         A(I,I)=(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
1087                         J=N/2+Index(N_x,i-1,j);%ux: i,j
1088                         A(I,J)=-(2*(2*miu(i-2,j-1)+lambda(i-2,j-1))/dx);
1089                         J=Index(N_x,i-1,j+1);%uy: i+1,j
1090                         A(I,J)=(3/4)*lambda(i-2,j-1)/dy;
1091                         J=Index(N_x,i-1,j-1);%uy: i-1,j
1092                         A(I,J)=-(3/4)*lambda(i-2,j-1)/dy;
1093                         J=Index(N_x,i-2,j+1);%uy: i+1,j-1
1094                         A(I,J)=(-1/4)*lambda(i-2,j-1)/dy;
1095                         J=Index(N_x,i-2,j-1);%uy: i-1,j-1
1096                         A(I,J)=(+1/4)*lambda(i-2,j-1)/dy;
1097                         f(I)=f(I)+sigma_xx_r(j-1);
1098                    %sigma_xy=0 for uy: i+1/2,j
1099                         I=Index(N_x,i,j);%uy: i+1/2,j
1100                         A(I,I)=(2*miu(i-2,j-1)/dx);
1101                         J=Index(N_x,i-1,j);%uy: i,j
1102                         A(I,J)=-(2*miu(i-2,j-1)/dx);
1103                         J=N/2+Index(N_x,i-1,j+1);%ux: i,j+1
1104                         A(I,J)=(3*miu(i-2,j-1)/(4*dy));
```

```
                        J=N/2+Index(N_x,i-1,j-1);%ux: i,j-1
                        A(I,J)=-(3*miu(i-2,j-1)/(4*dy));
                        J=N/2+Index(N_x,i-2,j-1);%ux: i-1,j-1
                        A(I,J)=(miu(i-2,j-1)/(4*dy));
                        J=N/2+Index(N_x,i-2,j+1);%ux: i-1,j+1
                        A(I,J)=-(miu(i-2,j-1)/(4*dy));
                        f(I)=f(I)+sigma_xy_r(j-1);
                end



            elseif((2<j)&&(j<(N_y+1))&&i==(2))%Point 10: middle right hande side cells
                %1st equ: div.(sigma)y=0
                I=Index(N_x,i,j);%uy: i,j
                A(I,I)=-2*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2))-3*miu(i-1,j-1)/dx^2;
                J=Index(N_x,i,j+1);%uy: i,j+1
                A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
                J=Index(N_x,i,j-1);%uy: i,j-1
                A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dy^2);
                J=Index(N_x,i+1,j);%uy: i+1,j
                A(I,J)=(miu(i-1,j-1)/dx^2);
                J=Index(N_x,i-1,j);%uy: i-1/2,j
                A(I,J)=2*(miu(i-1,j-1)/dx^2);
                J=N/2+Index(N_x,i+1,j+1);%ux: i+1,j+1
                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=N/2+Index(N_x,i,j+1);%ux: i,j+1
                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=N/2+Index(N_x,i+1,j-1);%ux: i+1,j-1
                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=N/2+Index(N_x,i,j-1);%ux: i,j-1
                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                f(I)=f(I)+f_y(i-1,j-1);
                %2nd equ
                I=N/2+Index(N_x,i,j);%ux: i,j
                A(I,I)=-3*((2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2))-...
                    2*miu(i-1,j-1)/dy^2;
                J=N/2+Index(N_x,i,j+1);%ux: i,j+1
                A(I,J)=miu(i-1,j-1)/dy^2;
                J=N/2+Index(N_x,i,j-1);%ux: i,j-1
                A(I,J)=miu(i-1,j-1)/dy^2;
                J=N/2+Index(N_x,i+1,j);%ux: i+1,j
                A(I,J)=(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
                J=N/2+Index(N_x,i-1,j);%ux: i-1/2,j
                A(I,J)=2*(2*miu(i-1,j-1)+lambda(i-1,j-1))/(dx^2);
                J=Index(N_x,i+1,j+1);%uy: i+1,j+1
                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=Index(N_x,i,j+1);%uy: i,j+1
                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=Index(N_x,i+1,j-1);%uy: i+1,j-1
                A(I,J)=-(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                J=Index(N_x,i,j-1);%uy: i,j-1
                A(I,J)=(miu(i-1,j-1)+lambda(i-1,j-1))/(2*dy*dx);
                f(I)=f(I)+f_x(i-1,j-1);
            end
        end
    end
A=-A;% different convention displacement and stress is used
        %with compressive stresses being positive and extentional stresses
        % are also positive
u=A\f;
% u_y=u(1:N/2);
% u_x=u((N/2+1):end);
for j=2:(N_y+1)
    for i=2:(N_x+1)
        I=Index(N_x,i,j);
        Uy(i-1,j-1)=u(I);
```

```
1171            uy(i,j)=u(I);
1172            I=N/2+Index(N_x,i,j);
1173            Ux(i-1,j-1)=u(I);
1174            ux(i,j)=u(I);
1175        end
1176    end
1177    i=(N_x+2); %right interface
1178        for j=2:N_y+1
1179            I=Index(N_x,i,j);
1180            uy(i,j)=u(I);
1181            I=N/2+Index(N_x,i,j);
1182            ux(i,j)=u(I);
1183        end
1184    i=(1); %left interface
1185        for j=2:N_y+1
1186            I=Index(N_x,i,j);
1187            uy(i,j)=u(I);
1188            I=N/2+Index(N_x,i,j);
1189            ux(i,j)=u(I);
1190        end
1191    j=(N_y+2); %top interface
1192        for i=1:N_y
1193            I=Index(N_x,i,j);
1194            uy(i+1,j)=u(I);
1195            I=N/2+Index(N_x,i,j);
1196            ux(i+1,j)=u(I);
1197        end
1198    j=(1); %bottom interface
1199
1200        for i=3:N_y+2
1201            I=Index(N_x,i,j);
1202            uy(i-1,j)=u(I);
1203            I=N/2+Index(N_x,i,j);
1204            ux(i-1,j)=u(I);
1205        end
1206    % initialization
1207        dUx_dx=zeros(N_x+1,N_y);
1208        dUy_dx=zeros(N_x+1,N_y);
1209        dUy_dy=zeros(N_x,N_y+1);
1210        dUx_dy=zeros(N_x,N_y+1);
1211        dUx_dx_bar=zeros(N_x,N_y+1);
1212        dUy_dx_bar=zeros(N_x,N_y+1);
1213        dUy_dy_bar=zeros(N_x+1,N_y);
1214        dUx_dy_bar=zeros(N_x+1,N_y);
1215        Ux_bar=zeros(N_x+1,N_y);
1216        Uy_bar=zeros(N_x+1,N_y);
1217        stress_xx=zeros(N_x+1,N_y);
1218        stress_xy=zeros(N_x+1,N_y);
1219        stress_yy=zeros(N_x,N_y+1);
1220        stress_yx=zeros(N_x,N_y+1);
1221        div_x=zeros(N_x,N_y);
1222        div_y=zeros(N_x,N_y);
1223    % calculation
1224     for i=1:N_x+1
1225        j=2:N_y+1;
1226        dUx_dx(i,j-1)=(ux(i+1,j)-ux(i,j))/dx;
1227        dUy_dx(i,j-1)=(uy(i+1,j)-uy(i,j))/dx;
1228        if (i==1)||(i==N_x+1)
1229            dUx_dx(i,j-1)=2*dUx_dx(i,j-1);
1230            dUy_dx(i,j-1)=2*dUy_dx(i,j-1);
1231        end
1232
1233     end
1234     for j=1:N_y+1
1235        i=2:N_x+1;
1236        dUy_dy(i-1,j)=(uy(i,j+1)-uy(i,j))/dy;
```

```
    dUx_dy(i-1,j)=(ux(i,j+1)-ux(i,j))/dy;
    if (j==1)||(j==N_y+1)
        dUx_dy(i-1,j)=2*dUx_dy(i-1,j);
        dUy_dy(i-1,j)=2*dUy_dy(i-1,j);
    end
end



%Ux & Uy approximated at interfaces known as Ux_bar Uy_bar
for j=1:N_y+1
    if j==1
     for i=1:N_x+1
        if i==1
            Ux_bar(i,j)=7/4*(Ux(i,j))-1/4*(Ux(i+1,j)+Ux(i+1,j+1)+Ux(i,j+1));
            Uy_bar(i,j)=7/4*(Uy(i,j))-1/4*(Uy(i+1,j)+Uy(i+1,j+1)+Uy(i,j+1));
        elseif i==(N_x+1)
            Ux_bar(i,j)=7/4*(Ux(i-1,j))-1/4*(Ux(i-2,j)+Ux(i-2,j+1)+Ux(i-1,j+1));
            Uy_bar(i,j)=7/4*(Uy(i-1,j))-1/4*(Uy(i-2,j)+Uy(i-2,j+1)+Uy(i-1,j+1));
        else
            Ux_bar(i,j)=3/4*(Ux(i-1,j)+Ux(i,j))-1/4*(Ux(i,j+1)+Ux(i-1,j+1));
            Uy_bar(i,j)=3/4*(Uy(i-1,j)+Uy(i,j))-1/4*(Uy(i,j+1)+Uy(i-1,j+1));
        end
     end
    elseif j==N_y+1
    for i=1:N_x+1
        if i==1
            Ux_bar(i,j)=7/4*(Ux(i,j-1))-1/4*(Ux(i+1,j-1)+Ux(i+1,j-2)+Ux(i,j-2));
            Uy_bar(i,j)=7/4*(Uy(i,j-1))-1/4*(Uy(i+1,j-1)+Uy(i+1,j-2)+Uy(i,j-2));
        elseif i==(N_x+1)
            Ux_bar(i,j)=7/4*(Ux(i-1,j-1))-1/4*(Ux(i-2,j-1)+Ux(i-2,j-2)+Ux(i-1,j-2));
            Uy_bar(i,j)=7/4*(Uy(i-1,j-1))-1/4*(Uy(i-2,j-1)+Uy(i-2,j-2)+Uy(i-1,j-2));
        else
            Ux_bar(i,j)=3/4*(Ux(i-1,j-1)+Ux(i,j-1))-1/4*(Ux(i,j-2)+Ux(i-1,j-2));
            Uy_bar(i,j)=3/4*(Uy(i-1,j-1)+Uy(i,j-1))-1/4*(Uy(i,j-2)+Uy(i-1,j-2));
        end
    end

    else
        for i=1:N_x+1
            if i==1
                Ux_bar(i,j)=3/4*(Ux(i,j)+Ux(i,j-1))-1/4*(Ux(i+1,j)+Ux(i+1,j-1));
                Uy_bar(i,j)=3/4*(Uy(i,j)+Uy(i,j-1))-1/4*(Uy(i+1,j)+Uy(i+1,j-1));
            elseif i==(N_x+1)
                Ux_bar(i,j)=3/4*(Ux(i-1,j)+Ux(i-1,j-1))-1/4*(Ux(i-2,j)+Ux(i-2,j-1));
                Uy_bar(i,j)=3/4*(Uy(i-1,j)+Uy(i-1,j-1))-1/4*(Uy(i-2,j)+Uy(i-2,j-1));
            else
                Ux_bar(i,j)=(Ux(i,j)+Ux(i-1,j)+Ux(i,j-1)+Ux(i-1,j-1))/4;
                Uy_bar(i,j)=(Uy(i,j)+Uy(i-1,j)+Uy(i,j-1)+Uy(i-1,j-1))/4;
            end
        end
    end
end
for i=1:N_x
    dUx_dx_bar(i,:)=(Ux_bar(i+1,:)-Ux_bar(i,:))/dx;
    dUy_dx_bar(i,:)=(Uy_bar(i+1,:)-Uy_bar(i,:))/dx;
end
for j=1:N_y
    dUy_dy_bar(:,j)=(Uy_bar(:,j+1)-Uy_bar(:,j))/dy;
    dUx_dy_bar(:,j)=(Ux_bar(:,j+1)-Ux_bar(:,j))/dy;
end

[lambdaH_x,lambdaH_y,miuH_x,miuH_y]=Lambda_miu_Harmonic(N_x,N_y,lambda,miu);
% Stress Calculation
    for i=1:N_x+1
       stress_xx(i,:)=-((2*miuH_x(i,:)+lambdaH_x(i,:)).*dUx_dx(i,:)+...
```

```
            lambdaH_x(i,:).*dUy_dy_bar(i,:));
        stress_xy(i,:)=-(miuH_x(i,:).*(dUy_dx(i,:)+dUx_dy_bar(i,:)));
    end
    for j=1:N_y+1
        stress_yy(:,j)=-((2*miuH_y(:,j)+lambdaH_y(:,j)).*dUy_dy(:,j)+...
            lambdaH_y(:,j).*dUx_dx_bar(:,j));
        stress_yx(:,j)=-(miuH_y(:,j).*(dUx_dy(:,j)+dUy_dx_bar(:,j)));
    end

