

Numerisk modellering av elastisk deformasjon av sprekker

Tor Erik Bergum Wollmann

Masteroppgave i anvendt matematikk

Matematisk-Naturvitenskapelig fakultet, Universitetet i Bergen



Juni 3, 2024

Annerkjennelser

Jeg vil takke veileder min, Ivar Stefansson. Uten hans visjon for prosjektet, gode forklaringer, kodehjelp og forståelse av både PorePy og bergmekanikk ville ikke denne oppgaven blitt noe av.

Jeg vil også takke medlemmene av PorePy brukergruppen på UiB. Takk til Omar, Ingrid, Yura, Eirik og alle andre som har hjulpet meg med å forstå PorePy, numpy, og alle småtingene som har gitt meg problemer i kodingen.

Abstrakt

Bakgrunn: Sprekker i stein introduserer kompleksiteter under deformasjon og dette er et område der det fortsatt er mye vi ikke forstår. På UiB blir det kontinuerlig utviklet et verktøy kalt PorePy som kan kjøre komplekse numeriske simuleringer for å prøve å øke forståelsen rundt slike komplekse systemer.

Metode: I denne oppgaven introduserer vi et skille mellom elastisk og plastisk deformasjon som tidligere ikke eksisterer i PorePy. Vi lager matematiske modeller og implementerer disse ved å kode dem i Python som en utvidet del av PorePy rammeverket. Vi lager også en testsuite som kan automatisk verifisere resultatene våre.

Resultat: Det er blitt implementert et skille mellom elastisk og plastisk deformasjon og introdusert tangentiell elastisk forskyvning, for både 2 dimensjonale og 3 dimensjonale mekaniske simuleringer i PorePy.

Konklusjon: In-situ vil det være et elastisk-plastisk skille i deformasjonen som skjer i stein. Den modifiserte modellen vil derfor være et steg mot mer realistiske simuleringer. I fremtiden, hvis man vil ha enda mer realistiske simuleringer der man tar hensyn til destruktiv deformasjon, vil det være naturlig å bygge på et slikt elastisk-plastisk skille som det nå er uproblematisk å legge til i PorePy.

Innhold

Annerkjennelser.....	2
Abstrakt.....	3
1 Introduksjon.....	7
1.1 Bidrag.....	7
1.2 Disposisjon.....	8
2 Bakgrunnsteori.....	9
2.1 Bergmekanikk.....	9
2.2 Deformasjon i stein.....	10
2.2.1 Elastisitet.....	11
2.2.2 Friksjon.....	11
2.3 Materialegenskaper.....	12
2.3.1 Stivhet.....	13
2.3.2 Styrke.....	13
2.3.3 Ruhet.....	14
2.4 Ordliste.....	14
3 Modell.....	16
3.1 Geometri.....	16
3.2 Formulering av modell.....	17
3.3 Randbetingelser.....	20
4 Implementering.....	21
4.1 PorePy.....	21
4.2 Modifisering.....	23
5 Resultater.....	25
5.1 Testing.....	25
5.1.1 2 dimensjoner.....	26
5.1.2 3 dimensjoner.....	28
5.1.3 Testresultater.....	29
5.2 Kildekode.....	29
6 Diskusjon og konklusjoner.....	31
Bibliografi.....	32

Symbolliste

σ	Spenning
E	Elastisk modul, materialkonstant, relatert til stivhet
ϵ	Strekk
F	Kraft
μ	Friksjonskoeffisient, materialkonstant
N	Normalkraft
ϕ	Friksjonsvinkel, materialkonstant
UCS	«Uniaxial Compressive Strength», materialkonstant, relater til styrke
JCS	«Joint Compressive Strength», en materialkonstant, relatert til styrke
JRC	«Joint Roughness Coefficient», materialkonstant, relatert til ruhet
u	Forskyvning
n	Normalvektor
λ	Trekk
K	Modellens stivhetsparameter
b	Friksjonsgrense
ζ	Positiv, reel skalar
g	Åpningsfunksjon, beskriver hvordan en sprekk i kontakt vil åpne seg
θ	Dilatasjonsvinkel, beskriver hvor mye materialet øker/minker i volum ved skjæring
a	Sprekkåpning, distanse mellom sprekkoverflatene
Ω	Domene, PorePy notasjon
Γ	Grensesnitt, ligger mellom domene
Π	Operator, tilsvarer en projeksjon fra domene til grensesnitt
Ξ	Operator, tilsvarer en projeksjon fra grensesnitt til domene

1 Introduksjon

Bergmekanikk som forskningsfelt har ikke vært en distinkt vitenskapelig disiplin mer enn 60 år (Jaeger, Cook, & Zimmerman, 2007), men har fått økende viktighet i moderne tid etter hvert som vi begynner mer og mer komplekse, ambisiøse prosjekt i naturen rundt oss.

Deformasjonen av stein med sprekker i er et område der det er mye vi ikke forstår, samtidig som at det har stor innflytelse over moderne prosjekter som oljeboring eller karbondioksidinjeksjon. I denne oppgaven vil vi se på Python rammeverket utviklet av forskningsgruppen for porøse medier ved UiB for å øke for forståelse av de komplekse dynamiske prosessene gjennom numeriske simuleringer, og de matematiske modellene som ligger bak. Dette rammeverket er i aktiv utvikling som en del av et forskningsprosjekt, og målet vårt med denne oppgaven blir å introdusere både nye samt å modifisere eksisterende modeller til en av simuleringsmodulene i dette rammeverket for å få mer realistisk data ut ifra simuleringene dette brukes til.

«It has been claimed – correctly – that rock masses are the single most complex of engineering materials utilized by man.” - (Barton, 2013, p. 5)

1.1 Bidrag

Vi har laget en oppdatert modell for PorePy som deler forskyvning i sprekken opp i en plastisk og en elastisk del, basert på en lignende oppdeling presentert av White (2014). Den matematiske modellen er blitt kodet inn i funksjoner som kan implementeres i PorePy, og det er laget en tilhørende testsuite i Pytest. Det er planer om å implementer en full versjon av modellen presentert av White (2014) i PorePy og bidraget til denne oppgaven vil være første steget i denne prosessen, gitt at den blir innlemmet som en del av PorePy rammeverket.

1.2 Disposisjon

Kapittel 1 inneholder motivasjonen bak oppgaven og oppsettet av oppgaveartikkelen samt artikkelens bidrag.

Kapittel 2 er ment å gi en introduksjon bergmekanikken som ligger bak modellene som settes opp i senere kapitler. Det vil snakke om deformasjon i stein og hvordan dette kan regnes ut ved hjelp av å sette opp en Hookes lov og hvordan dette blir påvirket av friksjon gjennom en Coulomb-type friksjonslov og hvilke materialparameter vi må tenkt på når vi setter opp modellene våre senere. Til slutt her er det en ordliste der man kan se hvilke engelske fagbegrep som gir opphav til de norske ordene brukt i oppgaven.

Kapittel 3 presenterer hvordan de matematiske modellene er satt opp i PorePy, samt hvordan forskjellige geometrier på problemer blir behandlet i PorePy. Her introduseres også de nye ligningene som er laget for å introdusere plastisk og elastisk deformasjon.

Kapittel 4 handler om hvordan strukturen ser ut på kodesiden av PorePy og hvordan vi har implementert den modifiserte matematiske modellen inn i denne strukturen.

Kapittel 5 inneholder testene som er brukt for å verifisere koden, resultatene av disse. Her er det også linker til GitHub-depotet der koden ligger, forklaringer av dette depotet og instruksjoner for hvordan man kan reprodusere resultatene presentert.

Kapittel 6 vil inneholde en diskusjon av resultatene presentert i kapittel 5, noen av manglene med måten dette er gjort på, samt peke ut hvordan dette kan brukes som byggestein for videre utvikling av PorePy som en realistisk simulator av deformasjon av stein.

2 Bakgrunnsteori

Dette kapittelet er ment å introdusere bergmekanikk og stein som materiale for å gi en kontekst om modellene som er brukt senere. Delene av dette kapittelet som handler om bergmekanikk er sterkt basert på (Jaeger, Cook, & Zimmerman, 2007) sin bok «Fundamentals of Rock Mechanics».

Seksjon 2.1 handler om hva som gjør bergmekanikk spesielt relativt til annen mekanikk, samt hvilke moment man må tenke på hvis man vil gjøre beregninger på stein.

I 2.2 vil det så bli diskutert hvordan deformasjon i stein arter seg, og hvilke moment man må tenke på når man skal modellere dette.

2.3 vil deretter handle om materialparameterne som introduseres i seksjon 2.2 og hvor de vanligvis kommer fra.

Fordi bergmekanikk som fagfelt hovedsakelig er på engelsk, og de fleste av kildene i oppgaven er på engelsk, er det ofte ikke veldefinerte norske oversettelser av de engelske fagbegrepene. Derfor vil det i seksjon 2.4 være en ordliste der fagbegrepene som brukes i dette og senere kapittel relateres til de engelske begrepene.

2.1 Bergmekanikk

Bergmekanikk er i utgangspunktet en videreføring av de fundamentale begrepene i fasthetslære, men med spesifikt stein som medium. Det betyr at man som i fasthetslære snakker om kreftene som virker på materialet i form av en stresstensor, disse spenningene fører til at det oppstår trekraft som igjen fører til en deformasjon i form av strekk eller forflytning.

Når det er snakk om stein som medium er det en del egenskaper som vil spesielt påvirke hvordan lovene fra fasthetslæren oppfører seg, og en av de viktigste er tilstedeværelsen av sprekker i stein (Jaeger, Cook, & Zimmerman, 2007).



BILDE 1: ET EKSEMPEL DER MAN TYDELIG KAN SE SPREKKER OG DISKONTINUITETER I EN STEIN.

