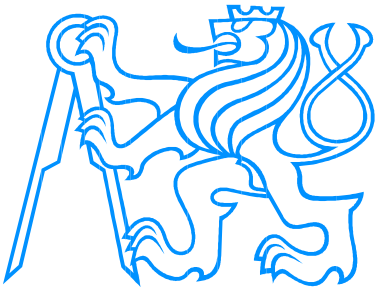
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ
V PRAZE
Fakulta stavební
Katedra mechaniky



Konstitutivní model Cam-Clay

Constitutive Model Cam-Clay

BAKALÁŘSKÁ PRÁCE

Ondřej Faltus

Studijní program: Stavební inženýrství
Studijní obor: Konstrukce a dopravní stavby

Vedoucí práce: Ing. Martin Horák, Ph.D.

Praha, 2018

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: Faltus Jméno: Ondřej Osobní číslo: *439041*

Zadávající katedra: Katedra mechaniky

Studijní program: Stavební inženýrství

Studijní obor: Konstrukce a dopravní stavby

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce: Konstitutivní model Cam-Clay

Název bakalářské práce anglicky: Constitutive model Cam-Clay

Pokyny pro vypracování:
Cílem práce je implementace modelu Cam-Clay do programového prostředí OOFEM. Chování modelu bude ověřeno na jednoduchých příkladech na úrovni materiálového bodu a model bude následně aplikován na výpočet složitější konstrukce, např. stability svahu.

Seznam doporučené literatury:
Simo, Hughes: Computational inelasticity, 2006, Springer Science & Business Media
Jirásek, Bažant: Inelastic analysis of structures, 2002, John Wiley & Sons
Dokumentace k programu OOFEM dostupná na stránkách www.oofem.org

Jméno vedoucího bakalářské práce: Ing. Martin Horák, Ph.D.

Datum zadání bakalářské práce: 23.2.2018 Termín odevzdání bakalářské práce: 27.5.2018
Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku

Podpis vedoucího práce Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

Beru na vědomí, že jsem povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v bakalářské práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce" a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací".

23. 02. 2018

Datum převzetí zadání Podpis studenta(ky)

Čestné prohlášení

Prohlašuji, že jsem tuto bakalářskou práci na téma "Konstitutivní model Cam-Clay" vypracoval pod odborným vedením Ing. Martina Horáka, PhD., samostatně a citoval veškerou použitou literaturu.

V Praze dne 27. 5. 2018

Ondřej Faltus

Poděkování

V první řadě patří poděkování Ing. Martinu Horákovi, PhD., za odborné vedení při této práci, zvláště za uvedení do problematiky programu OOFEM a za vskutku mimořádnou trpělivost a nasazení při testování mého kódu.

Děkuji i všem lidem v mém okolí, kteří mi poskytovali přátelskou podporu během chvil, kdy se tato práce rodila.

Abstrakt

Práce se zabývá modifikovaným materiálovým modelem Cam-Clay. Model je představen a popsán, je odvozen zákon zpevnění. Jsou představeny algoritmy pro implementaci tohoto modelu do výpočetního software OOFEM, provedená implementace je podrobně popsána. Funkčnost výsledného kódu je otestována na několika příkladech o různém stupni složitosti.

Abstract

The thesis deals with the Modified Cam-Clay model. The model is introduced and described, a hardening law is derived. Algorithms are presented serving for this model's implementation into the OOFEM finite element computational software. The implementation is described in detail. The resulting code's functionality is tested on several examples of varying degrees of complexity.

Klíčová slova

Modifikovaný model Cam-Clay, OOFEM, plasticita, návrat na plochu plasticity, projekce na nejbližší bod

Keywords

Modified Cam-Clay model, OOFEM, plasticity, return mapping, closest-point projection

Obsah

Seznam použitých symbolů

$\boldsymbol{\sigma}$ - matice napětí ve tvaru $(\sigma_x, \sigma_y, \sigma_z, \tau_{yz}, \tau_{xz}, \tau_{xy})^T$

σ_m - střední napětí dle vztahu $\sigma_m = \frac{1}{3}(\sigma_x + \sigma_y + \sigma_z)$

s_* - deviatorická část napětí na ose $*$ dle vztahu $s_* = \sigma_* - \sigma_m$

$\boldsymbol{s}$ - matice deviatorického napětí ve tvaru $(s_x, s_y, s_z, \tau_{yz}, \tau_{xz}, \tau_{xy})^T$