%% Calculating Divergence of stress
for i=1:N_x
    for j=1:N_y
        div_x(i,j)=(stress_xx(i+1,j)-stress_xx(i,j))/dx+(stress_yx(i,j+1)...
            -stress_yx(i,j))/dy-f_x(i,j);
        div_y(i,j)=(stress_xy(i+1,j)-stress_xy(i,j))/dx+(stress_yy(i,j+1)...
            -stress_yy(i,j))/dy-f_y(i,j);
    end
end
%% calculating diveregnce of displacement
    duxdx=zeros(N_x+1,N_y);
    duydy=zeros(N_x,N_y+1);
    % duxdx
        for j=1:N_y
            for i=1:N_x+1
                if ((i==1) || (i==N_x+1))
                    duxdx(i,j)=(ux(i+1,j+1)-ux(i,j+1))/(dx/2);
                else
                    duxdx(i,j)=(ux(i+1,j+1)-ux(i,j+1))/(dx);
                end
            end
        end
    % duydy
        for i=1:N_x
            for j=1:N_y+1
                if ((j==1) || (j==N_y+1))
                    duydy(i,j)=(uy(i+1,j+1)-uy(i+1,j))/(dy/2);
                else
                    duydy(i,j)=(uy(i+1,j+1)-uy(i+1,j))/(dy);
                end
            end
        end
     %divergance at cell centers
     duxdx_avg=zeros(N_x,N_y);
     duydy_avg=zeros(N_x,N_y);
      for i=1:N_x
          for j=1:N_y
              duxdx_avg(i,j)=(duxdx(i,j)+duxdx(i+1,j))/2;
              duydy_avg(i,j)=(duydy(i,j)+duydy(i,j+1))/2;
              divU(i,j)=duxdx_avg(i,j)+duydy_avg(i,j);
          end
      end
```

D.2 Codes for Fluid Solver

```
% Author:           Mitra Asadollahi #4412451
% Description:
% This script solve flow equation where the rheology law is Darcy's law.
% The solutions are valid for linearly compressible fluids.
global lambda_fl N_x N_y  BC_r_fl BC_l_fl BC_b_fl BC_t_fl  dx dy dt M b K_dr
  % setting BCs
    %if BC=2 no flow BC
    %if BC=1 constant pressure boundary condition
```

```
    % here only constant pressure condition is used
BC_r_fl=1;                  % flow BC at the right hand side
BC_l_fl=1;                  % flow BC at the left hand side
BC_b_fl=1;                  % flow BC at the bottom side
BC_t_fl=1;                  % flow BC at the top side
% Initialization
N_x=3;                      % number of grid points in x direction [-]
N_y=3;                      % number of grid points in y direction [-]
N_t=30;                     % number of timesteps
M=1;                        % stiffness of the medium [Pa]
L_x=1;                      % length of the medium in x direction [m]
L_y=1;                      % length of the medium in y direction [m]
t_end=0.5;                  % end time [s]
dx=L_x/N_x;                 % grid length [m]
dy=L_y/N_y;                 % grid length [m]
t_s=linspace(0,t_end,N_t);% discritized time vector[s]
dt=(t_end-0)/(N_t-1);       % time step [s]
x_c=dx/2:dx:L_x;            % grid cells in x direction [m]
y_c=dy/2:dy:L_y;            % grid cells in y direction [m]
x_i=0:dx:L_x;               % grid interface in x direction [m]
y_i=0:dy:L_y;               % grid interface in y direction [m]
p_ex=zeros(N_x,N_y,N_t);  % exact pressure solution [Pa]
p_nu=zeros(N_x,N_y,N_t);  % numerical pressure solution [Pa]
f=zeros(N_x,N_y,N_t);       % flow source terms [1/s]
dpdx_ex=zeros(N_x,N_y,N_t);%pressure gradients [Pa/m]
dpdy_nu=zeros(N_x,N_y,N_t);
dpdx_nu=zeros(N_x,N_y,N_t);
dpdy_ex=zeros(N_x,N_y,N_t);
P_r_fl=zeros(N_y,N_t);      %pressure [Pa]
P_l_fl=zeros(N_y,N_t);
P_b_fl=zeros(N_x,N_t);
P_t_fl=zeros(N_x,N_t);
% fluid flow value
lambda_fl=0.1;                      %fluid viscosity[Pa.s]
lambda_sl=0.1;                      %solid lame's parameter[Pa]
b=1;                                %biot coefficient[-]
miu=0.1;                            %shear modulus [Pa]
nu=lambda_sl/(2*(lambda_sl+miu));%possion's ratio [-]
K_dr=E/(3*(1-2*nu));                %drained bulk modulus [Pa]
% defining symbolic function
Df=0.0138;                          %consolidation coefficient
      % P=sin(y)*cos(x) % exp(-0.5^2*Df*t/L_x^2) %*exp(-0.5^2*Df*t/L_x^2);
          syms x y t
          P(x,y,t) =2*(lambda_sl+2*miu)*pi*sin(pi*x)*cos(pi*y)*sin(pi*t);
      % calculating second derivatives
          dp_xx = diff(P,x,2);
          dp_yy = diff(P,y,2);
      % 1st derivative
          dp_x = diff(P,x,1);
          dp_y = diff(P,y,1);
          dp_t= diff(P,t,1);

for t=1:N_t
      % calculating pressure values at BCs
      %right BC
          P_r_fl(:,t)=P(x_i(end),y_c,t_s(t));
      %left BC
          P_l_fl(:,t)=P(x_i(1),y_c,t_s(t));
      %bottom BC
          P_b_fl(:,t)=P(x_c,y_i(1),t_s(t));
      % top BC
          P_t_fl(:,t)=P(x_c,y_i(end),t_s(t));
% calculating exact solution & source term
  for i=1:length(x_c)
      p_ex(i,:,t)=P(x_c(i),y_c,t_s(t));
      for j=1:N_y
```

```
75                f(i,j,t)=(1/M+b^2/K_dr)*(dp_t(x_c(i),y_c(j),t_s(t)))-(lambda_fl)...
76                    *(dp_xx(x_c(i),y_c(j),t_s(t))+dp_yy(x_c(i),y_c(j),t_s(t)));
77            end
78            dpdx_ex(i,:,t)=dp_x(x_c(i),y_c,t_s(t));
79            dpdy_ex(:,i,t)=dp_y(x_c,y_c(i),t_s(t));
80        end
81
82    end
83  %initial codition
84  div_Vq=zeros(N_x,N_y,N_t);
85  div_Vq_ex=zeros(N_x,N_y,N_t);
86  p_nu(:,:,1)=p_ex(:,:,1);
87  % numerical solution
88  for t=2:N_t
89      Q=reshape(f(:,:,t),N_x*N_y,1);
90      [dpdy_last,Vy_last,div_V_q,A,C,q,p,dpdx_nu(:,:,t),dpdy_nu(:,:,t)...
91          ,LambdaH_y, LambdaH_x]=FlowSolver(Q,P_r_fl(:,t),P_l_fl(:,t),...
92          P_b_fl(:,t),P_t_fl(:,t),p_nu(:,:,t-1));
93      p_nu(:,:,t)=reshape(p,N_x,N_y);
94      div_Vq(:,:,t)=div_V_q;
95      %back calcylating equation from exact solution
96      [dpdx_eexx,dpdy_eexx,Vx_ex,Vy_ex]=GradientVelocity(N_x*N_y,N_x,...
97          N_y,reshape(p_ex(:,:,t),N_x*N_y,1),dx,dy,LambdaH_x,LambdaH_y,...
98          BC_r_fl,BC_l_fl,BC_t_fl,BC_b_fl,P_r_fl(:,t),P_l_fl(:,t),P_t_fl(:,t),...
99          P_b_fl(:,t));
100     [div_V_q]=Back_calc(N_x,N_y,Q,p_ex(:,:,t-1),p_ex(:,:,t),Vy_ex,...
101         Vx_ex,dx,dy);
102     div_Vq_ex(:,:,t)=div_V_q;
103 end
104
105 RMSD_p=zeros(N_t,1);
106 for t=1:N_t
107     figure(1);
108     hold on;
109     surf(y_c,x_c,p_ex(:,:,t));
110     xlabel('y axis');
111     ylabel('x axis');
112     title('P Exact Solution');
113     colorbar;
114     az = 30;
115     el = 0;
116     view(az, el);
117     figure(2);
118     hold on;
119     surf(y_c,x_c,p_nu(:,:,t));
120     xlabel('y axis');
121     ylabel('x axis');
122     title('P Numerical Solution');
123     colorbar;
124     az = 30;
125     el = 0;
126     view(az, el);
127     RMSD_p(t)=sqrt(mean(mean((p_ex(:,:,t)-p_nu(:,:,t)).^2)));
128     RMSD_dpdx(t)=sqrt(mean(mean((dpdx_ex(:,:,t)-dpdx_nu(:,:,t)).^2)));
129     RMSD_dpdy(t)=sqrt(mean(mean((dpdy_ex(:,:,t)-dpdy_nu(:,:,t)).^2)));
130 end
131 % error calculation: RMSD : sqrt(mean((y_exact-y_numerical)^2))
132     figure
133 surf(y_c,x_c,abs(p_ex(:,:,end)-p_nu(:,:,end)));
134 xlabel('y axis')
135 colorbar;
136 %% flow solver
137 function [dpdy,Vy,div_V_q,A,C,q,p,Dpdx,Dpdy,LambdaH_y, LambdaH_x]=...
138     FlowSolver(q,P_r_fl,P_l_fl,P_b_fl,P_t_fl,p_n)
139 global lambda_fl N_x N_y  BC_r_fl BC_l_fl BC_b_fl BC_t_fl  dx dy
140 % loc_wi loc_wj Pw PIw Well  L_x L_y
```

```
Dpdx=zeros(N_x,N_y);
Dpdy=zeros(N_x,N_y);
%% Input Values---------------------------------------------------------------
    % Input Values
        %phi=0.3; % Porosity, [frac.]
    % Discritization
        Nx = N_x;
        Ny = N_y;
        N  = N_x*N_y ; % Number of grid cells
    % Initializing Values
        Lambda = zeros( N,1);    % Lambda at Center of Grids
%% Homogeneous medium---------------------------------------------------------
    % Defining Lambda
        Lambda(1:N)=lambda_fl; % Homogeneous Reservoir
%% Harmonic Lambda Calculation------------------------------------------------
        [LambdaH_x,LambdaH_y]=Lambda_Harmonic(N,Nx,Ny,Lambda);

%% Transmissibility Calculation-----------------------------------------------
         [Tx,Ty] = Trans(dx,dy,Nx,Ny, Lambda);

 %% Defining Booundary Conditions---------------------------------------------
        % Pressure Boundary Condition already given in P_r_fl, etc.
        % No Flow Boundary Conditions
         [Ty,Tx]=BCs(Nx,Ny,Ty,Tx,BC_r_fl,BC_l_fl,BC_b_fl,BC_t_fl);
%% Main Script ---------------------------------------------------------------
% Building up Coeffient Matrix
        P_n=reshape(p_n,numel(p_n),1);
        q_f=q;
        [A,C,q]=Solution_MatV(N,q,Ty,Tx,Nx,Ny,BC_r_fl,P_r_fl,BC_l_fl,P_l_fl...
            ,BC_t_fl,P_t_fl,BC_b_fl,P_b_fl);
        p=(A+C)\(q+C*P_n);
%         p=(A)\(q);
% Building up Pressure Gradient & Velocity Calculation
    %dpdx & dpdy, velocity
        [dpdx,dpdy,Vx,Vy]=GradientVelocity(N,Nx,Ny,p,dx,dy,LambdaH_x,LambdaH_y...
            ,BC_r_fl,BC_l_fl,BC_t_fl,BC_b_fl,P_r_fl,P_l_fl,P_t_fl,P_b_fl);
% Back Calculating div(V)-q=0;
        [div_V_q]=Back_calc(Nx,Ny,q_f,p_n,reshape(p,N_x,N_y),Vy,Vx,dx,dy);
        for i=1:N_x
            for j=1:N_y
                I=Index_fl(N_x,N_y,i,j);
                I_t=Index_fl(N_x,N_y,i,j+1);
                Dpdy(i,j)=(dpdy(I)+dpdy(I_t))/2;
            end
        end
        for j=1:N_y
            for i=1:N_x
                I_r=(j-1)*(N_x+1)+i+1;
                I=(j-1)*(N_x+1)+i;
                Dpdx(i,j)=(dpdx(I)+dpdx(I_r))/2;
            end
        end
%% Solution Matrix
function[A,C,q]=Solution_MatV(N,q,Ty,Tx,Nx,Ny,BC_r_fl,P_r_fl,BC_l_fl...
        ,P_l_fl,BC_t_fl,P_t_fl,BC_b_fl,P_b_fl)
global M dt b K_dr
A = zeros(N,N);          % Coefficeint Matrix
C=(1/(M*dt)+b^2/(K_dr*dt)).*eye(N,N);  % Compressibility Vector
% implementing BC at none-boundary cells
                        for i=1:Nx
                         for j=1:Ny
                             I=Index_fl(Nx,Ny,i,j);
                                 if j>1
                                     Is=Index_fl(Nx,Ny,i,j-1);
                                      T=Ty(i,j);
                                      A(I,Is)=-T;
```

