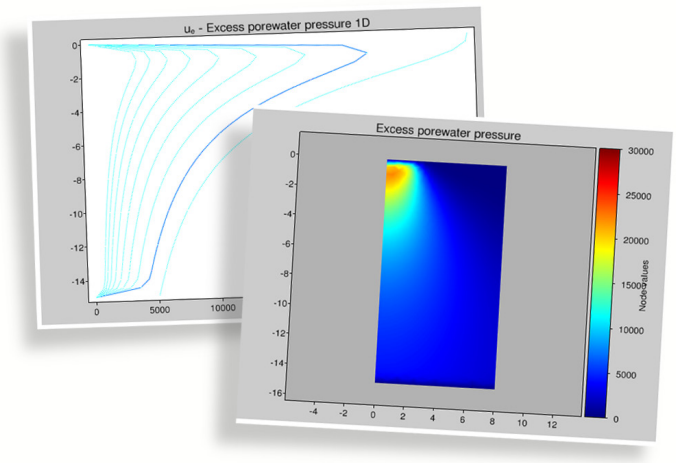
LUND UNIVERSITY



VisCon:
ETT VISUALISERINGSVERKTYG FÖR TVÅDIMENSIONELL KONSOLIDERING I UNDERVISNINGSSAMMANHANG

KARIN FORSMAN

Structural Mechanics

Master's Dissertation

DEPARTMENT OF CONSTRUCTION SCIENCES
DIVISION OF STRUCTURAL MECHANICS

ISRN LUTVDG/TVSM--17/5225--SE (1-99) | ISSN 0281-6679
MASTER'S DISSERTATION

VisCon: ETT VISUALISERINGSVERKTYG FÖR TVÅDIMENSIONELL KONSOLIDERING I UNDERVISNINGSSAMMANHANG

KARIN FORSMAN

Supervisors: Professor **OLA DAHLBLOM**, Div. of Structural Mechanics, LTH, together with **JONAS LINDEMANN**, PhD, Div. of Structural Mechanics, LTH & Lunarc and **ERIKA TUDISCO**, PhD, Div. of Geotechnical Engineering, LTH.

Examiner: **SUSANNE HEYDEN**, Senior Lecturer, Div. of Structural Mechanics, LTH.

Printed by Media-Tryck LU, Lund, Sweden, June 2017 *(Pl)*.

For information, address:
Division of Structural Mechanics,
Faculty of Engineering LTH, Lund University, Box 118, SE-221 00 Lund, Sweden.
Homepage: www.byggmek.lth.se

Förord

Det här arbetet utgör ett examensarbete, omfattande 30 högskolepoäng, och skrevs på avdelningen för Byggnadsmekanik vid Lunds Tekniska Högskola. Problemställningen utarbetades i samarbete med Erika Tudisco, Ola Dahlblom och Jonas Lindemann, som även varit handledare under arbetets gång. Jag vill passa på att tacka dem då de varit till stor hjälp i arbetets utformande och genomförande.

Karin Forsman

Juni 2017, Lund

Sammanfattning

Då ett jordlager belastas med en byggnad eller en anläggning ökar spänningarna i jorden. Initialt utgörs spänningsökningen av ett förhöjt porvattentryck. Allteftersom vatten transporteras kommer dock porvattenövertrycket att utjämnas. Det resulterar i en omfördelning av spänningarna. Förhållandet mellan dessa ges av effektivspänningsekvationen, som anger att en spänningsökning kan uttryckas som summan av porvattenövertrycket och effektivspänningen. Det innebär att då porvattenövertrycket minskar tilltar effektivspänningen. Processen då effektivspänningen ökar under en hydraulisk fördröjning refereras till som konsolidering. Effekten blir en ökad belastning på kornskelettet med efterföljande deformationer, så kallade sättningar. Jordens permeabilitet och vatteninnehåll avgör hur lång tid konsolideringen tar. I en jord med låg permeabilitet, så som lera, kan det ta lång tid innan sättningarna är fullt utvecklade.

I undervisningssammanhang är det vanligt att idealisera studier av konsolidering och enbart studera endimensionell vätskeströmning. Anledningen till den här förenklingen är att många studenter finner det tvådimensionella fallet svårt att förstå. Vidare kräver det tvådimensionella fallet mer komplicerade beräkningar som ofta förutsätter kunskap inom exempelvis finita elementmetoden. I det här arbetet har ett program, VisCon, utvecklats för att just underlätta förståelsen av tvådimensionell konsolidering. VisCon syftar till att öka förståelsen av fenomenet genom en tvådimensionell grafisk framställning av porvattenövertryckets tidsberoende, där beräkningen av porvattenövertrycket görs med hjälp av finita elementmetoden. Utöver den tvådimensionella framställningen ges endimensionella grafer för porvattenövertrycket och effektivspänningen vid valfri definierad sektion. Ett mål med VisCon är att förflytta studenternas fokus från att genomföra numeriska beräkningar till att fokusera på de faktiska geotekniska aspekterna av konsolideringen, så som att omvandla resultatet till sättningar.

I utvecklingen av datorprogram för undervisning är det viktigt att väga in begreppet användbarhet. Användbarheten kan mätas utifrån programmets anpassning, användaracceptans, användarkompetens samt användarvänlighet. Vidare har forskning visat att för att ett datorprogram ska fungera som ett pedagogiskt hjälpmedel måste kurslärare ges tillräckligt med tid för att själva lära sig programmet samt väva in det på ett genomtänkt sätt i undervisningen. För att VisCon ska kunna användas i undervisningssammanhang har en användarmanual samt tillhörande övningsmaterial tagits fram. Övningsmaterialet är utformat för att studenterna med hjälp av VisCon ska kunna genomföra sättningsberäkningar samt analysera beräkningsmodellens begränsningar.

Abstract

When a ground surface is subjected to a load an increased level of stress will occur in the soil underneath. As soil material has a limited permeability, an increased stress will initially result in an excess pore water pressure. However, as time passes the water will flow and the pore water pressure will even out. The relation between the total stress, the excess pore water pressure and the effective stress is given by the effective stress equation. It states that the total change in stress will always equal the sum of the excess pore water pressure and the effective stress. Consequently, when the excess pore water pressure diminishes the effective stress increases accordingly. This process is referred to as consolidation. It results in a greater load within the soil skeleton followed by settlements. The permeability and water content of the soil are determining characteristics of how long the consolidation process will take. In a soil with low permeability, such as clay, it can take a long time for the settlements to fully develop.

For educational purposes it is common to idealise the problem of consolidation and simply study one dimensional groundwater flow. The main reason behind this simplification is that many students struggle to picture the two dimensional case. Furthermore, a two dimensional model requires more complex calculations and often assumes knowledge within numerical solution methods such as the finite element method. The aim of this master's dissertation has been to enhance students understanding of two dimensional consolidation. In order to do so a program, VisCon, has been developed to visualise how the excess pore water pressure varies with time and space. The program is based on a two dimensional model solved by applying the finite element method. Besides the two dimensional visualisation of the results the program produces one dimensional graphs. These illustrate the excess pore water pressure as well as the effective stress for any defined section of the model. By assisting students with a numerical solution of the excess pore water pressure a second aim with VisCon has been to switch focus within education from numerical calculations to actual geotechnical aspects of consolidation.

When developing a program for educational purposes it is important to consider its usability. The usability can be determined by studying how well the program is adjusted to its area of application, considered target group, previous knowledge of users and whether it is user-friendly or not. Furthermore, for a program to function as pedagogic software is has to be integrated within the education in a deliberate way. To facilitate the use of VisCon a user's manual as well as exercises has been developed. The exercises are intended to guide students in using VisCon for calculating settlements in two dimensions as well as analysing restrictions of the implemented model.

Innehåll

Kapitel 1

Inledning

I det här arbetet har ett beräknings- och visualiseringsverktyg för tvådimensionell konsolidering utvecklats. Programmet är avsett för att användas i undervisningssammanhang. Detta kapitel ger en översikt av fenomenet konsolidering. I programutvecklingen har vidare hänsyn tagits till forskning rörande datorn som pedagogiskt hjälpmedel. I det här avsnittet ges därför även en introduktion av datorns roll i undervisningssammanhang. Utifrån bakgrundsfakta formuleras arbetets syfte, mål och metod. För att möjliggöra arbetet inom given tidsram har en del avgränsningar gjorts. Dessa, samt avsedd målgrupp, presenteras vidare i det här kapitlet. Även en disposition över rapporten ges för en ökad förståelse av arbetets struktur och omfattning.

1.1 Bakgrund

1.1.1 Konsolidering

Då ett jordlager belastas med en byggnad eller en anläggning ökar spänningarna i jorden. Initialt utgörs spänningsökningen av ett förhöjt porvattentryck. Allteftersom vatten transporteras kommer dock porvattenövertrycket att utjämnas. Det resulterar i en omfördelning av spänningarna. Förhållandet mellan dessa ges av effektivspänningsekvationen, som anger att en spänningsökning kan uttryckas som summan av porvattenövertrycket och effektivspänningen. Det innebär att då porvattenövertrycket minskar tilltar effektivspänningen. Processen då effektivspänningen ökar under en hydraulisk fördröjning refereras till som konsolidering. Effekten blir en ökad belastning på kornskelettet med efterföljande deformationer, så kallade sättningar. Jordens permeabilitet och vatteninnehåll avgör hur lång tid konsolideringen tar. I en jord med låg permeabilitet, så som lera, kan det ta lång tid innan sättningarna

är fullt utvecklade [21].

I undervisningssammanhang är det vanligt att idealisera studier av konsolidering och enbart studera endimensionell vätskeströmning. Anledningen till den här förenklingen är att många studenter finner det tvådimensionella fallet svårt att greppa. Vidare kräver det tvådimensionella fallet mer komplicerade beräkningar som ofta förutsätter något djupare kunskap inom exempelvis finita elementmetoden. Det är därför av intresse att på ett lättöverskådligt sätt beräkna och illustrera konsolidering i två dimensioner.

1.1.2 Datorstödd undervisning

Den så kallade datoriseringen inleddes under 1950-talet, en tid då komplexa och tidskrävande processer började lösas med hjälp av datorer. Datorn vann snabbt en allt viktigare roll på många fronter inom samhället. Inom den svenska skolundervisningen introducerades datorn för första gången under 1970-talet [12]. Datorinnehavet i svenska skolor har sedan dess ökat markant. År 1993 var det genomsnittliga datorinnehavet en dator per 25 elever. År 2012 hade många kommuner som målsättning att i åtminstone en skola inom kommunen tillgodose en dator per elev. Idag har de allra flesta elever en egen bärbar dator utöver de gemensamma stationära datorer som skolan tillhandahåller.

I och med den snabba datorutvecklingen och kunskapsutbytet via internet var förhoppningarna på datorns roll inom undervisningen stora [13]. I den så kallade IT-propositionen från 1995 klarlades tydligt att lärare på högskolenivå skulle använda datorer som ett pedagogiskt hjälpmedel i en allt större utsträckning än vad som gjorts tidigare [5]. De stora satsningarna som har gjorts på datorstödd undervisning har dock inte visat någon entydig effekt som generellt tyder på att datorn skulle vara ett effektivt pedagogiskt hjälpmedel [13]. För att öka den pedagogiska funktionen är det viktigt att fokusera på hur nya program ska involveras i undervisningen [5].

1.2 Syfte, mål och metod

Syfte

Det övergripande syftet med det här arbetet är att underlätta studenters förståelse för tvådimensionell konsolidering. Arbetet omfattar utveckling av befintliga rutiner i CALFEM[1] för Python [15] samt förslag på nya rutiner anpassade för den här typen

[1]CALFEM står för Computer Aided Learning of the Finite Element Method och är en samling rutiner som kan användas för beräkning och visualisering med finita elementmetoden. [6]

av beräkningsverktyg.

Mål

1. Utveckla ett verktyg för beräkning och visualisering av tvådimensionell konsolidering. Verktyget skall inkludera ett lättanvänt grafiskt gränssnitt, med syftet att underlätta för studenter att förstå fenomenet.

2. Ta fram övningsmaterial kopplat till programmet för att på ett lättare sätt inkludera det i undervisningssammanhang.

3. Föreslå tillägg av rutiner till CALFEM för Python anpassade för den här typen av utveckling.

4. Utvärdera hur CALFEM för Python fungerar som underliggande plattform för utveckling av denna typ av programvara.

Metod

Arbetet inleds med en litteraturstudie där bakomliggande teori inom geoteknik studeras. Även forskning kring datorn som ett pedagogiskt hjälpmedel undersöks. Därefter utvecklas ett beräkningsverktyg för porvattenövertryckets variation i tid och rum. Programmet utvecklas i programmeringsspråket Python med hjälp av rutiner från de olika biblioteken CALFEM för Python [15], VisVis [4], PyQt [2] och Numpy [1]. Ett lättanvänt grafiskt gränssnitt för visualisering av resultatet utvecklas i gränssnittsbiblioteket Qt [2]. En utveckling av rutiner i CALFEM för Python [15] samt förslag på nya tillägg görs. Vidare utformas kursmaterial kopplat till programmet, där konsolideringen kopplas till resulterande sättningar och utrymme ges för en analys av beräkningsmodellens begränsningar.

1.3 Avgränsningar

För att möjliggöra arbetet inom given tidsram görs vissa avgränsningar. Medan förändringen i porvattentryck beräknas numeriskt används analytiska uttryck för att beskriva spänningsfördelningen utifrån ett givet lastfall. Vidare visar det utvecklade programmet endast hur porvattenövertrycket samt effektivspänningen beror av tiden. Ingen visualisering av sättningarna görs. Dock presenteras i övningsbeskrivningarna hur användaren, utifrån beräknat porvattenövertryck och antagen spänningsfördelning, kan bestämma resulterande töjningar och sättningar. Dessa beräkningar

avser endast sättningar till följd av konsolidering och inte det motsatta fenomenet, där porvattenövertrycket ökar och orsakar svällning.

Ytterligare en avgränsning är att beräkningsmodellen förutsätter ett tvådimensionellt grundvattenflöde. Det begränsar antalet analyserade lastfall. Lastfallet som kommer att analyseras är ett där lasten är jämnt utbredd med en begränsad bredd i planet och oändlig utsträckning i en tredje dimension.

Grundvattenytan antas också sammanfalla med marknivån. Som en följd av detta bortser arbetet från effekten av kapillär stighöjd och dess inverkan på spänningsfördelningen. I kopplingen till sättningsberäkningar studeras enbart deformationen som sker till följd av en förändrad effektivspänning. Med andra ord behandlas ej dynamiska beräkningar, inverkan av släntstabilitet, erosion eller brott i denna koppling.

1.4 Avsedd målgrupp

Målgruppen för det här arbetet är civilingenjörer med grundläggande kunskap inom geoteknik. För att fullt förstå härledningar och beräkningar förutsätts dock kunskaper inom finita elementmetoden. En civilingenjör med en annan bakgrund än den beskrivna, utan kunskap inom varken finita elementmetoden eller geoteknik, ska förstå arbetets teoretiska bakgrund i grova drag. Vidare ska denne, med hjälp av beskrivning i manual och övningskompendium, kunna tillämpa det utvecklade programmet för att lösa enklare konsolideringsproblem.

1.5 Disposition

Det här arbetet bygger på kunskaper inom geoteknik, finita elementmetoden, pedagogik och programmering. Arbetet inleds med tre mer teoribaserade kapitel, kapitel 2 – 4, där grundläggande geotekniska begrepp, finita elementmetoden samt datorn som pedagogiskt verktyg presenteras. Därefter presenteras det utvecklade programmet i kapitel 5 – 6 och därefter en diskussion samt avslutande kommentarer. Upplägget för rapporten är:

Kapitel 2: Grundläggande geotekniska begrepp

Det här kapitlet behandlar kortfattat hur jord är uppbyggd, hur spänningsfördelningen ser ut i en jordprofil, konsolideringsteori samt hur sättningar uppstår. Differentialekvationen för tvådimensionell konsolidering presenteras.

Kapitel 3: FE-metoden för 2D-konsolidering

I kapitel 3 presenteras grunderna inom finita elementmetoden kopplade till tvådimensionell konsolidering. Differentialekvationen som presenterats i kapitel 2 skrivs om till en FE-formulering och en löses med hjälp av en tidsapproximation.

Kapitel 4: Datorn som pedagogiskt verktyg

Det här kapitlet syftar till att lyfta hur datorprogram bör utvecklas för att fungera som pedagogiskt hjälpmedel i en skolmiljö. Vidare syftar det till att undersöka hur programvaror bäst inkluderas i undervisningssammanhang. I kapitlet definieras begreppen kunskap samt användbarhet och olika undervisningsformer presenteras.

Kapitel 5: VisCon - Programbeskrivning

I det här kapitlet beskrivs beräkningsmodellen som implementerats i VisCon. Vidare ges en förkortad version av programmets manual och tillhörande övningsuppgifter presenteras.

Kapitel 6: VisCon - Implementering

Kapitel 6 fokuserar på att beskriva hur VisCon implementerats i programmeringsspråket Python. Det inleds med en presentation av utvecklingsmiljön följt av en beskrivning av huvudprogrammets uppbyggnad. Därefter följer en förklaring av de rutiner som utvecklats för att möjliggöra den här typen beräknings- och visualiseringsverktyg.

Kapitel 7: Diskussion

I kapitel 7 diskuteras programmet utifrån dess utformning som ett pedagogiskt verktyg. Även val vid implementering av rutiner analyseras.

Kapitel 8: Avslutande kommentarer

I det sista kapitlet dras slutsatser utifrån målen som formulerades i det inledande kapitlet. Vidare presenteras förslag på vidare arbete.

Kapitel 2

Grundläggande geotekniska begrepp

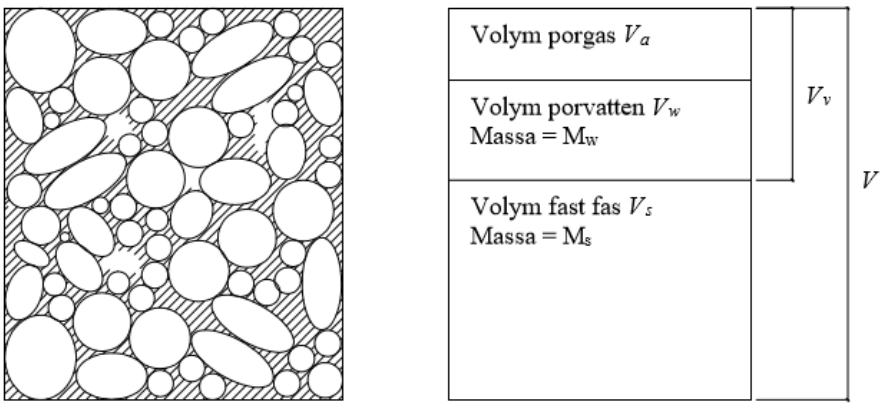
Jord är ett inhomogent material och dess egenskaper beror av flera faktorer, däribland förekomsten av grundvatten. I det här kapitlet behandlas grundläggande teori kring jords materialegenskaper. Inledningsvis presenteras dess uppbyggnad, där innehåll och grundläggande begrepp definieras. Därefter introduceras spänningar följt av deformationsegenskaper. En förklaring av konsolidering ges, där Terzaghis differentialekvation för konsolidering i två dimensioner introduceras. Avslutningsvis kopplas förändringen i porvattenövertryck till resulterande sättningar

2.1 Jordmateriallära

2.1.1 Grundläggande uppbyggnad

Jord består, beroende på vatteninnehåll, av två eller tre faser. I ett helt torrt material utgörs jorden av en fast fas samt en gasfas. Innehåller jorden vatten tillkommer en vätskefas, och om jorden är fullt vattenmättad finns det således ingen gasfas [8]. Den fasta fasen utgörs av mineraler eller lerpartiklar av varierande storlek. I vissa fall innehåller jorden även organiska material. Hålrummen som bildas mellan de fasta partiklarna kallas för porer, och är antingen fyllda med porvatten eller porgas. I figur 2.1a visas hur en jord kan vara uppbyggd, och i figur 2.1b illustreras en separering av de olika faserna med tillhörande beteckningar [7]. Det är nödvändigt att definiera en jords sammansättning för att i ett senare skede kunna analysera dess mekaniska egenskaper. Utifrån definitionerna i figur 2.1b introduceras nedan ett antal grundläggande begrepp som används inom geotekniken.

Förhållandet mellan den totala volymen, V, och volymen porer, V_v, definieras som



Figur 2.1: *a) En typisk sammansättning av det imhogogena materialet jord som består av två till tre faser. b) En separering av den fasta, flytande och gasfasen samt beteckningar för dess volymer och massor.*

jordens porositet, n, enligt [8]:

$$n = \frac{V_v}{V} \tag{2.1}$$

Förhållandet mellan volymen porer, V_v, och volymen av den fasta fasen, V_s, definieras som jordens portal, e, enligt [8]:

$$e = \frac{V_v}{V_s} \tag{2.2}$$

Mellan porositet och portal råder sambandet [11]:

$$e = \frac{n}{1-n} \tag{2.3}$$

Man skiljer ofta på begreppen friktionsjord och kohesionsjord. Den avgörande skillnaden mellan dessa jordklassificeringar är hur bindningen mellan de olika partiklarna ser ut. I en friktionsjord utgörs, som namnet avslöjar, krafterna mellan jordens fasta partiklar av friktion. Friktionsjordar är grovkorniga jordar. I en kohesionsjord överförs kraften mellan partiklarna genom kohesion. Kohesion verkar mellan små partiklar i finkorniga jordar. Leror är vanligt förekommande bland kohesionsjordar [19]. Vidare i detta arbete kommer huvudfokus att ligga på kohesionsjordar.

2.1.2 Porvatten och permeabilitet

Porerna i jorden är antingen fyllda med porgas, porvatten eller både och. Eftersom en jords deformationsegenskaper beror på dess vatteninnehåll är det av intresse att veta huruvida jorden är mättad eller ej. Man brukar göra ett antagande om att jorden är fullt vattenmättad under grundvattennivån. Även över grundvattenytan förekommer vatten i porerna, vilket beror på infiltration, kemiska bindningar samt kapillär stigning. Hur högt vattnet stiger över grundvattenytan beror på porernas storlek samt vatteninnehåll [8]. I finporiga material, så som leror, uppvisas en hög kapillär stighöjd som oftast överstiger 8 m [7]. I det här arbetet bortses dock från denna effekt.

Eftersom jord är ett poröst material tillåts vätska att transporteras genom det. Flödet i en jord beskrivs vanligen med Darcys lag. Darcys lag anger flödeshastigheten, v, som en funktion av permeabiliteten k, och tryckskillnaden, i, i respektive riktning. Tryckskillnaden kan även beskrivas som gradienten av det studerade områdets hydrauliska höjd, h. Analyserar man en tvådimensionell provkropp i xz-planet ges flödeshastigheten i respektive riktning av [8]:

$$v_x = k_x i_x = -k_x \frac{\partial h}{\partial x} \tag{2.4}$$

$$v_z = k_z i_z = -k_z \frac{\partial h}{\partial z} \tag{2.5}$$

Permeabiliteten, k, är ett mått på hur lätt vätska flödar genom ett visst medium och mäts i [m/s]. Permeabiliteten beror på portalet, e, definierat enligt ekvation 2.2. Vidare påverkas permeabiliteten av porernas storlek, exempelvis är de enskilda porerna i lera mindre än i sand vilket resulterar i en längre permeabilitet för lera [8]. Permeabiliteten kan variera med strömningsriktning, vilket främst beror på permeabla skikt i jordlagerföljden som påverkar vilken riktning vattnet kommer att flöda i. Vanligtvis resulterar sådana lager i en ökad vertikal strömning [7]. Temperaturen i jorden påverkar porvätskans viskositet och har således även den viss inverkan på permeabiliteten. I en jord med låg permeabilitet fås låga flödeshastigheter [8].

2.2 Spänningar i jord

2.2.1 Effektivspänningsekvationen

Jordens egentyngd

Jordens egentyngd, γ_s, orsakar vertikala spänningar, σ_z som ökar med djupet, h, enligt:

$$\sigma_z = \gamma_s h \tag{2.6}$$

Notera att marknivån anges som referenspunkt med positiv riktning nedåt, samt att tryckspänning är definierad som positiv [8].

Porvattentryck

Utöver jordens egentyngd påverkar grundvattnet spänningsfördelningen i jorden. Grundvattennivån definieras som den nivå där atmosfärtryck råder, det vill säga där porvattentrycket, u, är noll. Det statiska porvattentrycket, u_s, ökar därifrån linjärt enligt:

$$u_s = \gamma_w h \tag{2.7}$$

I ekvationen anger h djupet, med grundvattennivån som referenspunkt och positiv riktning nedåt, samt γ_w vattnets tyngd [8].

Vattnets egentyngd ges av $\gamma_w = \rho_w g$ där g är gravitationskonstanten $g = 9.81$ m/s^2 och ρ_w är porvattnets densitet. Porvattnets densitet anges vanligen som $\rho_w = 1000$ kg/m^3 men beroende på om det innehåller stora mängder lösta salter kan detta värde vara något högre [21].