I_1 - první invariant napětí dle vztahu $I_1 = \sigma_x + \sigma_y + \sigma_z$

J_2 - druhý invariant napětí dle vztahu $J_2 = \frac{1}{6}((\sigma_x - \sigma_y)^2 + (\sigma_y - \sigma_z)^2 + (\sigma_x - \sigma_z)^2) + \tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2$

$\boldsymbol{\varepsilon}$ - matice deformace ve tvaru $(\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{yz}, \gamma_{xz}, \gamma_{xy})^T$

$\boldsymbol{\varepsilon}_e$ - elastická deformace

$\boldsymbol{\varepsilon}_p$ - plastická deformace

$\boldsymbol{g}(\boldsymbol{\sigma})$ - plastický gradient

$\boldsymbol{\delta}$ - škálovací matice ve tvaru $(1, 1, 1, 0, 0, 0)^T$

$\boldsymbol{P}$ - škálovací matice ve tvaru $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}$

1 Úvod

Hlavním cílem této práce je rozšíření výpočtového softwaru OOFEM, který je vyvíjen (mimo jiné) na katedře mechaniky Fakulty stavební ČVUT v Praze (viz [7]). Tento software pro numerické výpočty metodou konečných prvků v současné době obsahuje moduly umožňující výpočet pomocí řady materiálů, nicméně modul, který by zpracovával některý z modelů Cam-Clay, zatím chybí. Vzhledem k významu, jaký má model Cam-Clay v geotechnickém inženýrství, se zdá jeho doplnění nanejvýš vhodné. Konkrétním modelem, kterým se zabývá tato práce, je modifikovaný model Cam-Clay, vhodný zejména z důvodu eliptického tvaru plochy plasticity (diskutováno v části 2.4) a tím dané jednoduchosti numerické implementace.

1.1 Teorie víceosé plasticity

Teoretickou základnou pro elastoplastické numerické výpočty je teorie víceosé plasticity. V této části uvedeme některé základní zákonitosti a předpoklady této teorie pro pozdější referenci. Rozhodně si však tato část neklade za cíl zpracovávat toto téma podrobně.

Teorie víceosé plasticity rozšiřuje jednorozměrnou plasticitní teorii (pružnoplastický model) o víceosou napjatost. Skalární veličiny přejdou na veličiny tenzorové a z jednoznačného stanovení meze plasticity se stává problematické stanovení tzv. *funkce plasticity*, která vyjadřuje povrch v prostoru napětí, na němž dochází k plastizaci [6].

Hlavní myšlenky plasticitní teorie se dají shrnout do několika matematicky vyjádřených vztahů. Je to zejména *zobecněný Hookův zákon* [6], který popisuje elastické chování, ve tvaru

$$\boldsymbol{\sigma} = \boldsymbol{D}_e \boldsymbol{\varepsilon}_e \tag{1}$$

kde

$\boldsymbol{\sigma}$ je sloupcová matice napětí
$\boldsymbol{D}_e$ je matice pružné tuhosti
$\boldsymbol{\varepsilon}_e$ je sloupcová matice elastické deformace

a poté tři základní rovnice plastického chování. Jsou to *podmínka plastické přípustnosti* ve tvaru

$$f(\boldsymbol{\sigma}) \leq 0 \tag{2}$$

kde

$f(\boldsymbol{\sigma})$ je funkce plasticity

která omezuje napětí tak, aby se nedostalo mimo přípustnou oblast uvnitř povrchu definovaného funkcí plasticity [6], *zákon plastického přetváření* ve tvaru

$$\dot{\boldsymbol{\varepsilon}}_p = \dot{\lambda}\boldsymbol{g}(\boldsymbol{\sigma}) \tag{3}$$

kde

$\dot{\boldsymbol{\varepsilon}}_p$ je rychlost (časová derivace) plastické deformace
$\dot{\lambda}$ je rychlost (časová derivace) plastického násobitele udávajícího její velikost
$\boldsymbol{g}(\boldsymbol{\sigma})$ je plastický gradient udávající její směr

který udává způsob, jakým je řízen směr plastické deformace v závislosti na napětí [6], a konečně *podmínka komplementarity* ve tvaru

$$\dot{\lambda} f(\boldsymbol{\sigma}) = 0 \tag{4}$$

která omezuje plastické přetváření pouze na stav, kdy je napětí na ploše plasticity [6].