```
207                                     A(I,I)=A(I,I)+T;
208                                 end
209                                 if i>1
210                                     Iw=Index_fl(Nx,Ny,i-1,j);
211                                     T=Tx(i,j);
212                                     A(I,Iw)=-T;
213                                     A(I,I)=A(I,I)+T;
214                                 end
215                                 if i<Nx
216                                     Ie=Index_fl(Nx,Ny,i+1,j);
217                                     T=Tx(i+1,j);
218                                     A(I,Ie)=-T;
219                                     A(I,I)=A(I,I)+T;
220                                 end
221                                 if j<Ny
222                                     In=Index_fl(Nx,Ny,i,j+1);
223                                     T=Ty(i,j+1);
224                                     A(I,In)=-T;
225                                     A(I,I)=A(I,I)+T;
226                                 end
227
228                             end
229                         end
230 %          Defining Solution Vector
231 % implementing BC at LHS
232 i=1;
233                         for j=1:Ny
234                             I=Index_fl(Nx,Ny,i,j);
235                             A(I,I)=A(I,I)+Tx(i,j);
236                             if BC_r_fl==1
237                             q(I)=q(I)+Tx(i,j)*P_l_fl(j);
238                             end
239                         end
240 % implementing BC at RHS
241 i=Nx+1;
242                         for j=1:Ny
243                             I=Index_fl(Nx,Ny,i-1,j);
244                             A(I,I)=A(I,I)+Tx(i,j);
245                             if BC_l_fl==1
246                             q(I)=q(I)+Tx(i,j)*P_r_fl(j);
247                             end
248                         end
249 % implementing BC at THS
250 j=Ny+1;
251                         for i=1:Nx
252                             I=Index_fl(Nx,Ny,i,j-1);
253                             A(I,I)=A(I,I)+Ty(i,j);
254                             if BC_t_fl==1
255                             q(I)=q(I)+Ty(i,j)*P_t_fl(i);
256                             end
257                         end
258 % implementing BC at BHS
259 j=1;
260                         for i=1:Nx
261                             I=Index_fl(Nx,Ny,i,j);
262                             A(I,I)=A(I,I)+Ty(i,j);
263                             if BC_b_fl==1
264                             q(I)=q(I)+Ty(i,j)*P_b_fl(i);
265                             end
266                         end
267 end
```

D.3 Codes for Fixed Strain Sequentially Poroelasticity Solver

```
%Author:            Mitra Asadollahi #4412451
%Description:
% This code is used to sequentially couple the finite volume poroelastic
% solid deformation code and the finite volume linearly compressible fluid
% flow through fixed stress scheme. The output of this code are the
% equilibrium total and effective stresses, as well as, fluid pressure and
% flow rates. The governing equations are back calculated to check the
% correctness of implementation of the code.
global dt  Cv nu F nu_undr B x_c y_c dx dy b N_t K_dr BC_b BC_t BC_l BC_r...
    N_x N_y L_x L_y  lambda_fl  BC_r_fl BC_l_fl BC_b_fl BC_t_fl miu lambda_sl...
    M  P_r_fl  P_l_fl  P_t_fl  P_b_fl LambdaH_x LambdaH_y N y_i x_i t_obs
 % setting BCs for flow----------------------------------------------------------
    %if BC=2 no flow BC
    %if BC=1 constant pressure boundary condition
    % here only constant pressure condition is used
          BC_r_fl=1;
          BC_l_fl=2;
          BC_b_fl=2;
          BC_t_fl=2;
  % setting BCs for mechanics
    % 1: [ux uy]=[c1 c2];
    % 2: u.n=c1; (sigma.n).m=c2;
    % 3: sigma.n=[c1 c2]
        BC_b=2;              %  boundary condition at the bottom interface
        BC_t=3;              %  boundary condition at the top interface
        BC_l=2;              %  boundary condition at the left interface
        BC_r=3;              %  boundary condition at the right interface
 % medium properties
 % solid & fluid properties
    E=10^9;                          % young modulus [Pa]
    nu=0.35;                         % poisson ratio [-]
    lambda_sl=(E*nu)/((1+nu)*(1-2*nu));% lame's constant [Pa]
    miu=E/(2*(1+nu));                % shear modulus [Pa]
    perm_f=9.87*10^-16;              % fluid permeability [m^2]
    vis_fl=9.81*10^-5;               % fluid viscosity [Pa.s]
    lambda_fl=perm_f/vis_fl;         % mobility ratio
    phi=0.375;                       % porosity [-]
    K_dr=E/(3*(1-2*nu));             % drained bulk modulus [Pa]
    K_s=10^100; %3.6*10^10;          % solid bulk modulus [Pa]
    b=1-K_dr/K_s;                    % biot coefficient [-] : range [0.1,1]
cf=4.4*10^-10;                       % fluid compressibility [1/Pa]
    S=(phi*cf+(b-phi)/K_s);
    M=S^-1;                          % biot modulus [Pa]: range [10^4,10^14]
    K_oneD=K_dr+(4/3)*miu;           % uniaxial bulk modulus [Pa]
    Cv=lambda_fl*M*K_oneD/(K_oneD+b^2*M); % Consolidation Coefficient[m^2/s]
    K_f =1/cf;                       % fluid bulk modulus [Pa]
    B=b*M/(K_dr+b^2*M);              % skempton's coeficient [-]
    nu_undr=(3*nu+b*B*(1-2*nu))/(3-b*B*(1-2*nu)); % undrained poisson ratio [-]
    K_undr=K_dr+b^2*M;               % undrained bulk modulus [Pa]
    %0<alpha<0.5 dt/dx^2<=vis_fl/(2*(1-2*alpha)*(perm_f))*(1/M+b^2/K_dr)
    dtdxx_st=(vis_fl/(2*perm_f))*(1/M+b^2/K_dr);
    % domain description
      L_x=100;
      L_y=100;
      N_x=20;
      N_y=20;
      N_t=201;                           % number of timesteps [-]
      dx=L_x/N_x;
      dy=L_y/N_y;
      t_c=L_x^2/Cv;                      % charachteristic time [s]
      t_end=t_c;
      F=2*10^4;                          %force implemented at top
      load=(F/L_x);%psa
      tau=b^2*M/K_dr;                     % coupling strength
```

```
65        dt=t_end/(N_t-1);
66        x_c=dx/2:dx:L_x;
67        y_c=dy/2:dy:L_y;
68        x_i=0:dx:L_x;
69        y_i=0:dy:L_y;
70        t_s=linspace(0,t_end,N_t);
71
72    % initialization
73       div_u_in=zeros(N_x,N_y);
74       Ux_in=zeros(N_x+1,N_y);
75       Uy_in=zeros(N_x,N_y+1);
76       f_x=zeros(N_x,N_y,N_t);
77       f_y=zeros(N_x,N_y,N_t);
78       str_xx_l=zeros(N_y,N_t);
79       str_xx_r=zeros(N_y,N_t);
80       str_xy_l=zeros(N_y,N_t);
81       str_xy_r=zeros(N_y,N_t);
82       u_x_l=zeros(N_y,N_t);
83       u_x_r=zeros(N_y,N_t);
84       u_y_l=zeros(N_y,N_t);
85       u_y_r=zeros(N_y,N_t);
86       str_yy_b=zeros(N_x,N_t);
87       str_yy_t=load*ones(N_x,N_t);
88       str_yx_b=zeros(N_x,N_t);
89       str_yx_t=zeros(N_x,N_t);
90       u_x_b=zeros(N_x,N_t);
91       u_x_t=zeros(N_x,N_t);
92       u_y_b=zeros(N_x,N_t);
93       u_y_t=zeros(N_x,N_t);
94       P_r_fl=zeros(N_y,N_t);
95       P_l_fl=zeros(N_y,N_t);
96       P_t_fl=zeros(N_x,N_t);
97       P_b_fl=zeros(N_x,N_t);
98       p_ex=zeros(N_x,N_y,N_t);
99       f=zeros(N_x,N_y,N_t);
100      divU_ex=zeros(N_x,N_y,N_t);
101      U_yi=zeros(N_x,N_y+1);
102      U_xi=zeros(N_x+1,N_y);
103      t_obs=10;
104      jump=round( (N_t-1)/t_obs);
105   % Homogeneous medium----------------------------------------------------------
106      % Defining Lambda
107           N=N_x*N_y;
108           Lambda(1:N)=lambda_fl; % Homogeneous Reservoir
109  % Harmonic Lambda Calculation-------------------------------------------------
110     [LambdaH_x,LambdaH_y]=Lambda_Harmonic(N_x*N_y,N_x,N_y,Lambda);
111  %% constructing numerical solution--------------------------------------------
112  if 1<=tau
113  if (dt/dx^2>dtdxx_st)||(dt/dy^2>dtdxx_st)
114  display('coupling scheme is unstable either coarsen grids or refine timesteps');
115  end
116  end
117   % initial condition
118    t=1;
119      %Initial Pressure
120           p_ana=(1/(3*L_x))*B*(1+nu_undr)*F;
121      %Initial Displacement
122      for i=1:N_x+1
123           Ux_in(i,:)=(F*nu_undr/(2*miu*L_x)).*x_i(i);
124      end
125      for j=1:N_y+1
126           Uy_in(:,j)=-(F*(1-nu_undr)/(2*miu*L_x)).*y_i(j);
127      end
128    % divergence of displacement
129    for i=1:N_x
130        for j=1:N_y
```

```
131             div_u_in(i,j)=(Ux_in(i+1,j)-Ux_in(i,j))/dx+(Uy_in(i,j+1)...
132                 -Uy_in(i,j))/dy;
133         end
134     end
135     % pressure
136     p_in=p_ana*ones(N_x,N_y);
137             figure(7);
138             surf(y_c,x_c,p_in);
139             xlabel('x');
140             ylabel('y');
141             title('initial pressure at cell centers');
142             colorbar;
143             figure(8);
144             surf(y_c,x_c,div_u_in);
145             xlabel('x');
146             ylabel('y');
147             title('initial \nabla \cdot u at cell centers');
148             colorbar;
149     %initial condition is given to numerical solver as first guess
150     [A_f,A_s,f_f,f_s,stress_xx,stress_yy,dpdx_nu,dpdy_nu,div_u_nu,count,...
151         Ux_nu,Uy_nu,P,flow_equ,solid_equ_x,solid_equ_y,convg_p,convg_ux,...
152         convg_uy]=Poromechanic_Solver(div_u_in,p_in,f_x,f_y,f,P_r_fl,P_l_fl,...
153         P_b_fl,P_t_fl,str_xx_r,str_xy_r,str_yx_t,str_yy_t,str_yx_b,str_yy_b,...
154         str_xx_l,str_xy_l,u_y_r,u_x_r,u_y_l,u_x_l,u_y_t,u_x_t,u_y_b,u_x_b);
155     % error calculation
156 %% plotting ------------------------------------------------------------
157   t=2:N_t;
158 Uy_nu(5,:,1)=-(F*(1-nu_undr)/(2*miu*L_x)).*y_c;
159 Ux_nu(:,5,1)=(F*nu_undr/(2*miu*L_x)).*x_c;
160             figure(9);
161             i=1:jump:N_t;
162             hold on;
163             pp(:,i)=P(:,5,i);
164             plot(x_c, pp(:,i),'*-');
165             xlabel('y');
166             ylabel('x');
167             hold on;
168             title('P Numerical Solution');
169             colorbar;
170
171
172             figure(10);
173             i=1:jump:N_t;
174             hold on;
175             pp(:,i)=Uy_nu(5,:,i);
176             plot(y_c,pp(:,i),'*-');
177             xlabel('y');
178             ylabel('x');
179             title('U_y Numerical Solution');
180             colorbar;
181
182             figure(11);
183             i=1:jump:N_t;
184             hold on;
185             pp(:,i)=Ux_nu(:,5,i);
186             plot(x_c,pp(:,i),'*-');
187             xlabel('y');
188             ylabel('x');
189             title('U_x Numerical Solution');
190             colorbar;
191     hold off;
192 function [A_f,A_s,f_f,f_s,stress_xx,stress_yy,dpdx,dpdy,div_u,count,Ux,Uy,...
193     P,flow_equ,solid_equ_x,solid_equ_y,convg_p,convg_ux,convg_uy]=...
194     Poromechanic_Solver(div_u_in,p_in,f_x,f_y,f,P_r_fl,P_l_fl,P_b_fl,P_t_fl,...
195     str_xx_r,str_xy_r,str_yx_t,str_yy_t,str_yx_b,str_yy_b,str_xx_l,str_xy_l...
196     ,u_y_r,u_x_r,u_y_l,u_x_l,u_y_t,u_x_t,u_y_b,u_x_b)
```