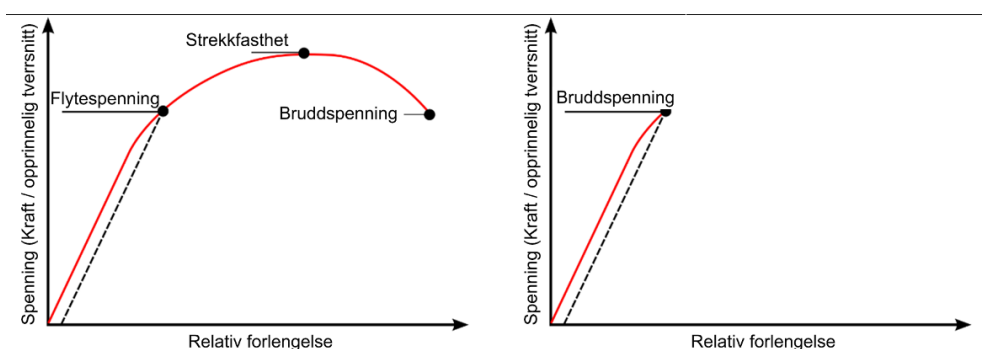
BILDE HENTET FRA [HTTPS://COMMONS.WIKIMEDIA.ORG/WIKI/FILE:CRACKS_AT_SUNRISE-ON-SEA,_EASTERN_CAPE.JPG](https://commons.wikimedia.org/wiki/File:Cracks_at_sunrise-on-sea,_eastern_cape.jpg)

Disse sprekke er diskontinuiteter i materialet som oppstår mens materialet deformeres under spenning. Man kan dele opp disse diskontinuitetene etter størrelse, retning og om det har forekommet bevegelse på tvers av diskontinuiteten (Palmström, 2002), men det finnes ikke noen standardisert passende oversettelse for fagbegrepene som brukes til denne differensieringen, med ett spesifikt unntak. Når det er forekommet bevegelse på tvers av sprekkeretning er det referert til som en forkastning (forekommer bl.a. ved jordskjelv), mens alt annet kalles bare for «sprekker» (for denne avhandlingen sin del er det heller ikke relevant å skille etter størrelse).

Det at stein er fylt av slike sprekker gir dem egenskaper som påvirker hvordan materialet oppfører seg når det påføres en spenning (Jaeger, Cook, & Zimmerman, 2007).

2.2 Deformasjon i stein

Materialer under spenning vil undergå elastisk deformasjon til det nås en flytespenning. Etter nådd flytespenning vil det som kalles duktile materialer oppleve plastisk, permanent deformasjon til det oppnås en svikt ved bruddspenning (Swallowe, 1999). En av egenskapene til stein er at det er et skjørt materiale, det betyr at bruddspenning og flytespenning inntreffer samtidig (eller tilnærmet samtidig) (Jaeger, Cook, & Zimmerman, 2007). Det betyr at steiner ikke vil oppleve et duktilt område, og vil ikke kunne permanent deformeres uten at det oppnår svikt og vil heller deformeres destruktivt.

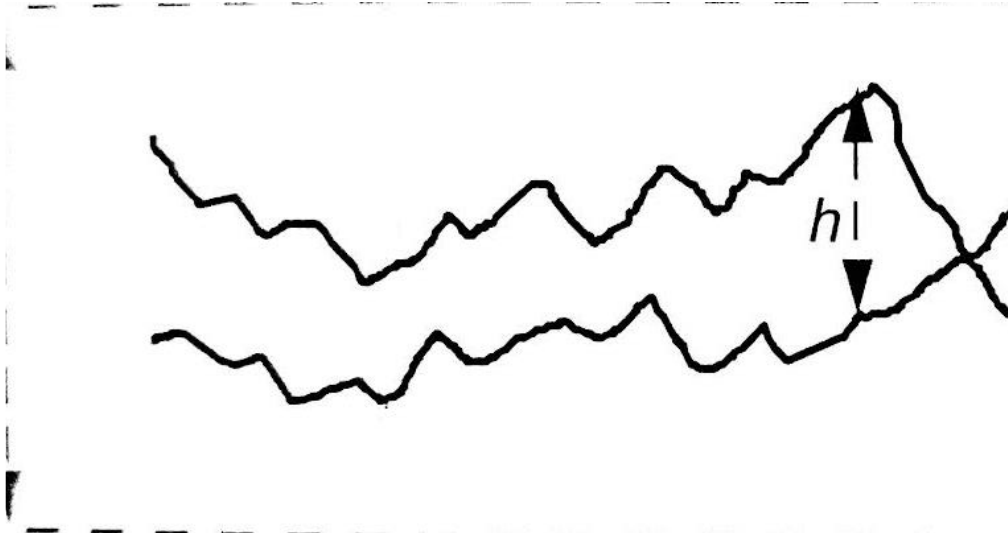


BILDE 2: EKSEMPEL PÅ SPENNING-FORLENGELSE KURVE FOR DUKTILE MATERIALER (VENSTRE) OG STEIN (HØYRE)

BILDE HENTET FRA [HTTPS://MEDIA.SNL.NO/MEDIA/151095/STANDARD_FORLENGELSE.PNG](https://media.snl.no/media/151095/STANDARD_FORLENGELSE.PNG)

Generelt vil dette være sant for stein, og for denne oppgavens del vil vi anta at dette stemmer, men for helhetens skyld kan det merkes at noen steiner vil oppføre seg duktilt hvis det holdes under høyt trykk (ofte $> 100\text{MPa}$) eller under høye temperaturer (dette gjelder færre steiner, men f.eks. noen kalkstein opptrer duktilt ved 500°C (Jaeger, Cook, & Zimmerman, 2007)).

For å modellere stein under spenning må vi altså ha en konstitutiv lov for elastisitet, men i tillegg trenger vi å tenke på selve sprekke. Sprekkene har nemlig den egenskapen at de kan være åpne, dvs. at det ikke er kontakt mellom sprekkeoverflatene, eller lukket, altså at sprekkeoverflatene er i kontakt (på Bilde 3 under kan man se at avstanden h mellom sprekkeoverflatene går mot 0 i noen punkter). Derfor må vi også ha en konstitutiv lov for friksjon der sprekken er lukket.



BILDE 3: EN SPREKK SOM ER DELVIS LUKKET. VI KAN SE AT I NOEN PUNKT VIL SPREKKHØYDEN h GÅ TIL 0.

BILDET HENTET FRA (JAEGER, COOK, & ZIMMERMAN, 2007)

2.2.1 Elastisitet

Elastisitet er altså det vi kaller forholdet mellom strekket og spenningen over steinen. Det er flere måter å relatere dette forholdet (f.eks. porøelastisk der sprekker og porer i stein kan være fylt med fluid som vil påvirke elastisiteten) der den enkleste er et lineært elastisk forhold slik som sees på Bilde 2. Når man definerer dette som et lineært forhold kalles det for Hookes lov, og hvis man ser på strekk og stress i samme retning vil den se slik ut:

$$\sigma = E \cdot \epsilon. \quad (1)$$

Her er σ stress og ϵ er strekk, eller relativ deformasjon av materialet. E er en materialparameter kalt elastisk modul, som handler om hvor stivt materialet er (stivere materiale vil bøye seg mindre) og representerer hvor bratt kurven i Bilde 2 er. Hva denne parameteren blir kalt varier med hvordan loven er formulert, men det relevante her er at det er en materialparameter (Jaeger, Cook, & Zimmerman, 2007).

2.2.2 Friksjon

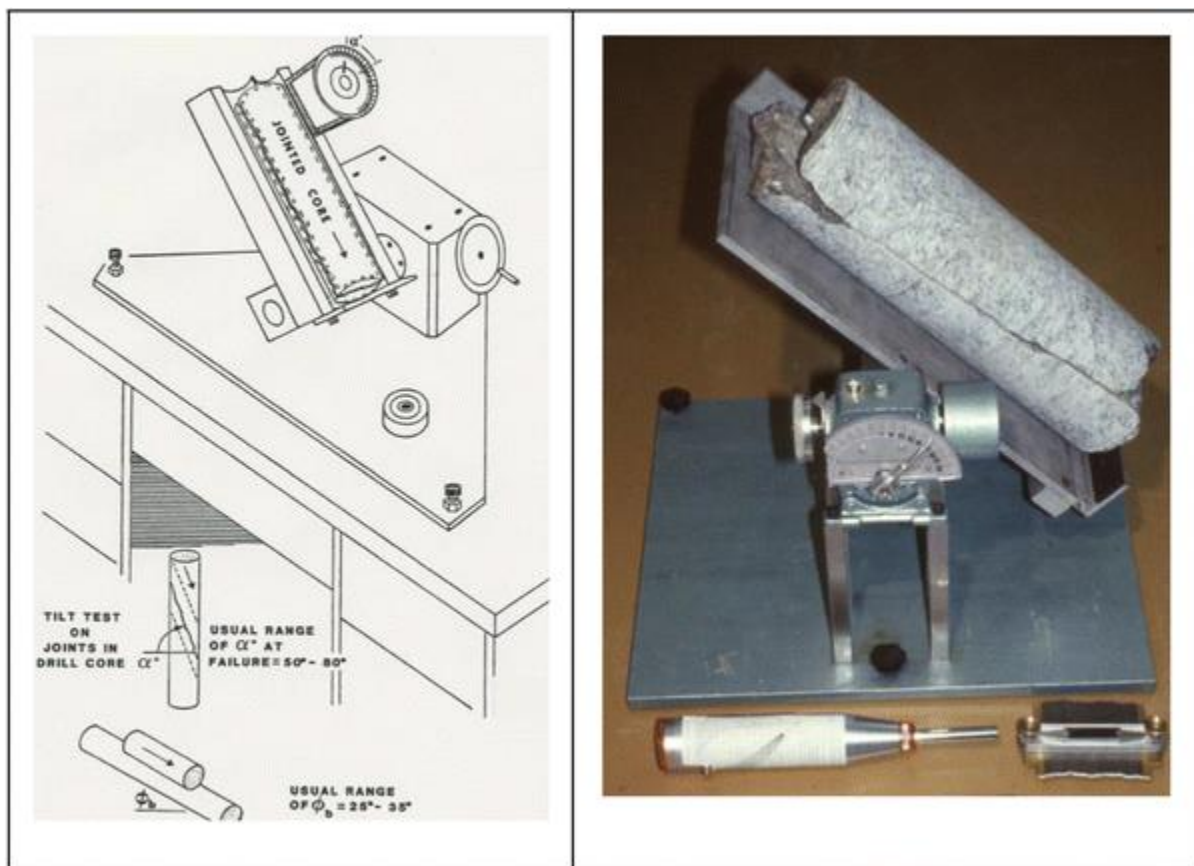
Hvis sprekken inne i steinen er i lukket konfigurasjon betyr det at overflatene som gnisser mot hverandre vil oppleve friksjon som vil virke i motsatt retning av bevegelsesretningen. Det finnes flere måter å definere friksjonen, en veldig vanlig en er det som kalles en Coulomb type friksjonslov (Jaeger, Cook, & Zimmerman, 2007):

$$F = \mu N. \quad (2)$$

Denne typen lov sier at friksjonskraften F henger sammen normalkraften N ved friksjonsparameteren μ . For stein er det vanligste å uttrykke friksjonskoeffisienten som en vinkel (Barton, 2021) ved å bytte friksjonskoeffisienten (μ) med en friksjonsvinkel (ϕ):

$$\mu = \tan \phi. \quad (3)$$

Man bruker en vinkel fordi dette er en empirisk parameter som man finner ved hjelp av en tilte-test (se Bilde 4). Friksjonsvinkelen tilsvarer da den vinkelen der tyngdekraften overskrider friksjonskraften i steinprøven og det forekommer deformasjon. Når man ser på steinprøver med sprekker i seg vil denne vinkelen være påvirket av ruheten til sprekken. Det vil si at denne nye friksjonsvinkelparameteren vil bestå av både av en materiell komponent (ϕ_g , grunnleggende friksjonsvinkel (Barton & Choubey, 1977)) samt en ruhetskomponent som er individuell til sprekken.