Porvattentrycket verkar i samtliga riktningar, men antas vara tillräckligt litet för att inte komprimera jordens fasta partiklar på en materialnivå. Vatten som förekommer ovanför grundvattenytan, på grund av kapillär stigning, resulterar i ett negativt tryck [8]. Som konstaterats bortser dock det här arbetet från denna effekt.

Effektivspänning

Effektivspänningen, σ'_z, är spänningen inom jordens fasta fas, det vill säga belastningen på själva kornskelettet. Den beror av totalspänningen, σ_z, samt porvattentrycket, u, och ges av [8]:

$$\sigma'_z = \sigma_z - u \tag{2.8}$$

Ekvationen ovan refereras till som effektivspänningsekvationen. Då en vattenmättad jord utsätts för en ökad belastning sker initialt en ökning av porvattentrycket. Porvattenövertrycket, u_e, uppstår till följd av att vattnet antas inkompressibelt. Med tiden kommer dock vatten att transporteras för att utjämna porvattenövertrycket. Flödet kan, som konstaterats tidigare, beskrivas med Darcys lag och är en process under vilken effektivspänningen successivt ökar. Då en statisk porvattentrycksfördelning, u_s, återfåtts utgörs den ökade belastningen, $\Delta\sigma_z$, enbart av en ökad effektivspänning $\Delta\sigma'_z$. Det tidsberoende sambandet ges av [20]:

$$\Delta\sigma'_z(t) = \Delta\sigma_z - u_e(t) \tag{2.9}$$

I grundläggningssammanhang skiljer man på begreppen dränerad och odränerad. Avgörande för huruvida en jord är dränerad eller ej är om det finns ett porvattenövertryck, u_e. Om det finns sägs jorden vara odränerad. Har tryckfördelningen emellertid återgått till den statiska porvattentrycksfördelningen, u_s, sägs jorden vara dränerad [8].

2.2.2 Spänningsfördelning vid strimlelast

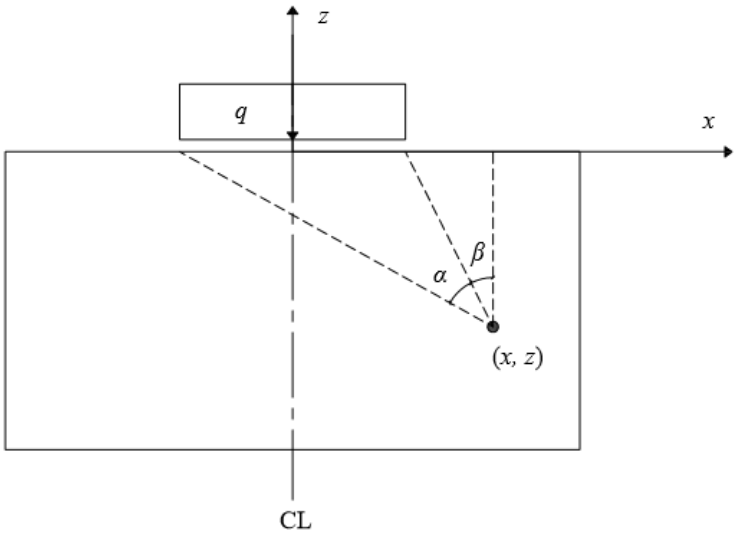
Hur spänningarna i jorden fördelar sig under en applicerad last beror på aktuellt lastfall. Studerar man en jämnt utbredd last, q, med en begränsad bredd i xz-planet men en oändlig utsträckning i y-led, enligt figur 2.2, fås normal- och skjuvspänningar enligt:

$$\sigma_z = \frac{q}{\pi}(\alpha + \sin\alpha\cos(\alpha + 2\beta)) \tag{2.10}$$

$$\sigma_x = \frac{q}{\pi}(\alpha - \sin\alpha\cos(\alpha + 2\beta)) \tag{2.11}$$

$$\tau_{xz} = \frac{q}{\pi}(\sin\alpha\cos(\alpha + 2\beta)) \tag{2.12}$$

Uttrycken är baserade på elasticitetsteori och vinklarna α och β mäts i [rad] och bestäms utifrån en punkts koordinater, (x, z), med definitioner enligt figur 2.2. De analytiska uttrycken i ekvation 2.10 – 2.12 kan alltså användas för att beräkna spänningarna i valfri punkt för en last q [Pa] [8].



Figur 2.2: *Definition av vinklarna α och β som används i ekvation 2.10 – 2.12 vid bestämning av spänningsfördelningen i en jordprofil belastad med en strimlelast av storleken q Pa.*

2.3 Primär konsolidering

2.3.1 Definition

Då en odränerad jord med låg permeabilitet genomgår en process där porerna dräneras på porvätska, utan att ersättas med porgas, genomgås en process som kallas konsolidering. Det är en volymminskning under en hydraulisk fördröjning. För leror, där permeabiliteten är låg och vatteninnehållet högt, kan den här processen ta lång tid då det tar tid för porvattenövertrycket att utjämnas på grund av ett långsamt flöde [20]. Är porerna fyllda med luft och jorden uppbyggd av grovkorniga partiklar går denna process betydligt snabbare [8].

2.3.2 Terzaghis differentialekvation för konsolidering i 2D

För att beskriva konsolideringen behövs en ekvation som anger hur det ökade porvattentrycket u_e, orsakat av en spänningsökning, $\Delta\sigma_z$, beror av tid och rum, $u_e = f(x, z, t)$. Terzaghi har utvecklat en teori som beskriver just detta och som bygger på följande antaganden [20]:

- Jorden är fullt vattenmättad, det vill säga odränerad och samtliga porer är fyllda med porvätska.

- Flödet i jorden går att beskriva med hjälp av Darcys lag.
- Porvattnet samt jordens fasta fas antas vara inkompressibla på materialnivå.
- Konsolideringens tidsberoende beror endast på fördröjningen av vätskans flöde beskriven av Darcys lag.

Utifrån bland annat jordens egenskaper beskrivna i ekvation 2.1 – 2.3 samt Darcys lag enligt ekvation 2.4 – 2.5 kan en differentialekvation som beskriver konsolideringsförloppet härledas. Enligt Terzaghis teori ges denna av [20]:

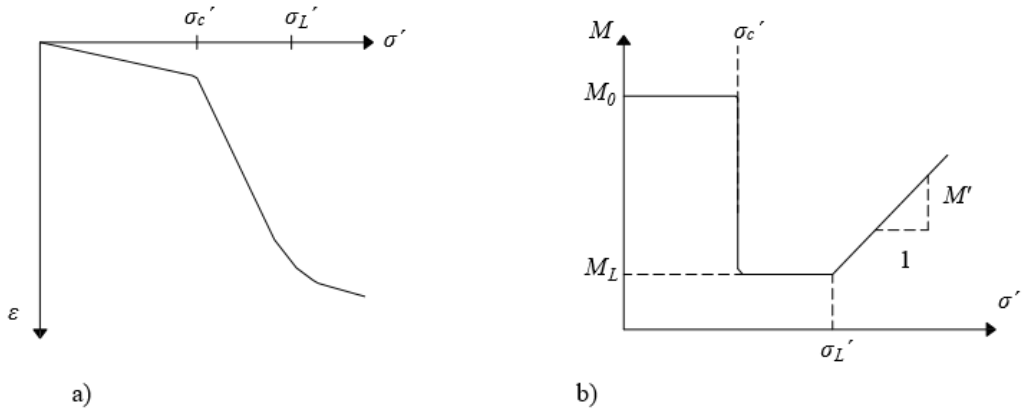
$$\frac{1}{M}\frac{\partial u_e}{\partial t} = \frac{\partial}{\partial x}\left(\frac{k_x}{\gamma_w}\frac{\partial u_e}{\partial x}\right) + \frac{\partial}{\partial z}\left(\frac{k_z}{\gamma_w}\frac{\partial u_e}{\partial z}\right) \tag{2.13}$$

där M anger jordens kompressionsmodul i [Pa] och γ_w porvattnets tunghet i [N/m^3].

2.4 Sättningar till följd av primär konsolidering

2.4.1 Deformationsegenskaper

Vid en ökad belastning, $\Delta\sigma_z$, sker en kompression, δ_z, av jorden, vilket innebär att kornpartiklarna förs närmre varandra och volymen porer, V_v, minskar. Hur mycket en jord deformeras beror på dess sammansättning samt vilka laster den utsatts för i ett tidigare skede [8].



Figur 2.3: *Spännings-deformationsdiagram samt kompressionsmodul beroende på spänningen i en jord.*

Man brukar skilja på om jorden är normalkonsoliderad eller överkonsoliderad. Innebörden av att en jord är överkonsoliderad är att den vid något tillfälle utsatts

för en högre belastning än den aktuella. Är belastningen densamma som förkonsolideringstrycket, det vill säga den spänning den mest belastats med, sägs den vara normalkonsoliderad [7]. I figur 2.3a visas spännings-deformationssamband över den vertikala effektivspänningen, σ'_z, och töjningen ε_z. Markerat är förkonsolideringstrycket, σ'_c, samt spänningen vid vilken deformationssambandet upphör att vara linjärt, σ'_L. Detta samband kan fås utifrån tester så som CRS-försök[1]. Utifrån lutningen på spännings-deformationssambandet i figur 2.3a kan kompressionsmodulen M bestämmas enligt [18]:

$$M = \frac{\Delta\sigma'_z}{\Delta\varepsilon_z} \tag{2.14}$$

Kompressionsmodulens, M, variation med effektivspänningen, σ'_z, visas i figur 2.3b. Från figuren går det att konstatera att följande kompressionsmoduler tillämpas inom angivna spänningsintervall:

$$M = M_0 \text{ för } \sigma'_z \leq \sigma'_c \tag{2.15}$$

$$M = M_L \text{ för } \sigma'_c < \sigma'_z \leq \sigma'_L \tag{2.16}$$

$$M = M_L + (\sigma'_z - \sigma'_L)M' \text{ för } \sigma'_z > \sigma'_L \tag{2.17}$$

Ibland beskrivs jordens deformationsegenskaper istället i form av elasticitetsmodulen, E. Har man kompressionsmodulen, M, samt tvärkontraktionstalet, ν, kan man bestämma elasticitetsmodulen utifrån [7]:

$$E = \frac{(1+\nu)(1-2\nu)}{1-\nu}M \tag{2.18}$$

2.4.2 Töjningar och sättningar

Utifrån deformationsegenskaper kan jordens vertikala töjning, ε_z, bestämmas vid en viss belastning. Då jorden är överkonsoliderad, det vill säga den vertikala effektivspänningen efter pålastningen, $\sigma'_{z,1} = \sigma'_{z,0} + \Delta\sigma'_z$, är mindre eller lika med förkonsolideringstrycket, $\sigma'_{z,1} \leq \sigma'_c$, erhålls följande uttryck för töjningen ε_z:

$$\varepsilon_z = \frac{\Delta\sigma'_z}{M_0} \tag{2.19}$$

Då den vertikala spänningen efter pålastningen, $\sigma'_{z,1}$, ligger mellan förkonsolideringstrycket och den spänning för vilket deformationssambandet enligt ödometerförsök upphör vara linjärt, $\sigma'_c < \sigma'_{z,1} \leq \sigma'_L$, ges den vertikala töjningen, ε_z, av:

[1]CRS innebär att ett spännings-deformationsdiagram tas fram genom ett deformationsstyrt ödometerförsök [18].

$$\varepsilon_z = \frac{\sigma'_c - \sigma'_{z,0}}{M_0} + \frac{\sigma'_{z,1} - \sigma'_c}{M_L} \tag{2.20}$$

Då den vertikala spänningen efter pålastningen, $\sigma'_{z,1}$, överstiger denna spänningsnivå, det vill säga $\sigma'_{z,1} > \sigma'_L$ ges den vertikala töjningen ε_z av:

$$\varepsilon_z = \frac{\sigma'_c - \sigma'_{z,0}}{M_0} + \frac{\sigma'_L - \sigma'_c}{M_L} + \frac{1}{M'} \ln\left(1 + (\sigma'_{z,1} - \sigma'_L)\frac{M'}{M_L}\right) \tag{2.21}$$

Sättningarna som uppstår till följd av töjningarna i ekvation 2.19 - 2.21 fås genom integration med avseende på jordens djup enligt:

$$\delta_z = \int_0^h \varepsilon_z dz \tag{2.22}$$

Kapitel 3

FE-metoden för 2D-konsolidering

Finita elementmetoden är en numerisk metod som tillämpas för att lösa godtyckliga differentialekvationer. Tillvägagångssättet bygger på att man utifrån ett fysikaliskt fenomen formulerar en differentialekvation. För att lösa problemet görs en indelning i finita element och genom en approximativ metod bestäms variationen inom de olika elementen. Den generella lösningsmetoden för ett finita elementproblem kan delas in i följande steg [14]:

- Definition av modell
- Elementindelning och formfunktioner
- Lösning av ekvationssystem

I det här kapitlet beskrivs finita elementmetoden för fallet tvådimensionell konsolidering utifrån det här upplägget. För mer allmän teori kring finita elementmetoden hänvisas till *Introduction to the Finite element method* av Ottosen och Petersson [14].

3.1 Definition av modell

3.1.1 Differentialekvation

Konsolideringsförloppet i två dimensioner kan utifrån ekvation 2.13 beskrivas enligt [20]:

$$\frac{\partial}{\partial x}\left(k_x \frac{\partial u_e}{\partial x}\right) + \frac{\partial}{\partial z}\left(k_z \frac{\partial u_e}{\partial z}\right) - \frac{\gamma_w}{M}\frac{\partial u_e}{\partial t} = 0 \tag{3.1}$$

Det är en partiell differentialekvation av första ordningen, där vattnets tunghet, γ_w, samt jordens kompressionsmodul, M, antagits konstanta över elementen. Det som skiljer differentialekvationen från en stationär differentialekvation är den tidsberoende termen [20].

3.1.2 Svag formulering

Första steget för att erhålla en FE-formulering utifrån en differentialekvation är att skriva om differentialekvationen på den så kallade svaga formen. För detta skrivs ekvation 3.1 först på matrisform:

$$\begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial z} \end{bmatrix} \begin{bmatrix} k_x & 0 \\ 0 & k_z \end{bmatrix} \begin{bmatrix} \frac{\partial u_e}{\partial x} \\ \frac{\partial u_e}{\partial z} \end{bmatrix} - \frac{\gamma_w}{M} \frac{\partial u_e}{\partial t} = 0 \tag{3.2}$$

Med:

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial z} \end{bmatrix} \tag{3.3}$$

och:

$$\mathbf{D} = \begin{bmatrix} k_x & 0 \\ 0 & k_z \end{bmatrix} \tag{3.4}$$

kan ekvation 3.2 skrivas som:

$$\nabla^T \mathbf{D} \nabla u_e - \frac{\gamma_w}{M} \frac{\partial u_e}{\partial t} = 0 \tag{3.5}$$

Ekvation 3.5 multipliceras med en godtycklig och tidsoberoende viktfunktion, $v_m(x, z)$, samt integreras över området A [14]:

$$\int_A v_m \nabla^T \mathbf{D} \nabla u_e dA - \int_A v_m \frac{\gamma_w}{M} \frac{\partial u_e}{\partial t} dA = 0 \tag{3.6}$$

Green Gauss teorem är definierat enligt [14]:

$$\int_A v_m \nabla^T \mathbf{D} \nabla u_e dA = \oint_L v_m q_n dL - \int_A (\nabla v_m)^T \mathbf{q} dA \tag{3.7}$$

där q_n är flödet vinkelrätt mot ytan, det vill säga fluxvektorn projicerad på en normal, $\mathbf{n}$, till denna. Genom en omformulering via Green Gauss teorem av första termen i ekvation 3.6 samt en indelning av randvillkoren längs L_h och L_g, fås den svaga formuleringen enligt:

$$\int_A (\nabla v_m)^T \mathbf{D} \nabla u_e dA + \int_A v_m \frac{\gamma_w}{M} \frac{\partial u_e}{\partial t} dA = \oint_{L_h} v_m h dL + \oint_{L_g} v_m q_n dL \tag{3.8}$$

där $u_e = g$ är känt längs randen L_g och $q_n = h$ är känt längs randen L_h [14] .

3.1.3 FE-formulering

Porvattenövertrycket, $u_e(x, z, t)$, inom ett element kan approximeras med hjälp av den tidsoberoende formfunktionsmatrisen, $\mathbf{N}(x, z)$, samt nodernas tidsberoende porvattenövertryck, $\mathbf{a}^e(t)$, enligt:

$$u_e(x, z, t) = \mathbf{N}(x, z)\mathbf{a}^e(t) \tag{3.9}$$

Under förutsättningen att $\nabla \mathbf{N} = \mathbf{B}$ fås:

$$\nabla u_e = \nabla \mathbf{N} \mathbf{a}^e = \mathbf{B}\mathbf{a}^e \tag{3.10}$$

I enlighet med Galerkins metod väljs viktfunktionerna $v_m(x, z) = v_m^T(x, z)$ enligt:

$$v_m = \mathbf{N}\mathbf{c} = (\mathbf{N}\mathbf{c})^T = \mathbf{c}^T \mathbf{N}^T \tag{3.11}$$

där $\mathbf{N}(x, z)$ ges av formfunktionerna och $\mathbf{c}$ är en godtycklig matris. Vidare erhålls då:

$$(\nabla v_m)^T = (\mathbf{B}\mathbf{c})^T = \mathbf{c}^T \mathbf{B}^T \tag{3.12}$$

Porvattenövertryckets derivata kan uttryckas enligt [16]:

$$\frac{\partial u_e}{\partial t} = \mathbf{N} \frac{\partial \mathbf{a}}{\partial t} = \mathbf{N}\dot{\mathbf{a}} \tag{3.13}$$

Substitution av ekvation 3.9 – 3.13 i ekvation 3.8 ger:

$$\mathbf{c}^T \left(\int_A \mathbf{B}^T \mathbf{D} \mathbf{B} dA \mathbf{a} + \int_A \mathbf{N}^T \frac{\gamma_w}{M} \mathbf{N} dA \dot{\mathbf{a}} - \oint_{L_h} \mathbf{N}^T h dL - \oint_{L_g} \mathbf{N}^T q_n dL \right) = 0 \tag{3.14}$$

med villkoret $u_e = g$ på ytan L_g.

Eftersom $\mathbf{c}^T$ är en godtycklig matris kan denna strykas och ekvation 3.14 skrivas på formen [14]:

$$\mathbf{Ka} + \mathbf{C\dot{a}} = \mathbf{f} \tag{3.15}$$

där

$$\mathbf{K} = \int_A \mathbf{B}^T \mathbf{D} \mathbf{B} \; dA \tag{3.16}$$

$$\mathbf{C} = \int_A \mathbf{N}^T \frac{\gamma_w}{M} \mathbf{N} \; dA \tag{3.17}$$

$$\mathbf{f} = \oint_{L_h} \mathbf{N}^T h dL + \oint_{L_g} \mathbf{N}^T q_n dL \tag{3.18}$$

3.2 Elementindelning och formfunktioner

Eftersom lastfallet som kommer att undersökas i det här arbetet är begränsat i xz-planet men med en stor utsträckning i y-led kan fördelningen anses lika för varje tvärsnitt i xz-planet längs med y-axeln. Med andra ord antas porvattnet endast flöda i två dimensioner [17].

Området som finita elementmetoden ska tillämpas för delas in i mindre områden, så kallade finita element. I tvådimensionella problem väljs vanligen triangulära eller kvadratiska element, avgränsade av definierade nodpunkter. I det här arbetet görs approximationen att porvattenövertrycket kommer att variera linjärt inom tvådimensionella trenodselement, som är avgränsade med raka linjer [17].

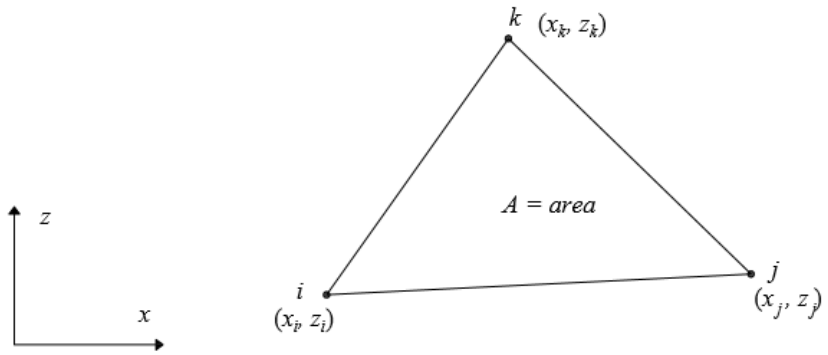
En approximation av tryckfördelningen inom respektive element görs. Fördelningen beskrivs med formfunktioner som utgörs av enkla polynom som beskriver fördelningen över elementet mellan nodernas värde. Porvattenövertrycket för ett element kan beskrivas som en viktning av formfunktionerna multiplicerade med nodvärderna enligt ekvation 3.9. Formfunktionerna, $\mathbf{N^e}$, bygger på antagen fördelning inom elementen och för ett trenodselement med en linjär fördelning och koordinater enligt figur 3.1 fås formfunktionsmatrisen enligt [14]:

$$\mathbf{N}^e = \begin{bmatrix} N_i^e & N_j^e & N_k^e \end{bmatrix} \tag{3.19}$$

$$N_i^e = \frac{1}{2A} \left[x_j z_k - x_k z_j + (z_j - z_k)x + (x_k - x_j)z \right] \tag{3.20}$$

$$N_j^e = \frac{1}{2A} \left[x_k z_i - x_i z_k + (z_k - z_i)x + (x_i - x_k)z \right] \tag{3.21}$$

$$N_k^e = \frac{1}{2A} \left[x_i z_j - x_j z_i + (z_i - z_j)x + (x_j - x_i)z \right] \tag{3.22}$$



Figur 3.1: *Definition av koordinater för ett triangulärt element med tre noder.*

Vid val av antal element, det vill säga hur fin elementindelning som görs, tas hänsyn till om det är ett område där tryckfördelningen ändras mycket och således kräver en finare indelning. Antalet element avgör noggrannheten av approximationen. Dock innebär en finare elementindelning en snabbt växande beräkningstid på grund av stora matriser. Ett sätt att lösa problemet är att ha olika fin elementindelning i olika delar av geometrin [17].

Utifrån approximerade formfunktioner inom respektive element kan elementstyvhetsmatris $\mathbf{K}^e$, elementkapacitansmatris $\mathbf{C}^e$ och elementlastvektor $\mathbf{f}^e$ bestämmas enligt ekvation 3.16 - 3.18. För att erhålla de globala matriserna som beskriver det sammansatta systemet genomförs en så kallad assemblering. Det är en process där elementmatriserna slås ihop till en global matris som beskriver hela systemet. Den utgår från systemets så kallade topologimatris som anger kopplingen mellan element och frihetsgrader. Då elementindelningen görs är det viktigt att försöka numrera noderna på ett sätt som genererar en låg bandbredd. Det uppfylls genom att ha en så liten skillnad som möjligt mellan nodnumren inom varje element [14].

3.3 Lösning av ekvationssystem

För att beskriva tvådimensionell konsolidering har ekvation 3.15 härletts. För att lösa denna måste en approximation av porvattenövertryckets tidsvariation göras. Antas en linjär variation med tiden ges porvattenövertrycket enligt [16]:

$$\mathbf{a}(t) = \frac{\mathbf{a}_{i+1} - \mathbf{a}_i}{\Delta t}(t - t_i) + \mathbf{a}_i \tag{3.23}$$

där Δt är längden på definierat tidsinkrement. Indelningen i tidsinkrement blir avgörande för approximationens korrekthet. Baserat på den här approximationen kan

följande samband härledas för den nya porvattenövertrycksfördelningen, $\mathbf{a}_{i+1}$, efter tidssteget Δt [16]:

$$\hat{\mathbf{K}}\mathbf{a}_{i+1} = \hat{\mathbf{f}} \tag{3.24}$$

där

$$\hat{\mathbf{K}} = (\mathbf{C} + \Delta t \Theta \mathbf{K}) \tag{3.25}$$

$$\hat{\mathbf{f}} = \left[(\mathbf{C} - \Delta t \mathbf{K}(1-\Theta))\,\mathbf{a}_i + \Delta t \bar{\mathbf{f}}\right] \tag{3.26}$$

$$\bar{\mathbf{f}} = \mathbf{f}_i + \Theta(\mathbf{f}_{i+1} - \mathbf{f}_i) \tag{3.27}$$

Parametern Θ kallas för viktparametern. Val av $\Theta = 0$ refereras till som Forward Euler-metoden och är en implicit metod som ger bra resultat om tillräckligt små tidssteg väljs. Crank Nicholsons metod innebär ett val av $\Theta = 0.5$ och är en explicit metod. Backward Euler innebär ett val av $\Theta = 1$ är även det en explicit metod, där styvhetsmatrisens invers beräknas i varje steg [16].