```
197 global N_x N_y N_t b x_c y_c L_x L_y y_i x_i
198 % poromechanic solver follows the algorithm below to find the solution
199 % iteratively:
200 % 1. div_u is initially set to be equal to the previous timestep value
201 % 2. solve for pressure in: (1/M)dpdt+div(-lambda_fl*grad(p))=f-b*(d(div(u))/dt);
202 % 3. solve for displacement in:
203 % -(2*miu+lambda)*ddux_xx-(miu+lambda)*dduy_xy-miu*ddux_yy=f_x-b*dpdx;
204 % 4. update convergence criteria
205 % setting parameters
206     max_itr = 50; %iteration number
207     thrEr=10^-3;  %threshold error
208     stress_xx=zeros(N_x+1,N_y,N_t);
209     stress_yy=zeros(N_x,N_y+1,N_t);
210 % initialization
211     P=zeros(N_x,N_y,N_t);
212     Ux=zeros(N_x,N_y,N_t);
213     Uy=zeros(N_x,N_y,N_t);
214     div_u=zeros(N_x,N_y,N_t);
215     dpdx=zeros(N_x,N_y,N_t);
216     dpdy=zeros(N_x,N_y,N_t);
217     flow_equ=zeros(N_x,N_y,N_t);
218     solid_equ_x=zeros(N_x,N_y,N_t);
219     solid_equ_y=zeros(N_x,N_y,N_t);
220     convg_p=zeros(max_itr,N_t);
221     convg_ux=zeros(max_itr,N_t);
222     convg_uy=zeros(max_itr,N_t);
223     f_x_n=zeros(N_x,N_y,N_t);
224     f_y_n=zeros(N_x,N_y,N_t);
225     fx=zeros(N_x+2,N_y+2,N_t);
226     fy=zeros(N_x+2,N_y+2,N_t);
227 % initial condition : div_u & p_n are needed
228     t = 1;
229     div_u(:,:,t)=div_u_in;
230     P(:,:,t)=p_in;
231  while(t < N_t )
232      t = t+1;
233      count=0;
234      dP_0=1;
235      criteria_p=1;
236      criteria_ux=1;
237      criteria_uy=1;
238      criteria_itr_nr=1;
239     while ((criteria_p || criteria_uy || criteria_ux) && criteria_itr_nr)
240             count=count+1;
241     % 1. div_u is initially set to be equal to the previous timestep value
242         if count ==1 %refers to nr. 1 in solution algorithm
243             div_u(:,:,t)=div_u(:,:,t-1);%divU(nu)=divU(n-1) at first iteration
244         end
245
246     % 2. solve for pressure in: (1/M)dpdt+div(-lambda_fl*grad(p))=f-b*(d(div(u))/dt);
247         % constructing div.u(nu)-div.u(n-1)
248         q_s=(reshape(div_u(:,:,t),numel(div_u(:,:,t)),1))-...
249             (reshape(div_u(:,:,t-1),numel(div_u(:,:,t)),1));
250         % fluid source terms
251         q=reshape(f(:,:,t),N_x*N_y,1);
252         [A_f,f_f,div_V_q,p,dpdx(:,:,t),dpdy(:,:,t)]=FlowSolver(q,q_s,...
253             P_r_fl(:,t),P_l_fl(:,t),P_b_fl(:,t),P_t_fl(:,t),P(:,:,t-1));
254         flow_equ(:,:,t)=div_V_q;
255         P(:,:,t)=reshape(p,N_x,N_y);
256    % 3. solve for displacement in:
257         % -(2*miu+lambda)*ddux_xx-(miu+lambda)*dduy_xy-miu*ddux_yy=f_x-b*dpdx;
258         %mechanical part
259         f_x_n(:,:,t)=f_x(:,:,t)-b*dpdx(:,:,t);
260         f_y_n(:,:,t)=f_y(:,:,t)-b*dpdy(:,:,t);
261         [p_xx,p_yy]=P_conv(P(:,:,t),P_r_fl(:,t),P_l_fl(:,t),P_b_fl(:,t),...
262             P_t_fl(:,t));
```

```
263         %pressure at top
264         ef_pt=p_yy(:,N_y+1);
265         %pressure at bottom
266         ef_pb=p_yy(:,1);
267         %pressure at right
268         ef_pr=p_xx(N_x+1,:);
269         %pressure at left
270         ef_pl=p_xx(1,:);
271         [A_s,f_s,dUx_dx_bar, dUy_dy_bar,duxdx_avg,duydy_avg,fx(:,:,t),...
272             fy(:,:,t),stress_xx(:,:,t),stress_yy(:,:,t),U_x,U_y,div_x,div_y,...
273             divU]=DispSolver_verification(ef_pt,ef_pb,ef_pr,ef_pl,str_xx_r(:,t)...
274             ,str_xy_r(:,t),str_yx_t(:,t),str_yy_t(:,t),str_yx_b(:,t),...
275             str_yy_b(:,t),str_xx_l(:,t),str_xy_l(:,t),u_y_r(:,t),u_x_r(:,t)...
276             ,u_y_l(:,t),u_x_l(:,t),u_y_t(:,t),u_x_t(:,t),u_y_b(:,t),...
277             u_x_b(:,t),f_x_n(:,:,t),f_y_n(:,:,t));
278         solid_equ_x(:,:,t)=div_x;
279         solid_equ_y(:,:,t)=div_y;
280         Ux(:,:,t)=U_x;
281         Uy(:,:,t)=U_y;
282         div_u(:,:,t)=divU;
283     %4. updating convergence criteria
284         %iteration number should be less than maximum allowable nr. of itr.
285             criteria_itr_nr= (count < max_itr);
286         % change in pressure solution should be minimal
287             P_nu_plus=P(:,:,t);
288             if count==1  %at first iteration P_nu is set as P(:,t-1);
289                 P_nu=P(:,:,t-1);
290                 dP_0=norm ((P_nu-P_nu_plus),2);
291             end
292             dP=norm ((P_nu-P_nu_plus),2);
293             convg_p(count,t)=dP/dP_0;
294             P_nu=P_nu_plus;
295             criteria_p=(dP)>(thrEr*dP_0);
296         % change in ux solution should be minimal
297             ux_nu_plus=Ux(:,:,t);
298             if count==1
299                 ux_nu=Ux(:,:,t-1);
300                 dux_0=norm ((ux_nu-ux_nu_plus),2);
301             end
302             dux= norm ((ux_nu_plus-ux_nu),2);
303             ux_nu=ux_nu_plus;
304             convg_ux(count,t)=dux/dux_0;
305             criteria_ux=(dux>(thrEr*dux_0));
306         % change in uy should be minimal
307             uy_nu_plus=Uy(:,:,t);
308             if count==1
309                 uy_nu=Uy(:,:,t-1);
310                 duy_0=norm ((uy_nu-uy_nu_plus),2);
311             end
312             duy= norm ((uy_nu_plus-uy_nu),2);
313             uy_nu=uy_nu_plus;
314             convg_uy(count,t)=duy/duy_0;
315             criteria_uy=(duy>(thrEr*duy_0));
316 
317     end
318     if count==max_itr
319         display('system didnot converge');
320         break;
321     end
322 end
```

D.4 Codes for Fully Coupled Poroelasticity Solver

This code is private.

D.5 Codes for Synthetic Test Cases

The code for sequentially coupled method is presented here. For fully coupled method implementation would be similar.

```
%Author:            Mitra Asadollahi #4412451
%Description:
%This script generates source term and boundary and initial conditions from
%symbolic exact solution functions from the Biot's equations. Later, these
%source terms and conditions are feeded into the numerical solver and
%consequently numerical solutions are obtained. The error analysis
%conducted on this code can illustrate convergence and accuracy of the
%discritization.
global dt dx dy b K_dr N_t BC_b BC_t BC_l BC_r N_x N_y L_x L_y  lambda_fl ...
    BC_r_fl BC_l_fl BC_b_fl BC_t_fl miu lambda_sl M y_c x_c y_i x_i
 %% setting BCs----------------------------------------------------------
  % setting BCs for fluid
    %if BC=2 no flow BC
    %if BC=1 constant pressure boundary condition
    % here only constant pressure condition is used
          BC_r_fl=1;
          BC_l_fl=2;
          BC_b_fl=1;
          BC_t_fl=1;
  % setting BCs for mechanics
    % 1: [ux uy]=[c1 c2];
    % 2: u.n=c1; (sigma.n).m=c2;
    % 3: sigma.n=[c1 c2]
        BC_b=1;      %  boundary condition at the bottom interface
        BC_t=1;      %  boundary condition at the top interface
        BC_l=1;      %  boundary condition at the left interface
        BC_r=1;      %  boundary condition at the right interface
 % properties
      miu=0.1;            % shear modulus [Pa]
      lambda_sl=0.1;      % lame's constant[Pa]
      lambda_fl=0.1;      % fluid mobility[Pa]
      M=1;                % biot modulus[Pa]
      b=1;                % biot coefficient[-]
n=0;
for m=2%10:5:20
    n=n+1;
  % domain description
      L_x=1;            % length of the domain in x direction [m]
      L_y=1;            % length of the domain in y direction [m]
      T=0.5;            % end time [s]
      N_x=m;            % number of grid points in x direction [-]
      N_y=m;            % number of grid points in y direction [-]
      N_t=11;           % number of time steps [-]
      dx=L_x/N_x;       % grid size [m]
      dy=L_y/N_y;       % grid size [m]
      nu=lambda_sl/(2*(miu+lambda_sl));%possion's ratio [-]
      E=miu*(3*lambda_sl+2*miu)/(miu+lambda_sl);% elastic modulus[Pa]
      K_dr=E/(3*(1-2*nu));%drained bulk modulus [Pa]
      dt=T/(N_t-1);       %time steps [s]
      x_c=dx/2:dx:L_x;    % center of grid points in x direction [m]
      y_c=dy/2:dy:L_y;    % center of grid points in y direction [m]
      x_i=0:dx:L_x;       % interface of grid in x direction [m]
```

```
53        y_i=0:dy:L_y;        % interface of grid in y direction [m]
54        t_s=linspace(0,T,N_t);% time vector
55
56    % initialization
57      Ux_ex=zeros(N_x,N_y,N_t); %displacements exact and numerical[m]
58      Uy_ex=zeros(N_x,N_y,N_t);
59      Ux=zeros(N_x+1,N_y);
60      Uy=zeros(N_x,N_y+1);
61      f_x=zeros(N_x,N_y,N_t); %source terms to equilibrium equation [Pa/m]
62      f_y=zeros(N_x,N_y,N_t);
63      str_xx_l=zeros(N_y,N_t);% boundary stresses and displacement values [Pa]
64      str_xx_r=zeros(N_y,N_t);
65      str_xy_l=zeros(N_y,N_t);
66      str_xy_r=zeros(N_y,N_t);
67      u_x_l=zeros(N_y,N_t);
68      u_x_r=zeros(N_y,N_t);
69      u_y_l=zeros(N_y,N_t);
70      u_y_r=zeros(N_y,N_t);
71      str_yy_b=zeros(N_x,N_t);
72      str_yy_t=zeros(N_x,N_t);
73      str_yx_b=zeros(N_x,N_t);
74      str_yx_t=zeros(N_x,N_t);
75      u_x_b=zeros(N_x,N_t);
76      u_x_t=zeros(N_x,N_t);
77      u_y_b=zeros(N_x,N_t);
78      u_y_t=zeros(N_x,N_t);
79      P_r_fl=zeros(N_y,N_t);
80      P_l_fl=zeros(N_y,N_t);
81      P_t_fl=zeros(N_x,N_t);
82      P_b_fl=zeros(N_x,N_t);
83      p_ex=zeros(N_x,N_y,N_t);
84      f=zeros(N_x,N_y,N_t);
85      dpdx_ex=zeros(N_x,N_y,N_t);
86      dpdy_ex=zeros(N_x,N_y,N_t);
87      divU_ex=zeros(N_x,N_y,N_t);
88      U_yi=zeros(N_x,N_y+1);
89      U_xi=zeros(N_x+1,N_y);
90    %% definig solution functions-------------------------------------------------
91      % pressure
92          % P=sin(y)*cos(x)*cos(t)
93              syms x y t
94              P(x,y,t) =-20*(lambda_sl+2*miu)*pi*sin(pi*x)*cos(pi*y)*sin(pi*t);
95          % calculating second derivatives
96              dp_xx = diff(P,x,2);
97              dp_yy = diff(P,y,2);
98          % 1st derivative
99              dp_x = diff(P,x,1);
100             dp_y = diff(P,y,1);
101             dp_t = diff(P,t,1);
102     % displacement
103         % ux=sin(y)*cos(x)
104             syms x y t
105             u_x(x,y,t) =10*sin(pi*y)*cos(pi*x)*sin(pi*t);
106         % uy=y^2*x^3
107             syms x y t
108             u_y(x,y,t) =10*sin(pi*x)*cos(pi*y)*sin(pi*t);
109             % calculating second derivatives
110             df_ux_xx = diff(u_x,x,2);
111             df_ux_yy = diff(u_x,y,2);
112             df_uy_xx = diff(u_y,x,2);
113             df_uy_yy = diff(u_y,y,2);
114             df_uy_xy = diff(diff(u_y,x,1),y,1);
115             df_ux_xy = diff(diff(u_x,x,1),y,1);
116             % calculating first derivatives
117             df_ux_x = diff(u_x,x,1);
118             df_ux_y = diff(u_x,y,1);
```