BILDE 4: VENSTRE: SKISSE AV ET TILTE-TEST OPPSETT, HØYRE: BILDE AV ET TILTE-TEST OPPSETT. EN SCHMIDT-HAMMER KAN SEES VED SIDEN AV OPPSETTET. BILDET HENTET FRA [HTTPS://DOI.ORG/10.1007/S41062-016-0011-1](https://doi.org/10.1007/s41062-016-0011-1)

2.3 Materialelegenskaper

Selv om man kan snakke om stein som en gruppe materiale som deler relativt like egenskaper (f.eks. tilstedeværelsen av sprekker gjennom materialet, eller en stress-strekk kurve som ligner den i Bilde 2), har det også en del egenskaper som vil variere ut ifra hvilken stein man snakker om. For sprekkeformasjoner er det hovedsakelig stivhet og ruhet som vil påvirke hvordan deformasjonen arter seg (Lei & Barton, 2022).

2.3.1 Stivhet

Et materiales stivhet er et mål på hvor elastisk materialet opptrer (hvis man ser på kurven i Bilde 2) vil lavere stigningstall tilsvare høyere elastisitet). Antar man et lineært forhold slik som er gjort i ligning (1) vil dette være en form for modul som blir funnet empirisk (gjerne med et oppsett lignende Bilde 5). Formen på modulen (og Hookes loven) vil være avhengig av hvordan spenning blir påført og hvordan den relative forflytningen i materialet utartes, men de er alltid empiriske materialparametere som blir funnet ved at man regner ut stigningstallet på spenning-forlengelses kurven (Bilde 2) (Jaeger, Cook, & Zimmerman, 2007).

2.3.2 Styrke

For å se på deformasjon i materialet må man også snakke om materialets styrke. Styrken er et mål på hvor mye strekk steinen kan påføres før den når flytespenning, dvs. for materialer som ikke er duktile vil styrken påvirke hvor mye spenning materialet tåler før det svikter (Lei & Barton, 2022). For stein bruker man hovedsakelig to forskjellige styrker for materialet, nemlig UCS («Uniaxial compressive strenght», hvor mye spenning en sylinder kan motstå før den begynner å kollapse(Bilde 5)) og JCS («Joint compressive strenght», stivheten til sprekkeoverflatene, vanligvis målt med en Schmidt-hammer slik som sees i Bilde 4 (Lei & Barton, 2022)) (Goel & Singh, 2011). I nye, ferske sprekker, vil disse verdiene være tilnærmet like (Goel & Singh, 2011), men etter hvert som sprekken forvitrer vil JCS falle til mye lavere verdier (Jaeger, Cook, & Zimmerman, 2007).

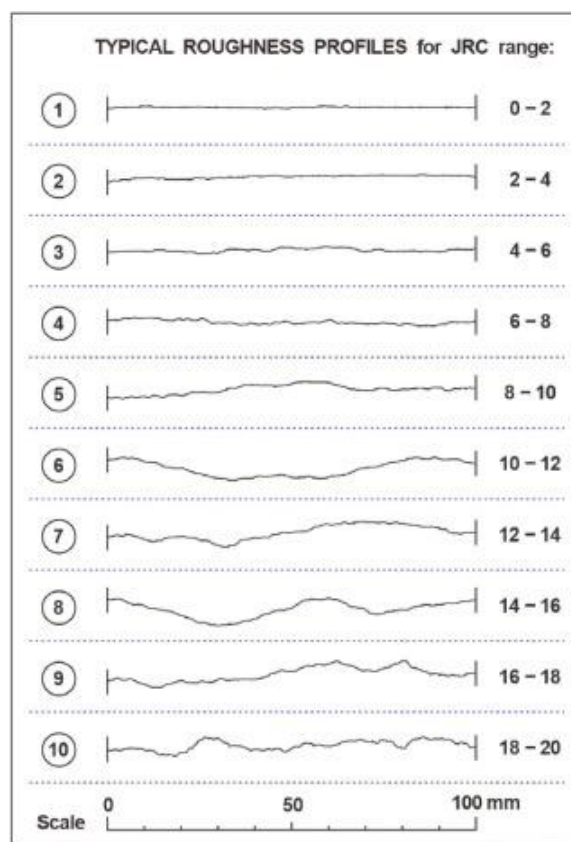


BILDE 5:ET VANLIG TESTOPPSETT FOR Å FINNE UCS EMPIRISK. ELASTISITET KAN FINNES PÅ SAMME MÅTE.

BILDE HENTET FRA [HTTPS://DOI.ORG/10.1016/J.JRMMS.2005.09.014](https://doi.org/10.1016/j.jrmms.2005.09.014)

2.3.3 Ruhet

Sprekkes ruhet er et mål på hvor glatt sprekken er. Konvensjonen er å referere til ruheten som JRC («Joint roughness coefficient») (Barton, Wang, & Yong, 2023), og man ser da på ruheten som en skalar mellom 0 og 20 der lavere tall tilsvarer glattere sprekke. Det er flere måter å estimere JRC, den vanligste er å bruke referanseruheter og sammenligne hvordan sprekken ser ut relativt til dem (Grasseli & Egger, 2003) (referanseprofilene kan sees i Bilde 6). For større sprekker (karakteristisk lengde på over 10 cm) er det også vanlig å regne JRC ved å se på amplituden til sprekken (Barton, Wang, & Yong, 2023). Denne ruhetsparameteren kommer inn i modeller som en del av friksjonsvinkelen som nevnt i 2.2.2 og dermed i friksjonsgrensen.



BILDE 6: 10 REFERANSEPROFILER FOR RUHET BILDE HENTET FRA (LEI & BARTON, 2022)

2.4 Ordliste

Under finner man en alfabetisk ordliste (sortert etter norsk). Oversettelser av begrep er så godt som mulig basert på Store Norske Leksikon, men det er ikke alltid at det finnes relevante begrep der heller, og det er da oversatt etter beste evne.

Norsk, oversatt	Engelsk fagbegrep	Forklaring
Betingelse om ingen-gjennomtrenging	Nonpenetration condition	
Brudd/svikt	Failure	Fra SNL
Fast-glide	Stick-slip	
Fasthetslære	Continuum Mechanics	Fra SNL
Fjernkrefter	Body forces	Fra SNL
Flytespenning	Yield stress	Fra SNL
Forflytning	Displacement	
Forskyvningsmodul	Shear modulus	
Friksjonsgrense	Friction bound	
Grensesnitt	Interface	
Grunnleggende friksjonsvinkel	Basic angle of friction	
Konstitutiv	Constitutive	
Kontaktkrefter	Contact mechanics	Fra SNL
Poroelastisk	Poroelasticity	
Randbetingelser	Boundary conditions	
Rutenett	Grid	
Spenning, stress	Stress	Fra SNL.
Sprekk	Joint, fracture	Ofte brukes disse fagbegrepene til å beskrive samme fenomen, forskjellen ligger i størrelsen på dem. På norsk er det ikke noen ord som oppfanger denne distinksjonen så ordene er oversatt til det samme.
Sprekknnettverk	Fracture network	
Strekk	Strain	Fra SNL
Trekraft	Traction	
Tverrgående	Transverse	

3 Modell

Hensikten i dette kapittelet vil være å gjøre rede for de matematiske modellene som brukes i oppgaven. I utgangspunktet bygger modelleringene jeg lager videre på de allerede eksisterende matematiske modellene som ligger bak PorePy rammeverket utviklet av UiBs porøse medier gruppe. Dette kapittelet er derfor sterkt basert på tidligere artikler ((Ivar Stefansson, 2024), (Keilegavlen, et al., 2020)) om PorePy rammeverket skrevet av PorePy utviklere.

Modellene er av mikset dimensjon så seksjon 3.1 vil begynne med å forklare basisen bak hvordan geometrien i modellene behandles.

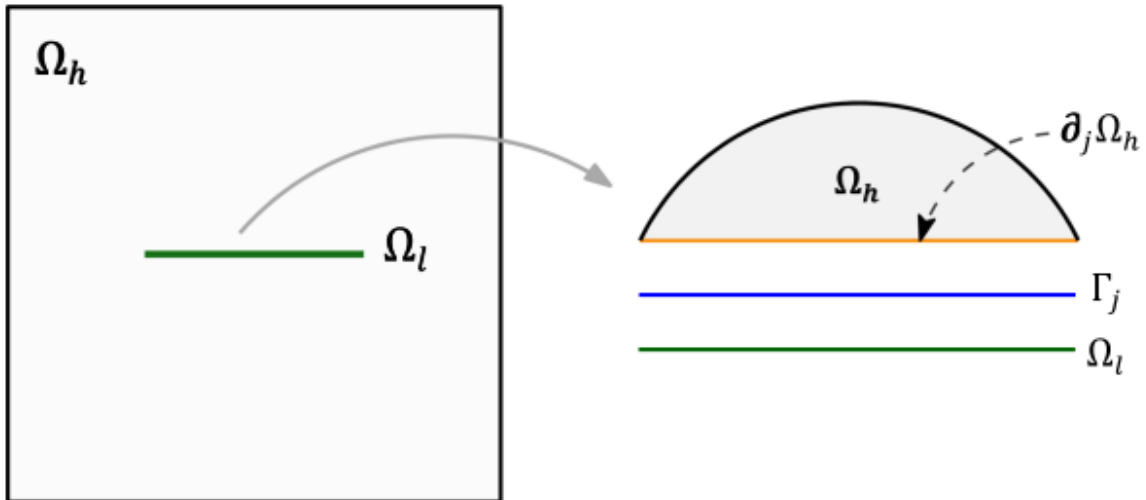
3.2 vil snakke om hvilke ligninger som ligger bak modellen, samt hvilke fysiske relasjoner som gjelder for simuleringen. Flere av ligningene presentert her er slik de allerede er presentert i PorePy, det som er nytt er den tangentielle elastiske deformasjonen, og dermed også elastisk-plastisk oppdelingen. Tanken om en elastisk-plastisk oppdeling er ikke et nytt konsept (eks. se (Lei & Barton, 2022)). Den spesifikke oppdelingen som diskuteres her er basert på (White, 2014).