Kapitel 4

Datorn som pedagogiskt verktyg

I föregående kapitel har grundläggande geotekniska begrepp samt finita elementmetoden för tvådimensionell konsolidering presenterats. I utvecklingen av en programvara är det dock lika viktigt att studera hur datorn faktiskt kan och bör användas som ett pedagogiskt verktyg för att få en grund att stå på gällande dess utformning och tillämpning. I detta kapitel presenteras forskning kring datorn som pedagogiskt verktyg, där kapitlet inleds med en diskussion kring olika typer av kunskap och hur dessa kan införskaffas med datorn som hjälpmedel. Vidare förs en diskussion kring interaktionen mellan dator och människa samt hur datorprogram bör involveras i undervisningssammanhang. Sonja Farkell-Bååthes avhandling *Datorn som pedagogiskt hjälpmedel* [10] har använts som nyckelreferens i detta kapitel då hon på ett överskådligt sätt sammanfattar både historia och forskningsresultat rörande datorn som pedagogiskt hjälpmedel. Önskas vidare läsning kring detta hänvisas därför till hennes avhandling.

4.1 Definition av kunskap

Kunskap som begrepp kan delas in i fyra olika huvudgrupper. Man skiljer på kunskap i form av fakta, förståelse, färdighet och förtrogenhet. Nedan följer en kort förklaring av respektive begrepp [10].

Fakta

Med faktakunskap avses kunskap i form av information och regler som kan memoreras. Det kan till exempel omfatta definitioner. I och med datorns introducerande i undervisningen har möjligheten till inhämtning av faktakunskaper expanderat. In-

ternet är en viktig källa där faktakunskaper sprids. Dock innebär denna nya källa av information även ett ökat behov av källkritik [10].

Förståelse

Förståelse är en kunskap som bygger på att man kan koppla samman olika typer av faktakunskaper och utifrån dessa identifiera strukturer och sammanhang. Förståelsekunskaperna kan förbättras med hjälp av datorn som pedagogiskt hjälpmedel tack vare grafiska framställningar av händelser och koncept. Med hjälp av datorprogram inom undervisningen kan man modellera komplexa fenomen, och utifrån detta analysera resultatet och dra slutsatser [10].

Färdighet

En färdighet är en praktisk kunskap som till exempel att utföra tankeoperationer. Det kan beskrivas som när man vet hur något ska utföras och genomför det. Pedagogiska datorprogram har visat sig vara framgångsrika i sammanhang av färdighetsinlärning om läraren givits tillräckligt med tid och resurser för att inkludera programvaran i undervisningen på ett givande sätt [10].

Förtrogenhet

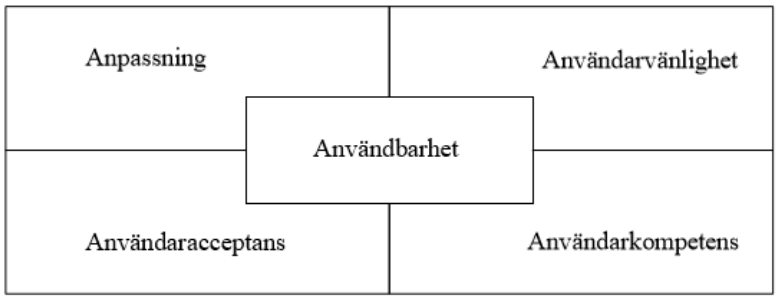
Förtrogenhet är kunskap som sitter i ryggmärgen. I användning av datorprogram kan det förklaras genom att man baserat på tidigare kunskaper om andra datorprogram kan ta reda på hur ett nytt program fungerar. Detta då man redan bekantat sig med en viss struktur. Det innebär i praktiken att man många gånger kan förstå och använda ett nytt program tack vare att man känner igen funktioner [10].

4.2 Samspel mellan dator och människa

Anledningen till att man vill använda datorer i undervisningssammanhang är att de möjliggör en ökad produktivitet; man kan genomföra beräkningar snabbare, enklare och i många fall med ett reducerat antal felkällor då många räknefel utesluts [10]. Vidare öppnar datoranvändandet upp för nya typer av inlärning. En stor styrka ligger i visualisering och olika typer av grafisk framställning. Simuleringar hjälper till att öka förståelsen för komplexa fenomen [5].

För att användandet av datorer ska fungera friktionsfritt krävs dock en fungerande

interaktion mellan människa och dator. I relationen mellan dessa parter är människans roll att vara flexibel och kunna ta in och analysera resultat från datorn, där datorn i sin tur utgör ett redskap uppbyggt utifrån en samling regler och villkor. Ett programs användbarhet är avgörande för denna interaktion. Användbarhet kan beskrivas utifrån fyra nyckelkategorier enligt figur 4.1; anpassning, användarvänlighet, användaracceptans och användarkompetens [10]. Nedan presenteras dessa mer ingående.



Figur 4.1: *För att en pedagogisk programvara ska ha en hög användbarhet krävs en hög anpassning, användarvänlighet, användaracceptans och användarkompetens.*

Anpassning

Anpassning rör hur väl programmet är utformat efter aktuellt användningsområde och avsedda användare. Ett program speciellt framtaget för en viss funktion och målgrupp har en bättre anpassning än ett program som är mer generellt. Överflödiga funktioner riskerar exempelvis att missleda användarens uppmärksamhet. Det har bevisats att begränsade program, det vill säga demoversioner av program där endast vissa funktioner finns tillgängliga, bidrar till en snabbare inlärning. Istället för att användaren drunknar i funktioner är det mer troligt att denna lyckas uppehålla ett intresse och bevara en hög självkänsla, vilket är viktigt för programmets pedagogiska funktion [10].

Användarvänlighet

För att mäta ett programs användbarhet definieras vidare begreppet användarvänlighet. Det bygger på att programmet ska ha en hög tillgänglighet och exempelvis vara lätt att ladda ned och använda på flera olika datorer. Vidare bygger användarvänlighet på att programmet ska byggas utifrån en lagom mängd information som gör

att användaren kan följa med och förstå vad som händer i varje steg av programmet. Då användaren får för mycket information att processa i varje steg blir programmet svårare att förstå. Exempelvis är det lätt för användaren att förlora kopplingen mellan vad som händer med fördröjda funktioner, till exempel där en angiven parameter visar utslag först i ett senare skede. Direkt återkoppling mellan funktioner och programreaktioner har visat sig vara det mest effektiva för inlärningen. För en hög användarvänlighet är det slutligen viktigt att erbjuda goda hjälpfunktioner som användaren kan ta till då denne stöter på problem [10].

Användaracceptans

Användaracceptans handlar kort sagt om att användaren ska acceptera programmet och se det som en givande inslag i utbildningen. Det krävs ett intresse och en tro på att programmet kommer att resultera i en ökad kunskap för att användaren ska lägga tid och energi på inlärningsprocessen. Utan användaracceptans kommer användaren inte att kunna inhämta kunskap ur ett program på grund av bristande motivation [10].

Användarkompetens

Ytterligare en förutsättning för att ett program ska ha en hög användbarhet är att användaren har tillräckligt med kunskaper för att använda programmet. Begreppet användarkompetens går således hand i hand med begreppet anpassning; programmet måste vara utformat efter användarna samtidigt som användarna måste ha tillräckliga kunskaper för programmet. Det har visat sig att en snabbare inlärningsprocess erhålls då man tar tillvara användarens tidigare kunskaper. Exempelvis lär sig användaren ett nytt program betydligt snabbare då denne känner igen funktioner och utformning från andra programvaror [10].

4.3 Tillämpning i undervisningssammanhang

För att faktiskt dra nyttja av pedagogiska programvaror är det viktigt att de inkluderas i undervisningen på rätt sätt. Forskning har nämligen visat att det finns goda förutsättningar för många pedagogiska hjälpmedel, men att det som haltar är att lärarna inte ges tillräcklig tid för att inkludera verktygen i undervisningen på ett givande sätt [10].

Man har länge vetat att den bästa inlärningsprocessen fås då eleverna lär sig via ett problembaserat lärande där de själva får formulera hypoteser och frågeställningar.

Vid en sådan inlärning är det dock otroligt viktigt att eleverna ges goda möjligheter att få svar på sina frågor. Hur de får svar på sina frågor rörande användning och utformning varierar beroende på hur programmet inkluderas i undervisningen. Nedan nämns tre alternativa metoder; föreläsningar, utforskning och observation [10].

Ett sätt att inkludera en programvara i utbildningen är att introducera det på en föreläsning. Det är ett bra sätt då eleverna befinner sig i ett forum där de ges möjlighet att ställa frågor till läraren. Forskning visar att inlärningen sker bäst då undervisningen hålls i små grupper. Det gynnar även inlärningsprocessen då grupperna är homogena och eleverna inom gruppen är på en liknande kunskapsnivå [10].

Ett andra sätt att inkludera en programvara är lärandeformen utforskning. Det innebär att eleverna lär sig på egen hand genom att testa sig fram och självständigt experimentera med programmet. Eleverna lär sig genom att se vilka händelser och resultat olika funktioner i programmet genererar. Lärandeformen utforskning medför stora nackdelar då den tillämpas för alltför omfattande och komplexa program, där det finns väldigt många funktioner och varierade tillämpningar. Användaren riskerar att drunkna i programmet och mötas av en känsla av hopplöshet då inlärningen blir övermäktig. Ska inlärningsprocessen ske via utforskning är det således viktigt att programmet har en hög användarvänlighet och en god anpassning. För att möjliggöra lärandeformen utforskning bör en manual tillhandahållas, där användaren introduceras till programmet men också, vilket är det allra viktigaste, kan hitta lösningen till eventuella felsituationer [10].

Ett tredje sätt att inkludera programvaran är att grunda inlärningen på observation. Det är en lärandeform där eleverna lär sig genom att observera någon annan använda programmet ifråga. För att detta ska fungera optimalt förutsätts funktioner med en direkt koppling, då det annars blir svårt för eleverna att förstå olika kommandon om ingen momentan respons ges [10].

Avslutningsvis är elevernas motivation vital för en god inlärning. Studenternas motivation kan styras genom olika typer av belöningar, både inre och yttre. Även om yttre belöningar har visat en viss effektekt är det de inre belöningarna som är allra viktigast för en hög motivation. En inre belöning kan vara att studenten känner att den pedagogiska programvaran verkligen hjälpt den egna förståelsen inom aktuellt område [10].

Kapitel 5

VisCon - Programbeskrivning

VisCon är ett program utvecklat inom det här arbetet för att beräkna porvattenövertryckets variation i tid och rum. Utifrån beräknade nodvärden visualiseras resultatet i två dimensioner med hjälp av en färgskala. Vidare visualiseras resultatet med endimensionella grafer. De endimensionella graferna visas för valfri definierad sektion av modellen, antingen ett vertikalt eller horisontellt snitt. Det här kapitlet klargör antaganden och teori bakom den implementerade beräkningsmodellen. Därefter följer en kort version av programmets manual, som visar användargränssnittets design och beskriver hur programmet kan användas. I slutet av kapitlet beskrivs hur övningsuppgifter kopplade till VisCon utformats. Dessa samt ett räkneexempel återfinns i bilaga B.

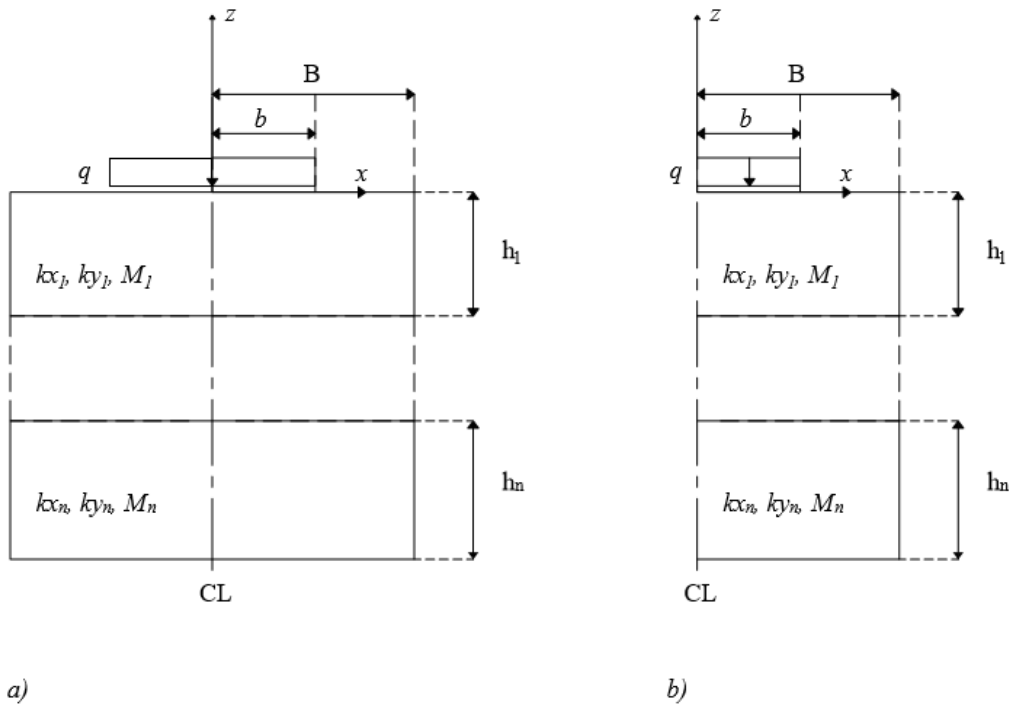
5.1 Beräkningsmodell

5.1.1 Geometri, geotekniska egenskaper och lastfall

Geometrin för beräkningsmodellen som VisCon har utvecklats för visas i figur 5.1a. Då modellen antas symmetrisk analyserar programmet bara en symmetrihalva enligt figur 5.1b. I figuren visas variabler för geometri, elementegenskaper och last. Den beskrivna lasten antas jämnt utbredd.

Programmet tillåter en analys av en jordprofil med ett obegränsat antal jordlager, dock antas djup h, permeabilitet k och kompressionsmodul M konstanta inom respektive lager. Permeabiliteten kan tilldelas ett värde för flöde i x-led, k_x, och ett annat i z-led, k_z.

Ovanstående parametrar går att variera och anges då aktuell modell skapas i använ-



Figur 5.1: *a) Beräkningsmodellen som analyseras med hjälp av visualiseringsverktyget VisCon. b) Då modellen är symmetrisk analyseras endast ena symmetrihalvan av jordprofilen.*

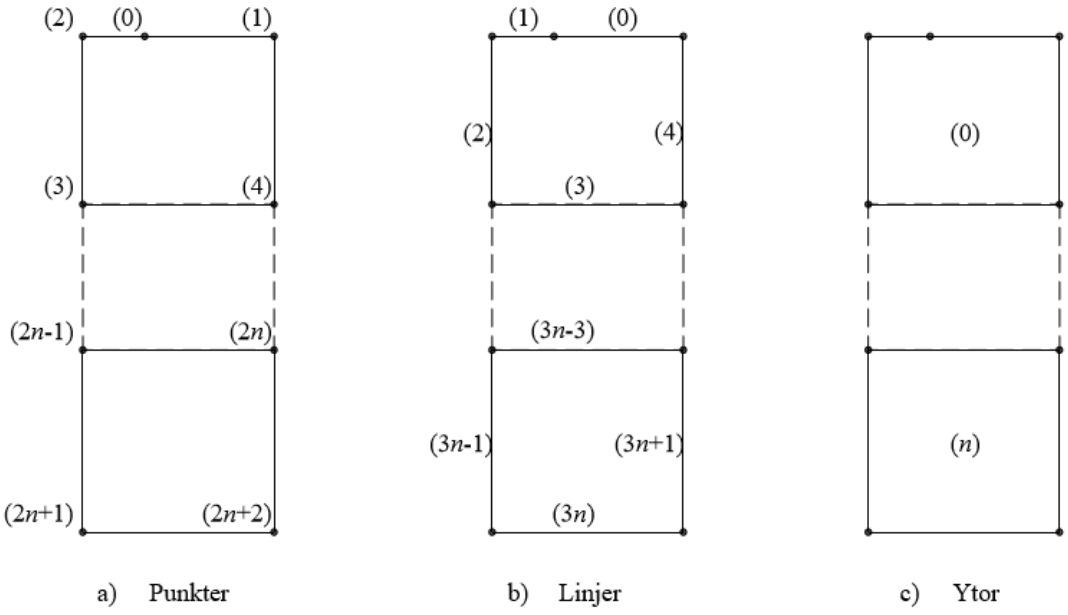
dargränssnittet. Då programmet initieras är dock en fördefinierad modell angiven med värden enligt tabell 5.1. Detta för att programmet ska kunna köras direkt utan utfärdande av felmeddelanden.

Tabell 5.1: *Variabler för modellens lastfall samt geometriska och geotekniska egenskaper. I tabellen presenteras vidare värden för den fördefinierade modellen.*

Variabel	**Fördefinierat värde**	**Enhet**
Lastens bredd, b	2	[m]
Modellens bredd, B	8	[m]
Lastens storlek, q	$30 \cdot 10^3$	[Pa]
Antal jordlager	1	[st]
Djup jordlager, h	15	[m]
Permeabilitet [k_x, k_z]	[$1.0 \cdot 10^{-9}$, $1.0 \cdot 10^{-9}$]	[m/s]
Kompressionsmodul M	$900 \cdot 10^3$	[Pa]
Tunghet porvatten γ_w	$10 \cdot 10^3$	[N/m^3]

5.1.2 Elementindelning

Utifrån definierad geometri, jordegenskaper och lastfall skapas en finita elementmodell. Det görs genom en nätgenerering av jordprofilen. Nätgenereringen görs med rutiner i CALFEM för Python, se mer i avsnitt 6.1. Då modellen i VisCon utgår från en geometri som beror av antalet jordlager, n, skapas en geometri-instans som en funktion av denna variabel. Figur 5.2 visar hur modellens punkter, linjer och ytor numreras beroende på antalet lager. Numreringen av punkter, linjer och ytor tilldelas nummer med start på 0.



Figur 5.2: *För att möjliggöra elementindelningen skapas en geometri-instans där modellen definieras utifrån punkter, linjer och ytor.*

Till CALFEM för Pythons nätgenereringsrutin måste elementtyp, maximal storlek och antal frihetsgrader per nod anges. I VisCon tillåts användaren variera elementstorleken mellan värdena 0.2 och 1.0. Elementstorlekens värde påverkar hur elementindelningen görs. Det angivna värdet multiplicerat med modellens bredd och djup avgör antalet element i respektive riktning. I den fördefinierade modellen, som användargränssnittet uppdateras med vid programmets initiering, är elementstorleken tilldelad värdet 0.5. Modellens elementtyp och antal frihetsgrader per nod går ej att ändra via gränssnittet.

5.1.3 Randvillkor

För att definiera en fullständig finita elementmodell måste randvillkor anges. Följande randvillkor är implementerade i VisCon:

- Flödet vinkelrätt mot centrumlinjen antas vara noll. Detta då modellen är symmetrisk och porvattnet antas flöda symmetriskt utifrån symmetrilinjen.
- Vid tiden $t > 0$ antas porvattenövertrycket $u_e = 0$ längs modellens övre rand. Det innebär att porvattnet kan flöda fritt vinkelrätt mot denna yta.
- Modellens grund kan tilldelas två olika randvillkor beroende på om jordlagret antas vila på en yta som anses permeabel eller ej. Antas grunden permeabel är porvattenövertrycket $u_e = 0$. Det innebär med andra ord att porvattnet kan flöda fritt vinkelrätt mot denna yta. Ett exempel på detta är om det modellerade jordlagret vilar på en grund av grovkornig sand. Om grunden ej modelleras som permeabel antas flödet vinkelrätt mot denna yta vara noll. Det kan till exempel vara ett jordlager som vilar ovanpå en solid berggrund.
- Flödet vinkelrätt mot randen på motsatt sida från centrumlinjen antas vara noll. Det kan antingen symbolisera en impermeabel yta eller att bredden på jordprofilen är så pass stor att flödet vid den här randen är väldigt litet, $q \approx 0$. För att detta antagande ska vara riktig måste en tillräcklig bredd på jordprofilen, B, väljas.

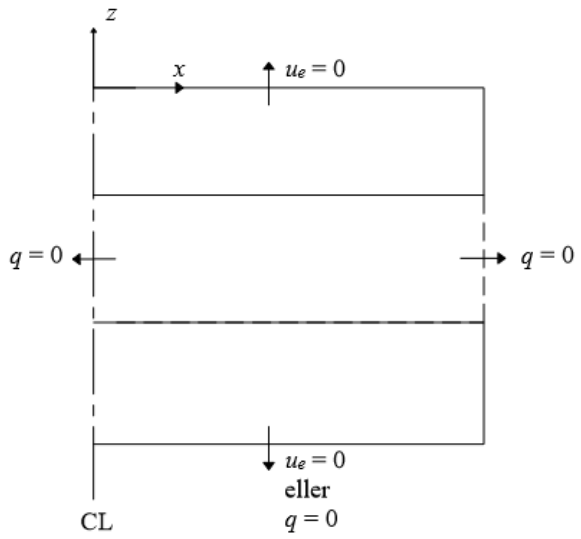
Figur 5.3 visar randvillkoren.

5.1.4 Ekvationssystem och elementbeskrivning

VisCons syfte är som konstaterat att beskriva porvattenövertryckets variation i tid och rum, $u_e(t, x, z)$. För att göra detta nyttjas Terzaghi's differentialekvation enligt ekvation 3.1, där härledning och antaganden återfinns i avsnitt 2.3.2. Antagandena vid härledningen av denna differentialekvation ligger till grund även för modellen implementerad i VisCon.

I avsnitt 3.1.3 utvecklades FE-formuleringen för differentialekvationen varpå ekvationssystemet enligt ekvation 3.15 – 3.18 erhölls. Formfunktionerna som väljs för att beskriva porvattenövertryckets variation inom respektive element antas linjära, och elementen ges av tvådimensionella triangulära trenodselement med en frihetsgrad per nod.

För att finna en tidsapproximation till ekvation 3.15 måste en tidsapproximation införas, vilket beskrivs i avsnitt 3.3. Enligt backward difference metoden och last-



Figur 5.3: *Beräkningsmodellen med definierade randvillkor för flöde och porvattenövertryck vid $t > 0$.*

vektorn $\mathbf{f}$=$\mathbf{0}$ fås systemet:

$$\hat{\mathbf{K}}\mathbf{a}_{i+1} = \hat{\mathbf{f}} \tag{5.1}$$

där

$$\hat{\mathbf{K}} = \mathbf{C} + \Delta t\mathbf{K} \tag{5.2}$$

$$\hat{\mathbf{f}} = \mathbf{C}\mathbf{a}_i \tag{5.3}$$

För att lösa detta ekvationssystem krävs att tidsinkrement, simuleringstid och minsta antal tidssteg för visualiseringen definieras. Dessa parametrar samt vilka värden de tilldelas enligt den fördefinierade modellen anges i tabell 5.2. Vidare måste den initiala spänningsfördelningen anges, det vill säga den spänningsfördelning som motsvarar $\mathbf{a}_{t=0}$. Denna fås med hjälp av de analytiska uttrycken givna för en strimlelast enligt avsnitt 2.2.2.