```
            df_uy_y = diff(u_y,y,1);
            df_uy_x = diff(u_y,x,1);
            dtdxx_st=(1/(2*lambda_fl))*(1/M+b^2/K_dr);

%% constructing BCs----------------------------------------------------------
 for t=1:N_t
% calculating stresses at the interfaces

  % x=0; sigma_xx sigma_xy
       str_xx_l(:,t)=-lambda_sl*df_uy_y(x_i(1),y_c,t_s(t))-(2*miu+...
           lambda_sl)*df_ux_x(x_i(1),y_c,t_s(t));
       str_xy_l(:,t)=-miu*df_uy_x(x_i(1),y_c,t_s(t))-miu*df_ux_y(x_i(1),...
           y_c,t_s(t));
       u_x_l(:,t)=u_x(x_i(1),y_c,t_s(t));
       u_y_l(:,t)=u_y(x_i(1),y_c,t_s(t));
  % x=L_x; sigma_xx sigma_xy
       str_xx_r(:,t)=-lambda_sl*df_uy_y(x_i(end),y_c,t_s(t))-(2*miu+...
           lambda_sl)*df_ux_x(x_i(end),y_c,t_s(t));
       str_xy_r(:,t)=-miu*df_uy_x(x_i(end),y_c,t_s(t))-miu*...
           df_ux_y(x_i(end),y_c,t_s(t));
       u_x_r(:,t)=u_x(x_i(end),y_c,t_s(t));
       u_y_r(:,t)=u_y(x_i(end),y_c,t_s(t));
  % y=0; sigma_yy sigma_yx
       str_yy_b(:,t)=-lambda_sl*df_ux_x(x_c,y_i(1),t_s(t))-(2*miu+...
           lambda_sl)*df_uy_y(x_c,y_i(1),t_s(t));
       str_yx_b(:,t)=-miu*df_uy_x(x_c,y_i(1),t_s(t))-miu*df_ux_y(x_c,...
           y_i(1),t_s(t));
       u_x_b(:,t)=u_x(x_c,y_i(1),t_s(t));
       u_y_b(:,t)=u_y(x_c,y_i(1),t_s(t));
  % y=L_y; sigma_yy sigma_yx
      str_yy_t(:,t)=-lambda_sl*df_ux_x(x_c,y_i(end),t_s(t))-(2*miu+...
          lambda_sl)*df_uy_y(x_c,y_i(end),t_s(t));
      str_yx_t(:,t)=-miu*df_uy_x(x_c,y_i(end),t_s(t))-miu*df_ux_y(x_c,...
          y_i(end),t_s(t));
       u_x_t(:,t)=u_x(x_c,y_i(end),t_s(t));
       u_y_t(:,t)=u_y(x_c,y_i(end),t_s(t));
    % calculating pressure values at BCs
       %right BC
           P_r_fl(:,t)=P(x_i(end),y_c,t_s(t));
       %left BC
           P_l_fl(:,t)=P(x_i(1),y_c,t_s(t));
       %bottom BC
           P_b_fl(:,t)=P(x_c,y_i(1),t_s(t));
       % top BC
           P_t_fl(:,t)=P(x_c,y_i(end),t_s(t));
%% constructing source terms, exact solution--------------------------------
       %divergence of displacement at cell centers
        for i=1:N_x
            for j=1:N_y
                divU_ex(i,j,t)=df_ux_x(x_c(i),y_c(j),t_s(t))+...
                    df_uy_y(x_c(i),y_c(j),t_s(t));
            end
        end
       for i=1:length(x_c)
            Ux_ex(i,:,t)=u_x(x_c(i),y_c,t_s(t));
            Uy_ex(i,:,t)=u_y(x_c(i),y_c,t_s(t));
            p_ex(i,:,t)=P(x_c(i),y_c,t_s(t));
            if t==1
            else
               for j=1:N_y
                   f(i,j,t)=(1/M+b^2/K_dr)*dp_t(x_c(i),y_c(j),t_s(t))+...
                       (b/dt)*(divU_ex(i,j,t)-divU_ex(i,j,t-1))-...
                       (lambda_fl)*(dp_xx(x_c(i),y_c(j),t_s(t))+...
                       dp_yy(x_c(i),y_c(j),t_s(t)));
                   f_x(i,j,t)=-((lambda_sl+miu)*(df_ux_xx(x_c(i),y_c(j),...
                       t_s(t))+df_uy_xy(x_c(i),y_c(j),t_s(t)))...
```

```
                        +miu*(df_ux_xx(x_c(i),y_c(j),t_s(t))+df_ux_yy(x_c(i),...
                        y_c(j),t_s(t))))+b*dp_x(x_c(i),y_c(j),t_s(t));
                       f_y(i,j,t)=-((lambda_sl+miu)*(df_uy_yy(x_c(i),y_c(j),...
                            t_s(t))+df_ux_xy(x_c(i),y_c(j),t_s(t)))...
                        +miu*(df_uy_xx(x_c(i),y_c(j),t_s(t))+df_uy_yy(x_c(i)...
                        ,y_c(j),t_s(t))))+b*dp_y(x_c(i),y_c(j),t_s(t));
                    end
                end
                dpdx_ex(i,:,t)=dp_x(x_c(i),y_c,t_s(t));
                dpdy_ex(:,i,t)=dp_y(x_c,y_c(i),t_s(t));
            end


    end

    %% constructing numerical solution-----------------------------------------
   % initial condition
    t=1;
    % divergence of displacement
    div_u_in=divU_ex(:,:,1);
    % pressure
    p_in=p_ex(:,:,t);
    %initial condition is given to numerical solver as first guess
    [stress_xx,stress_yy,dpdx,dpdy,div_u,count,Ux_nu,Uy_nu,p_nu,flow_equ,...
        solid_equ_x,solid_equ_y,convg_p,convg_ux,convg_uy]=Poromechanic_Solver...
        (div_u_in,p_in,f_x,f_y,f,P_r_fl,P_l_fl,P_b_fl,P_t_fl,str_xx_r,...
        str_xy_r,str_yx_t,str_yy_t,str_yx_b,str_yy_b,str_xx_l,str_xy_l,...
        u_y_r,u_x_r,u_y_l,u_x_l,u_y_t,u_x_t,u_y_b,u_x_b);
% error calculation RMSE
for t=N_t
    RMSD_p(n)=sqrt(mean(mean((p_ex(:,:,t)-p_nu(:,:,t)).^2)));
    RMSD_ux(n)=sqrt(mean(mean((Ux_ex(:,:,t)-Ux_nu(:,:,t)).^2)));
    RMSD_uy(n)=sqrt(mean(mean((Uy_ex(:,:,t)-Uy_nu(:,:,t)).^2)));
    RMSD_divu(n)=sqrt(mean(mean((divU_ex(:,:,t)-div_u_nu(:,:,t)).^2)));
    RMSD_dpx(n)=sqrt(mean(mean((dpdx_ex(:,:,t)-dpdx_nu(:,:,t)).^2)));
    RMSD_dpy(n)=sqrt(mean(mean((dpdy_ex(:,:,t)-dpdy_nu(:,:,t)).^2)));
    DX(n)=L_x/N_x;
    DY(n)=L_y/N_y;
end
end
 figure;
plot(log(DX),log(RMSD_ux));
m_ux = polyfit(log(DX),log(RMSD_ux),1); %order of accuracy
title('Consistency Check for Ux');
xlabel('Ln(dx)');
ylabel('Ln(RMSD_ux)');
figure;
plot(log(DY),log(RMSD_uy));
m_uy = polyfit(log(DY),log(RMSD_uy),1); %order of accuracy
title('Consistency Check for Uy');
xlabel('Ln(dy)');
ylabel('Ln(RMSD_uy)');
figure;
plot(log(DX),log(RMSD_p));
m_p = polyfit(log(DX),log(RMSD_p),1); %order of accuracy
title('Consistency Check for Pressure');
xlabel('Ln(dx)');
ylabel('Ln(RMSD_p)');
```

D.6 Codes for Analytical Solution for Mandel Test Case

```
function [flow_equ,solid_equ_x,solid_equ_y,P_twD,Ux_twD,Uy_twD]=Analytical_Mandel
```

```
%-------------------------------------------------------------------------
% Author:                Mitra Asadollahi #4412451
% Description:
%       Mandel test case is one bench mark test case for plane strain
%       poroelasticity problem in 2D. Homogeneous case for fluid and solid.
%       Assuming linear-compressiblility for fluid and solid particles.
%       fluid problem BCs:
%                                no flow BC at left, top, and bottom
%                                drainage BC, i.e. p=0, at right BC
%       solid problem BCs:
%                               BC at top: constant normal stress
%                               BC at right: traction free surface
%                               BC at right and bottom: normal displacement
%                               to the surface being equal to zero
%                               *shear stress zero outside the boundaries
%-------------------------------------------------------------------------
global dt Cv miu nu N_x N_y  N_t L_x B nu_undr F x_c y_c b dx dy L_y N...
    P_r_fl P_l_fl P_t_fl P_b_fl
%initialization ---------------------------------------------------------
    eta=(1-nu)/(nu_undr-nu);            % coefficient in characteristic equ.
    P = zeros(N_x,N_t);                 % pressure in 1D
    Ux=zeros(N_x,N_t);                  % u_x in 1D
    Uy=zeros(N_y,N_t);                  % u_y in 1D
    solid_equ_x=zeros(N_x,N_y,N_t);     % back calculated (\nabla.\sigma)_x
    solid_equ_y=zeros(N_x,N_y,N_t);     % back calculated (\nabla.\sigma)_y
    flow_equ=zeros(N_x,N_y,N_t);        % back calculattion of storage equation
    divU=zeros(N_x,N_y,N_t);            % \nabla.u
    P_twD=zeros(N_x,N_y,N_t);           % pressure in 2D
    Ux_twD=zeros(N_x,N_y,N_t);          % u_x in 2D
    Uy_twD=zeros(N_x,N_y,N_t);          % u_y in 2D
    f_x=zeros(N_x,N_y,N_t);             % body forces acting as source term
    f_y=zeros(N_x,N_y,N_t);             % body forces acting as source term
    q_s=zeros(N_x,N_y,N_t);             % source terms from solid in fluid equ.
% initial condition-------------------------------------------------------
   x=[0 x_c L_x];
   y=[0 y_c L_y];
    t=1;
    %Initial Pressure
        P(:,t)=(1/(3*L_x))*B*(1+nu_undr)*F;
        P_twD(:,:,t)=(1/(3*L_x))*B*(1+nu_undr)*F;
    %Initial Displacement & Stress (assuming linear displacement)
        Ux(:,t)=(F*nu_undr/(2*miu*L_x)).*x_c;
        Uy(:,t)=-(F*(1-nu_undr)/(2*miu*L_x)).*y_c;

  for j=1:length(x_c)
        Ux_twD(:,j,t)=Ux(:,t);
        Uy_twD(j,:,t)=Uy(:,t);
  end