Kapittel 3 avsluttes av seksjon 3.3 der det gis en gjennomgang av hvordan randbetingelser behandles i kontekst av de tidligere seksjonene.

3.1 Geometri

En av de vesentlige egenskapene ved sprekker i stein er at de har et veldig skjevfordelt størrelsesforhold. Vi kan derfor behandle sprekken som en dimensjon lavere enn det omliggende domenet. Hvis vi ser på en stein i 3 dimensjoner vil sprekken da opptre som et 2-dimensjonalt plan. På samme måte vil skjæringspunktet mellom to sprekker da opptre som en linje (altså 1 dimensjonalt objekt), og krysningen av forskjellige skjæringspunkt vil da være et 0 dimensjonalt objekt (et punkt). Avhengig av hvor mange forskjellige sprekker som er i steinen vil modellens domene altså inneholde subdomener som har forskjellige dimensjoner (et domene av dimensjon $d = 3$ kan ha subdomener av dimensjon $d_s = d - i$, $i \in \{0,1,2,3\}$).

Domener kobles sammen med subdomener av en lavere eller høyere dimensjon ved å sette inn et grensesnitt. Randverdien til domenet Ω_i vil vi kalle $\partial\Omega_i$ og den delen av randen som kobles mot subdomenet blir $\partial_j\Omega_i \subseteq \partial\Omega_i$ og koblingen skjer via grensesnittet Γ_j .



BILDE 7: DOMENE Ω_h HENGER SAMMEN MED SUBDOMENET Ω_l VED ET GRENSESNIITT Γ_j

FIGUR HENTET FRA (KEILEGAVLEN, ET AL., 2020) [HTTPS://DOI.ORG/10.1007/S10596-020-10002-5](https://doi.org/10.1007/s10596-020-10002-5)

I koblingen mellom domenet og grensesnittet spesifiserer vi om verdier projiseres fra domenet Ω_i til grensesnittet Γ_j ved å kalle dette for en operasjon Π_j^i , og den inverse operasjonen for Ξ_j^i . Dette er basisen for geometrien som brukes for å løse ligningene i modellen.

3.2 Formulering av modell

Utgangspunktet for modellen er å simulere oppførselen til sprekkenettverk som naturlig opptrer i stein. For å sette opp en fysisk realistisk modell må vi lage oss en konserveringslov for momentbalansen. Det gjør vi ved å sette opp den totale stresstensoren σ mot summen av fjernkrefter F på følgende vis:

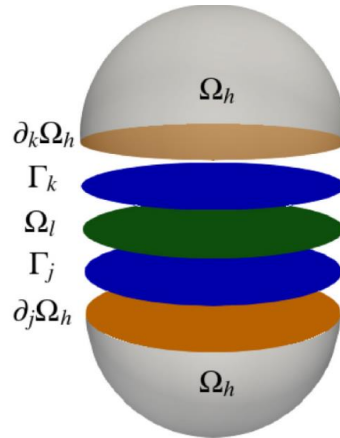
$$-\nabla \cdot \sigma = F, \quad (4)$$

som løses over domenet vårt med høyest dimensjon (vanligvis $d_i = 3$, men tenkelig også med $d_i = 2$ (da blir sprekken et 1d domene)). I tillegg formulerer vi en Hookes lov for matrisen vår:

$$F = -E \cdot \nabla' u. \quad (5)$$

Der ∇' er en symmetrisk gradient ($\nabla' u = \frac{1}{2}(\nabla u + (\nabla u)^T)$), vi kan da se på $\nabla' u$ som strekkensoren så lenge vi antar relativt liten deformasjon. Betydningen blir da at vi regner med at steinen som utgjør domenet er lineært elastisk slik at deformasjonen i domenet (her representert ved u , eller forflytning i domenet) er proporsjonal med kraften som påføres domenet ved en elastisk modulus E , en materialkonstant.

Fordi vi ser på spreknettverk må vi også ta hensyn til kontaktkreftene som opptrer i sprekken. Siden vi ser på sprekker som en lavere dimensjon fra domenet rundt dem gjør vi dette ved å lage grensesnitt mellom domene.



BILDE 8: EKSEMPEL PÅ KOBLINGEN MELLOM TO SUBDOMENER Ω_h OG Ω_l GJENNOM GRENSESNITTENE Γ_k OG Γ_j

FIGUR HENTET FRA (IVAR STEFANSSON, 2024) [HTTPS://DOI.ORG/10.1016/J.RINAM.2023.100428](https://doi.org/10.1016/j.rinam.2023.100428)

Sett ut ifra figuren over definerer vi en normalvektor, \mathbf{n}_l , til sprekken Ω_l , som er identisk til en tilsvarende vektor i Ω_h , nemlig \mathbf{n}_h . Retningen på \mathbf{n}_l blir så utgangspunktet for kontakttrekk i sprekken λ_l . Deretter blir kontakttrekket mellom sprekkeoverflatene balansert ved å sette opp følgende:

$$\Pi_j^h \sigma_h * \mathbf{n}_h = \Pi_j^l \lambda_l, \quad (6)$$

$$-\Pi_k^h \sigma_h * \mathbf{n}_h = \Pi_k^l \lambda_l. \quad (7)$$

I tillegg er det også ikke alltid kontakt mellom sprekkeoverflatene så vi definerer spranget mellom grensesnittene som:

$$[[\mathbf{u}]] = \Xi_k^l \mathbf{u}_k - \Xi_j^l \mathbf{u}_j. \quad (8)$$

Her er \mathbf{u} igjen forflytningen i domenet vårt, en vektor som består av $[[\mathbf{u}]]_{\parallel}$ og $[[\mathbf{u}]]_{\perp}$ definert ut ifra \mathbf{n}_l , der $[[\mathbf{u}]]_{\parallel}$ er i kontakttrekkets retning og $[[\mathbf{u}]]_{\perp}$ er ortogonalt på denne. I tillegg består \mathbf{u} av en plastisk og en elastisk komponent for å representere flytespenningen i det lineært elastiske domenet vårt:

$$\mathbf{u} = \mathbf{u}^e + \mathbf{u}^p. \quad (9)$$

Vi bruker så definisjonen vår av \mathbf{u} (hvis det ikke er presisert med hevet skrift eller senket skrift snakker vi om total forflytning) til å lage en betingelse om ingen-gjennomtrenging, fordi sprekkeoverflatene kan ikke penetrere gjennom hverandre:

$$\begin{aligned} \llbracket \mathbf{u} \rrbracket_{\perp} - g &\geq 0, \\ \lambda_{\perp} &\leq 0, \\ \lambda_{\perp} (\llbracket \mathbf{u} \rrbracket_{\perp} - g) &= 0. \end{aligned} \tag{10}$$

Her er g avstanden mellom sprekkeoverflatene som er delvis i kontakt, definert senere gjennom det vi kaller åpningsfunksjonen, ligning (14). Den andre ulikheten i (10) refererer til at ut ifra definisjonen vår av trekraften vil komprimerende kontakttrekk være en negativ verdi. Med dette blir friksjonsmodellen til

$$\begin{aligned} \|\lambda_{\parallel}\| &\leq b, \\ \|\lambda_{\parallel}\| < b &\rightarrow \llbracket \dot{\mathbf{u}}^p \rrbracket_{\parallel} = 0, \\ \|\lambda_{\parallel}\| = b &\rightarrow \exists \zeta \in \mathbb{R}^+ : \llbracket \dot{\mathbf{u}}^p \rrbracket_{\parallel} = \zeta \lambda_{\parallel}. \end{aligned} \tag{11}$$

Betydningen her blir da at den tangentielle plastiske forflytningen $\llbracket \dot{\mathbf{u}}^p \rrbracket_{\parallel}$ vil ikke oppstå så lenge kraften er bundet av friksjonsgrensen b , og når det først blir forflytning vil den være parallell med trekraften. Tanken er at sprekken vil enten være fastlåst eller i glidning, der den plastiske forflytningen tilsvarer glidningen til sprekken.

Ut ifra vår tidligere oppdeling (9) vil det også være en elastisk fase som tillater noe forflytning i den fastlåste fasen før friksjonsgrensen nåes:

$$\lambda_{\parallel} = K_{\parallel} \llbracket \mathbf{u}^e \rrbracket_{\parallel}. \tag{12}$$

Her er K_{\parallel} den tangentielle stivheten, en parameter som gjør at vi kan modellere den inkrementelle forskyvningen i det elastiske regimet ved en Hookes lov. Man kan se på det som den tangentielle delen av materialets forskyvningsmodul. Siden verdien på denne vil variere avhengig av egenskaper i materialet antar vi for enkelhetens skyld at denne vil ha en gitt skalar verdi.

I ligning (11) vil prikknotasjonen brukt i $\dot{\mathbf{u}}$ tilsa at forflytningen skjer i inkrement relativt til et referansepunkt. Der \mathbf{u} er samlet forflytning over domenet vil $\dot{\mathbf{u}}$ oppdateres etter hvert som den numeriske modellen itereres.

For å ta hensyn til at modellen kan fortsatt være i elastisk spenn etter å ha gått inn i en fastlåst-fase setter vi en klausul der inkrement blir gjort ifra en referanseramme uten elastisk stress:

Når den plastiske forflytningen først inntreffer antar vi at friksjonen følger en Coulomb type friksjonslov med en konstant friksjonskoeffisient μ

$$b = -\mu \lambda_{\perp}. \tag{13}$$

Vi trenger også en funksjon som representerer hvordan sprekken vil åpne seg

$$g = g^0 + \tan\theta \|\llbracket \mathbf{u}^p \rrbracket_{\parallel}\| + \frac{\Delta u_{max} \lambda_{\perp}}{\Delta u_{max} K_{\perp} - \lambda_{\perp}}. \tag{14}$$

Her er g^0 åpningen i sprekken når det ikke blir påført noe trekk eller stress krefter, θ er dilatasjonsvinkelen som påvirker dilatasjonen i steinen ($\theta > 0$ betyr at materialets volum øker under skjærspenning, mens $\theta \leq 0$ tilsvarer en sammentrekning), K_{\perp} er normalstivheten per areal og Δu_{max} er den maksimale lukningen av sprekken (sprekken er punktvis enten åpen eller lukket). Maksimal lukket tilsvarer hele sprekken lukket). Endring i g vil selvfølgelig føre til endringer i ligning (10), og gir oss en sprekkåpning a , som tilsvarer

$$a = a^0 + \llbracket \mathbf{u} \rrbracket_{\perp}. \quad (15)$$

De aller fleste av ligningene presentert over er allerede eksisterende i PorePy, og det som er blitt modifisert er det som relaterer seg til den elastisk-plastiske oppdeling. Vil derfor presisere at ligning (9) og (12) er de nyintroduserte ligningene, mens ligning (11) og (14) er allerede eksisterende i PorePy men modifisert til å bruke plastisk deformasjon (før denne elastisk-plastiske oppdelingen brukte disse bare «deformasjon»). Alle andre ligninger presentert i dette kapitlet er ligninger sånn som de eksisterer i PorePy rammeverket før denne artikkelen ble skrevet.