Tabell 5.2: *Parametrar som måste definieras vid genomförande av tidsimuleringen, samt värden på dessa för den fördefinierade modellen.*

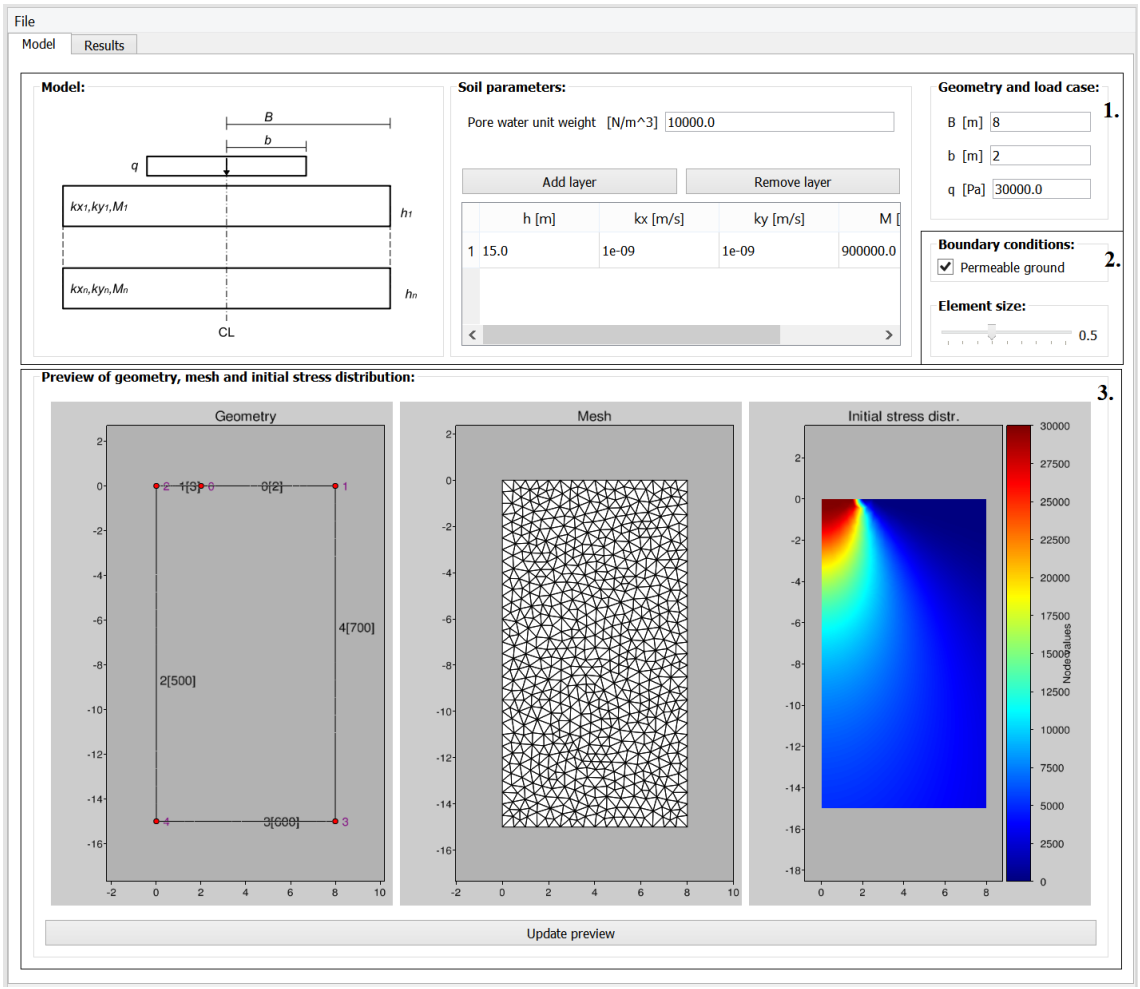
Simuleringstid T	2	[år]
Tidsinkrement, Δt	0.025	[år]
Minsta antal visualiserade tidssteg av resultat	8	[st]

5.2 Manual

En fristående användarmanual med en utförlig beskrivning av programmets funktioner och en sammanfattning av dess implementering, syfte och bakgrund finns att hitta i bilaga A. Denna är skriven på engelska. Nedan följer dock en kortfattad manual avsedd för att sammanfatta och illustrera de viktigaste funktionerna hos visualiseringsverktyget VisCon.

Flik 1: Modellbeskrivning

Det första fönstret som öppnas då VisCon initieras visas i figur 5.4.



Figur 5.4: *Programmet VisCons huvudfönster är avsett för att ange parametrar för aktuell modell samt förhandsgranskning av dess geometri, elementindelning och initiala spänningsfördelning.*

1. I rutan markerad 1 i figur 5.4 styrs den indata som presenterades i avsnitt 5.1.1, det vill säga jordprofilens geometri och egenskaper samt lastens utbredning och storlek.

2. Elementindelning och randvillkor, som presenterades i avsnitt 5.1.2 respektive 5.1.3, kan styras genom att ange elementstorlek och permeabilitet för modellens grund i området markerat 2 i figur 5.4

3. I programmets första fönster möjliggörs även en förhandsvisualisering av den inmatade modellens geometri, elementindelning samt initiala spänningsfördelning, se figur 5.4 markerat 3. Denna uppdateras genom att klicka på knappen "Update Preview".

Flik 2: Tidsimulering och visualisering

Då flik 2, "*Results*", klickas på genomför programmet en elementindelning, FEM-beräkning, tidsimulering samt extrahering av värden för porvattenövertrycket till de endimensionella graferna. Därefter öppnas fönstret som visas i figur 5.5.

1. I rutan markerad 1 i figur 5.5 korrigeras inparametrar rörande tidsimuleringen. Dessa presenterades i avsnitt 5.1.4. I fönstret finns vidare två knappar för att uppdatera visualiseringen. Knappen *Update Time simulation* uppdaterar hela tidsimuleringen genom att genomföra både en tidsapproximation samt extrahera värden för de endimensionella graferna vid valt antal sektioner. Knappen *Update 1D-plot* genomför bara extraktionen av värden till de endimensionella graferna utifrån valt antal sektioner. Det innebär alltså bara en framställning av nya värden för de nya sektionerna, baserat på samma beräkningsresultat.

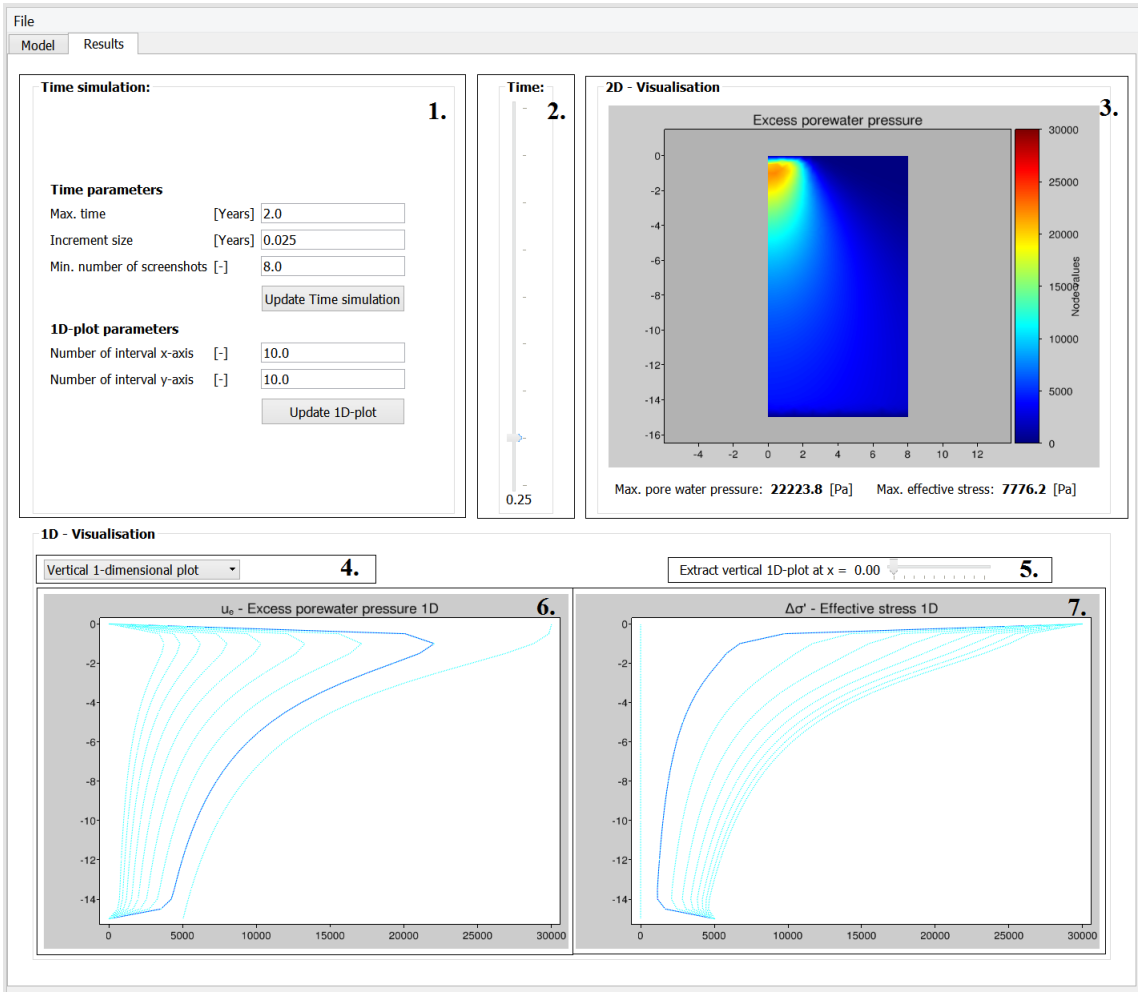
2. I rutan markerad 2 justeras med ett tidreglage för vilken tidpunkt den aktuella tvådimensionella samt endimensionella visualiseringen visas.

3. I rutan markerad 3 visas porvattenövertrycket, u_e, för den tid som angetts med tidreglaget. Vidare anges, i text, maximalt porvattenövertryck och maximal effektivspänning. Värt att notera är kopplingen mellan dessa värden som ges av effektivspänningsekvationen; summan av effektivspänningen, σ'_z, och porvattenövertrycket, u_e, ger den totala spänningsökningen, $\Delta\sigma$.

4. Rullgardinsmenyn i rutan markerad med 4 anger om de endimensionella graferna ska visas för ett vertikalt eller horisontellt snitt av modellen.

5. I rutan markerad 5 anges för vilket x- alternativt z-värde som värden längs en sektion ska extraheras till de endimensionella graferna.

6. Grafen i ruta 6 visar porvattenövertrycket vid den sektion som angivits med



Figur 5.5: *Visualiseringverktyget VisCons andra fönster är avsett för att styra parametrar till tidsimuleringen samt visualisera konsolideringsprocessen både tvådimensionellt och endimensionellt.*

reglaget i ruta 5. En grafisk framställning ges för samtliga definierade tidssteg, men den graf som är markerad med en mörkare blå färg representerar den tidpunkt som angivits med tidreglaget i ruta 2.

7. Grafen i ruta 7 visar effektivspänningen på motsvarande sätt som porvattenövertrycket visas i ruta 6.

5.3 Övningsmaterial

Till programmet VisCon har övningsuppgifter utarbetats. Dessa återfinns i bilaga B. I den bilagan finns även en beskrivning av hur VisCon är tänkt att användas

vid beräkning av sättningar samt ett tillämpat räkneexempel. Övningsuppgifterna är utformade för att behandla följande:

- En jämförelsestudie där ett konsolideringsproblem löses med både en endimensionell och en tvådimensionell beräkningsmodell.
- En undersökning av hur resulterande sättningar beror av antal beräkningslager samt hur många lager man behöver för att uppnå en konvergerande lösning.
- En studie av hur strimlelastens bredd påverkar vilken typ av beräkningsmodell som bör tillämpas.
- En jämförelse av hur permeabiliteten för grunden under jordlagret påverkar porvattenövertryckets variation i tid och rum.
- En förståelse för hur sättningarna varierar utifrån centrumlinjen, samt en förmåga att utifrån beräknade värden visualisera sättningarna i två dimensioner.
- En undersökning av hur val av elementindelning, jordprofilens bredd och tidsinkrement påverkar beräkningsmodellen och resultatet i programmet VisCon.

Kapitel 6

VisCon - Implementering

I det här kapitlet beskrivs uppbyggnaden av koden i VisCon samt implementeringen i Python. Kapitlet inleds med en kort redogörelse av utvecklingsmiljön samt rutiner och programvaror som programmet är uppbyggt kring. Därefter presenteras själva huvudprogrammets upplägg utifrån ett klassdiagram samt en kortfattad beskrivning av respektive klass syfte. För att möjliggöra utvecklingen av VisCon har en del nya rutiner behövt utvecklas från grunden, alternativt implementeras från CALFEM [6] för Matlab till CALFEM för Python. Dessa sammanfattas i det sista avsnittet i det här kapitlet. Källkoden för huvudprogrammet har utelämnats i den här rapporten, dock återfinns källkoden för de ny rutinerna i bilaga C.

6.1 Utvecklingsmiljö

Python är ett objektorienterat programmeringsspråk för allmänbruk som finns tillgängligt att ladda ned fritt från Pythons officiella hemsida. Programmeringsspråket utvecklades av Guido van Rossum och publicerades första gången 1991. Tanken med Python var att skapa ett programmeringsspråk där fokus låg på användarvänlighet och en ren användarlayout [3]. I utvecklingen av VisCon har Python version 3.5.3 använts som utvecklingsmiljö. Som komplement i programutvecklingen har olika paket och rutiner använts för att hantera beräkning, gränssnitt och visualisering. Nedan följer en kortfattad presentation av dessa.

CALFEM för Python

CALFEM, det vill säga Computer Aided Learning of the Finite Element Method, är en interaktiv programvara utvecklad på avdelningen för Byggnadsmekanik vid

Lunds Tekniska Högskola. CALFEM innehåller rutiner utvecklade för Matlab och tillämpas för att lösa finita elementproblem. Programmets bakomliggande syfte är att underlätta undervisning inom finita elementmetoden. I manualen för CALFEM finns förklaringar av samtliga utvecklade rutiner samt exempel där dessa tillämpas. [6]

CALFEM för Python är en implementering av CALFEM men i programmeringsspråket Python istället för Matlab. CALFEM för Python utvecklades 2010 inom ramen för ett examensarbete. Majoriteten av funktionerna i CALFEM har implementerats i CALFEM för Python men det finns fortfarande en del rutiner som saknas. För en full förteckning över vilka rutiner som implementerats hänvisas till examensarbetet *Implementation of CALFEM for Python* skrivet av Andreas Ottosson [15]. Elementindelnings- och visualiseringsrutinerna i CALFEM för Python utvecklades 2013 inom ramen för ett annat examensarbete. I detta nyttjades mjukvaran GMSH och visualiseringsbiblioteket VisVis för att implementera förbättrade elementindelnings- och visualiseringsrutiner [9].

PyQt

PyQt är är ett så kallat GUI-bibliotek, där GUI står för Graphical User Interface. Med hjälp av PyQt kan ett gränssnitt byggas upp utifrån så kallade widgets, vilket kan vara exempelvis tabeller, reglage, textrader eller knappar. Händelser, så som då en widget aktiveras eller ändras, kan med hjälp av PyQt kopplas till funktioner definierade i programkoden. Användargränssnittet kan antingen byggas upp i kod eller skapas som en UI-fil, User interface, med hjälp av QT-designtool. PyQt5 och QT-designtool version 5.7.0. användes i utvecklingen av användargränssnittet i VisCon [2].

Visvis

VisVis är ett bibliotek för Python som är utvecklat för att möjliggöra visualisering av endimensionella upp till fyrdimensionella data. Visvis är objektorienterat och är implementerat med syntax snarlik den som används vid visualisering i Matlab. I Visvis kan allt från grafer utifrån givna punkter till bilder och skuggade elementindelningar visualiseras [4].

NumPy

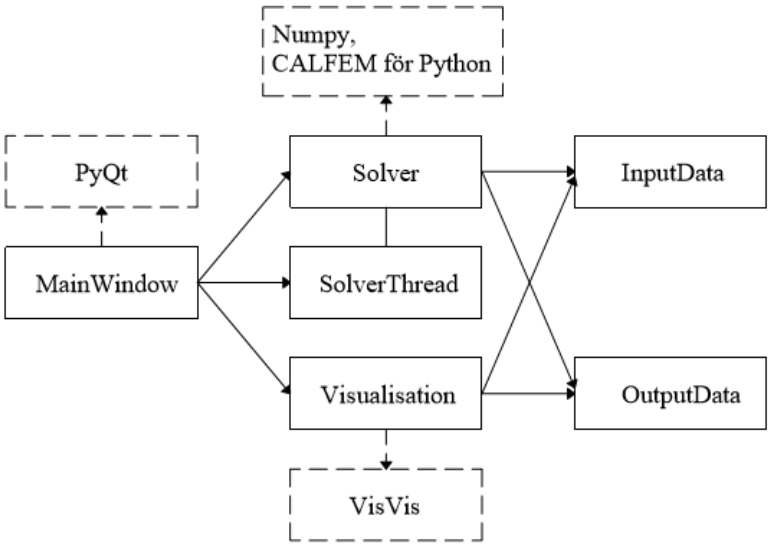
NumPy är en modul utvecklad för Python som kan hantera matrisoperationer och innehåller ett stort matematiskt bibliotek. NumPy är utvecklat utifrån en tidigare

modul kallad Numeric och publicerades i namnet NumPy år 2006. Med hjälp av NumPy kan olika matrisoperationer genomföras på ett smidigt sätt [1].

6.2 Implementering av huvudprogram

6.2.1 Klassdiagram

Huvudprogrammet, VisCon, bygger på 6 stycken klasser och därutöver 4 importerade bibliotek med rutiner presenterade i avsnitt 6.1. I figur 6.1 visas ett diagram för programmets klasser. De heldragna linjerna symboliserar huvudprogrammets 6 klasser och kopplingen mellan dessa. De streckade rutorna samt linjerna visar hur de olika biblioteken beskrivna i avsnitt 6.1 använts i programmet. I följande text presenteras klassernas huvudsakliga innehåll och syfte.

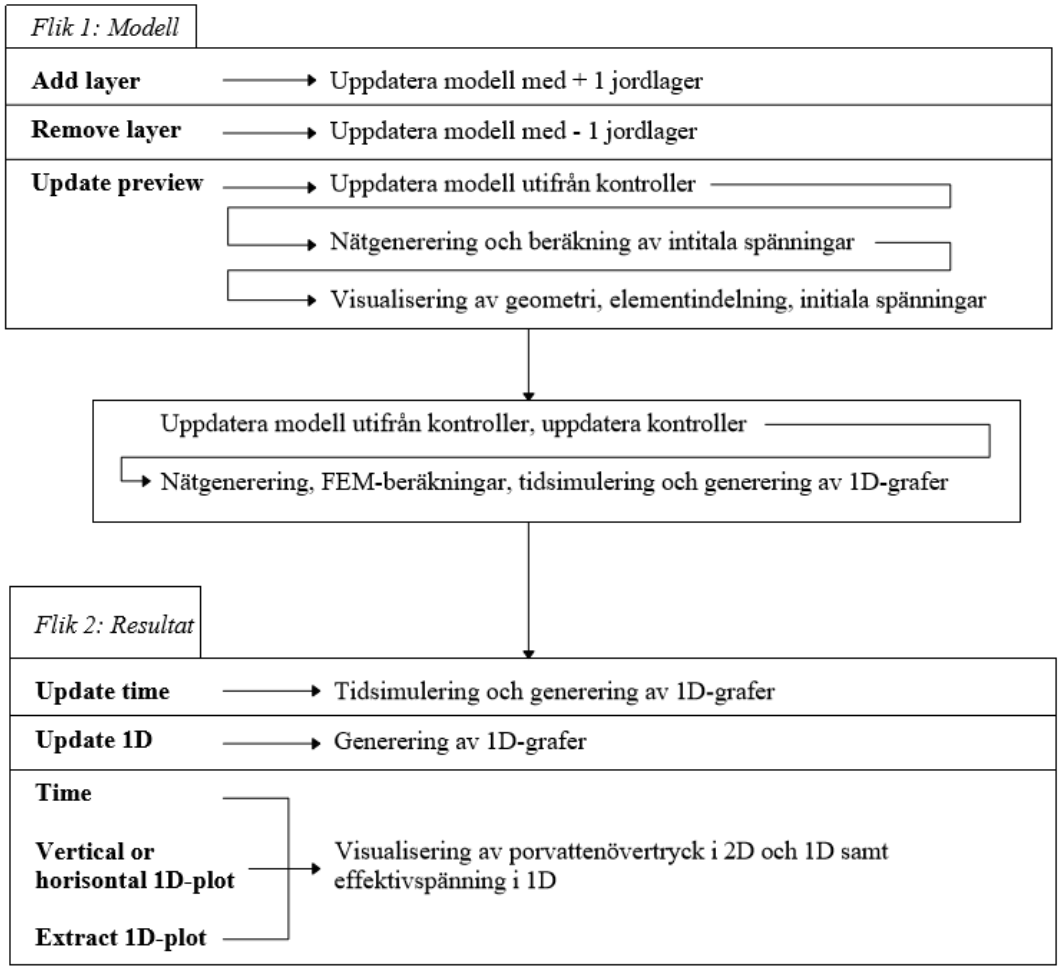


Figur 6.1: *Klassdiagram som visar hur programmets respektive klasser är kopplade och vidare hur importerade bibliotek används av dessa.*

6.2.2 Klass: MainWindow

Klassen MainWindow implementerar användargränssnittet. Användargränssnittet beskrivs i en UI-fil skapad i Qt-designer vilken läses in från PyQt. Klassen MainWindow sköter införandet av vissa delar i huvudfönstret som ej definierats i Qt-designer.

Användargränssnittet är uppbyggt utifrån två flikar, *Model* och *Results*, se figur 5.4 respektive 5.5. Som tidigare nämnts är den första fliken avsedd för modellens indata rörande geometri och lastfall. I denna möjliggörs även en förhandsvisualisering av beräkningsmodellens geometri, elementindelning samt den initiala spänningsfördelningen. Den andra fliken är avsedd för att styra tidsimuleringen samt visualisera resultatet i både en och två dimensioner. Klassen MainWindow hanterar samtliga händelser kopplade till objekt i gränssnittet och anropar olika klasser och funktioner för beräkning och visualisering. I figur 6.2 visas en schematisk bild över händelser som aktiveras vid interaktion med användargränssnittet.



Figur 6.2: *En schematisk förklaring av hur händelser är kopplade till olika knappar och reglage i användargränssnittet.*

6.2.3 Klass: InputData

I klassen InputData lagras modellens inparametrar. Vid programmets initiering uppdateras gränssnittets kontroller med en fördefinierad modell som sedan kan korrigeras efter önskade indata. Vidare är det i klassen InputData som en geometri-instans skapas utifrån beräkningsmodellens geometri.

6.2.4 Klass: OutputData

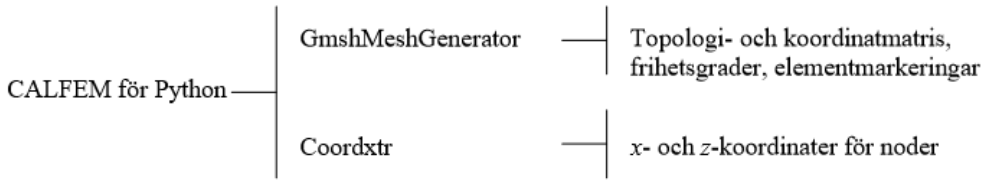
I klassen OutputData lagras variabler för genererad data, både skalärer, matriser och vektorer. Variablerna är tomma till dess att beräkningarna genomförts och de tilldelats värden.

6.2.5 Klass: Solver

Klassen Solver hanterar samtliga beräkningar och utgör VisCons beräkningsverktyg. Klassen är indelad i fyra olika funktioner som sköter de olika beräkningsstegen; executeMesh, executeFEM, executeTime och executeExtr. Dessa funktioner presenteras mer ingående nedan.

executeMesh

Den första funktionen i klassen sköter elementindelningen. Det görs genom att klassen GmshMeshGenerator i CALFEM för Python anropas. Utifrån given geometriinstans, elementstorlek, elementtyp och antal frihetsgrader per nod skapas bland annat systemets topologimatris, en matris med elementens koordinater, frihetsgrader och elementmarkeringar, det vill säga en markering av vilka noder som tillhör

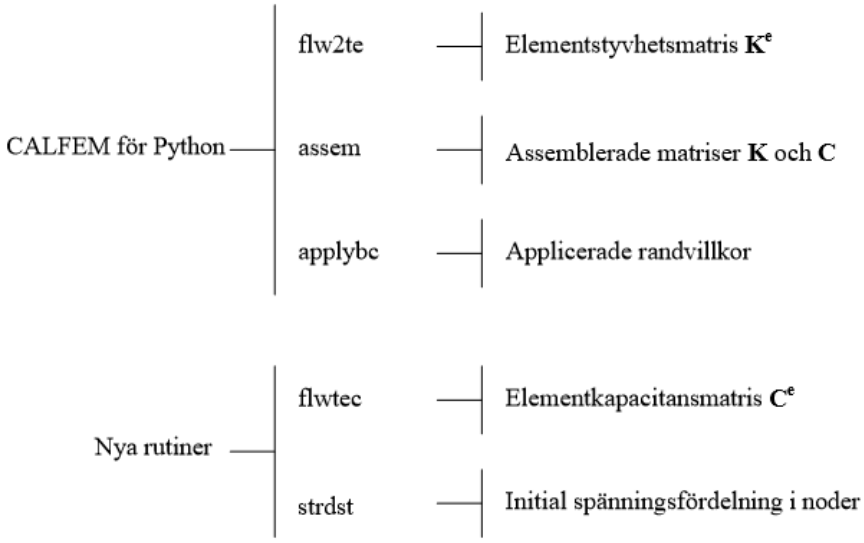


Figur 6.3: *En sammanfattning av vilka bibliotek och funktioner som nyttjas av funktionen executeMesh i klassen Solver.*

definierade ränder. Från CALFEM för Python har vidare en funktion kallad coordxtr använts för att få ut samtliga noders x- respektive z-koordinater elementvis. I figur 6.3 sammanfattas vilka bibliotek och funktioner som nyttjas och för vilka ändamål.

executeFEM

Den andra funktionen i klassen Solver, executeFEM, sköter framtagningen av elementstyvhetsmatriserna $\mathbf{K}^e$, elementkapacitansmatriserna $\mathbf{C}^e$ samt assembleringen av dessa. Detta görs med hjälp av rutinerna flw2te och assem i CALFEM för Python samt rutinen flwtec som implementerats inom ramen av detta examensarbete. För en mer utförlig beskrivning av rutinen flwtec se avsnitt 6.3.1.