% timestep=2-end----------------------------------------------------------
% finding alpha_n
% positive root of tan(alpha_n)=((1-nu)/(nu_undr-nu))*alpha_n
No_roots=363; %max no. of roots used to plot analytical solution
a_n=zeros(length(No_roots));
No_roots=363;
range=[1e-12 pi/2-1e-12];
for i=1:No_roots %roots of tan(x)=eta*x (characteristic equ.)
    a_n(i)=fzero(@(x_c)tan(x_c)-eta*x_c,range);
    range=range+pi;
end
sum_p=zeros(length(x_c),N_t);
sum_ux_one=zeros(length(x_c),N_t);
sum_ux_two=zeros(length(x_c),N_t);
sum_uy=zeros(length(y_c),N_t);
t=1;
T(t)=(t-1)*dt;
```

```
for t=2:N_t
    T(t)=(t-1)*dt;
    for j=1:length(x_c)
        % the summation terms
        for i=1:length(a_n)
            sum_p(j,t)=sum_p(j,t)+((sin(a_n(i))/(a_n(i)-sin(a_n(i))*cos(a_n(i))))*...
                (cos((a_n(i)*x_c(j))/L_x)-cos(a_n(i)))*exp(-(Cv*a_n(i)^2*T(t))/L_x^2));
            sum_ux_one(j,t)=sum_ux_one(j,t)+( (sin(a_n(i))*cos(a_n(i)))/( a_n(i)-...
                sin(a_n(i))*cos(a_n(i))) )*exp(-(Cv*a_n(i)^2*T(t))/L_x^2);
            sum_ux_two(j,t)=sum_ux_two(j,t)+((cos(a_n(i))/(a_n(i)-sin(a_n(i))*...
                cos(a_n(i))))*sin(a_n(i)*x_c(j)/L_x)*exp(-(Cv*a_n(i)^2*T(t))/L_x^2));
            sum_uy(j,t)=sum_uy(j,t)+((sin(a_n(i))*cos(a_n(i)))/(a_n(i)-sin(a_n(i))*...
                cos(a_n(i))) )*exp(-(Cv*a_n(i)^2*T(t))/L_x^2);

        end
    end
end
sumtD_ux_one=zeros(length(x_c),N_t);
sumtD_ux_two=zeros(length(x_c),N_t);
sumtD_uy=zeros(length(y_c),N_t);
for t=2:N_t
    T(t)=(t-1)*dt;
    for j=1:length(x_c)
        % the summation terms
        for i=1:length(a_n)
        sumtD_ux_one(j,t)=sumtD_ux_one(j,t)+((sin(a_n(i))*cos(a_n(i)))/(a_n(i)-...
            sin(a_n(i))*cos(a_n(i))))*exp(-(Cv*a_n(i)^2*T(t))/L_x^2);
        sumtD_ux_two(j,t)=sumtD_ux_two(j,t)+( (cos(a_n(i))/(a_n(i)-sin(a_n(i)))...
            *cos(a_n(i))))*sin(a_n(i)*x_c(j)/L_x)*exp(-(Cv*a_n(i)^2*T(t))/L_x^2));
        sumtD_uy(j,t)=sumtD_uy(j,t)+((sin(a_n(i))*cos(a_n(i)))/( a_n(i)-...
            sin(a_n(i))*cos(a_n(i))))*exp(-(Cv*a_n(i)^2*T(t))/L_x^2);
        end
    end
end
% building up solution through time----------------------------------------
% 1D
for t=2:N_t
 for j=1:length(x_c)
    P(j,t)=2*P(j,1)*sum_p(j,t);
    P_twD(j,:,t)=P(j,t);
    Ux(j,t)=(F*nu/(2*miu*L_x)-(F*nu_undr/(miu*L_x))*sum_ux_one(j,t))*x_c(j)+...
        (F/miu)*sum_ux_two(j,t);
    Uy(j,t)=(-F*(1-nu)/(2*miu*L_x)+F*((1-nu_undr)/(miu*L_x))*sum_uy(j,t))*y_c(j);
 end
end


% 2D Analytical
for t=2:N_t
 for j=1:length(x_c)
   Ux_twD(j,:,t)=(F*nu/(2*miu*L_x)-(F*nu_undr/(miu*L_x))*sumtD_ux_one(j,t))...
       *x_c(j)+(F/miu)*sumtD_ux_two(j,t);
   Uy_twD(:,j,t)=(-F*(1-nu)/(2*miu*L_x)+F*((1-nu_undr)/(miu*L_x))*...
       sumtD_uy(j,t))*y_c(j);
 end
end
for t=2:N_t
    T(t)=(t-1)*dt;
    for j=1:length(x)
        % the summation terms
        for i=1:length(a_n)
            sumtD_uxD_one(j,t)=sumtD_uxD_one(j,t)+( (sin(a_n(i))*cos(a_n(i)))/...
              (a_n(i)-sin(a_n(i))*cos(a_n(i))))*exp(-(Cv*a_n(i)^2*T(t))/L_x^2);
            sumtD_uxD_two(j,t)=sumtD_uxD_two(j,t)+((cos(a_n(i))/(a_n(i)-...
                sin(a_n(i))*cos(a_n(i))))*sin(a_n(i)*x(j)/L_x)*exp(-(Cv*a_n(i)^2*...
            T(t))/L_x^2));
```