3.3 Randbetingelser

Dette systemet av ligninger blir så initialisert ved å modifisere randbetingelser, samt materielle konstanter og andre initialbetingelser. På de eksterne kantene (altså randbetingelser på overflaten av steinen) kan vi sette enten Neumann eller Dirichlet randbetingelser. For randbetingelser av typen Dirichlet setter vi verdier for forflytting, \mathbf{u}_i ($d_i \in \{1,2,3\}$). Ved Neuman randbetingelse setter vi verdier for trekk og stress $\sigma_i * \mathbf{n}_i$ ($d_i \in \{1,2,3\}$).

I grensesnittet mellom subdomener (interne randbetingelser) setter vi en Dirichlet type betingelse, der vi håndhever kontinuitet og bevaring av forflytning:

$$\mathbf{u}_i = \Xi_j^i \mathbf{u}_j. \quad (16)$$

4 Implementering

Dette kapitlet dreier seg om implementeringen av modellen i det allerede eksisterende PorePy rammeverket.

Seksjon 4.1 vil presentere oppbygningen av PorePy-rammeverket, både hvordan strukturen ser ut, hvordan det kan modifiseres og eksempel på hvordan et simuleringsoppsett kan se ut.

4.2 vil ta for seg spesifikt hvordan modellen som allerede er i PorePy er blitt modifisert for å implementere tangentiell elastisk deformasjon.

4.1 PorePy

PorePy er skrevet i Python og bygget opp i en objekt-orientert struktur. Dette betyr at PorePy i bunn og grunn er en modulær samling av konstitutive lover, domenegeneratorer og simuleringsverktøy. En konsekvens av at det har denne strukturen er at det er relativt enkelt å modifisere individuelle modeller ved at man endrer arvestrukturen i systemet. Eksempelvis kan man si at PorePy består av 5 simuleringsmoduler:

Momentum Balance

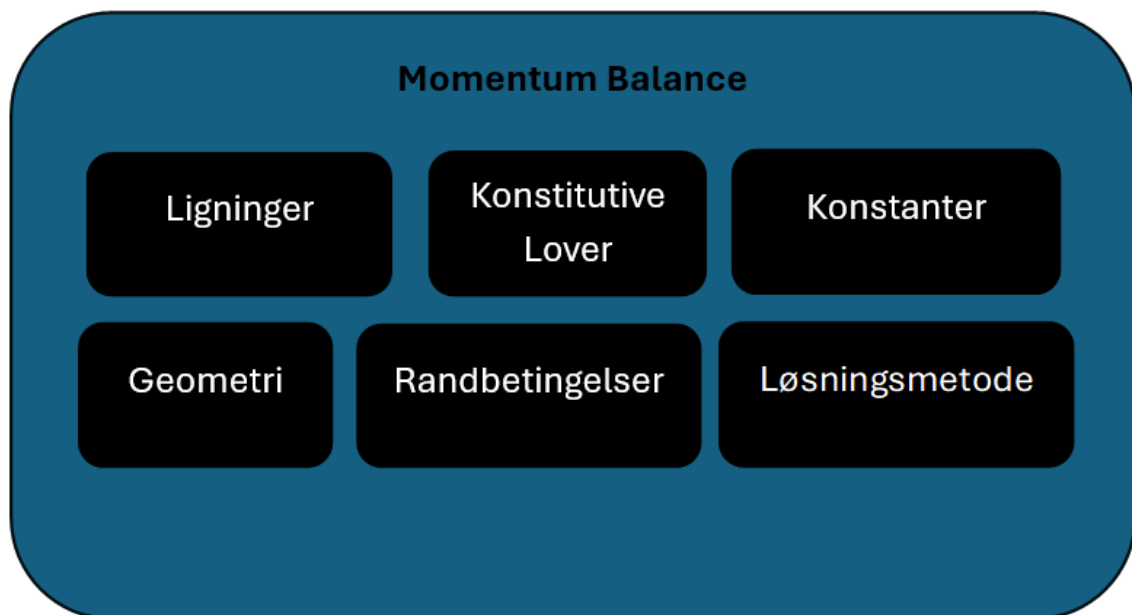
Fluid Mass Balance

Energy Balance

Poromechanics

Thermoporomechanics

Alle disse modulene består igjen av flere moduler under seg som inneholder bl.a. ligninger, konstitutive lover, se Bilde 9 nedenfor.

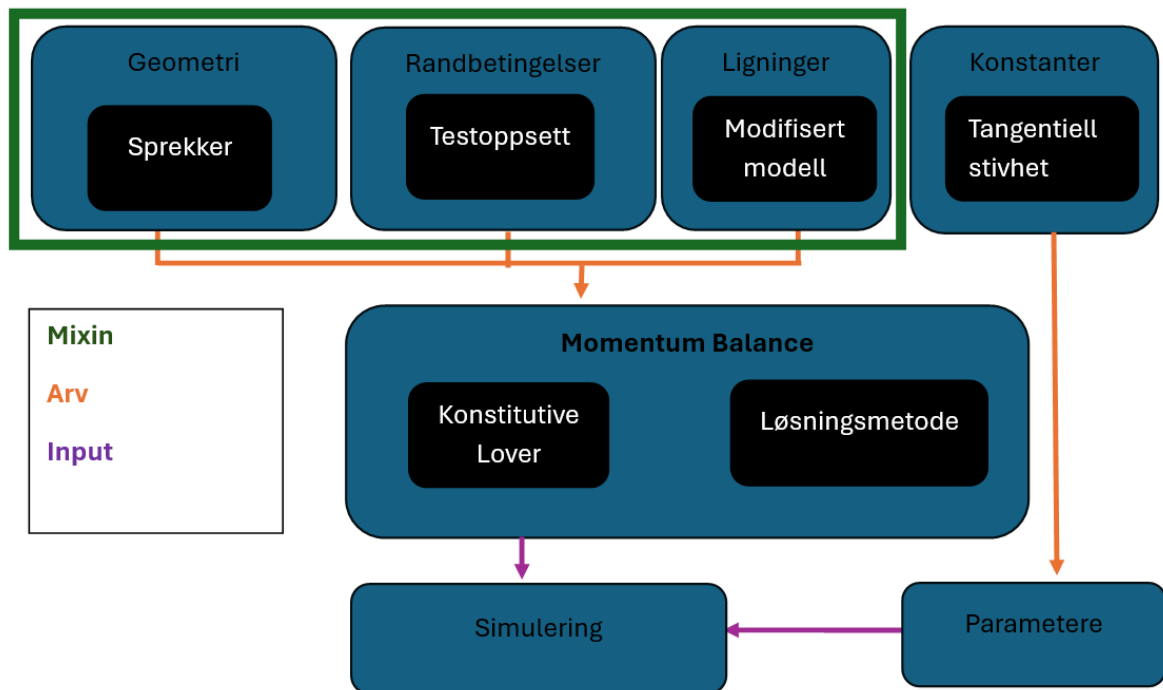


BILDE 9: STRUKTUREN PÅ EN AV SIMULERINGSMODULENE I POREPY

Her kan vi se samlingen av tingene i modulen «Momentum Balance». Av undergruppene her genererer «Geometri» en miksetdimensjonalt rutenett med sprekker, «Konstanter» inneholder parametere som er ment å fange opp karakteristikk med materialet (dette kan være ting som stivhet, temperatur, tetthet, osv.), «Ligninger» inneholder hvilke ligninger som løses i simuleringen, «Randbetingelser» setter randbetingelsene, både eksterne (på domenet) og interne (på sprekken/sprekkene), «Konstitutive lover» inneholder relasjonene mellom de forskjellige verdiene, og «Løsningsmetode» inneholder de numeriske prosedyrene som modulen bruker.

For å kjøre en simulering skriver man det som kalles en mixin-klasse og mikser den med en av de eksisterende modellklassene. Mixin-klassen inneholder de elementene man har lyst på i simuleringen sin og den eksisterende simuleringmodulen arver disse (man kan se mixin-klassen markert i grønt på Bilde 10).

På grunn av den objekt-orienterte strukturen kan man enkelt dra ut og modifisere individuelle deler for å passe inn i den simuleringen man har lyst til å kjøre. Nedenfor på Bilde 10 kan man se oppsettet som er brukt for å teste implementasjonen av den oppdaterte modellen:



BILDE 10: EKSEMPELOPPSETT PÅ EN SIMULERING I POREPY. EN MIXIN-KLASSE INNEHOLDER DE MODIFISERTE DELENE OG GIR DEM SOM ARV TIL DEN EKSISTERENDE MODULEN MOMENTUM BALANCE.

I dette eksempelet kan man se at geometrien, randbetingelsene, ligningene og konstanter er de delene av MomentumBalance modulen som er blitt modifisert for å kjøre simuleringene. I praksis er det laget tilpassete mixin-klasser som arves inn i en simulerings-klasse. Det siste leddet merket parametere vil inneholde eventuelt andre parametere man vil ha i simuleringen sin, dette kan være ting som tidssteg i den numeriske simuleringen, gittertype, cellestørrelse i gitteret, eller spesifikke verdier på konstanter for å nevne et par. Dette leddet kan kuttes ut, men det er ment å inneholde parameterne man gjerne vil kunne endre på når man kjører flere simuleringer.

4.2 Modifisering

Som kan sees på Bilde 10 vil modifiseringen av modellen skje ved at det lages et par nye ligninger i ligningsmodulen, samt at det introduseres en ny parameter i konstanter-modulen. Dette er bare en simpel skalar og representerer stivheten i tangentiell retning ($K_{||}$ i ligning (12)). I den oppdaterte variabellisten er den implementert som `tangential_fracture_stiffness` og har en standardverdi på 1,0.