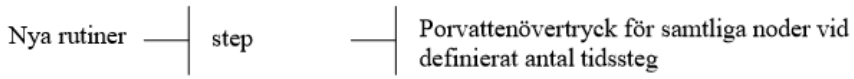


Figur 6.4: *En sammanfattning av vilka bibliotek och funktioner som nyttjas av funktionen executeFEM i klassen Solver*

Funktionen executeFEM genererar även en vektor för den initiala spänningsfördelningen orsakad av en jämnt utbredd last q. För detta används den implementerade rutinen strdst som finns förklarad i avsnitt 6.3.2. Randvillkor definieras vidare på aktuella ytor med funktionen applybc i CALFEM för Python. I figur 6.4 sammanfattas vilka bibliotek och funktioner som nyttjas samt för vilka ändamål.

executeTime

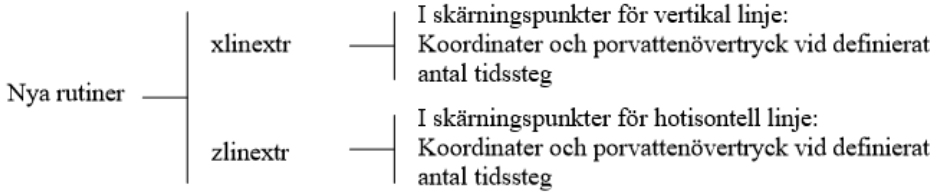
Den tredje funktionen i klassen Solver genomför tidsimuleringen. Tidsimuleringen är en approximation som beror av definierade värden i MainWindow. Baserat på valda tidsimuleringsparametrar samt beräknad styvhetsmatris, kapacitansmatris, initial spänningsfördelning samt randvillkor genomförs tidstegandet. Detta genomförs med den nya implementerade rutinen Step som finns beskriven i avsnitt 6.3.3. I figur 6.5 sammanfattas vilka bibliotek och funktioner som nyttjas samt för vilka ändamål.



Figur 6.5: *En sammanfattning av vilka bibliotek och funktioner som nyttjas av funktionen executeTime i klassen Solver.*

executeExtr

Klassen Solvers fjärde och sista klass sköter genereringen av de endimensionella graferna. Den nyttjar rutinerna xlinextr och zlinextr för att interpolera fram värden för porvattenövertrycket mellan noder som ligger intill en vald sektionslinje av modellen. Metoden för detta ges i beskrivningen av rutinerna i avsnitt 6.3.4 respektive 6.3.5. I figur 6.6 sammanfattas vilka bibliotek och funktioner som nyttjas samt för vilka ändamål.



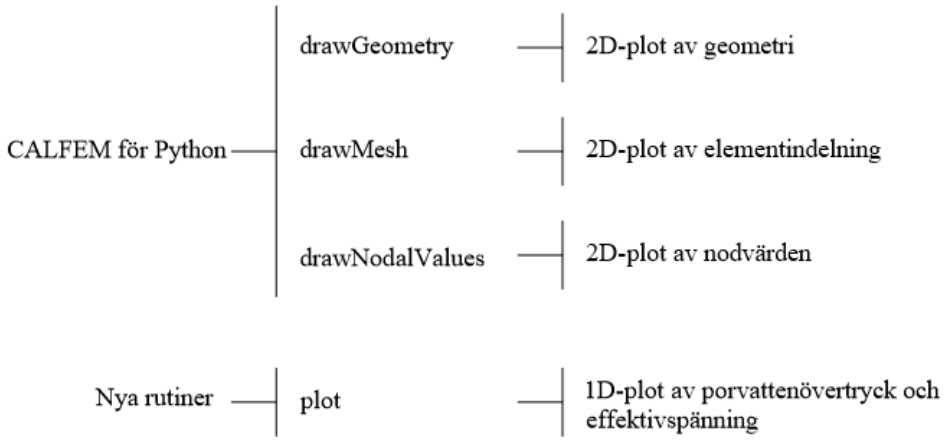
Figur 6.6: *En sammanfattning av vilka bibliotek och funktioner som nyttjas av funktionen executeExtr i klassen Solver.*

6.2.6 Klass: SolverThread

Klassen SolverThread hanterar beräkningarna i bakgrunden så att gränssnittet inte låser sig då beräkningarna genomförs. En så kallad progress bar är kopplad till klassen för att informera användaren om vilket beräkningssteg som genomförs för tillfället.

6.2.7 Klass: Visualisation

Klassen Visualisation hanterar visualiseringen av resultatet efter det att beräkningarna genomförts. Visualiseringen sker både med hjälp av endimensionella och tvådimensionella grafer. Klassen är kopplad till CALFEM för Python, som har utvecklade visualiseringsrutiner för geometri, elementindelning och nodvärden i två dimensioner. För visualisering av de framtagna värdena för de endimensionella graferna används enkla visualiseringsfunktioner i VisVis. I figur 6.7 sammanfattas vilka bibliotek och funktioner som nyttjas samt för vilka ändamål.



Figur 6.7: *En sammanfattning av vilka bibliotek och funktioner som nyttjas i klassen Visualisation.*

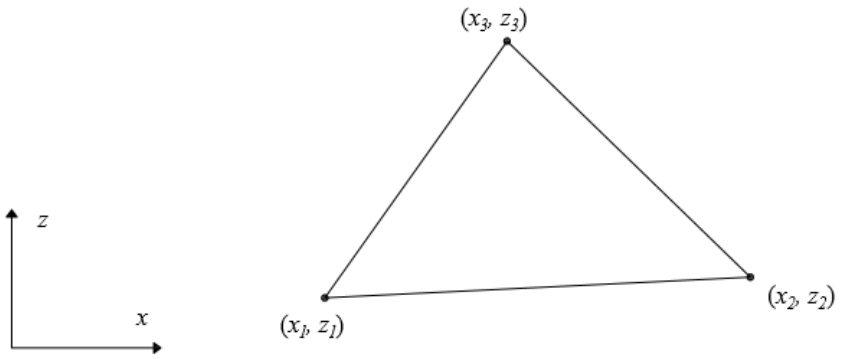
6.3 Implementering av nya rutiner

I utvecklingen av VisCon identifierades ett behov av vissa rutiner för beräkning och visualisering. Nedan följer en presentation av rutiner som implementerats från CALFEM för Matlab till CALFEM för Python samt de nya rutiner som utvecklats för detta arbete.

6.3.1 Funktion: flwtec

Syfte

Syftet med funktionen är att beräkna elementkapacitansmatrisen, $\mathbf{C}^e$, för ett tvådimensionellt element med tre noder.



Figur 6.8: *Definition av koordinater för ett element med tre noder.*

Syntax

```
Ce = flwtec(ex, ez, gamma, M)
```

Indata

Indata för funktionen definieras enligt tabell 6.1, utifrån ett element enligt figur 6.8.

Utdata

Funktionen returnerar elementets kapacitansmatris $\mathbf{C}^e$. Dim($\mathbf{C}^e$)=[3x3].

Tabell 6.1: *Indata till funktionen flwtec.*

gamma	=	Porvattnets tunghet γ_w		[N/m^3]
M	=	Elementets kompressionsmodul		[Pa]
ex	=	Elementets x-nodkoordinater	[x_1,x_2,x_3]	[m]
ez	=	Elementets z-nodkoordinater	[z_1,z_2,z_3]	[m]

Teori och referens till källkod

Teorin bakom det här avsnittet behandlades i avsnitt 3.3. Där konstaterades att för en partiell differentialekvation enligt ekvation 3.15 ges elementkapacitansmatrisen $\mathbf{C}^e$ enligt:

$$\mathbf{C}^e = \int_A \mathbf{N}^T \frac{\gamma_w}{M} \mathbf{N} \; dA \tag{6.1}$$

För ett tvådimensionellt element med 3 noder kan elementkapacitansmatrisen skrivas som [16]:

$$\mathbf{C}^e = \frac{A\gamma_w}{M} \begin{bmatrix} 1/6 & 1/12 & 1/12 \\ 1/12 & 1/6 & 1/12 \\ 1/12 & 1/12 & 1/6 \end{bmatrix} \tag{6.2}$$

där A är arean för elementet i [m^2]. Baserat på detta uttryck beräknar funktionen flwtec elementets kapacitansmatris. För fullständig beräkningskod se bilaga C.1.

6.3.2 Funktion: strdst

Syfte

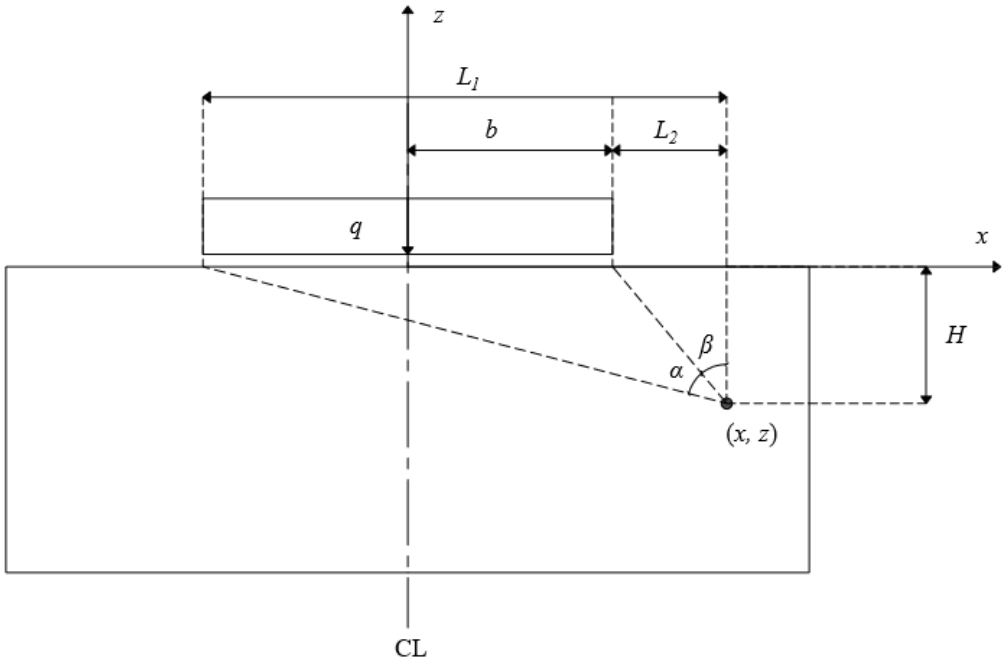
Syftet med funktionen är att beräkna den vertikala spänningsfördelningen $\Delta\sigma_z$ [Pa] vid en jämnt utbredd last, q [Pa], enligt figur 6.9. Funktionen är tillämpbar för en symmetrisk tvådimensionell modell.

Syntax

```
sigmaz = strdist(coords, ndof, q, B, b, hvec)
```

Indata

Indata ges enligt tabell 6.2.



Figur 6.9: *Definition av mått för att, med hjälp av funktionen strdst, beräkna spänningen i valfri punkt i en jordprofil.*

Tabell 6.2: *Indata till funktionen strdstr*

coords	=	Nodernas koordinater	[[x_1, z_1]; ... ;[x_n, z_n]]	[m]
ndof	=	Antal frihetsgrader		
q	=	Lastens magnitud		[Pa]
b	=	Lastens bredd, mätt från C.L.		[m]
hvec	=	Vektor med jordlagerdjup	[[h_1]; ... ;[h_n]]	[m]

Utdata

Funktionen returnerar en vektor, **sigmaz**, innehållande vertikala spänningen $\Delta\sigma_z$ i samtliga noder. Dim(**sigmaz**)= [ndof x 1].

Teori och källkod

Teorin bakom den här funktionen behandlas i avsnitt 2.2.2. Eftersom det är den vertikala spänningsfördelningen som är av intresse för utvecklingen av huvudprogrammet VisCon är det spänningsuttrycket för $\Delta\sigma_z$ som har implementerats i den här rutinen. För att beräkna spänningsfördelningen i samtliga noder utgår funktionen från definierade koordinater samt lastens storlek och utbredning. Modellen är

definierad enligt figur 6.9. Med dessa definitioner erhålls följande uttryck för vinklarna α och β då $H \neq 0$:

$$\beta = \arctan(L_2/H) \tag{6.3}$$
$$\alpha = \arctan(L_1/H) - \beta \tag{6.4}$$

där

$$L_1 = x + b \tag{6.5}$$
$$L_2 = x - b \tag{6.6}$$
$$H = |\, z \,| \tag{6.7}$$

Utifrån dessa värden ges den vertikala spänningen enligt:

$$\Delta\sigma_z = \frac{q}{\pi}(\alpha + \sin\alpha\cos(\alpha + 2\beta)) \tag{6.8}$$

Om $H = 0$ och $x \leq b$, det vill säga för en punkt belägen under lasten längs med markytan, erhålls direkt $\Delta\sigma_z = q$. Om $H = 0$ och $x > b$, det vill säga för en punkt belägen utanför lasten längs med markytan, fås $\Delta\sigma_z = 0$. I bilaga C.2 återfinns funktionens kompletta källkod.

6.3.3 Funktion: step

Syfte

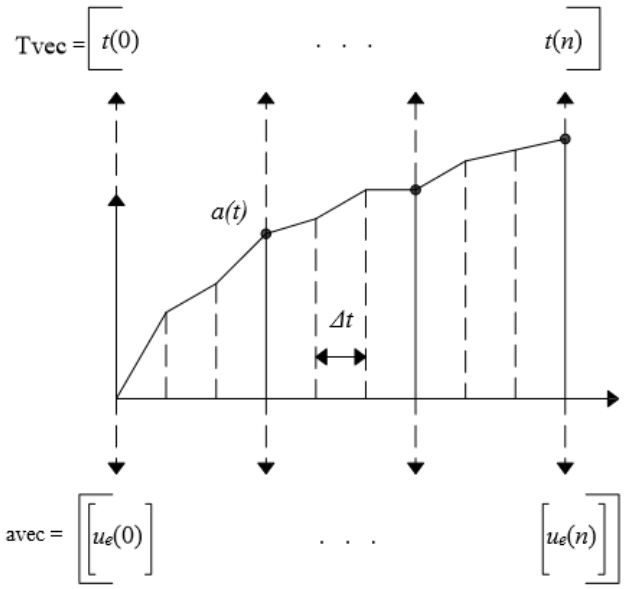
Syftet med funktionen är att, med hjälp av en tidsapproximation baserad på backward difference-metoden erhålla en lösning till en partiell differentialekvation $u_e(t, x, z)$ vid ett valt antal definierade tidpunkter. Figur 6.10 illustrerar indata och utdata samt principen för tidsapproximationen.

Syntax

```
avec, Tvec = step(K, C, a0, bc, bcval, dt, T, Tnr):
```

Indata

För funktionen anges indata enligt tabell 6.3. I indatan är det möjligt att ange tidsinkrementens storlek samt den totala simuleringstiden. Vidare anges för hur många tidssteg som nodvärderna **a**(t) ska lagras i den returnerade matrisen **avec**.



Figur 6.10: *Funktionen step genomför en tidsimulering beroende på valda tidsinkrement. För angivet antal tidpunkter, Tnr, sparas resultatet för porvattenövertrycket i matrisen avec och och motsvarande tidpunkter i vektorn Tvec.*

Tabell 6.3: *Indata för funktionen step*

K	=	Styvhetsmatris för modellen	
C	=	kapacitansmatris för modellen	
a0	=	Initial spänningsfördelning vid t = 0	
bc	=	Noder med angivna randvillkor	
bcval	=	Värden för randvillkor	[Pa]
dt	=	Tidsinkrement	[år]
T	=	Simuleringstid	[år]
Tnr	=	Antal tidssteg för visualisering	

Utdata

Funktionen returnerar en matris, **avec**, innehållande nodvärden för samtliga noder vid tidpunkterna som anges i den returnerade vektorn **Tvec**. Dim(**avec**) = [ndof x Tnr+1], Dim(**Tvec**) = [1 x Tnr+1].

Teori och källkod

Teorin bakom den här funktionen behandlas i avsnitt 3.3. Enligt informationen som återfinns i det kapitlet kan en ekvation enligt ekvation 3.15 lösas med en tidsapproximation. Med hjälp av backward difference-metoden och en yttre last $\mathbf{f}= \mathbf{0}$ fås:

$$\hat{\mathbf{K}}\mathbf{a}_{i+1} = \hat{\mathbf{f}} \tag{6.9}$$

där

$$\hat{\mathbf{K}} = \mathbf{C} + \Delta t\mathbf{K} \tag{6.10}$$

$$\hat{\mathbf{f}} = \mathbf{C}\mathbf{a}_i \tag{6.11}$$

Funktionen stegar sig alltså fram till en lösning $\mathbf{a}_{i+1}$ genom att bestämma nya styvhetsmatriser och lastmatriser vid varje tidsteg Δt. Approximationen utgår från att $\mathbf{a}_{t=0}$ ges av den spänningsfördelning som angivits i a0.

Källkoden återfinns i sin helhet i bilaga C.3. Observera att en liknande funktion, step, finns implementerad i CALFEM för Matlab, se definition i CALFEM a Finite Element Toolbox [6]. Dock skiljer sig dessa åt i indata och utdata.

6.3.4 Funktion: xlinextr

Syfte

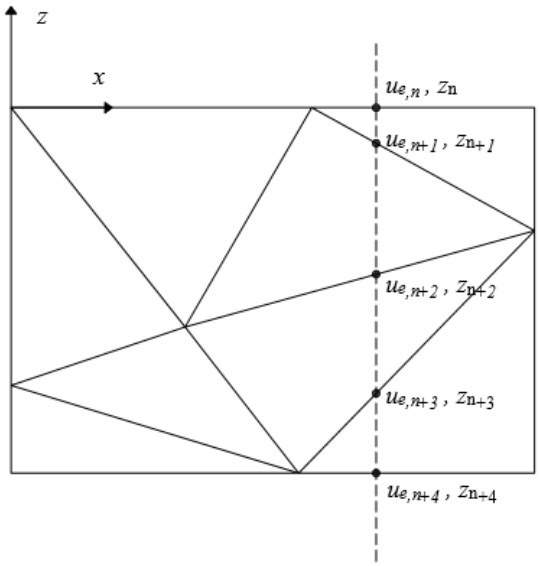
Syftet med funktionen är att skapa en vektor med z-värden samt en matris med funktionsvärden utifrån de punkter där en vertikal linje vid ett definierat x-värde skär elementnätet, se figur 6.11. Funktionen förutsätter en modell med tvådimensionella trenodselement och en frihetsgrad per nod. Vidare förutsätter den linjära formfunktioner.

Syntax

```
yvec, uvec = xlinextr(xl, surfx, ex, ez, ndof, edof,avec)
```

Indata

Indata ges av tabell 6.4.



Figur 6.11: *Funktionen xlinextr plockar ut koordinater som skär elementindelningen längs en definierad vertikal linje samt interpolerar fram porvattenövertryck och koordinater för dessa punkter. Detta görs för samtliga tidpunkter definierade i avec.*

Tabell 6.4: *Indata för funktionen xlinextr*

xl	=	x-värde för definierad sektion	[m]
surfx	=	x-värden för vertikala jordlager	[m]
ex	=	Matris med elementens x-koordinater. [[$x_{1,1}$,$x_{1,2}$,$x_{1,3}$];...;[$x_{n,1}$,$x_{n,2}$,$x_{n,3}$]]	[m]
ez	=	Matris med elementens y-koordinater. [[$z_{1,1}$,$z_{1,2}$,$z_{1,3}$];...;[$z_{n,1}$,$z_{n,2}$,$z_{n,3}$]]	[m]
ndof	=	Antal frihetsgrader	
edof	=	Topologimatris	
avec	=	Matris med nodvärden för definierade tidssteg.	[Pa]

Utdata

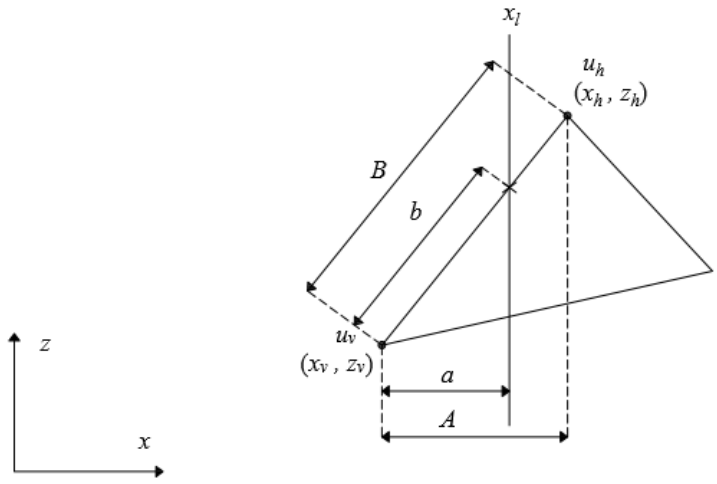
Funktionen returnerar en vektor **zvec** innehållande z-väden för samtliga skärningspunkter längs med den vertikala linjen. Den returnerar vidare en matris **uvec** som innehåller värden för porvattenövertrycket i elemenindelningens skärningspunkter, vid samtliga definierade tidssteg i indata **avec**. Dim(**zvec**) = [(Antal skärningspunkter för xl), 1], dim(**uvec**) = [(Antal skärningspunkter för xl), antal kolonner i **avec**].

Teori och källkod

Den här funktionen utgår ifrån ett antagandet om linjära formfunktioner och förutsätter med andra ord att det är en linjär variation mellan nodvärdena. Utifrån detta bygger funktionen på två huvudprinciper beskrivna nedan.

1. Plocka ut element
Det första steget bygger på att funktionen avgör vilka element som korsar den definierade sektionslinjen. Principen utgår från att söka igenom matrisen **ex** som innehåller samtliga elements x-koordinater elementvis. Ligger det största respektive minsta x-värdet på varsin sida om den definierade vertikala linjen, xl, sparas dess elementindex. Efter detta plockas samtliga x- respektive z-koordinater och nodvärden ut för de indexerade elementen.

2. Elementvis interpolering.
I nästa steg genomförs en interpolering, där z- respektive u_e-värden bestäms för de två punkterna där varje elementet skär den definierade sektionslinjen xl. Det görs genom att först avgöra vilka punkter som ligger till höger respektive till vänster om linjen. Utifrån detta nyttjas likformighet och utifrån figur 6.12 härleds följande samband, där x_l anger x-värdet för den definierade sektionslinjen:



Figur 6.12: *För att genomföra en interpolering för punkterna där linjen xl skär elementet kan måtten från denna figur utnyttjas i kombination med likformighet.*

$$\frac{b}{B} = \frac{a}{A} = \frac{x_l - x_v}{x_h - x_v} \tag{6.12}$$

Utifrån interpolering mellan den vänstra noden, indexerad v, och den högra noden,

indexerad h, samt förhållandet b/B i ekvation 6.12 fås:

$$u_e = u_v + (u_h - u_v)\frac{b}{B} = u_v + (u_h - u_v)\frac{x_l - x_v}{x_h - x_v} \tag{6.13}$$

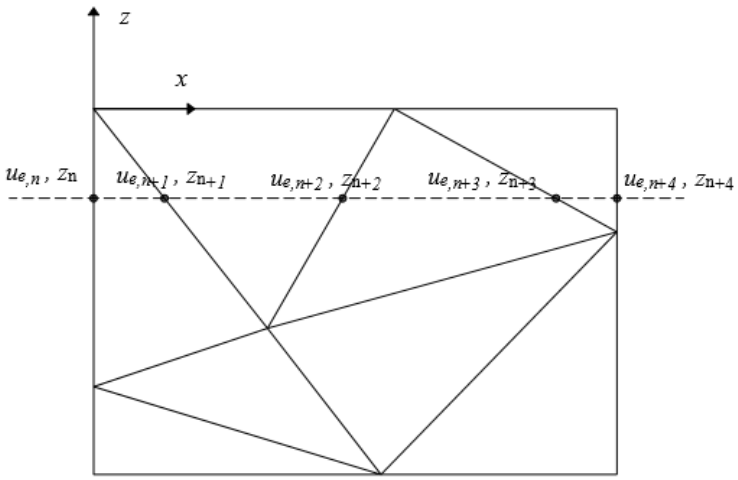
$$z = z_v + (z_h - z_v)\frac{b}{B} = z_v + (z_h - z_v)\frac{x_l - x_v}{x_h - x_v} \tag{6.14}$$

Interpoleringen genomförs två gånger för samtliga element som skär den definierade linjen. Källkoden för funktionen återfinns i sin helhet i bilaga C.4.