```
134                 sumtD_uyD(j,t)=sumtD_uyD(j,t)+((sin(a_n(i))*cos(a_n(i)))/(a_n(i)-...
135                     sin(a_n(i))*cos(a_n(i))))* exp(-(Cv*a_n(i)^2*T(t))/L_x^2);
136
137             end
138         end
139     end
140     % Back Calculation-------------------------------------------------------------
141     % Building up Pressure Gradient & Velocity Calculation
142     t=1;
143     [divU(:,:,t)]=div_u(ux_twD(:,:,t),uy_twD(:,:,t));
144     for t=2:N_t
145         %dpdx & dpdy, velocity
146                 [dpdx,dpdy,Vx,Vy]=GradientVelocity(reshape(P_twD(:,:,t),N_x*N_y,1),...
147                     P_r_fl,P_l_fl,P_t_fl,P_b_fl);
148                 f_x(:,:,t)=-b*dpdx;
149                 f_y(:,:,t)=-b*dpdy;
150          %b*d(div(U))/dt
151                  [divU(:,:,t)]=div_u(Ux_twD(:,:,t),Uy_twD(:,:,t));
152                  q_s=(reshape(divU(:,:,t),N_x*N_y,1))-(reshape(divU(:,:,t-1),N_x*N_y,1));
153                  q_s=(b/dt)*eye(N,N)*q_s;
154                  q_f=zeros(N_x*N_y,1);
155     % %b*d(div(u))/dt+(1/M+b^2/K_dr)*dpdt+div.(-lambda_fl*grad P)=0
156             [flow_equ(:,:,t)]=Back_calc(N_x,N_y,q_f,q_s,P_twD(:,:,t-1),P_twD(:,:,t)...
157                 ,Vy,Vx,dx,dy);
158
159     %div.(sigma)=0
160             [~,~,~,~,div_x,div_y]=Back_cal_solid(ux_twD,uy_twD,Ux_twD,Uy_twD,f_x,f_y);
161             solid_equ_x(:,:,t)=div_x;
162             solid_equ_y(:,:,t)=div_y;
163
164     end
165     % Plotting---------------------------------------------------------------------
166
167      %pressure
168      figure(1);
169
170           t= 1:jump:(N_t);
171              plot(x_c,P(:,t));
172              title('$$Analytical$$ $$Pressure$$ $$Solution$$ to $$Mandel$$ $$Problem$$'...
173                  ,'interpreter','latex','FontSize', 18);
174              xlabel('$$\bf{X},m $$ ','interpreter','latex','FontSize', 18);
175              ylabel('$$\bf{Pressure}, Pa $$','interpreter','latex','FontSize', 18);
176      %U_x
177              figure(2);
178              t= 1:jump:(N_t);
179              plot(x_c,Ux(:,t));
180              title('$$Analytical$$ $$U_x$$ $$Solution$$ to $$Mandel$$ $$Problem$$',...
181                  'interpreter','latex','FontSize', 18);
182              xlabel('$$\bf{X},m $$ ','interpreter','latex','FontSize', 18);
183              ylabel('$$\bf{U_x}, m $$ ','interpreter','latex','FontSize', 18);
184      %U_y
185              figure(3);
186              t= 1:jump:(N_t);
187              plot(y_c,Uy(:,t));
188              title('$$Analytical$$ $$U_y$$ $$Solution$$ to $$Mandel$$ $$Problem$$',...
189                  'interpreter','latex','FontSize', 18);
190              xlabel('$$\bf{Y},m $$ ','interpreter','latex','FontSize', 18);
191              ylabel('$$\bf{U_y}, m $$','interpreter','latex','FontSize', 18);
192
193     end
```

Nomenclature

Parameters related to Fluid

λ_t fluid mobility, k/μ_f, $[m^2/(Pa.s)]$

μ_f fluid viscosity, $[Pa.s]$

ρ_f fluid density, $[kg/m^3]$

C_f fluid compressibility, $[1/Pa]$

K_f fluid bulk modulus, equivalent to $1/C_f$, $[Pa]$

p pressure, $[Pa]$

q source term for fluid,$[1/s]$

$v_{x,y,z}$ fluid velocity in x,y,z directions, $[m/s]$

Abbreviated Terms

(LS) X-FEM (Least Square) Extended Finite Element Method

BEM Boundary Element Method

CC FVM Cell Centred-Cell Centred Finite Volume Method

CQ-BEM Convolution Quadrature Boundary Element Method

FDM Finite Difference Method

FEM Finite Element Method

FVM Finite Volume Method

GFEM Generalized Finite Element Method and Galerkin Finite Element Method

VC FVM Vertex Centred-Cell Centred Finite Volume Method

VV FVM Vertex Centred-Vertex Centred Finite Volume Method

Parameters for Additional Physics

ρ_b bulk density $[kg/m^3]$

$\sigma_{\theta\theta}$ normal angular stress $[Pa]$

$\mathbf{g}$ gravity force vector, $[m/s^2]$

k_α effective permeability of the phase α $[m^2]$

s^* total intrinsic heat capacity $[(J.Kg)/(m^3.K)]$

S_α saturation of the phase α $[-]$

T_0 initial temperature $[K]$

p_α pressure of the phase α $[Pa]$

α_f fluid thermal expansion coefficient $[1/K]$

α_s solid thermal expansion coefficient $[1/K]$

α_t total thermal expansion coefficient $[1/K]$

β thermal expansion factor $[Pa/K]$

$\Delta\rho$ difference between density of solid and fluid $[kg/m^3]$

ϕ_0 initial porosity value $[-]$

σ_{rr} normal radial stress $[Pa]$

$\varepsilon_{\theta\theta}$ pure angular strain $[-]$

ε_{rr} pure radial strain $[-]$

A, $A^{'}$, $B^{'}$, B positive equation constants $[-]$

C_f fluid heat capacity $[J/K]$

C_s solid heat capacity $[J/K]$

D p wave modulus,$(2\mu(1-\nu))/(1-2\nu)$, $[Pa]$

$d\boldsymbol{\varepsilon}^p$ incremental plastic strain tensor $[-]$

g_x/g_y gravitational field components $[m/s^2]$

h layer thickness $[m]$

K^* thermal conductivity $[N/(m.K)]$

k_0 initial permeability value $[m^2]$

K_f^* fluid thermal conductivity $[N/(m.K)]$

K_s^* solid thermal conductivity $[N/(m.K)]$

M gravity parameter $[-]$

q_T heat source term $[W]$

q_α source term of the phase α $[1/s]$

s intrinsic heat capacity $[(J.Kg)/(m^3.K)]$

u_θ angular displacement $[m]$

u_r radial displacement $[m]$

Parameters connecting Solid and Fluid

$\sigma^{'}$ effective stress, $[Pa]$

τ coupling strength, $[-]$

b Biot coefficient, formulated as $(1 - \frac{C_s}{C_m})$, varying between ϕ to 1, $[-]$

c consolidation coefficient, $[m^2/s]$

C_m matrix compressibility, equivalent to $1/K_{dr}$, $[1/Pa]$

K_{dr} drained bulk modulus, equivalent to $1/C_m$, $[Pa]$

S storativity, $[1/Pa]$

t_c characteristic time, $[s]$

V bulk volume, $[m^3]$

Parameters related to Solid

λ lame's first parameter showing incomprehensibility of the system, $[Pa]$

μ lame's second parameter or shear modulus showing rigidity of the system, $[Pa]$

ν poisson ratio, $[-]$

ϕ porosity, [-]

ρ_s solid density, $[kg/m^3]$

σ total stress, $[Pa]$

C_s solid compressibility, $[1/Pa]$

c_v consolidation coefficient, $[m^2/s]$

E elastic modulus, $[Pa]$

f_x source terms in solid equation in x direction, $[Pa/m]$

f_y source terms in solid equation in y direction, $[Pa/m]$

k absolute permeability of the medium, $[m^2]$

K_{1D} uniaxial bulk modulus, $[Pa]$

K_s solid bulk modulus, equivalent to $1/C_s$, $[Pa]$

u displacement, $[m]$

u_x displacement in x direction, $[m]$

u_y displacement in y direction, $[m]$

V_s solid volume, $[m^3]$

w solid particle velocity, $[m/s]$

Mathematical Operators

φ^ν φ at iteration numbered ν

$\left[\frac{\partial \psi}{\partial t}\right]^n$ derivative of ψ in time, i.e. $\frac{\psi^{n+1} - \psi^n}{\Delta t}$

$\Delta\psi$ variation of ψ

δ Kronecker delta

$\ell^2(\psi)$ 2-norm/Euclidean norm of ψ

ϵ smallest non-zero positive value possible

$\int$ integral operator

$\overline{\psi}_{\mathbf{i,j}}$ represents average value of ψ in cell i, j

∇ gradient operator

$\nabla.$ divergence operator

$\oint$ integral operator over closed surface

$\Omega_{i,j}$ area over the control volume of cell i, j

$\partial\psi/\partial i$ first derivative of ψ with respect to i

$\partial^2\psi/\partial i^2$ second derivative of ψ with respect to i

$\sqrt[2]{\psi}$ second root of ψ

$\sum$ summation

$\tilde{\psi}$ value of ψ estimated based on shape function

ζ threshold error

I identity matrix

m normal vector to n, i.e. normal vector the surface

n normal vector to the surface

$O(h^p)$ shows order of accuracy of p; in other words when h is halved, error is expected to be smaller with order of 0.5^p.

S represents surface

$tr(m)$ trace of the m matrix

V represents volume

References

Y. Abousleiman, A. H.-D. Cheng, L. Cui, E. Detournay, and J.-C. Roegiers. Mandel's problem revisited. *Geótechnique*, 1996.

D. R. Allen. Physical changes of reservoir properties caused by subsidence and repressurizing operations, wilmington field, california. *Journal of Petroleum Technology*, 20(1), 1968.

R. Asadi and B. Ataie-Ashtiani. Numerical modeling of subsidence in saturated porous media: A mass conservative method. *Journal of Hydrology*, 2016.

R. Asadi, B. Ataie-Ashtiani, and C. T. Simmons. Finite volume coupling strategies for the solution of a biot consolidation model. *Computers and Geotechnics*, 55:494–505, 2014.

I. Babuška. The finite element method with lagrangian multipliers. *Numerische Mathematik*, 20(3):179–192, 1973.

M. Bagheri and A. Settari. Modeling of geomechanics in naturally fractured reservoirs. *Reservoir Simulation Symposium*, 2005.

M. Bai and Y. Abousleiman. Thermoporoelastic coupling with application to consolidation. *International Journal for Numerical and Analytical Methods in Geomechanics*, 21(2):121–132, 1997.

P. K. Banerjee and R. Butterfield. Boundary element method in engineering science. *McGraw-Hill, London*, 1981.

L. Banjai and M. Schanz. Boundary-element modeling of three-dimensional anisotropic viscoelastic solids finite element method. *Journal of Advanced Materials*, 2015.

M. Bataee, S. Irawan, S. Ridha, H. Hematpour, Z. Hamdi, et al. Wellbore failure during water-alternating-gas injection by use of flow-stress coupling method. *SPE Journal*, 2016.

J. Bear. *Dynamics of fluids in porous media*. Courier Corporation, 2013.

E. Bemer, M. Boutéca, O. Vincké, N. Hoteit, and O. Ozanam. Poromechanics: From linear to nonlinear poroelasticity and poroviscoelasticity. *Oil & Gas Science and Technology*, 56(6): 531–544, 2001.

L. Berger, R. Bordas, D. Kay, and S. Tavener. A stabilized finite element method for nonlinear poroelasticity.

N. Berkowitz. *An introduction to coal technology*. Elsevier, 2012.

M. A. Biot. Consolidation settlement under a rectangular load distribution. *Journal of Applied Physics*, 1941a.

M. A. Biot. General theory of three-dimensional consolidation. *Journal of Applied Physics*, 1941b.

M. A. Biot. Theory of elasticity and consolidation for a porous anisotropic solid. *Journal of Applied Physics*, 1955.

M. A. Biot. General solutions of the equations of elasticity and consolidation for a porous material. *Journal of Applied Mechanics*, 1956a.

M. A. Biot. Thermoelastisity and irreversible thermodynamics. *Journal of Applied Physics*, 1956b.

M. A. Biot. Nonlinear and semilinear rheology of porous solids. *Journal of Geophysical Research*, 1962.

M. A. Biot and Willis D. G. The elastic coefficients of the theory of consolidation. *Journal of Applied Mechanics*, 1957.

T. I. Bjørnarå, J. M. Nordbotten, and J. Park. Vertically integrated models for coupled two-phase flow and geomechanics in porous media. *Water Resources Research*, 2016.

M. Borregales, F. A. Radu, K. Kumar, and J. M. Nordbotten. Numerical convergence of iterative coupling for non-linear biot's model. *Book of Abstracts Computational Methods in Applied Mathematics (CMAM-7) July 31-August 6, 2016 University of Jyväskylä, Finland*, page 53, 2016.

W. E. Boyce and R. C. DiPrima. Elementary differential equations and boundary value problems. *Jhon Wiley, New York*, 1977.

W_F Brace, JB Walsh, and WT Frangos. Permeability of granite under high pressure. *Journal of Geophysical research*, 73(6):2225–2236, 1968.

F. Brezzi and K. J. Bathe. A discourse on the stability conditions for mixed finite element formulations. *Computer methods in applied mechanics and engineering*, 82(1-3):27–57, 1990.

E. C. Bryant, J. Hwang, M. M. Sharma, et al. Arbitrary fracture propagation in heterogeneous poroelastic formations using a finite volume-based cohesive zone model. In *SPE Hydraulic Fracturing Technology Conference*. Society of Petroleum Engineers, 2015.

Y. Caillabet, P. Fabrie, P. Landerau, B. Noetinger, and M. Quintard. Implementation of finite-volume method for determination of effective parameters in fissured porous media. *Numerical Methods Partial Defferential Equations*, 2000.

P. Cardiff, Ž. Tuković, H. Jasak, and A. Ivanković. A block-coupled finite volume methodology for linear elasticity and unstructured meshes. *Computers & Structures*, 175:100–122, 2016.

N. Castelletto, J. A. White, and H. A. Tchelepi. Accuracy and convergence properties of the fixed-stress iterative solution of two-way coupled poromechanics. *International Journal for Numerical and Analytical Methods in Geomechanics*, 39(14):1593–1618, 2015.

D. Chapelle and K. J. Bathe. The inf-sup test. *Computers & structures*, 47(4):537–545, 1993.

A. H.-D. Cheng and E. Detournay. A direct boundary element method for plane strain poroelasticity. *International Journal of Numerical and Analytical Methods in Geomechanics*, 1988.

Alexander H-D Cheng, E Detournay, and Y Abousleiman. *Poroelasticity*, volume 27. Springer, 2016.

O. Coussy. Revisiting the constitutive equations of unsaturated porous solids using a lagrangian saturation concept. *International Journal for Numerical and Analytical Methods in Geomechanics*, 31(15):1675–1694, 2007.

S. L. Crouch and A. M. Starfield. Boundary element method in solid mechanics: With applications in rock mechanics and geological engineering. *Allen and Unwin, London*, 1983.

J D Bredehoeft and BB Hanshaw. On the maintenance of anomalous fluid pressures: I. thick sedimentary sequences. *Geological Society of America Bulletin*, 79(9):1097–1106, 1968.

R. Deb and P. Jenny. Numerical modeling of flow-mechanics coupling in a fractured reservoir with porous matrix. *41st Workshop on Geothermal Reservoir Engineering*, 2016.

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

E. Detournay and A. Peirce. On the moving boundary conditions for a hydraulic fracture. *International Journal of Engineering Science*, 84:147–155, 2014.

F. Doster, J. M. Nordbotten, et al. Full pressure coupling for geo-mechanical multi-phase multi-component flow simulations. In *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers, 2015.

W. Ehlers. Theoretical and numerical methods in continuum mechanics of porous materials. *IUTAM Symposium*, 1999.

Terry Engelder and Alfred Lacazette. Natural hydraulic fracturing. *Rock joints: Rotterdam, AA Balkema*, pages 35–44, 1990.

M. Ferronato, G. Gambolati, and P. Teatini. Ill-conditioning of finite element poroelasticity equations. *International Journal of Solids and Structures*, 38(34):5995–6014, 2001.

L. Formaggia, F. Saleri, and A. Veneziani. *Solving numerical PDEs: Problems, applications, exercises*. Springer Science & Business Media, 2012.

T. T. Garipov, M. Karimi-Fard, and H. A. Tchelepi. Discrete fracture model for coupled flow and geomechanics. *Journal of Computational Geoscience*, 2016.

F. J. Gaspar, F. J. Lisbona, and P. N. Vabishchevich. Staggered grid discretizations for the quasi-static biot's consolidation problem. *Applied numerical mathematics*, 56(6):888–898, 2006a.

F. J. Gaspar, F.J. Lisbona, C. W. Oosterlee, and P. N. Vabishchevich. An efficient multigrid solver for a reformulated version of the poroelasticity system. *Computer methods in applied mechanics and engineering*, 196(8):1447–1457, 2007.

F. J. Gaspar, J. L. Gracia, F. J. Lisbona, and C.W. Oosterlee. Distributive smoothers in multigrid for problems with dominating grad–div operators. *Numerical linear algebra with applications*, 15(8):661–683, 2008a.