Modifiseringene av ligningene er litt mer involverte og er gjort ved å introdusere to nye funksjoner og er gjort ved at det elastisk-plastiske skillet fra ligning (9) er implementert på følgende måte:

```
1 def plastic_displacement_jump(self, subdomains: list
2   [pp.Grid]) -> pp.ad.Operator:
3     """Return an operator that represents the plastic
4     component of the displacement jump."""
5     total_jump = self.displacement_jump(subdomains)
6     elastic_jump = self.elastic_displacement_jump(sub
7     domains)
8     return total_jump - elastic_jump
```

BILDE 11: BILDE AV KODEN SOM DELER HOPPET I ELASTISK OG PLASTISK DEL

Variabelen `displacement_jump` er allerede en eksisterende variabel i PorePy som regnes ut via ligning (8), mens `elastic_displacement_jump` blir en ny størrelse.

Fordi normal elastisk deformasjon allerede er implementert i PorePy vil denne `elastic_displacement_jump` være summen av den allerede eksisterende normal-deformasjonen og den nye tangentielle elastiske deformasjonen. Normal-deformasjonen er basert på en projeksjon av kontakttrekket i normal retning, så vi lager den tangentielle på lignende vis:

```

1 class ElastoPlasticFractureGap:
2     def elastic_displacement_jump(self, subdomains: list[pp.Grid]) -> pp.ad.Operator:
3         """Return an operator that represents the elastic component of the displacement jump."""
4         basis= self.basis(subdomains, dim=self.nd) # type: ignore[call-arg]
5         local_basis = self.basis(subdomains, dim=self.nd - 1)
6         tangential_to_nd = pp.ad.sum_operator_list(
7             e_nd @ e_f.T for e_nd, e_f in zip(basis[:-1], local_basis)
8         )
9         normal_to_nd = basis[-1]
10
11         nd_vec_to_tangential = self.tangential_component(subdomains)
12         t_t = nd_vec_to_tangential @ self.contact_traction(subdomains)
13         K_t = self.solid.tangential_fracture_stiffness()
14         u_t = t_t / K_t
15         # Broadcast to number of cells in case elastic normal deformation is scalar.
16         nc = sum([sd.num_cells for sd in subdomains])
17         u_n = self.elastic_normal_fracture_deformation(subdomains) * pp.ad.DenseArray(np.ones(nc))
18         return tangential_to_nd @ u_t + normal_to_nd @ u_n

```

BILDE 12:KODEN SOM LAGER DEN ELASTISKE DEFORMASJONEN

I Bilde 12 kan man på linje 14 se hvordan ligning (12) er kodet. K_{\parallel} er her kalt K_t og blir ganske enkelt hentet fra den tidligere nevnte variabelmodulen. I tillegg vil det komme inn ett ekstra steg her fordi u_n og u_t er projeksjoner av kontakttrekket i normal og tangentiell retning og vil derfor være en dimensjon lavere enn det man forventer displacement_jump å være. De er derfor pakket inn i et ekstra lag med kode som transformerer u_n og u_t tilbake til den dimensjonen som forventet ved at de prosjekteres på en nulltensor av riktig dimensjon (dette blir behandlet av linje 3-8 og 16-17 på Bilde 12 over).

I tillegg til disse to nye funksjonene er det lagt til oppdateringer til eksisterende funksjoner i fra å bruke displacement_jump til å bruke den nye plastic_displacement_jump, slik som beskrevet i ligning (10), (11) og (14).

5 Resultater

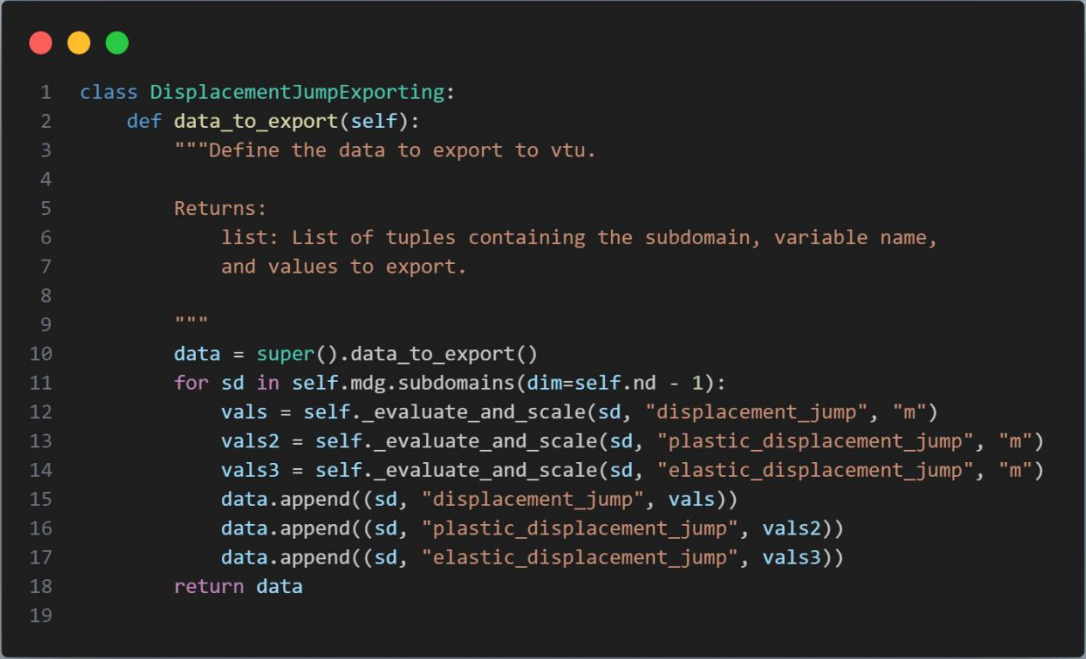
Dette kapitlet handler om den verifisering av implementeringen diskutert i kapittel 4 og hvordan koden er satt opp.

Seksjon 5.1 snakker om hvilke testoppsett som er brukt i verifiseringstestene, samt resultatene av disse.

5.2 presenterer GitHub-depotet, både det som inneholder koden i fra denne oppgaven, og PorePy-depotet. Her er det også instruksjoner om hvordan testene kan repliseres lokalt.

5.1 Testing

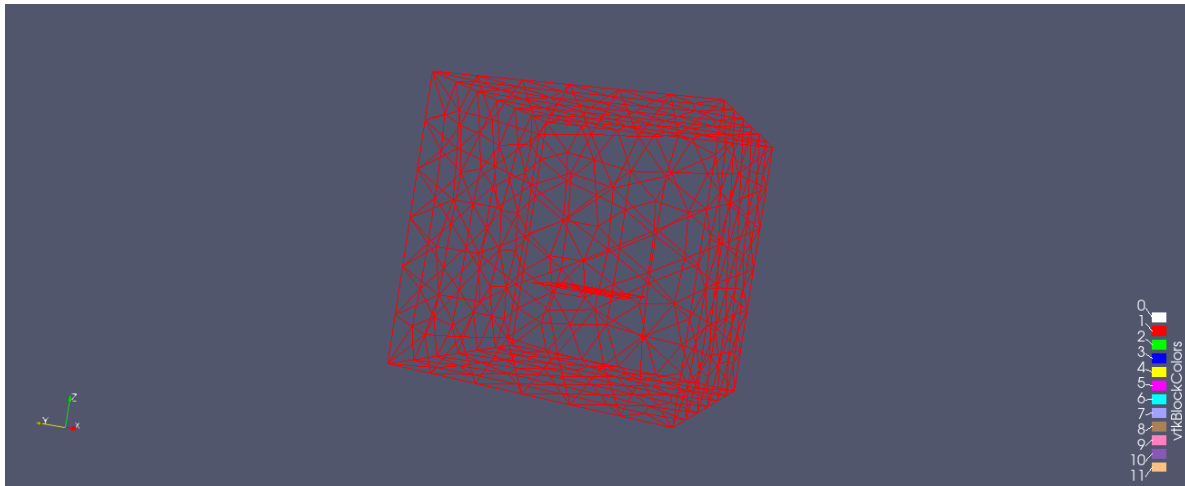
Det er blitt utført en rekke tester for å se at simuleringene med den modifiserte modellen faktisk er en implementasjon av en tangentiell elastisk forflytning konsistent med hvordan det vil oppføre seg i in-situ stein i naturen. Bilde 10 gir et forenklet bilde av hvordan oppsettet til disse testene ser ut. Som forsikring om at koden i seg selv fungerer er det gjort blitt gjort initiale tester der resultatene er eksportert til ParaView for analyse. ParaView er et dataanalyse- og visualiseringsprogram, og alle visualiseringene nedenfor er generert med ParaView. PorePy er laget slikt at det automatisk genereres filer som kan importeres i ParaView når man kjører en simulering, og det er enkelt å inkludere verdiene fra de nye funksjonene (se Bilde 13 under).



```
1 class DisplacementJumpExporting:
2     def data_to_export(self):
3         """Define the data to export to vtU.
4
5         Returns:
6             list: List of tuples containing the subdomain, variable name,
7                 and values to export.
8
9         """
10        data = super().data_to_export()
11        for sd in self.mdg.subdomains(dim=self.nd - 1):
12            vals = self._evaluate_and_scale(sd, "displacement_jump", "m")
13            vals2 = self._evaluate_and_scale(sd, "plastic_displacement_jump", "m")
14            vals3 = self._evaluate_and_scale(sd, "elastic_displacement_jump", "m")
15            data.append((sd, "displacement_jump", vals))
16            data.append((sd, "plastic_displacement_jump", vals2))
17            data.append((sd, "elastic_displacement_jump", vals3))
18        return data
19
```

BILDE 13:KODEN SOM LAGER EN EKSPORTERINGSFIL TIL PARAVIEW. VI KAN SE AT VI LAGER EN LISTE SOM LEGGER TIL VERDIENE VÅRE TIL DEN ALLEREDE EKISTERENDE LISTEN AV DATA SOM EKSPORTERES (LINJE 10, 15-17). DISSE NYE VERDIENE ER DEFINERT PÅ SPREKKEN SÅ VI HENTER DEM UT FRA SUBDOMENER MED DIMENSJON D-1 (LINJE 11).

Verifisering av at ligningene produserer riktige verdier er gjort ved å lage en testsuite i Pytest. Pytest er en modul i Python som gjør det mulig å programmere tester som kan kjøres automatisk. I PorePy er det laget flere pytester (da kalt en testsuite) som kjøres automatisk hver gang koden oppdateres for å verifisere at de forskjellige modulene fortsatt fungerer som forventet.



BILDE 14: EKSEMPEL AV ET 3D OPPSETT EKSPORTERT TIL PARAVIEW.