6.3.5 Funktion: zlinextr

Syfte

Syftet med funktionen är att skapa en vektor med x-värden samt en matris med funktionsvärden utifrån punkter där en horisontell linje vid ett definierat z-värde skär elementnätet, se figur 6.13. Funktionen förutsätter en modell med tvådimensionella trenodselement och en frihetsgrad per nod. Vidare förutsätter den linjära formfunktioner.



Figur 6.13: *Funktionen zlinextr plockar ut punkter som skär elementindelningen längs en definierad horisontell linje samt bestämmer porvattenövertryck och koordinater för dessa punkter. Detta görs för samtliga tidpunkter definierade i avec.*

Syntax

Följande syntax används för att anropa funktionen:

```
xvec, uvec = zlinextr(zl, surfz, ex,ez, ndof, edof,avec)
```

Indata

Indata ges av tabell 6.5.

Tabell 6.5: *Indata för funktionen zlinextr*

zl	=	x-värde för definierad sektion	[m]
surfz	=	z-värden för horisontella jordlager	[m]
ex	=	Matris med elementens x-koordinater. [[$x_{1,1}$,$x_{1,2}$,$x_{1,3}$];...;[$x_{n,1}$,$x_{n,2}$,$x_{n,3}$]]	[m]
ez	=	Matris med elementens y-koordinater. [[$z_{1,1}$,$z_{1,2}$,$z_{1,3}$];...;[$z_{n,1}$,$z_{n,2}$,$z_{n,3}$]]	[m]
ndof	=	Antal frihetsgrader	
edof	=	Topologimatris	
avec	=	Matris med nodvärden för definierade tidssteg.	[Pa]

Utdata

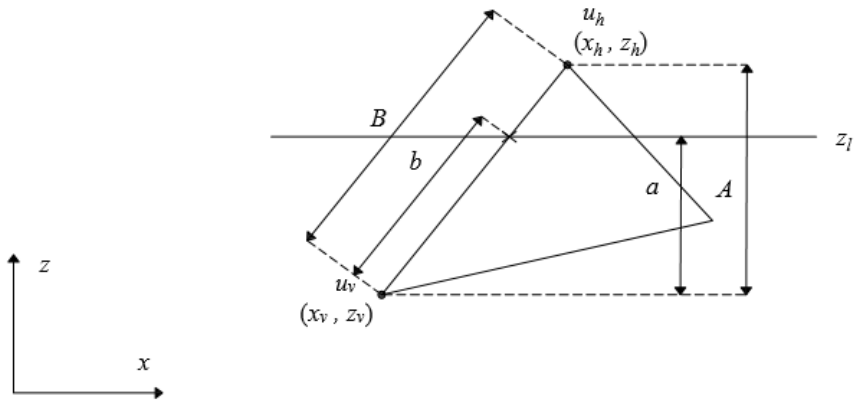
Funktionen returnerar en vektor **xvec** innehållande x-väden för samtliga skärningspunkter längs med den horisontella linjen zl. Den returnerar vidare en matris **uvec** som innehåller värden för porvattenövertrycket i elementindelningens skärningspunkter, vid samtliga definierade tidssteg i indata **avec**. Dim(**xvec**) = [(Antal skärningspunkter för zl), 1], dim(**uvec**) = [(Antal skärningspunkter för zl), antal kolonner i **avec**].

Teori och källkod

Uppbyggnaden av den här funktionen följer samma principer som funktionen xlinextr, därför hänvisas till teorin för denna i avsnitt 6.3.4. Då värden plockas ut från en horisontell linje fås dock uttrycken för porvattenövertryck och x-koordinat i punkterna som skär den definierade linjen från figur 6.14 enligt:

$$\frac{b}{B} = \frac{a}{A} = \frac{z_l - z_v}{z_h - z_v} \tag{6.15}$$

vilket via interpolering ger:



Figur 6.14: *För att genomföra en interpolering för skärningspunkterna kan måtten från denna figur utnyttjas i kombination med likformighet.*

$$u = u_v + (u_h - u_v)\frac{b}{B} = u_v + (u_h - u_v)\frac{z_l - z_v}{z_h - z_v} \tag{6.16}$$

$$x = x_v + (x_h - x_v)\frac{b}{B} = x_v + (x_h - x_v)\frac{z_l - z_v}{z_h - z_v} \tag{6.17}$$

där z_l anger z-koordinaten för den definierade horisontella linjen.

Kapitel 7

Diskussion

VisCon är ett visualiseringsverktyg avsett för användning i undervisningssammanhang. Tanken med VisCon är att på ett avskalat sätt understödja studenterna med en beräkning samt grafisk framställning av porvattenövertryckets variation i två dimensioner, eftersom nuvarande kurslitteratur [21] endast ger en endimensionell framställning av problemet. I följande avsnitt diskuteras val i utformningen av VisCon, där både VisCon som ett pedagogiskt verktyg och implementeringen av själva programvaran analyseras. Programmet är ej utvecklat för att användas i dimensioneringssammanhang. För sådana beräkningar finns redan utvecklade FEM-program som löser liknande problemställning är och dessa är av betydligt mer avancerad karaktär.

7.1 VisCon som ett pedagogiskt verktyg

7.1.1 Användbarhet

Anpassning

I utvecklingen av ett pedagogiskt verktyg avsett för undervisning krävs, som konstaterats i kapitel 4, en hög användbarhet. I utvecklingen av VisCon fokuserades på att uppnå detta. Det första begreppet som anses definiera användbarhet är anpassning. Forskning har visat att begränsade program ofta resulterar i en snabbare inlärningsprocess. Därför har många onödiga funktioner och finesser skalats bort i användargränssnittet för VisCon. Till skillnad från de program som redan existerar för beräkning av konsolidering har endast de mest nödvändiga funktionerna implementerats i detta visualiseringsverktyg.

Problemställningen till det här arbetet utarbetades i samarbete med ansvarig kurslärare, Erika Tudisco, i kursen *Grundläggningsteknik VGTN01* vid Lunds Tekniska Högskola. Bakgrunden till problemställningen var att läraren identifierat ett behov av ett program för att visualisera konsolidering. Att VisCon utvecklats direkt efter aktuellt användningsområde, med de avsedda användarna i ett tydligt fokus, ger visualiseringsverktyget väldigt goda förutsättningar ur ett pedagogiskt perspektiv. Genom att kursläraren har varit med och påverkat utformningen av programmet är det mer troligt att programmet följer övrig undervisning gällande beteckningar, berörda områden, teckenkonventioner med mera. Detta gör det lättare både för läraren att väva in programmet som ett naturligt inslag i utbildningen och för eleverna att följa med och känna att det fyller en tydlig funktion inom utbildningen.

Användarvänlighet

I valet av utvecklingsmiljö har programvaror och bibliotek som finns fritt tillgängliga valts. Detta då en hög användarvänlighet bygger på att programmet finns tillgängligt för vem som helst, när som helst. I undervisningssammanhang är det vidare viktigt att skala bort ekonomiska aspekter då kunskapen ska finnas tillgänglig oberoende av ekonomiska förutsättningar. Existerande programvaror för liknande beräkningar och visualiseringar har ofta höga licenskostnader som gör dem ekonomiskt oförsvarbara för universitetet och definitivt för gemene man.

När man diskuterar användarvänlighet ser man vidare till att programmet ska vara uppbyggt på ett sådant sätt att användaren får lagom mycket information att processa i varje steg av programmet. Det här är anledningen till att användargränssnittet delats in i olika beräkningsområden. Den första fliken har en tydlig utformning för indata gällande geometri, geoteknik och lastfall, medan den andra fokuserar på tidsimuleringen. För att göra VisCon mer användarvänligt hade ytterligare indelningar kunnat göras. Vidare hade en direkt återkoppling av inmatning i användargränssnittet kunnat implementeras. Till exempel skulle modellens framställning i förhandsgranskningsfigurerna kunna ändras då en parameter ändras i indatan. Dock undveks detta för att ej göra programmet långsammare. I utformningsprocessen gäller det alltså att hitta en lämplig balans mellan att hålla programmet enkelt, för en känsla av att det är lättanvänt, och att implementera mer avancerade funktioner. I VisCon finns exempelvis möjligheten att förhandsgranska modellen men det kräver att användaren instruerar programmet att uppdatera figurerna. I flik två, där resultaten visualiseras, uppdateras figurerna momentant då tidreglaget ändras. Samma sak gäller då olika sektioner för de endimensionella graferna markeras. För att förbättra förståelsen av hur de endimensionella graferna plockas ut hade till exempel sektionspilar kunnat visas i den tvådimensionella modellen. Det hade även gjort programmet mer lättanvänt om man ej behövt förhandsdefiniera antalet sektioner för visualiseringen, men återigen har detta inte implementerats för att ej sakta ned visualiseringsprocessen.

För ett programs användarvänlighet är det vidare viktigt med goda hjälpfunktioner. Detta för att användaren lätt ska kunna ta sig ur felsituationer. I VisCon har varningsmeddelanden implementerats för orimliga värden som skulle generera ogiltiga beräkningar. I felmeddelandena anges krav på vilket värde den felaktigt inmatade parametern måste ha för att accepteras. Användaren ges utöver detta en fördefinierad beräkningsmodell för att från början få se exempel på rimliga värden för modellens parametrar.

Användaracceptans

För att öka användaracceptansen av VisCon är det av yttersta vikt att läraren lyckas inkludera programmet i sin utbildning på ett bra sätt. Som nämnts innan har programmet VisCon goda förutsättningar för detta då programmet har en hög anpassning och är utformat efter kursen som det ska användas i. För att höja den inre motivationen gäller det dock att verkligen få studenterna att känna att programmet hjälper dem i deras förståelse av konsolidering.

I utformningen av VisCon var målet först att ta fram ett verktyg som visualiserar tidsberoende sättningar. Målet kom dock att ändras under arbetets gång. Programmet utvecklades till att enbart visualisera porvattenövertryckets variation i tid och rum samt en endimensionell illustration av effektivspänningens variation. Att vidareutveckla programmet till att innefatta en koppling mellan beräknade spänningar och resulterande töjningar och sättningar hade varit ett enkelt steg eftersom programmet redan utvecklats till att plocka ut spänningsvärden längs en godtycklig linje. Dock utelämnades denna beräkning i ett pedagogisk syfte; VisCon ger inte användaren hela lösningen utan utifrån beräknade värden på porvattenövertrycket får denne själv koppla resultatet till sättningar. Istället för att få hela svaret beräknat av programmet behöver användaren inhämta resultat från programmet och koppla ihop med sättningsteori. För att lösa övningsuppgifterna som utvecklats till programmet VisCon krävs alltså fortfarande en förståelse för effektivspänningsekvationen och deformationsegenskaper, utan att för den sakens skull kräva någon kunskap inom numeriska lösningsmetoder. Detta då finita elementmetoden sällan är ett krav inom kurser i grundläggningsteknik. Förhoppningen är att denna utformning ska bidra till en ökad inre motivation och att programmet utgör en länk i inlärningen snarare än en fullständig lösning. Målet är att användaren inte bara ska inhämta faktakunskaper utan även utveckla djupare förståelsekunskaper i och med visualiseringen och de egna kopplingarna.

Användarkompetens

Vad det gäller användarkompetens har VisCon goda förutsättningar då programmet har en hög anpassning och är utvecklat kursspecifikt. Syftet med programmet är att,

utöver visualisering, assistera studenterna med tvådimensionella numeriska beräkningar som de ej förväntas kunna. I utformningen av VisCon var det därför viktigt att hålla val av indata enkelt för att användarna ska förstå samtliga inparametrar. I användargränssnittet tillåts fortfarande val av elementstorlek och tidsinkrement för att styra hur god tidsapproximation som erhålls. Anledningen till att dessa parametrar hålls öppna för användaren är att de är av vikt för förståelsen av modellens begränsningar. I övningsuppgifterna öppnas upp för en reflektion kring hur modellen beror av dessa.

Användargränssnittet för VisCon är tänkt att efterlikna andra program. Till exempel finns en meny där en ny modell kan öppnas, aktuell modell sparas, sparas som eller en redan sparad modell öppnas. Vidare har väldigt enkla knappar och reglage implementerats. Detta då det är lättare för användaren att ta till sig ett nytt program då det följer en bekant struktur. Att resultatet ges i form av endimensionella grafer är vidare för att studenterna ska känna igen resultatet från illustrationer i övrig undervisning. Förhoppningen är att studenterna ska koppla hur den tvådimensionella visualiseringen relaterar till den endimensionella för att, med hjälp av VisCon, förstå teorin på ett bättre sätt.

7.1.2 Tillämpning i undervisning

Undervisningsform

Att introducera ett datorprogram i undervisningen innebär oftast vissa praktiska svårigheter som måste hanteras. Till exempel förutsätter det övningstillfällen där studenterna har tillgång till både datorer och lärarhjälp. Ett möjligt sätt att inkludera VisCon i undervisningen är genom att introducera programmet på en föreläsning, där inlärningen sker genom observation. Detta kan därefter följas upp vid ett övningstillfälle i en datorsal där studenterna får lära sig via utforskning, med en assisterande lärare som finns tillgänglig för att svara på frågor som uppstår. Detta skulle ge studenterna en grundläggande förståelse för programmets uppbyggnad, följt av ett tillfälle av problembaserat lärande där de får lära sig genom egen problemformulering. Ett annat alternativ skulle vara att skapa en videoföreläsning som förklarar konceptet konsolidering samt hur VisCon kan tillämpas. Det skulle utgöra en form av observationsinlärning.

En risk då datorprogram inkluderas i undervisningssammanhang är att studenterna känner en låg motivation då programmet ej kan dyka upp vid ett examinationstillfälle. I längden är det viktigast att bygga ett program utifrån den inre motivationen men för att studenterna ska ge programmet en chans kan det ibland krävas en yttre motivation för att överhuvudtaget börja utforska programmet. Den yttre motivationen skulle kunna höjas genom att programmet inkluderas i en betygsatt inläm-

ningsuppgift. Den inre motivationen kan höjas genom att öka användaracceptansen, vilket diskuterades ovan.

VisCon syftar till en ökad förståelse för tvådimensionell konsolidering då ett behov av förbättrad undervisning inom detta område identifierats. Värt att diskutera är dock om ett visualiseringsverktyg är det som krävs, eller om det finns något annat sätt att förbättra förståelsen för konsolidering. Till exempel bör förbättringspotential i övrig undervisning samt kurslitteratur analyseras. I dagsläget är instruktionsvideor på nätet en vanlig källa till inlärning. I ett vidare arbete hade till exempel en sådan video som på ett illustrativt sätt utnyttjar visualisering av konceptet tillsammans med teori kunna förbättra studenternas förståelse inom området.

Utformning av övningsuppgifter

Att utveckla ett program som visualiserar någonting är en sak, att faktiskt ge det en tillämpning är en annan. För att assistera användningen av det utvecklade programmet har övningsuppgifter utformats. Detta för att studenterna faktiskt ska ta till sig vad programmet illustrerar och få en möjlighet att reflektera kring både resultat och beräkningsmodellens antaganden.

Bland de utvecklade uppgifterna får studenterna genomföra en tvådimensionell beräkning av en uppgift som de redan har löst med hjälp av en endimensionell modell. Tanken är att öppna upp för en diskussion kring hur korrekt den endimensionella modellen är i förhållande till den tvådimensionella utifrån ett givet lastfall. I en annan uppgift genomförs tvådimensionella beräkningar för en och samma uppgift men med en varierad bredd på den applicerade lasten. Detta för att vidare illustrera och öppna upp för en analys av när en tvådimensionell respektive en endimensionell analys är lämplig.

I sättningsberäkningar delas jordprofilen in i beräkningslager inom vilka en genomsnittlig spänning antas för hela lagertjockleken. Hur väl man lyckas fånga aktuell spänningsfördelning beror på hur fin lagerindelningen är. I övningsuppgift A2 öppnas upp för en diskussion kring hur många beräkningslager som behövs för att erhålla en konvergerande lösning. Den här kunskapen är viktig att ha med sig då en grov indelning snabbt ger en sämre approximation av verkligheten. Förhoppningen med denna uppgift är vidare att studenterna ska kunna studera de endimensionella graferna och utifrån dessa kunna avgöra hur många lager som behövs för olika delar av jordprofilen.

Andra faktorer som övningsuppgifterna behandlar är att illustrera hur porvattenövertrycket beror av grundens permeabilitet. Vidare finns uppgifter för att lyfta hur variationen ser ut längs med vertikala snitt på olika avstånd från centrumlinjen, där studenterna får skissa den deformerade modellen. Det är alltså här kopplingen till

den faktiska visualiseringen av de tvådimensionella tidsberoende sättningarna görs.

Uppgift A5 har som huvudsakligt syfte att få studenterna att reflektera kring de numeriska lösningarnas begränsningar. Programmet är beroende av tidsparametrar samt elementstorlek, och ger olika utdata beroende på hur dessa värden väljs. I utvecklingen av VisCon fanns initialt en tanke bakom att införa en begränsning av storleken på tidsinkrement. Dock har tidsinkrementets storlek lämnats fritt. Anledningen att tidssteg lämnades öppet är för att studenterna ska få en förståelse av möjliga felkällor samt att svaret är en approximation.

7.2 Implementering av VisCon

7.2.1 Utvecklingsmiljö

Utvecklingsmiljön som använts i utvecklingen av VisCon har fungerat bra. Rutinerna som återfinns i CALFEM för Python ligger till en god grund för det här arbetet då de möjliggör en smidig elementindelning, generering av elementmatriser, applicering av randvillkor och lösning av ekvationssystem. Är man bekant med funktioner i CALFEM för Matlab är det lätt att känna igen sig i CALFEM för Python då de utnyttjar liknande syntax och följer samma beräkningsstruktur. En faktor som skulle kunna ses över är dock numreringen i topologimatrisen edof. I numreringen av element startar både topologimatrisen i versionen för Matlab och Python på siffran 1. Det fyller en poäng att dessa matriser ser likadana ut, men samtidigt leder det även till en förvirring vid programmering i Python. Detta då Python är uppbyggt på ett annat sätt än Matlab med indexering som startar på siffran 0 istället för 1.

7.2.2 Nya rutiner

Eftersom alla funktioner som finns i CALFEM för Matlab inte finns i versionen för Python var vissa av dessa tvungna att implementeras i det här arbetet; flwtec och step. Funktionen som i det här arbetet namngivits flwtec är i stort sett hämtad direkt från CALFEM för Matlab, men där är den kombinerad med funktionen flw2te som bestämmer styvhetsmatrisen och lastvektorn. Funktionen step skiljer sig något mellan programmen då den implementerats efter önskemål från handledare i det här arbetet. Det som skiljer sig åt är hur man anger för vilka tidpunkter resultatet från tidsimuleringen ska sparas. I CALFEM för Matlab anges en separat vektor för dessa tidpunkter. I versionen av step implementerad i det här arbetet utgår funktionen från ett definierat antal tidpunkter, som sedan fördelas med jämna intervall över den totala simuleringstiden. Att ha en funktion som är uppbyggd på två olika sätt i

de olika versionerna av CALFEM kan vara något förvirrande och bör eventuellt ses över.

För funktionerna xlinextr och zlinextr har en metod implementerats där funktionsvärden och koordinater interpoleras för punkter där en godtycklig linje skär elementindelningen. I implementeringen av dessa funktioner diskuterades olika alternativ. Det alternativa förslaget var att redan vid definitionen av punkter, linjer och ytor i geometri-instansen definiera linjer för vilka nodvärden plockas ut till de endimensionella graferna. Denna implementering hade inneburit att man ej behövt interpolera värden. Med andra ord hade funktionerna xlinextr och zlinextr ej varit nödvändiga då nodvärden för sektionslinjerna kunnat plockas ut med hjälp av elementmarkeringar. Dock hade detta inneburit att linjerna hade behövt definieras på förhand. Om nya sektionslinjer önskats hade både nätgenerering, FEM-beräkningar och tidsimuleringen behövt utföras på nytt. För stora modeller med en hög noggrannhet hade detta inneburit en onödigt lång simuleringstid. Med xlinextr och ylinextr sköts interpoleringen av skärningspunkterna utanför övriga beräkningssteg. Man kan alltså ändra för vilka sektioner de endimensionella graferna visas utan att genomföra något annat än interpolering av redan genererade resultat.

Ytterligare en aspekt som diskuterats angående dessa funktioner är om interpoleringen ska ske direkt då slidern ändras eller ej. Detta hade inneburit att man ej på förhand behövt definiera antal sektionslinjer som slidern tillåter. Det hade reducerat antalet indata som användaren måste förhålla sig till, och resulterat i ett mer lättanvänt program. Dock hade det även inneburit ett långsammare program då interpoleringen skett i realtid. För att undvika detta gjordes extraheringen av värden till de endimensionella graferna ändå till en separat funktion, men där antalet sektionslinjer smidigt kan uppdateras utan att genomföra övriga beräkningssteg.

Kapitel 8

Avslutande kommentarer

8.1 Slutsatser

Det övergripande syftet med det här arbetet har varit att underlätta förståelsen för tvådimensionell konsolidering. Nedan presenteras några avslutande slutsatser utifrån målen som formulerades i avsnitt 1.2.

1. Den utvecklade programvaran VisCon beräknar och visualiserar porvattenövertrycket i två dimensioner. Vidare visualiseras porvattenövertrycket och effektivspänningen i en dimension för valfritt definierat snitt av modellen. Programmet har en god användbarhet ur aspekten anpassning då antalet funktioner skalats ned samt utformningen skett i samråd med ansvarig kurslärare. Användargränssnittet tar hänsyn till förkunskaper hos avsedda användare och har ett begränsat antal parametrar som går att variera. Felmeddelanden och indelning i olika beräkningsområden bidrar till programmets användarvänlighet. Utrymme för utveckling av en högre användbarhet finns bland annat genom att införa direkt återkoppling i gränssnittet.

2. Övningsmaterialet som utvecklats är uppbyggt för att studenterna med hjälp av VisCon ska kunna koppla porvattenövertryck till sättningar. Vidare är några uppgifter avsedda för att lyfta en diskussion kring beräkningsmodellens begränsningar. En avgörande faktor för hur datorn fungerar som ett pedagogiskt verktyg är hur väl läraren lyckas inkludera programvaran i undervisningen. Eftersom övningsuppgifter utvecklats för att direkt kunna appliceras i undervisningssammanhang har VisCon goda förutsättningar för just detta.

3. För att möjliggöra den här typen av visualiseringsverktyg krävs ett antal tillägg av rutiner i CALFEM för Python. Rutinerna som har utvecklats i det här arbetet beskrivs i kapitel 6.3. Bland dessa finns en rutin som beräknar

kapacitansmatrisen för ett tvådimensionellt trenodselement och en annan som beräknar initial spänningsfördelning för definierade koordinater. Vidare finns en rutin som baserat på globala matriser, initial spänningsfördelning och tidparametrar tillhandahåller en approximation av porvattenövertryckets variation i tid och rum. Två rutiner har även utvecklats för att få fram koordinater och porvattenövertryck längs valfri sektion av den tvådimensionella FE-modellen.

4. CALFEM möjliggör en enkel lösning av finita elementproblem. Det som tidigare saknat implementering i CALFEM för Python är möjligheten att lösa partiella differentialekvationer. I utvecklingen av programvaror så som VisCon är således Python-versionen ej fullständig. Med de nya rutinerna från det här arbetet och implementeringen av funktioner från Matlab-versionen till Python-versionen är dock utvecklingen av den här typen av programvaror möjlig.

För att slutligen vidga perspektivet bör nämnas att en potentiell risk med användning av programvaror, både i undervisning och arbetsliv, är att användarna ej förstår implementerad teori. Då den teoretiska kunskapen fattas saknas möjligheten att kontrollera beräkningar och genomföra enkla rimlighetsanalyser. Det här har blivit ett allt vanligare fenomen och är både hämmande för kunskapsutvecklingen och kan leda till stora risker om programmens tillförlitlighet aldrig ifrågasätts. Samtidigt behövs avancerade program för att möjliggöra omfattande beräkningar snabbt och ekonomiskt. Det handlar om att finna en lagom balans mellan vad datorn respektive individen beräknar och drar slutsatser kring. Kopplat till VisCon har denna avvägning gjorts genom att studenterna får beräknade värden för porvattenövertrycket, men själva måste koppla dessa till resulterande sättningar. På så vis behåller programmet en koppling till teorin som kräver att användaren har denna kunskap.