F. J Gaspar, F. J Lisbona, P. Matus, and V. T. K. Tuyen. Numerical methods for a one-dimensional non-linear biot's model. *Journal of Computational and Applied Mathematics*, 293:62–72, 2016.

F.J. Gaspar, F.J. Lisbona, and C.W. Oosterlee. On a decoupled algorithm for poroelasticity and its resolution by multigrid. *European Conference on Computational Fluid Dynamics ECCOMAS CFD*, 2006b.

F.J. Gaspar, F.J. Lisbona, and C.W. Oosterlee. A stabilized difference scheme for deformable porous media and its numerical resolution by multigrid methods. *Computing and Visualization in Science*, 11(2):67–76, 2008b.

S. Granet, P. Fabrie, P. Lemonnier, and M. Quintard. A two phase flow simulation of a fractured reservoir using a new fissure element method. *International Journal of Petroleum Sciences and Engineering*, 2001.

J. Greetsma. Problems of rock mechanics in petroleum production engineering. *1st ISRM Congress*, 1966.

J. Greetsma. A basic theory of subsidence due to reservoir compaction: The homogenous case. *NJG Netherlands Journal of Geosciences (previously Geologie en Mijnbouw)*, 1973a.

J. Greetsma. Land subsidence above compacting oil and gas reservoirs. *Journal of Petroleum Technology*, 1973b.

J. Guzmán and M. Olshanskii. Inf-sup stability of geometrically unfitted stokes finite elements. *arXiv preprint arXiv:1605.09681*, 2016.

B. C. Haimson and C. Fairhurst. Hydraulic fracturing in porous-permeable materials. *Journal of Petroleum Technology*, 1969.

H. Hermansen, L. K. Thomas, J. E. Sylte, and B. T. Aasboe. Twenty five years of ekofisk reservoir management. *SPE Annual Technical Conference and Exhibition*, 1997.

P.A. Hsieh, J.V. Tracy, C.E. Neuzil, J.D. Bredehoeft, and S.E. Silliman. A transient laboratory method for determining the hydraulic properties of 'tight' rocks-i theory. *International Journal of Rock Mechanics and Mining Sciences & Geomechanics*, 1981.

L.A. Igumnov and I.P. Markov. Wave propagation problems treated with convolution quadrature and bem. *Fast Boundary Element Methods in Engineering and Industrial Applications*, 2012.

C. E. Jacob. On the flow of water in an elastic artesian acquifer. *Transactions American Geophysical Union*, 1940.

B. Jha and R. Juanes. A locally conservative finite element framework for the simulation of coupled flow and reservoir geomechanics. *Annual Technical Conference and Exhibition*, 2006.

L. Jing. A review of techniques, advances and outstanding issues in numerical modelling for rock mechanics and rock engineering. *International Journal of Rock Mechanics & Mining Sciences*, 2003.

L. Jing and J.A. Hudson. Numerical methods in rock mechanics. *International Journal of Rock Mechanics & Mining Sciences*, 2002.

E. Keilegavlen and J. M. Nordbotten. Finite volume methods for elasticity with weak symmetry. *arXiv preprint arXiv:1512.01042*, 2015.

M. Kelkar. *Natural gas production engineering*. PennWell Books, 2008.

J. Kim, H. A. Tchelepi, and R. Juanes. Stability, accuracy and efficiency of sequential methods for coupled flow and geomechanics. *SPE Reservoir Simulation Symposium*, 2009.

J. Kim, H.A. Tchelepi, and R. Juanes. Stability and convergence of sequential methods for coupled flow and geomechanics: Fixed-stress and fixed-strain splits. *Computer Methods in Applied Mechanics and Engineering*, 2011a.

J. Kim, H.A. Tchelepi, and R. Juanes. Stability and convergence of sequential methods for coupled flow and geomechanics: Drained and undrained splits. *Computer Methods in Applied Mechanics and Engineering*, 2011b.

J. Kim et al. Unconditionally stable sequential schemes for thermoporomechanics: Undrained-adiabatic and extended fixed-stress splits. In *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers, 2015.

A. Koliji. geotechdata.info, 2016. URL http://www.geotechdata.info/parameter/soil-young's-modulus.html.

J. Korsawe, G. Starke, W. Wang, and O. Kolditz. Finite element analysis of poro-elastic consolidation in porous media: Standard and mixed approaches. *Journal of Computer Methods in Applied Mechanics and Engineering*, 2006.

A.R. Lamb, G.J. Gorman, O.R. Gosselin, and A. Onaisi. Coupled deformation and fluid flow in fractured porous media using dual permeability and explicitly defined fracture geometry. *Annual Conference and Exhibition*, 2010.

T. Lay and T. C. Wallace. Modern global seismology. 1995.

Q. Lei, J. P. Latham, J. Xiang, C. F. Tsang, P. Lang, and L. Guo. Effects of geomechanical changes on the validity of a discrete fracture network representation of a realistic two-dimensional fractured rock. *International Journal of Rock Mechanics and Mining Sciences*, 70:507–523, 2014.

R. W Lewis, B.A Schrefler, and N Abd R. A finite element analysis of multiphase immiscible flow in deforming porous media for subsurface systems. *Communications in numerical methods in engineering*, 14(2):135–149, 1998.

G. Li. A four-node plane parametric element based on quadrilateral area coordinate and its application to coupled solid-deformation/fluid flow simulation for porous geomaterials. *International Journal for Numerical and Analytical Methods in Geomechanics*, 2015.

P. Li, L. Zhang, and P. Li. A novel approach to eliminate error induced by cell to node projections. *International Journal for Numerical and Analytical Methods in Geomechanics*, 2015.

S. Li, X. Li, and D. Zhang. A fully coupled thermo-hydro-mechanical, three-dimensional model for hydraulic stimulation treatments. *Journal of Natural Gas Science and Engineering*, 34: 64–84, 2016.

P. Luo, C. Rodrigo, F.J. Gaspar, and C. W. Oosterlee. Multigrid method for nonlinear poroelasticity equations. *Computing and Visualization in Science*, pages 1–11, 2015.

J. Ma, G. Zhao, and N. Khalili. A fully coupled flow deformation model for elasto-plastic damage analysis in saturated fractured porous media. *Journal of International Plasticity*, 2016.

J Mandel. Consolidation des sols (étude mathématique)*. *Geotechnique*, 3(7):287–299, 1953.

WD. McCain. Properties of petroleum fluids: Petroleum pub. *Co. Millero, FJ, Ward, GK, and Chetirkin, PV, Relative*, 1973.

J. McNamee and R. E. Gibson. Displacement functions and linear transforms applied to diffusion through porous elastic media. *Journal of Mechanics and Applied Mathematics*, 1960.

C. C. Mei. Gravity effects in consolidation of layer of soft soil. *Journal of engineering mechanics*, 111(8):1038–1047, 1985.

A. Mikelić and M. F. Wheeler. Convergence of iterative coupling for coupled flow and geomechanics. *Computational Geosciences*, 17(3):455–461, 2013.

S. E. Minkoff, C. M. Stoneb, S. Bryant, M. Peszynskac, and M. F. Wheeler. Coupled fluid flow and geomechanical deformation modeling. *Journal of Petroleum Science and Engineering*, 2003.

H. N. Moghaddam, A. Keyhanib, and I. Aghayan. Modeling of crack propagation in layered structures using extended finite element method. *Journal of Civil Engineering*, 2016.

R Mourgues, JB Gressier, L Bodet, D Bureau, and Aurélien Gay. "basin scale" versus "localized" pore pressure/stress coupling–implications for trap integrity evaluation. *Marine and Petroleum Geology*, 28(5):1111–1121, 2011.

M. A Murad and A. F.D. Loula. Improved accuracy in finite element analysis of biot's consolidation problem. *Computer Methods in Applied Mechanics and Engineering*, 95(3):359–382, 1992.

M. Musivand Arzanfudi. *Computational Modeling of Multiphysics Multidomain Multiphase Flow in Porous Media*. PhD thesis, Delft University of Technology, 2016.

CE Neuzil and DW Pollock. Erosional unloading and fluid pressures in hydraulically" tight" rocks. *The Journal of Geology*, pages 179–193, 1983.

C.E. Neuzil, C. Cooley, S.E. Silliman, J.D. Bredehoeft, and P.A. Hsieh. A transient laboratory method for determining the hydraulic properties of 'tight' rocks-ii application. *International Journal of Rock Mechanics and Mining Sciences & Geomechanics*, 1981.

H. Nilsen, J. M. Nordbotten, and X. Raynaud. Comparison between cell-centered and nodal based discretization schemes for linear elasticity. *arXiv preprint arXiv:1604.08410*, 2016.

J. M. Nordbotten. Cell-centered finite volume discretizations for deformable porous media. *International Journal for Numerical Methods in Engineering*, 100(6):399–418, 2014a.

J. M. Nordbotten. Finite volume hydromechanical simulation in porous media. *Water resources research*, 50(5):4379–4394, 2014b.

J. M. Nordbotten. Stable cell-centered finite volume discretization for biot equations. *ARXIV*, 2015a.

J. M. Nordbotten. Convergence of a cell-centered finite volume discretization for linear elasticity. *SIAM Journal on Numerical Analysis*, 53(6):2605–2625, 2015b.

C. W. Oosterlee and F. J. Gaspar. Multigrid relaxation methods for systems of saddle point type. *Applied Numerical Mathematics*, 58(12):1933–1950, 2008.

N. Oreskes, K. Shrader-Frechette, K. Belitz, et al. Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, 263(5147):641–646, 1994.

I. Palmer, J. Mansoori, et al. How permeability depends on stress and pore pressure in coalbeds: a new model. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1996.

M. Perić, R. Kessler, and G. Scheuerer. Comparison of finite-volume numerical methods with staggered and colocated grids. *Computers & Fluids*, 16(4):389–403, 1988.

PetroWiki. petrowiki@spe.org, 2016. URL http:http://petrowiki.org/Oil_fluid_characteristics.

P. Ya Polubarinova-Kochina. Theory of the motion of ground water. 1977.

E. W. Pratt and D. W. Janson. Local subsidence of the goose creek oil flied. *Journal of Geology*, 1926.

BC. Punmia and A. K. Jain. *Soil mechanics and foundations*. Firewall Media, 2005.

V. Reichenberger, H. Jakobs, P. Bastian, and R. Helmig. A mixed-dimensional finite volume method for two-phase flow in fractured porous media. *Advances in Water Resources*, 29(7): 1020–1036, 2006.

J. R. Rice and M. P. Clearly. Some basic stress diffusion solutions for fluid saturated elastic porous media with compressible constituents. *Review of Geophysics and Space Physics*, 1976.

C. Rodrigo, F.J. Gaspar, X. Hu, and L.T. Zikatanov. Stability and monotonicity for some discretizations of the biot's consolidation model. *Computer Methods in Applied Mechanics and Engineering*, 298:183–204, 2016.

E. A. Roeloffs. Fault stability changes induced beneath a reservoir with cyclic variations in water level. *Journal of Geophysics Research*, 1988.

E. A. Roeloffs. Poroelastic techniques in the study of earth-quakes-related hydrologic phenomena. *Advanced Geophysics*, 1995.

J. W. Rudnicki. Slip on impermeable fault in a fluid-saturated rock mass. *Earthquake Source Mechanics*, 1986.

J. W. Rudnicki. Plane strain dislocations in linear elastic diffusive solids. *Journal of Applied Mechanics*, 1987.

J. W. Rudnicki and T.-C. Hsu. Pore pressure changes induced by slip on permeable and impermeable faults. *Journal of Geophysics Research*, 1988.

J. W. Rudnicki and E. A. Roeloffs. Plane-strain shear dislocations moving steadily in linear elastic diffusive solids. *Journal of Geophysics Research*, 1988.

J. B. Schoonbeek. Land subsidence as a result of natural gas extraction in the province of groningen. *SPE European Spring Meeting*, 1976.

P. Segall. Stress and subsidence resulting from subsurface fluid withdrawal in the epicentra region of the 1983 coalinga earthquake. *Journal of Geophysics Research*, 1985.

P. Segall. Earthquakes triggered by fluid extraction. *Geology*, 1989.

P. Segall, J.-R. Grasso, and A. Mossop. Poroelastic stressing and induced seismicity near the lacq gas field, southwestern france. *Journal of Geophysics Research*, 1994.

Dongho Shin and John C Strikwerda. Inf-sup conditions for finite-difference approximations of the stokes equations. *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics*, 39(01):121–134, 1997.

D. W. Simpson. Seismicity changes associated with reservoir loading. *Journal of Engineering Geology*, 1976.

I. S. Sokolnikoff, R. D. Specht, et al. *Mathematical theory of elasticity*, volume 83. McGraw-Hill New York, 1956.

M. Talebian. *Computational Modeling of Hydro-electro-mechanical Flow during CO_2 Geosequestration*. PhD thesis, Delft University of Technology, 2015.

T. Tang, O. Hededal, and P. Cardiff. On finite volume method implementation of poro-elasto-plasticity soil model. *International Journal for Numerical and Analytical Methods in Geomechanics*, 2015.

A. Tavakoli and M. Ferronato. On existence-uniqueness of the solution in a nonlinear biot's model. *Appl. Math. Inf. Sci*, 7:333–341, 2013.

K. Terzaghi. Theoretical soil mechanics. *Jhon Wiley, New York*, 1943.

D. Trans, L. Nghiem, and L. Buchanan. An overview of iteratively coupling between geomechanical deformation and fluid flow. *International Thermal Operations and Heavy Oil Symposium*, 2005.

T. Tsai, K. Chang, and L. Huang. Body force effect on consolidation of porous elastic media due to pumping. *Journal of the Chinese Institute of Engineers*, 29(1):75–82, 2006.

C. Tseng, T. Tsai, and L. Huang. Effects of body force on transient poroelastic consolidation due to groundwater pumping. *Environmental geology*, 54(7):1507–1516, 2008.

R. Vanselow. Relations between fem and fvm applied to the poisson equation. *Journal of Computing*, 1996.

P. A. Vermeer and A. Verruijt. An accuracy condition for consolidation by finite elements. *International Journal for numerical and analytical methods in geomechanics*, 5(1):1–14, 1981.

A. Verruijt. *Theory and Problems of Poroelascticity*. http://geo.verruijt.net, 2016.

J Walder and A Nur. Permeability measurement by the pulse-decay method: Effects of poroelastic phenomena and non-linear pore pressure diffusion. In *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts*, volume 23, pages 225–232. Elsevier, 1986.

H. F. Wang. Effects of deviatoric stress on undarined pore pressure response to fault slip. *Journal of Geophysics Research*, 1997.

H. F. Wang. *Theory of Linear Poroelasticity with Applications to Geomechanics and Hydrogeology*. Princeton University Press . Princeton and Oxford, 2000.

W. Wang, G. Kosakowski, and O. Kolditz. A parallel finite element scheme for thermo-hydro-mechanical (thm) coupled problems in porous media. *Computers & Geosciences*, 35(8): 1631–1641, 2009.

W. D. Wang, J. G. Wang, Z. L. Wang, ZL, and T. Nogami. An unequal-order radial interpolation meshless method for biot's consolidation theory. *Computers and Geotechnics*, 34(2):61–70, 2007.

T. P. Wihler. Locking-free adaptive discontinuous galerkin fem foe linear elasticity problems. *Mathematics of Computation*, 2006.

J. Yang, J. Han Y. Lei, and S. Meng. Enriched finite element method for three-dimensional viscoelastic interface crack problems. *Journal of Mechanical Science and Technology*, 2016.

X Yi, PP Valko, JE Russell, et al. Predicting critical drawdown for the onset of sand production. In *SPE International Symposium and Exhibition on Formation Damage Control*. Society of Petroleum Engineers, 2004.