For oss fungerer dette ved at vi gir testen verdier for randbetingelser og forventede verdier for displacement (dette er en verdi definert på domenet), elastic_displacement og plastic_displacement (disse er definert på sprekken). Testen kjører så en simulering med de gitte randbetingelsene og sjekker verdiene fra simuleringen opp mot de forventede verdiene gitt. For å lett kunne forutse hvilke verdier for forskyvning simuleringen vil gi velger vi å gjøre testene med noen spesielle parametere:

1. Testdomenet vil inneholde en tverrgående sprekke i midten av domenet.
2. Domenet vil være fullstendig stift
3. Sprekken vil være mest mulig elastisk
4. Sprekken vil ha en høy friksjonsgrense

I tillegg bruker vi randbetingelser som er Dirichlet på toppen og bunnen av y-aksen og Neumann på de resterende randene. Som Neumann betingelse påskriver vi 0 omliggende spenning, bunnen holdes i ro og vi vekselvis dytter (negativ y-forskyvning) og drar (positiv y-forskyvning) på toppranden. Toppranden blir påskrevet både normal og tangentiell forflytning (relativt til sprekken). Kombinasjonen av tverrgående sprekke, stift domene og elastisk sprekke kan vi forutse at all forskyvning fra toppranden vil bli overført til sprekken, da topphalvdelen av domenet vil «gli» på toppen av sprekken. Faktisk kan vi si at alle cellene på toppen av sprekken vil ha denne forflytningen.

5.1.1 2 dimensjoner

Det ene oppsettet til den 2-dimensjonale testen kan sees nedenfor på Bilde 16. I vårt dyttende tilfelle legger vi randbetingelsen på toppen til å være en forskyvning på $[1.0, -1.0]$. Ved parameterne spesifisert ovenfor forventer vi da at alle celler over sprekken vil ha en displacement på $[1, 0]$, da topphalvdelen av det stive domenet «gli» opp på den elastiske sprekken.

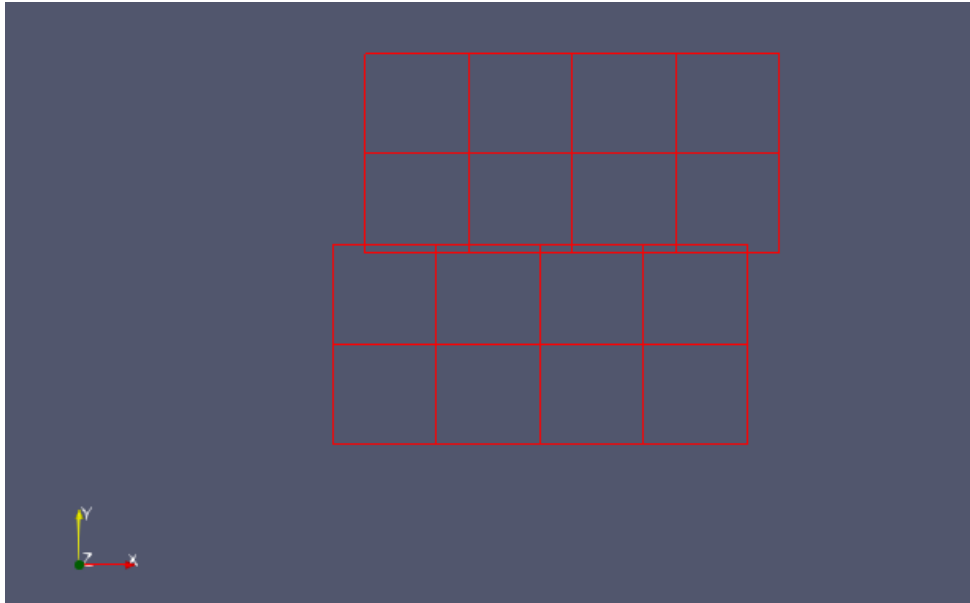
```

1  @pytest.mark.parametrize(
2      "north_displacement,u_e_expected,u_p_expected,u_x_expected",
3      [
4          ([1.0,-1.0],[-1,0], [0,0], [1]),
5          ([1.0,1.0],[0,0], [-1,1], [1]),
6      ],
7  )
8  def test_2d_single_fracture(north_displacement,u_e_expected,u_p_expected,u_x_expected):
9      """Test that the solution is qualitatively sound.
10
11     Parameters:
12         north_displacement (list): Value of displacement on the north boundary.
13         u_e_expected (list): Expected values of the elastic displacement jump in the x and y.
14             directions.
15         u_p_expected (list): Expected values of the plastic displacement jump in the x and y.
16             directions.
17         u_x_expected (list): Expected value of displacement in the x direction of cells above the fracture.
18
19     """
20     # Instantiate constants and store in params.
21     solid_vals = { "tangential_fracture_stiffness": 1e-5,
22                   "shear_modulus": 1e6,
23                   "lame_lambda": 1e3,
24                   }
25     solid = SolidConstantsWithTangentialStiffness(solid_vals)
26     params = {
27         "times_to_export": [], # Suppress output for tests
28         "material_constants": {"solid": solid},
29         "ux_north": north_displacement[0],
30         "uy_north": north_displacement[1],
31         "fracture_indices": [1],
32     }
33
34     # Create model and run simulation
35     setup = LinearModel(params)
36     pp.run_time_dependent_model(setup, params)

```

BILDE 15:HVORDAN PYTEST BRUKES FOR Å KJØRE SIMULERINGER. DENNE KODEN VISER OPPSETTET FOR 2 PYTESTSIMULERINGER MED FORSKJELLIGE RANDBETINGELSER OG FORVENTEDE VERDIER.

For elastisk_displacement forventer vi at pga. den høye friksjonsgrensen at det ikke vil være noen plastisk forflytning, og at forflytningen fra toppranden overføres til sprekken som elastisk forflytning. Her forventer vi da verdier på $[-1,0]$ og $[0,0]$ på elastic_displacement og plastic_displacement respektivt. På Bilde 15 over kan man se hvordan Pytest gjør dette for 2D simuleringen i koden. Vi gir den randbetingelser og forventede verdier ved gi den lister med sine respektive x og y verdier. North_displacement blir her randbetingelsen, u_e_expected er vår elastiske forskyvning, u_p_expected den plastiske. Den siste listen inneholder bare 1 verdi fordi i domenet generelt sjekker vi bare x verdien (og denne heter naturlig nok u_x_expected), da det er den verdien vi lett kan forutse. Bilde 15 viser bare frem til koden kjører simuleringen, men senere nede i koden hentes verdiene fra simuleringen ut og sammenlignes med de forventede verdiene.

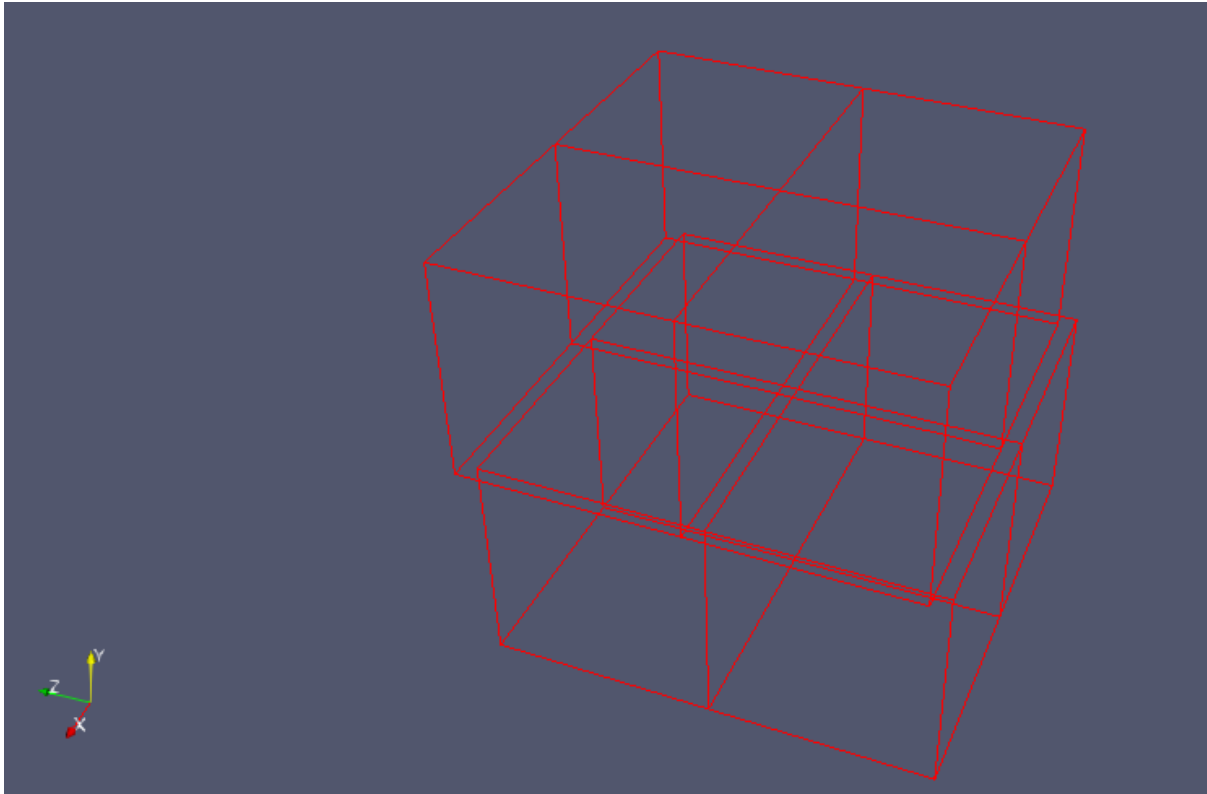


BILDE 16: EN AV OPPSETTENE BRUKT I PYTEST EKSPORTERT TIL PARAVIEW. I DEN ENDELIGE TESTKODEN ER DET BARE BRUKT 4 CELLER, MEN DET ER TESTET MED FLERE VARIANTER, F.EKS. MED 16 CELLER SLIK SOM PÅ DETTE BILDET.

Både testdomenet, firkantet domene med sidelengde 1 og en tverrgående sprekk, og randbetingelsene er versjoner av standardiserte klasser i PorePy og er ofte designet for å være enklest mulig (med automatiske tester som skal kjøres ofte er dette verdifullt).

5.1.2 3 dimensjoner

Det er også laget pytester for det 3-dimensjonale tilfellet og det er her også brukt et standardisert domene. Det er nå en kube med sidelengde 1, og sprekken er nå tverrgående i firkantplanet $[0,0.5,0]$, $[0,0.5,1]$, $[1,0.5,0]$ og $[1,0.5,1]$, som kan sees på Bilde 17 under. Sprekken er lagt i x-z planet slik at samme randbetingelse ifra 2D tilfellet kan gjenbrukes (altså at vi dytter/drar i toppen av y-aksen). Igjen kjører vi både en dytte og en dra test, med forskyvningen $[2, -1, 3]$ og $[2, 1, 3]$ som randbetingelser respektivt. I begge tilfellene forventer vi lignende resultater som i den 2-dimensjonale testen. For dytting betyr det at vi forventer ingen plastisk forskyvning og en elastisk forskyvning på $[2, 0, 3]$ i sprekkens og alle cellene over sprekkens. Ved drag forventer vi ingen elastisk forskyvning, en plastisk forskyvning på $[2, 1, 3]$ og samme forskyvning i cellene over sprekkens som i dyttetesten.