8.2 Vidare arbete

I ett vidare arbete skulle användargränssnitt, implementerad beräkningsmodell och funktioner i programmet VisCon kunna vidareutvecklas. Ett förslag på sådant arbete skulle vara att utöver den tvådimensionella modellen även implementera en endimensionell och en tredimensionell beräkningsmodell. Detta skulle ge användaren möjligheten att direkt i programmet experimentera och undersöka effekten av vald modell. Eftersom styvhets- och lastmatriser för både ett endimensionellt och tredimensionellt flöde utvecklats i CALFEM skulle det här arbetet bestå av att vidareutveckla de nya rutinerna i det här arbetet till att omfatta en respektive tre dimensioner. I ett vidare arbete hade man även kunnat införa ett enkelt sätt att genomföra och redovisa resultat av parameterstudier i VisCon.

I det här arbetet grundas den initiala spänningsfördelningen på analytiska uttryck.

I ett vidare arbete skulle denna fördelning kunna bestämmas med hjälp av finita elementmetoden. Fler förslag på utvecklingar är även att utveckla beräkningarna till att omfatta effekten av kapillär stigning, en varierad grundvattennivå samt en modell där jorden ej är fullt vattenmättad. Eftersom övningsuppgifter utvecklats som ett komplement till VisCon skulle det vara av intresse att analysera huruvida programmet i kombination med dessa bidrar till förståelsen av tvådimensionell konsolidering.

Litteraturförteckning

[1] *Numpy*. http://www.numpy.org/. [Hämtad: 2017-06-01].

[2] *PyQt*. http://pyqt.sourceforge.net/Docs/PyQt5/introduction.html. [Hämtad: 2017-06-01].

[3] *Python version 3.5.3*. https://docs.python.org/3/faq/general.html. [Hämtad: 2017-06-01].

[4] *Visvis*. https://pypi.python.org/pypi/visvis/1.8. [Hämtad: 2017-06-01].

[5] Ågren, Per Olof: *It i högre utbildning: rationalisering eller pedagogik?* Konferensbidrag: IT i universitetsundervisning, Umeå universitet, Umeå, Sverige.

[6] Austrell, Per Erik, Dahlblom, Ola, Lindemann, Jonas, Olsson, Anders, Olsson, Karl-Gunnar, Persson, Kent, Petersson, Hans, Ristinmaa, Matti, Sandberg, Göran och Wernberg, Per-Anders: *CALFEM-a finite element toolbox, version 3.4*. Studentlitteratur, Lund, Sverige, 2004.

[7] Avén, Sigurd: *Handboken Bygg G, Geoteknik*. Liber Förlag, Stockholm, Sverige, 1984.

[8] Craig, Robert F: *Craig's soil mechanics*. CRC Press, London, England, 2004.

[9] Edholm, Andreas: *Meshing and visualisation routines in the Python version of CALFEM*. Examensarbete, TVSM-5187, Byggnadsmekanik, Lunds universitet, Lund, Sverige, 2013.

[10] Farkell-Bååthe, Sonja: *Datorn som pedagogiskt hjälpmedel: effekter och erfarenheter av datorstöd i matematik*. Doktorsavhandling, Institutionen för individ, omvärld och lärande, Lärarhögskolan i Stockholm, Stockholm, Sverige, 2000.

[11] Larsson, Rolf: *Jords egenskaper*. Statens geotekniska institut, 2008.

[12] Nationalencyklopedin: *Datorisering*. http://www.ne.se/uppslagsverk-/encyklopedi/lång/datorisering. [Hämtad: 2017-06-01].

[13] Nationalencyklopedin: *Datorstödd undervisning.* http://www.ne.se-/uppslagsverk/encyklopedi/lång/datorstödd-undervisning. [Hämtad: 2017-06-01].

[14] Ottosen, Niels Saabye och Petersson, Hans: *Introduction to the Finite Element Method.* Prentice-Hall, New York, USA, 1992.

[15] Ottosson, Andreas: *Implementation of CALFEM for python.* Examensarbete, TVSM-5167, Byggnadsmekanik, Lunds universitet, Lund, Sverige, 2010.

[16] Persson, Kent: *Transient heat flow.* http://www.byggmek.lth.se/fileadmin-/byggnadsmekanik/education/courses/optional/VSMN25/Transient.pdf. [Hämtad: 2017-06-01].

[17] Potts, David M och Zdravković, Lidija: *Finite Element Analysis in Geotechnical Engineering: Theory.* Thomas Telford, London, England, 1999.

[18] Sällfors, G och Andréasson, L: *Geotekniska laboratorieanvisningar: Kompressionsegenskaper.* Statens råd för byggnadsforskning: Svensk byggtjänst, Volym 10.

[19] Statens geotekniska institut: *Jords hållfasthet.* http://www.swedgeo.se/-sv/kunskapscentrum/om-geoteknik-och-miljogeoteknik/geoteknik-och-markmiljo/jords-hallfasthet/skjuvhallfasthet/. [Hämtad: 2017-06-01].

[20] Terzaghi, Karl: *Theoretical soil mechanics.* John Wiley & Sons, New York, USA, 1943.

[21] Tudisco, Erika och Dahlblom, Ola: *Course literature in Foundation Engineering VGTN01.* LTH, Lunds universitet, Lund, Sverige, 2017.

Bilaga A

VisCon - Manual

A.1 Background

A.1.1 Aim with program

VisCon was developed in order to facilitate the understanding of two dimensional consolidation. The program visualises how the excess pore water pressure varies with time. It also allows the user to extract one dimensional plots of both excess pore water pressure and effective stress at any defined time step. These plots can be extracted vertically or horizontally at any defined section of the model. The program is intended to be used as a complement to ordinary course literature in order to enhance the understanding of consolidation in two dimensions.

A.1.2 Consolidation

When a ground surface is subjected to a load an increased level of stress will occur in the soil underneath. As soil material has a limited permeability, an increased stress will initially result in an excess pore water pressure. However, as time passes the water will flow and the pore water pressure will even out. The relation between the total stress $\Delta\sigma_z$, the excess porewater pressure u_e and the effective stress $\Delta\sigma'_z$ is given by the effective stress equation:

$$\Delta\sigma_z = u_e + \Delta\sigma'_z \tag{A.1}$$

The total change in stress will always equal the sum of the excess pore water pressure and the effective stress. When the pore water pressure reaches its static condition,

i.e. $u_e = 0$ the soil is considered consolidated and the total change in stress equals the effective stress, $\Delta\sigma_z = \Delta\sigma'_z$.

For further theoretical background the user is directed to the master's dissertation in which this program was developed. Non-Swedish-speaking users are directed to the following literature:

- Terzaghi, Karl: *Theoretical soil mechanics*. John Wiley & Sons, New York, 1943.
- Craig, Robert F: *Craig's soil mechanics*. CRC Press, London, 2004
- Tudisco, Erika och Dahlblom, Ola: *Course literature in Foundation Engineering*. LTH, Lund University, Lund, 2017.

A.1.3 Problem statement

Assumptions and limitations

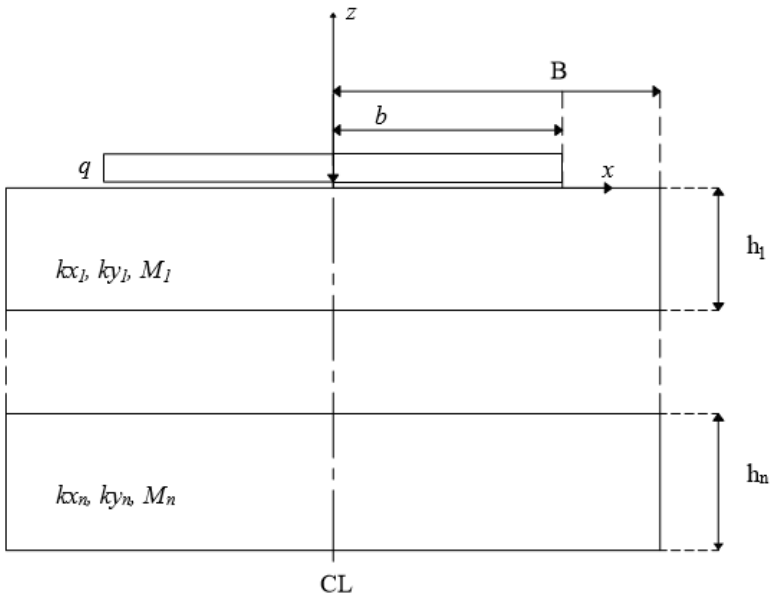
This program facilitates a numerical solution of the two dimensional consolidation equation. As the program is limited to solve two dimensional problems, it can only be applied when the groundwater flow is limited to the xz-plane. In other words, the program solves problems with a load of which the length is greater than the width. Within civil engineering such an application could typically be modelling a soil profile underneath a road embankment. However, it could also give an accurate approximation of the consolidation underneath a long building. This version of the program is limited to a symetrical case with uniformly distributed load.

In solving the differential equation of consolidation, a number of assumptions are needed. The ones used in this program are listed below:

- The soil is fully saturated.
- Darcy's law is applicable.
- The pore water, as well as the solid grains, are considered incompressible at a material level.
- The process of consolidation is dependent only on the delay of pore water flow described by Darcy's law.

Model

The model used in the program is shown in Figure A.1. The program allows the user to analyse a soil profile with an unlimited number of soil layers. The user can adjust the depth and soil characteristics within each layer. Furthermore, the user is allowed to adjust the magnitude and width of the load, as well as the width of the soil examined.



Figur A.1: *The model for which the program solves the differential equation of consolidation.*

A.1.4 Implementation

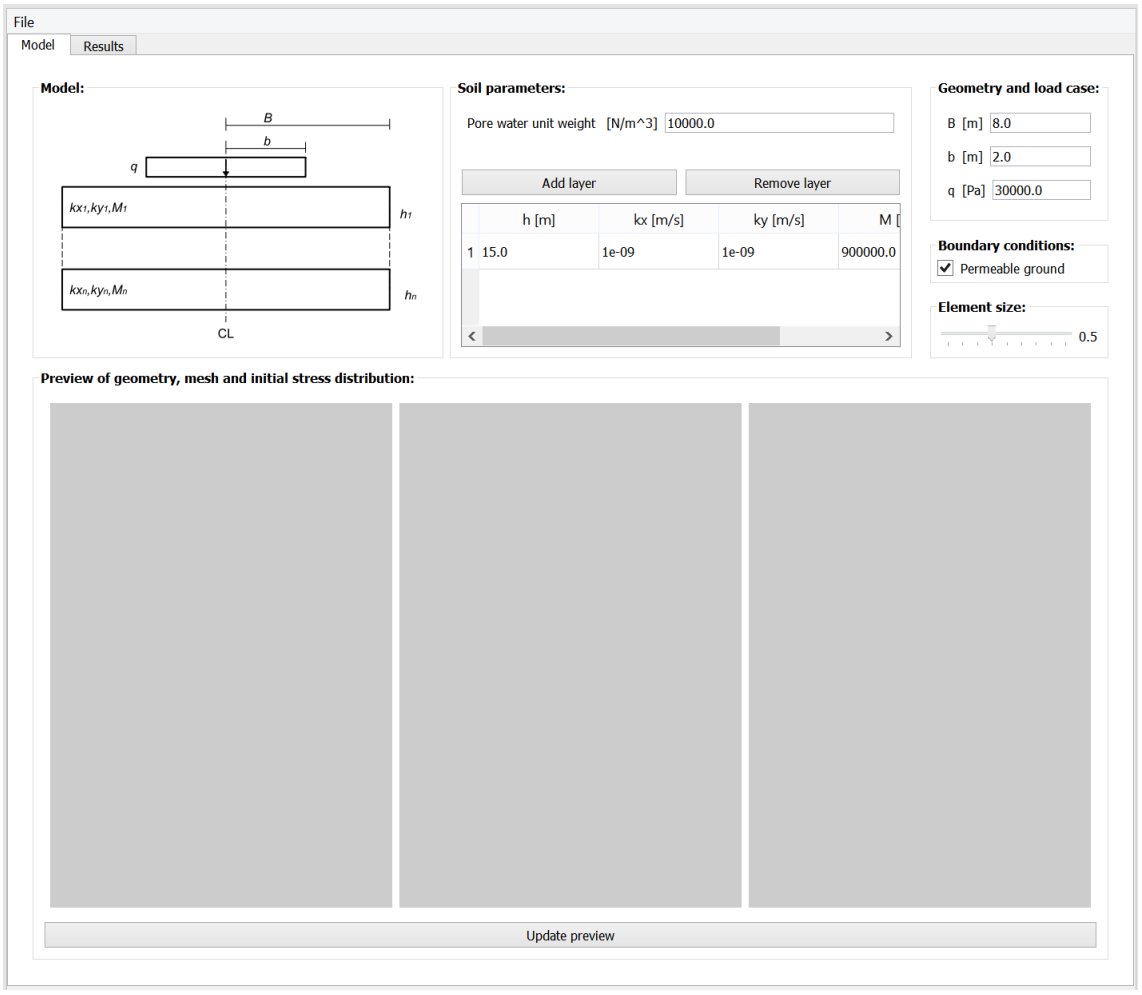
The visualisation tool is developed in Python with the use of VisVis, PyQt, Numpy and CALFEM for Python. It uses the finite element method in solving the differential equation decribing the process of consolidation. It is a numerical solution involving a time approximation according to the backward difference method.

A.2 How to use VisCon

A.2.1 Tab 1: Model

Overview

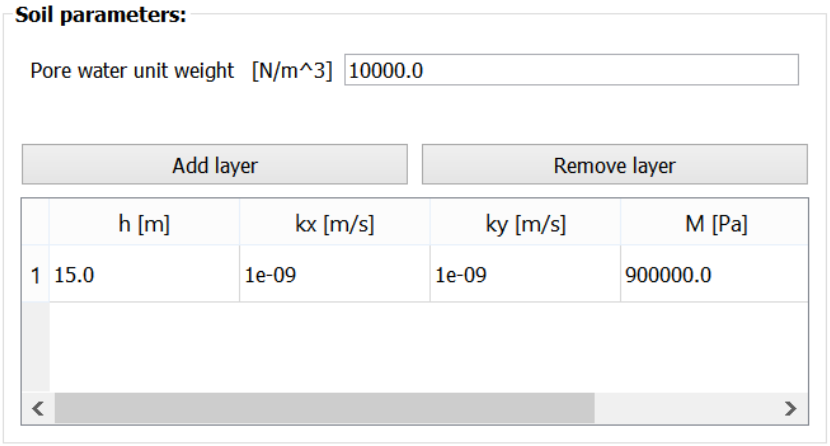
When the program is initiated the user interface shown in Figure A.2 will appear. The controls are updated with a pre-defined model.



Figur A.2: *The first interface of the Main Window, intended for defining the model.*

Soil parameters

In section Soil parameters, shown in Figure A.3, the user is allowed to enter desired soil parameters. The first field is intended for the pore water unit weight, γ_w, given by $\gamma_w = \rho_w g$ [N/m^3]. The pore water density, ρ_w, varies with the level of dissolved salts in the water, but is normally given by $\rho_w = 1000$ kg/m^3. The gravitational acceleration, g, is given by $g = 9.81$ m/s^2, i.e. the unit weigth is $\gamma_w = 1000 \cdot 9.81 = 9810$ N/m^3, or slightly higher if there is dissolved salts. Often $\gamma_w = 10\ 000$ N/m^3 can be assumed.



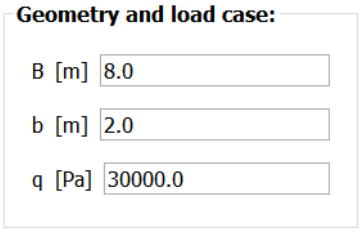
Figur A.3: *Section to assign parameters for pore water and soil profile.*

Next field in Soil parameters is intended for the soil profile. The user may adjust the number of soil layers by adding or removing layers to the table. For each layer it is possible to adjust depth h [m], permeability in x-direction k_x [m/s], permeability in z-direction k_z, and compression modulus M [Pa].

Geometry and load case

In section Geometry and load case, shown in figure A.4, it is possible to determine the geometry and load case. The model is symmetrical and the dimensions given are with reference to the centre line shown in Figure A.1.

B [m] is the width of the soil examined. It is important to choose a width of the soil considerably larger than the width of the load. The reason behind this is that the boundary condition implemented in this program assumes the flow perpendicular to the line opposite to the centre line to be zero. This boundary could also be represented by an impermeable structure surrounding the soil profile. b [m] is the width of the load in the z-direction. The load is assumend to be uniformly distributed

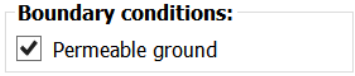


Figur A.4: *Section to assign geometry and load case.*

and infinite in the y-direction. The magnitude of the load is adjusted by the value of q [Pa].

Boundary conditions

The boundary conditions determine along which surfaces the pore water pressure should be zero at $t > 0$. The boundary conditions for the top surface are pre-defined and assumed to be zero, as water is free to flow perpendicular to its direction. It is not possible to change this condition in the program.



Figur A.5: *Section to assign boundary conditions.*

In section Boundary conditions, shown in Figure A.5, it is possible to choose if the bottom surface should be permeable or not. If the check box is ticked the bottom surface is permeable, allowing water to flow perpendicular to its direction.

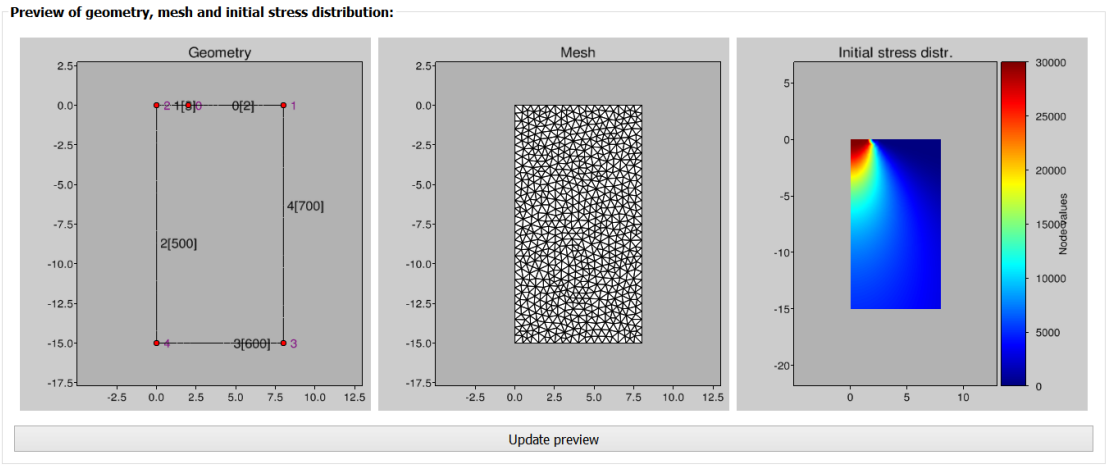
Element size

In section Element size it is possible to adjust the element size by controlling the slider, shown in Figure A.6. It is important to have a fine mesh since it results in a more precise solution. However, a fine mesh also results in large matrices and subsequently more time consuming calculations.

Figur A.6: *Section to control element size.*

Preview

In section Preview, shown in Figure A.7, it is possible to preview geometry, mesh and initial stress distrubution based on given input data. In order to update the preview, if any values have been changed, press the button *Update preview*.

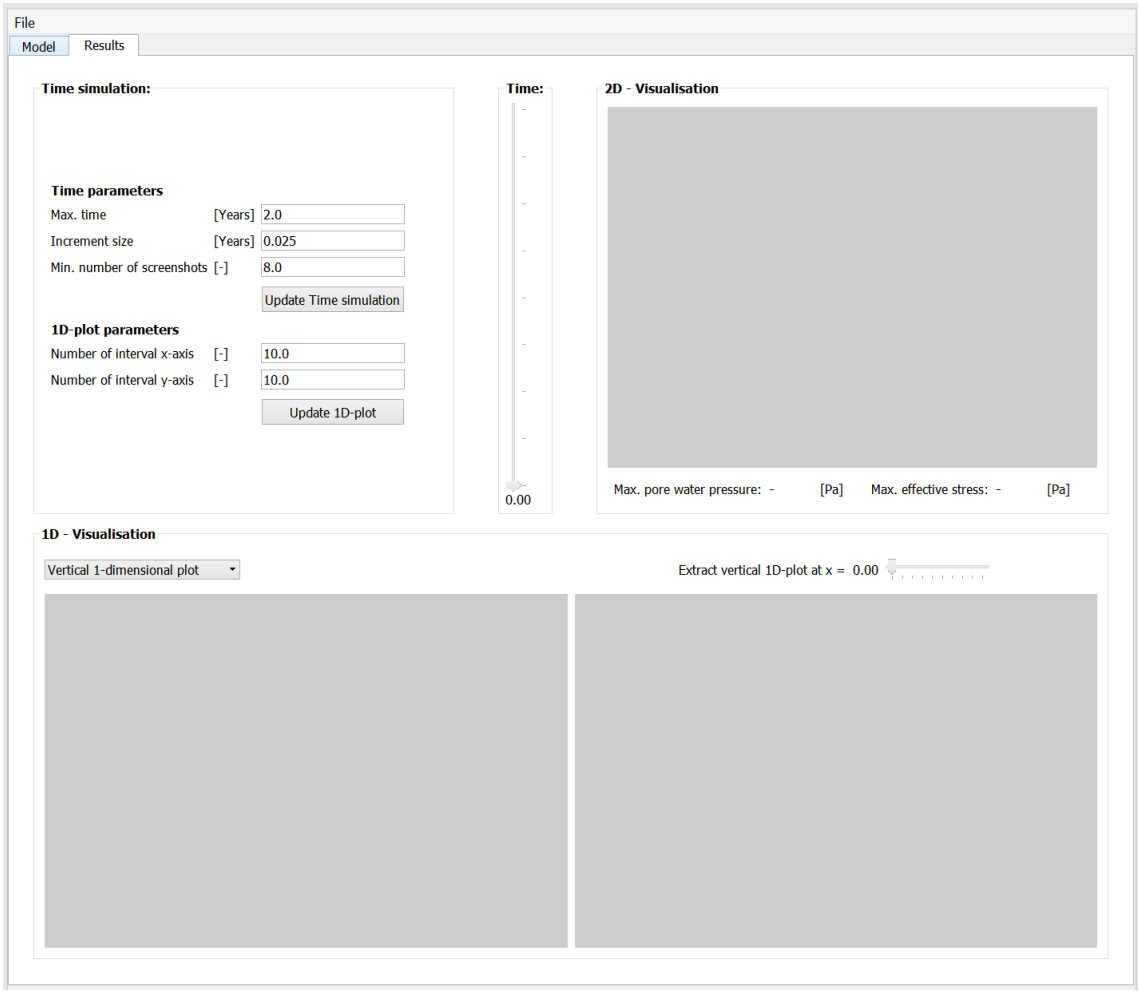


Figur A.7: *Preview based on given values for the model.*

A.2.2 Tab 2: Results

Overview

When the user clicks on the second tab, Results, the program will enter calculation mode. It will perform the meshing, generate the finite element matrices and perform a time simulation based on pre-defined values. Following, the user interface shown in Figure A.8 will appear.



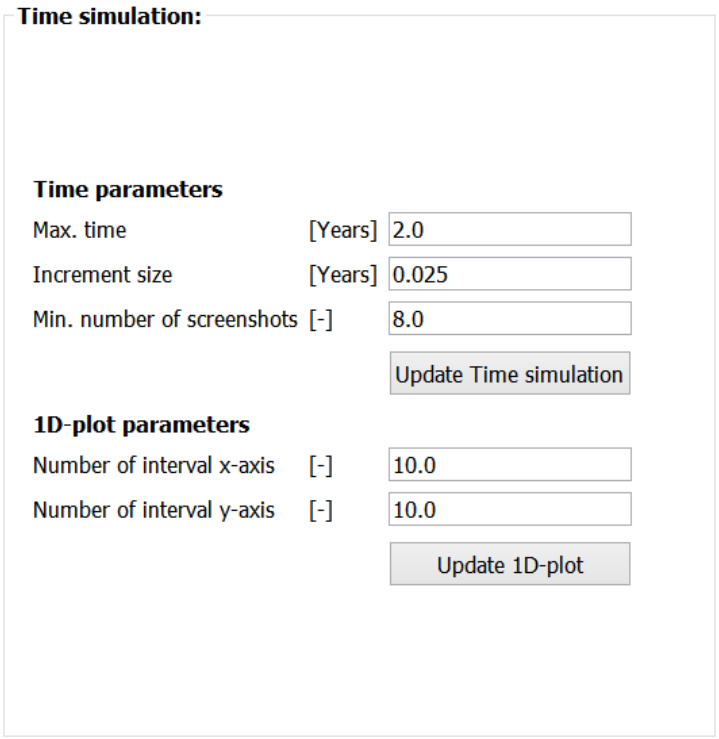
Figur A.8: *The second interface of the Main Window, intended for defining parameters for the time simulation as well as visualising the results.*

Time simulation

It is possible to control the time simulation by varying the values in section Time simulation, shown in Figure A.9. In the first part of this section the time parameters

are given. It is possible to decide how long the time simulation will be as well as the increment size of each time step. It is important not to choose a too large increment since it may result in a too rough time approximation. However, a small timestep will result in more time consuming calculations.