BILDE 17: VISUALISERING AV DET 3 DIMENSJONALE PYTEST OPPSETTET. SOM I 2D VIL TOPPCELLENE I DOMENET GLI I PÅ TOPPEN AV DEN ELASTISKE SPREKKEN I POSITIV Z OG Y RETNING.

5.1.3 Testresultater

Pytestsuiten gir de verdiene vi forventer, men for `plastic_displacement` og `elastic_displacement` ender det opp med at vi får negative tangentielle verdier istedenfor de positive verdiene vi forventet. F.eks. 3D dytting ender vi opp med `elastic_displacement` verdier på $[-2,0, -3]$ istedenfor $[2,0,3]$. Dette er fordi både `plastic_displacement` og `elastic_displacement` er verdier definert på sprekken og er derfor ikke i samme koordinatsystem som resten av domenet (siden sprekker er definert som en dimensjon mindre enn domenet rundt vil verdier definert på sprekken være ut ifra sitt eget lokale koordinatsystem). Om det lokale koordinatsystemet sprekken bruker er i samme retning som det globale koordinatsystemet domenet ligger i vil være tilsynelatende tilfeldig, men er spesifikt til rutenettet, da det kommer an på hvilken vei cellene i sprekken blir orientert under rutenettkonstruksjonen. Siden testsuiten bruker et predefinert rutenett der vi vet at det lokale koordinatsystemet er rotert i en slik retning at disse verdiene er negative istedenfor positive løser vi dette ved at vi hardkoder testen til å forvente negative verdier, da vi vet at for disse rutenettene er det det som tilsvarer fysiske riktige verdier.

5.2 Kildekode

GitHub er en internetbasert plattform designet for å brukes i kodebaserte prosjekt. PorePy er det man kaller et åpent-kildeprosjekt og koden ligger tilgjengelig på UiBs porøse media gruppe sin GitHub: <https://github.com/pmgbergen/porepy>.

Dette prosjektet kan man også få tilgang til på GitHub her: <https://github.com/TorWollmann/Master24>

På dette GitHub-depotet finner man filene `TestCase.py`, `constant_extensions.py`, `model_extensions.py`, `test_elastic_deformation.py` samt en `devcontainer` fil og en `.gitignore` fil (denne er bare relevant hvis du vil laste ting opp til depotet). `TestCase.py` er filen brukt til all

ParaView eksporteringen, både for den initiale testingen og for alle visualiseringsbildene i denne oppgaven. Det er prøvd å gi filene navn som er mest mulig selvforklarende, så `constant_extensions.py` er filen som inneholder den nye tangentielle stivhetsparameteren, `model_extensions.py` er filen som inneholder de modifiserte modellene (Bilde 11 og Bilde 12 er hentet her ifra) og `test_elastic_deformation.py` inneholder pytestene (ved å kalle den test kan den detekteres automatisk av Python). Devcontainer filen inneholder en «image» som kan lastes inn for å hente inn PorePy og replisere koden brukt i denne oppgaven lokalt (vi har brukt «Docker» som devcontainer). Bruk da commit «b36ad264f59b4e1b108848ef9414f0df39808829» i PorePy og commit: «778ddea5b71594ac377d93bad5ae92481814b42c» i Master24 depotet, da dette er versjonen brukt i denne oppgaven.

6 Diskusjon og konklusjoner

Hvis vi ser på resultatene fra seksjon 5.1.3 får vi verdier som stemmer overens med det vi forventer at simuleringene skal produsere. Dette vil være sant for de spesifikke parameterne vi har valgt (5.1) der sprekken kun befinner seg i en fastlåst fase og ikke opplever slipp og plastisk deformasjon gjennom en glide-fase. Vi kan gjøre dette siden vi kun har modifisert den elastiske tangentielle deformasjonen, og de andre momentene blir derfor allerede testet for i PorePy.

Vi har også møtt på «problemet» med at de lokale koordinatene i sprekken som brukes til å regne ut verdiene for forskyvning på sprekken ikke nødvendigvis ligger i samme retning som de globale koordinatene resten av domenet opererer på. I denne oppgaven har vi kommet rundt dette med å bare hard-kode verdier som tar hensyn til dette, men dette er en uelegant løsning som kun vil fungere så lenge vi vet hvilken vei det lokale koordinatsystemet er orientert, noe som betyr at vi må ha kjennskap til det spesifikke rutenettet vi bruker fra før av. I fremtiden kan det være aktuelt å implementere en rotasjonsfunksjon i PorePy som kan brukes til å transformere verdier fra det lokale koordinatsystemet til det globale og visa versa, men per i dag eksisterer dette ikke.

En annen begrensning med testene slik vi har satt dem opp nå er at det kun kjøres ett tidssteg, så vi får ikke verifisert at koden klarer å skille mellom u og \dot{u} (\dot{u} er den inkrementelle forskyvningen som oppdateres hvert tidssteg) siden disse vil ha samme verdi når vi bare har ett tidssteg. Dette er en limitasjon med å bruke den ferdiglagete, standardiserte, randbetingelsen i PorePy, og kan eventuelt fikses på med å legge inn en modifikasjon av denne randbetingelsen i testen.

Selv om å dele opp deformasjonen i et elastisk-plastisk forhold er et steg mot mer realistiske simuleringer er det fortsatt moment som mangler i modellene i PorePy. Hvis vi ser tilbake på figuren introdusert i kapittel 2, Bilde 2, forventer vi ikke bare en elastisk del, men også at det vil skje svikt når flytespenningen nåes. Når det skjer svikt i steinen på denne måten vil det innebære destruktiv skade som vil føre til at ruheten i sprekken blir brutt ned. Når dette skjer innebærer det at friksjonsgrensen vil bli lavere etter at svikt er nådd siden sprekken er blitt glattere. Dette vil gi hysteresese ved gjentatte forsøk på samme domene, i linje med hvordan in-situ stein vil oppføre seg (Bandis, Lumsden, & Barton, 1983). Dette vil gjøre systemet mye mer komplekst fordi det må introduseres en form for å spore skaden som oppstår, og lagre den videre i simuleringen. I den samme artikkelen (White, 2014) som er basis for de modifiserte ligningene brukt i denne oppgaven presenterer også White en modell for dette og det er planer om å se på om man kan bruke en lignende tilnærming for å utvide PorePy til å inkludere dette.

Hvis man i så fall bruker en tilnærming basert på (White, 2014), vil det måtte implementeres en elastisk-plastisk oppdeling i PorePy, og dette er en motivator for å få modellene og koden fra denne oppgaven akseptert som en offisiell del av PorePy rammeverket. Tanken om å bli en del av PorePy var også bakgrunnen bak noen av valgene gjort i testsuiten, som å bruke standardiserte randbetingelser og geometrier som kan forenkles mest mulig, da en automatisk testsuite er en av kravene for å være en del av PorePy.

Bibliografi

- Bandis, S. C., Lumsden, A. C., & Barton, N. R. (1983, Desember). Fundamentals of rock joint deformation. *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts*, ss. 249-268.
- Barton, N. R. (2013, August). Shear strength criteria for rock, rock joints, rockfill and rock masses: Problems and some solutions. *Journal of Rock Mechanics and Geotechnical Engineering*, ss. 249-261. <https://doi.org/10.1016/j.jrmge.2013.05.008>.
- Barton, N. R. (2021). *Details related to the ten JRC profiles and further work*. Oslo: NB&A.
- Barton, N. R., & Choubey, V. (1977, Desember). The shear strength of rock joints in theory and practice. *Rock Mechanics and Rock Engineering*, ss. 1-54. DOI: 10.1007/BF01261801.
- Barton, N. R., Wang, C., & Yong, R. (2023, Desember). Advances in joint roughness coefficient (JRC) and its engineering applications. *Journal of Rock Mechanics and Geotechnical Engineering*, ss. 3352-3379. <https://doi.org/10.1016/j.jrmge.2023.02.002>.
- Goel, R. K., & Singh, B. (2011). *Engineering Rock Mass Classification : Tunnelling, Foundations and Landslides*. Elsevier Science & Technology.
- Grasseli, G., & Egger, P. (2003, Januar). Constitutive law for the shear strength of rock joints based on three-dimensional surface parameters. *International Journal of Rock Mechanics and Mining Sciences*, ss. 25-40. [https://doi.org/10.1016/S1365-1609\(02\)00101-6](https://doi.org/10.1016/S1365-1609(02)00101-6).
- Ivar Stefansson, J. V. (2024, Februar). Flexible and rigorous numerical modelling of multiphysics processes in fractured porous media using PorePy. *Results in Applied Mathematics*, s. <https://doi.org/10.1016/j.rinam.2023.100428>.
- Jaeger, J. C., Cook, N. G., & Zimmerman, R. W. (2007). *Fundamentals of Rock Mechanics, fourth edition*. Blackwell Publishing Ltd.
- Keilegavlen, E., Berge, R., Fumageli, A., Starnoni, M., Stefansson, I., Varela, J., & Berre, I. (2020, Oktober). PorePy: an open-source software for simulation of multiphysics processes in fractured porous media. *Computational Geosciences*, ss. 243-265. <https://doi.org/10.1007/s10596-020-10002-5>.
- Lei, Q., & Barton, N. R. (2022, Mai). On the selection of joint constitutive models for geomechanics simulation of. *Computers and Geotechnics*, s. <https://doi.org/10.1016/j.compgeo.2022.104707>.
- Palmström, A. (2002). Measurement and characterization of rock mass jointing. I K. R. Saxena, & V. M. Sharma, *In-situ Characterization of Rocks*. A A Balkema Publishers.
- Swallowe, G. M. (1999). Yield and Plastic Deformation. I G. M. Swallowe, *Mechanical Properties and Testing of Polymers* (ss. 281-286). Springer Science+Business Media Dordrecht.
- White, J. A. (2014, Juli). Anisotropic damage of rock joints during cyclic loading: constitutive framework and numerical integration. *International Journal for Numerical and Analytical Methods in Geomechanics incorporating Mechanics of Cohesive-frictional Materials*, ss. 1036-1057. <https://doi.org/10.1002/nag.2247>.