Furthermore, it is possible to choose the minimum number of screenshots for the time simulation. The program will store and plot values of the excess pore water pressure for this number of time steps. The time steps will be equally distributed over the simulated time .



Figur A.9: *Section to enter values for the time simulation as well as the visualisation.*

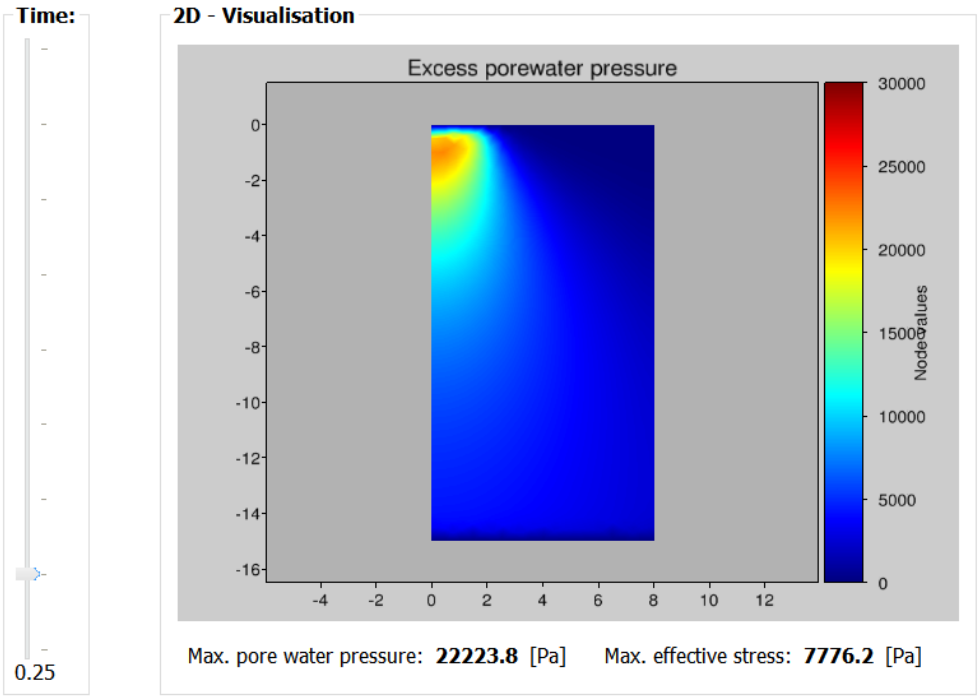
In the second part of this section it is possible to choose how many x- and z-values the program should extract one dimensional plots for. Based on the number the program will generate as many one dimensional plots, equally distributed over the x- and z-axis. The number connected to x will generate vertical one dimensional plots and the number for z will generate horizontal one dimensional plots.

Time

With this control it is possible to vary for which time step the current two and one dimensional visualisations are shown.

2D - Visualisation

In this section, shown in Figure A.10, the results of the transient analysis of the excess pore water pressure will be shown in two dimensions. The two dimensional figure shows the excess pore water pressure, plotted with a colour bar visible to the right side. Next to this figure a table gives the maximum excess pore water pressure and the maxium effective stress. According to the effective stress equation the sum of these will equal the total stress at all time steps.

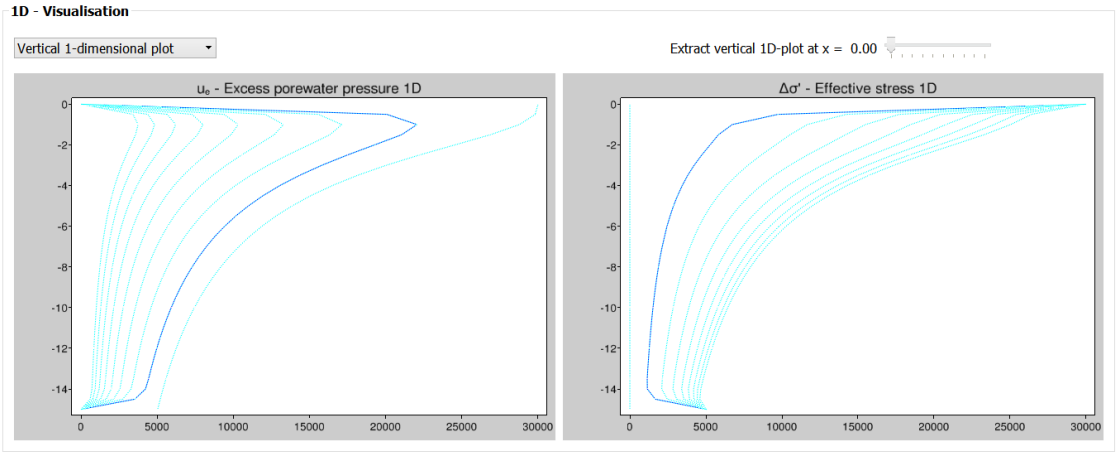


Figur A.10: *A two dimensional visualisation of the excess pore water pressure at a defined time step.*

The two dimensional colour plot and the maximum values will change accordingly to the current time step, defined in section Time.

1D - Visualisation

In the section for one dimensional visualisation, shown in Figure A.11 two one dimensional plots are extracted from the calculated excess pore water distrubution. It is possible to decide if the one dimensional plot should be a vertical or horizontal section in the drop-down menu. By the slider it is possible to choose for which x- or z-value the section should be plotted.



Figur A.11: *Extracted one dimensional plots, showing excess pore water pressure and effective stress at a defined section of the ground model.*

The first figure shows the excess pore water pressure at a chosen section, and the second figure shows the effective stress. A number of graphs will be visible, representing all defined plot time steps. The current time step is highlighted and it changes accordingly to the time step defined in section Time.

A.2.3 File mananger

New

Open a new file, with pre-defined values of the ground model as well as the time simulation.

Save

Save the current model.

Save as

Save the current model, giving it a new file name.

Open

Import an already existing model.

Exit

Exit the program.

Bilaga B

VisCon - Course material

B.1 How to apply Viscon

B.1.1 Method

1. Define model:

The initial step is to update VisCon with model data for the current problem, i.e. insert values for given soil profile and load case. Furthermore, when defining the model it is important to choose an element and time increment size small enough to achieve a converging solution.

2. Extract 1D-plot:

Next step is to define the maximum time for the time simulation, and make sure that the program record values of the excess pore water pressure at desired time steps. This is done by choosing a time increment dividable with desired length of time steps, and a number of screenshots catching these, see the example in B.1.2. Furthermore, the user should define intervals for the vertical and horisontal axis where one dimensional plots of the excess pore water pressure should be extracted.

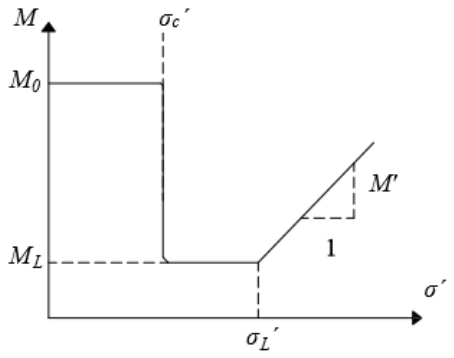
3. Calculate settlements:

The process of consolidation results in settlements. Settlements depend on the effective stress, i.e. the pressure carried by the soil skeleton. The total settlement δ_z

can be calculated by integrating the strain function over the depth at a certain section, according to:

$$\delta_z = \int_0^h \varepsilon_z dz \tag{B.1}$$

The strain ε_z depends on the consolidation level and is described by either Equation B.2, B.3 or B.4. $\sigma'_{z,0}$ is the effective stress before an increased stress and $\sigma'_{z,1}$ is the effective stress after an increased stress, i.e. $\sigma'_{z,1} = \sigma'_{z,0} + \Delta\sigma'_z$. Figure B.1 shows the relation between the compression modulus M and the effective stress σ'. σ'_c is referred to as the consolidation stress and σ'_L is the effective stress for which the compression modulus M is no longer constant.



Figur B.1: *Compression modulus M as a function of the effective stress σ' for clay.*

If $\sigma'_{z,1} \leq \sigma'_c$ the strain is given by:

$$\varepsilon_z = \frac{\Delta\sigma'_z}{M_0} \tag{B.2}$$

If $\sigma'_c < \sigma'_{z,1} \leq \sigma'_L$ the strain is given by:

$$\varepsilon_z = \frac{\sigma'_c - \sigma'_{z,0}}{M_0} + \frac{\sigma'_{z,1} - \sigma'_c}{M_L} \tag{B.3}$$

If $\sigma'_{z,1} > \sigma'_L$ the strain is given by:

$$\varepsilon_z = \frac{\sigma'_c - \sigma'_{z,0}}{M_0} + \frac{\sigma'_L - \sigma'_c}{M_L} + \frac{1}{M'} \ln\left(1 + (\sigma'_{z,1} - \sigma'_L)\frac{M'}{M_L}\right) \tag{B.4}$$

When calculating the total settlement it is common to divide the soil into a number of layers. For each layer the average effective stress is calculated and multiplied by the layer depth. If the soil profile is divided into n layers the total settlement can be expressed as a sum, according to:

$$\delta_z = \sum_{i=1}^{n} \varepsilon_{z,i} h_i \tag{B.5}$$

B.1.2 Example

Problem

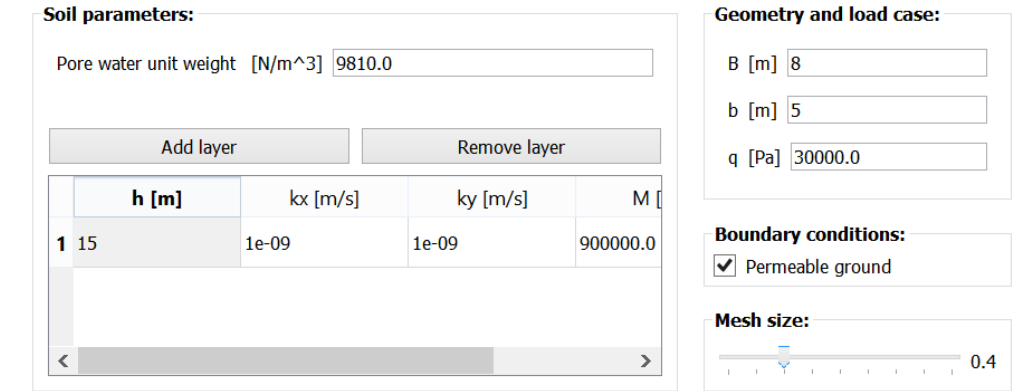
Consider a 15 m thick layer of clay resting on sand. The clay material has the permeability $k_x = k_z = 10^{-9}$ m/s and the compression modulus $M_L = 900$ kPa. Furthermore $\sigma'_{z,0} = \sigma'_c$. A road embankment is built on top of this layer. It has a width of 8 meters and is estimated to contribute with an increased load of 30 kPa. Calculate the settlement after 2 years and after a long time for a vertical section in the centre line as well as a vertical section 6 meter from the centre line. Do this by dividing the clay into five calculation layers.

1. Define model:

A 15 m layer is created in VisCon. For this layer the permeability is set to $k_x = k_z = 10^{-9}$ m/s and the compression modulus $M_L = 900$ kPa. The width of the load, with reference to the centre line, is $b = 5$ m. The width of the model is set to $B = 8$ m. The boundary conditions are set by ticking the box Permeable ground as the sand layer underneath the clay is considered permeable. See Figure B.2 for model input data. The element size is choosen as 0.5 and the time increment as $\Delta t = 0.05$ years.

2. Extract 1D -plot:

In order to catch the time step $t = 2$ years the maximum time is set to $T = 5$ years and the number of screenshots taken is set to 5. This will produce a a slider where the two dimensional and one dimensional visualisations can be shown for $t = 0$, 1, 2, 3, 4, 5 years. The value for the number of intervals along the x-axis is set to 5 in order to be able to extract a vertical 1D-plot at $x = 0$, 2, 4, 6, 8 m.



Figur B.2: *Input data when defining the model in example exercise.*

Tabell B.1: Computed effective stress $\Delta\sigma'_z$ after 2 years.

Depth z	$x = 0$ m	$x = 6$ m
1.5 m	25.6 kPa	4.2 kPa
4.5 m	22.0 kPa	9.2 kPa
7.5 m	17.3 kPa	9.7 kPa
10.5 m	13.7 kPa	9.3 kPa
13.5 m	11.0 kPa	8.5 kPa

When the time simulation is done the one dimensional plots will be available, by choosing values of the current time step and the x-coordinate for a selected section. The first assignment is to determine the settlements at $t = 2$ years for a vertical section at $x = 0$ m and $x = 6$ m. The time slider is set at 2 and the x-value slider at $x = 0$ m respective $x = 6$ m. The figure showing the effective stress is used and Table B.1 shows the values for $\Delta\sigma'_z$ after 2 years. It is possible to zoom and pan in the figure in order to determine values easier.

As the excess pore water pressure is assumed to be zero after a long time the effective stress then equals the initial stress distrubution $\Delta\sigma_z$. Hence, the values for the effective stress after a long time is given by studying the initial excess pore water pressure. The first figure is studied for a section at $x = 0$ m and $x = 6$ m. The values are given in Table B.2.

3. Calculate settlements:

As $\sigma'_{z,0} = \sigma'_c$ Equation B.2, in combination with Equation B.5, can be simplified and written as:

Tabell B.2: Computed effective stress $\Delta\sigma'_z$ after a long time.

Depth z	$x = 0$ m	$x = 6$ m
1.5 m	29.6 kPa	5.0 kPa
4.5 m	25.5 kPa	10.5 kPa
7.5 m	20.0 kPa	11.2 kPa
10.5 m	16.0 kPa	10.7 kPa
13.5 m	13.0 kPa	9.8 kPa

$$\delta = \sum_{i=1}^{n} \frac{\Delta\sigma'_{z,i}}{M_L} h_i \tag{B.6}$$

For $t = 2$ years and $x = 0$ m the following settlement is calculated:

$$\delta_{t=2,x=0} = \frac{25.6}{900}3.0 + \frac{22.0}{900}3.0 + \frac{17.3}{900}3.0 + \frac{13.7}{900}3.0 + \frac{11.0}{900}3.0 = 0.30 \text{ m}$$

For $t = 2$ years and $x = 6$ m the following settlement is calculated:

$$\delta_{t=2,x=6} = \frac{4.2}{900}3.0 + \frac{9.2}{900}3.0 + \frac{9.7}{900}3.0 + \frac{9.3}{900}3.0 + \frac{8.5}{900}3.0 = 0.14 \text{ m}$$

For $t = \infty$ years and $x = 0$ m the following settlement is calculated:

$$\delta_{t=\infty,x=0} = \frac{29.6}{900}3.0 + \frac{25.5}{900}3.0 + \frac{20.0}{900}3.0 + \frac{16.0}{900}3.0 + \frac{13.0}{900}3.0 = 0.35 \text{ m}$$

For $t = \infty$ years and $x = 6$ m the following settlement is calculated:

$$\delta_{t=\infty,x=6} = \frac{5.0}{900}3.0 + \frac{10.5}{900}3.0 + \frac{11.2}{900}3.0 + \frac{10.7}{900}3.0 + \frac{9.8}{900}3.0 = 0.16 \text{ m}$$

B.2 Exercises

A1

Consider a 15 m deep layer of clay, resting on sand. The clay has the material parameters $\sigma'_{z,0} = \sigma'_c$, $k_x = k_z = 5{\cdot}10^{-9}$ m/s and $M_L = 600$ kPa. A road embankment with a width of 14 m causes a surface load of 40 kPa.

a) Calculate the excess pore water pressure at the centre line after 5 years.

b) Calculate the total settlement after 5 years and after a long time. What is the avarage degree of consolidation after 5 years?

c) Compare the results in a) - b) with the results in Exercise 5-3 in *Course literature in Foundation Engineering VGTN01 2017* [21]. Make a comment upon the differences.

A2

Consider the model defined in Exercise A1. Calculate the settlements using $n = 1$, 2, 3 and 5 calculation layers. Plot the relationship between the number of layers and the calculated settlements. Make a comment upon when the solution convergens.

A3

Consider the model defined in Exercise A1, but loaded with a road embankment with the width of 2 m.

a) At the centre line, calculate the total settlement after 5 years. Compare the result with the results in Exercise A1 b) and Exercise 5-3 b) in *Course literature in Foundation Engineering VGTN01 2017* [21].

b) Is a one dimensional model appropriate for the load case described in this exercise?

c) Which are the limitations of a one dimensional versus a two dimensional model when calculating settlements?

A4

Consider the model defined in exercise A1 but with the clay layer resting on solid rock instead of sand.

a) Calculate the settlement after 5 years for a vertical section at $x = 0$ m, $x = 4$ m and $x = 8$ m.

b) Compare the result for $x = 0$ m with the result in Exercise A1 b).

c) Make a sketch of the deformed model.

A5

Try a model in VisCon with various values of time increment and element size. How will these parameters affect the solution? Furthermore, the flow, q, perpendicular to the right boundary is modelled as zero. Vary the width of the model, and discuss under which conditions this implementation is valid.

Bilaga C

Källkod: Nya rutiner

C.1 flwtec

```
def flwtec(ex, ey, rho, M):

    exm = np.asmatrix(ex)
    eym = np.asmatrix(ey)
    ci = np.hstack((np.ones([3,1]), np.transpose(exm),
        np.transpose(eym)))
    A = 0.5*np.linalg.det(ci)
    ce = np.matrix([[1/6, 1/12, 1/12],
                    [1/12, 1/6, 1/12],
                    [1/12, 1/12, 1/6]])
    Ce=rho/float(M)*A*ce
    return Ce
```

C.2 strdst

```
def strdist(coords, ndof, q, B, b, hvec):

    sigmaz = np.zeros([ndof,1])
    for i in range(ndof):
        x = coords[i, 0]
        y = coords[i, 1]
        A1 = x + b
        B1 = x - b
        C1 = - y
```

```
        if C1 != 0:
            beta = np.arctan(B1/C1)
            alpha = np.arctan(A1/C1)-beta
            sigmaz[i] = q/np.pi*(alpha+np.sin(alpha)*
                np.cos(alpha+2*beta))
        else:
            if x < b:
                beta = 2*np.pi
                alpha = 0
                sigmaz[i]= q
            else:
                beta = 2*np.pi
                alpha = 0
                sigmaz[i]= 0
    return sigmaz
```

C.3 step

```
def step(K, C, a, bc, bcval, dt, T, Tnr):

    n = int(T/dt)
    ndt = np.floor(T/dt/Tnr)
    Tvec = np.arange(ndt, n, ndt)
    Tvec = np.append(Tvec, n)

    theta = 1
    avec = a

    Kt = K
    Ct = C



    for i in range (n):

        Kt = Ct + dt*Kt
        f=(Ct+(1-theta)*Kt)*a
        a, r = cfc.solveq(Kt, f, bc)
        if (i+1) in Tvec:
            avec=np.hstack((avec, a))
    return avec, Tvec
```

C.4 xlinextr

```
def xlinextr(xl, surfx, ex,ey, ndof, edof,avec):

    xls = xl

    if xls == 0:
        xls = 0.01
    elif xls in surfx:
        xls = xls - 0.01


    nel = int(np.shape(edof)[0])
    elnr = []


    for i in range(nel):
        if np.min(ex[i,:]) < xls  < np.max(ex[i,:]):
            #Spara elementnummer
            elnr = np.append(elnr, i)


    nelc = int(np.shape(elnr)[0])
    nstep = int(np.shape(avec)[1])


    exdof = np.zeros([nelc,3])
    eydof = np.zeros([nelc,3])
    eu1dof = np.zeros([nelc,nstep])
    eu2dof = np.zeros([nelc,nstep])
    eu3dof = np.zeros([nelc,nstep])

    a = 0
    for i in elnr:

        exdof[a,:] = ex[i,:]
        eydof[a,:] = ey[i,:]


        dofs = edof[i,:]-1

        eu1dof[a,:] = avec[dofs[0],:]
        eu2dof[a,:] = avec[dofs[1],:]
```

```
        eu3dof[a,:] = avec[dofs[2],:]
        a = a +1


uvec = np.zeros([1,nstep])
yvec = []
for elex, eley, eleu1, eleu2, eleu3 in
    zip(exdof, eydof, eu1dof, eu2dof, eu3dof):
    eu = np.vstack((eleu1,eleu2,eleu3))

    exv = []
    eyv = []
    euv = np.zeros([1,nstep])


    exh = []
    eyh = []
    euh = np.zeros([1,nstep])


    for i in range(3):

        if elex[i] < xls:
            exv = np.append(exv, elex[i])
            eyv = np.append(eyv, eley[i])
            euv = np.vstack((euv, eu[i,:]))

        else:
            exh = np.append(exh, elex[i])
            eyh = np.append(eyh, eley[i])
            euh = np.vstack((euh, eu[i,:]))


    euv = euv[1::,:]
    euh = euh[1::,:]



    if np.shape(exv)[0] == 1:


        xv = exv[0]
        yv = eyv[0]
        uv = euv[0,:]
```

```
            for i in range(2):
                xh = exh[i]
                yh = eyh[i]
                uh = euh[i,:]

                y = yv + (yh - yv) * (xl - xv) / (xh - xv)
                u = uv + (uh - uv) * (xl - xv) / (xh - xv)

                yvec = np.append(yvec, y)
                uvec = np.vstack((uvec, u))

        elif np.shape(exv)[0] == 2:

            xh = exh[0]
            yh = eyh[0]
            uh = euh[0,:]



            for i in range(2):
                xv = exv[i]
                yv = eyv[i]
                uv = euv[i,:]

                y = yv + (yh - yv) * (xl - xv) / (xh - xv)
                u = uv + (uh - uv) * (xl - xv) / (xh - xv)


                yvec = np.append(yvec, y)
                uvec = np.vstack((uvec, u))

    uvec = uvec[1::,:]
    return yvec, uvec
```

C.5 zlinextr

```
def ylinextr(yl, surfy, ex, ey, ndof, edof,avec):

    yl = -yl #Bytt tecken vid byte x->y
```

```
yls = yl

if yls ==0:
    yls = - 0.001

elif -yls in surfy:
    yls = yls + 0.001

nel = int(np.shape(edof)[0])
elnr = []


for i in range(nel):
    if np.min(ey[i,:]) < yls  < np.max(ey[i,:]):
        #Spara elementnummer
        elnr = np.append(elnr, i)

nelc = int(np.shape(elnr)[0])
nstep = int(np.shape(avec)[1])


exdof = np.zeros([nelc,3])
eydof = np.zeros([nelc,3])
eu1dof = np.zeros([nelc,nstep])
eu2dof = np.zeros([nelc,nstep])
eu3dof = np.zeros([nelc,nstep])

a = 0
for i in elnr:

    exdof[a,:] = ex[i,:]
    eydof[a,:] = ey[i,:]


    dofs = edof[i,:]-1

    eu1dof[a,:] = avec[dofs[0],:]
    eu2dof[a,:] = avec[dofs[1],:]
    eu3dof[a,:] = avec[dofs[2],:]
    a = a +1


uvec = np.zeros([1,nstep])
xvec = []
```

```
for elex, eley, eleu1, eleu2, eleu3 in
    zip(exdof, eydof, eu1dof, eu2dof, eu3dof):
    eu = np.vstack((eleu1,eleu2,eleu3))

    exv = []
    eyv = []
    euv = np.zeros([1,nstep])


    exh = []
    eyh = []
    euh = np.zeros([1,nstep])


    for i in range(3):


        if eley[i] > yls:  #Bytt olikhet vid x->y
            exv = np.append(exv, elex[i])
            eyv = np.append(eyv, eley[i])
            euv = np.vstack((euv, eu[i,:]))

        else:
            exh = np.append(exh, elex[i])
            eyh = np.append(eyh, eley[i])
            euh = np.vstack((euh, eu[i,:]))


    euv = euv[1::,:]
    euh = euh[1::,:]


    if np.shape(exv)[0] == 1:


        xv = exv[0]
        yv = eyv[0]
        uv = euv[0,:]


        for i in range(2):
            xh = exh[i]
            yh = eyh[i]
            uh = euh[i,:]
```

```
            x = xv + (xh - xv) * (yl - yv) / (yh - yv)
            u = uv + (uh - uv) * (yl - yv) / (yh - yv)

            xvec = np.append(xvec, x)
            uvec = np.vstack((uvec, u))

    elif np.shape(exv)[0] == 2:

        xh = exh[0]
        yh = eyh[0]
        uh = euh[0,:]



        for i in range(2):
            xv = exv[i]
            yv = eyv[i]
            uv = euv[i,:]

            x = xv + (xh - xv) * (yl - yv) / (yh - yv)
            u = uv + (uh - uv) * (yl - yv) / (yh - yv)

            xvec = np.append(xvec, x)
            uvec = np.vstack((uvec, u))

uvec = uvec[1::,:]
return xvec, uvec
```