1.2 Poznámka o maticovém a tenzorovém zápisu

Narozdíl od jednoduchého skalárního vyjádření v případě 1D teorií jsou ve víceosé plasticitě napětí, deformace a další s nimi související veličiny obecně tenzorového charakteru [6]. Z hlediska implementace algoritmů do výpočtového softwaru je toto přístup značně nevhodný, neboť klade zbytečně vysoké nároky na strojovou paměť. Shodná informace, kterou ukladají tyto rozměrné tenzory, se dá vyjádřit kompaktním maticovým zápisem, kde napětí a deformace budou sloupcové matice o šesti prvcích a modul tuhosti matice čtvercová o 6×6 prvcích. Tento přístup je zcela ekvivalentní k tenzorovému zápisu používanému při formálním odvození algoritmů a je využit při praktické implementaci, z čehož také pramení některé komentáře o rozměrnosti výpočtů v kapitole 3.

2 Modifikovaný model Cam-Clay

2.1 Historie a význam

Otázka spojení teorie plasticity používané pro stavební materiály, jako je ocel a beton, s geotechnikou byla v popředí teoretického bádání v zejména v první polovině dvacátého století [8]. První formulace směřující ke skutečnému a použitelnému konstitutivnímu modelu pro zeminy se objevovaly v padesátých letech dvacátého století. Konečně v roce 1963 prezentovali Roscoe[1] a Schofield originální model Cam-Clay [8]. V roce 1968 následoval modifikovaný model Cam-Clay (Roscoe a Burland) [8]. První pokusy o numerickou implementaci se datují do let sedmdesátých [8].

Zásadní předností modifikovaného modelu Cam-Clay oproti modelu původnímu je eliptický tvar funkce plasticity v prostoru p-q (více viz 2.4); původní model namísto elipsy používal složitější křivku, která dokonce v místě hydrostatického tlaku měla ostrý zlom, což by přinášelo problémy se směrem plastického přetváření při takovém stavu napjatosti [8].

Inspirací pro model Cam-Clay (a v důsledku tedy i pro modifikovaný model Cam-Clay) je triaxiální zkouška vzorku jílovité zeminy [8]. Za pomoci určitých představ o chování vzorku při této zkoušce [4] je odvozen i zákon zpevnění (diskutováno v části 2.5). Je důležité poznamenat, že všechny popisované procesy se dějí pří dokonale odvodněné triaxiální zkoušce, pracujeme tedy vždy s efektivními, nikoli totálními, hodnotami napětí [12].

2.2 Parametry modelu

Jako každý materiálový model, i modifikovaný model Cam-Clay je určité matematické vyjádření představ o plasticitním chování materiálů [6], v tomto případě jílů ve stavu triaxiální napjatosti [8] [4]. Aby bylo možno provádět konkrétní výpočty, je třeba naškálovat model tak, aby odpovídal danému materiálu a situaci, v níž se tento nachází. K takovému škálování slouží následující parametry [4] [5]:

- sklon přímky kritického stavu M (více v částech 2.4 a 2.5)
- prekonsolidační tlak na počátku výpočtu $(p_c)_0$

[1] **Kenneth Harry Roscoe** (1914-1970) byl profesorem na Cambridge University, který svým životním dílem zásadně přispěl k vývoji plasticitních teorií v mechanice zemin.

Narozen v roce 1914, bojoval v druhé světové válce, jejíž valnou část strávil v německém zajetí. Po válce se na cambridgeské univerzitě věnoval studiu mechaniky zemin, roku 1968 byl jmenován profesorem. Zásadními články, které v oboru publikoval, byly *On the Yielding of Soils*, 1958 a několik článků formulujících modely Cam-Clay a modifikovaný Cam-Clay v 60. letech 20. století.

Zemřel předčasně při automobilové nehodě v dubnu roku 1970 [9].

- pórovitost zeminy e
- index normální konsolidace λ
- index bobtnání κ

Jak bude předvedeno v části 2.5, ukazuje se jako výhodné zkombinovat poslední tři parametry do nového parametru ϑ, který budeme nazývat *parametr zpevnění*. Vyjádříme ho jako:

$$\vartheta = \frac{1+e}{\lambda - \kappa} \tag{5}$$

Všechny tyto parametry jsou z hlediska výpočetního algoritmu jeho vstupy, které je třeba dodat zvenčí. Pro zjednodušení práce s programem jsem však do něj zapracoval určité implicitní hodnoty těchto parametrů, které budou brány v úvahu v případě, že při zadání úlohy nebude ten daný parametr zvláště specifikován. Tyto implicitní hodnoty, vypsané v tabulce 1, jsou dobrou ukázkou hodnot, v nichž se parametry pro běžné zeminy pohybují.

Parametr	Implicitní hodnota
M	1,2
$(p_c)_0$	100 kPa
e	0,2
λ	0,066 $(\ln \mathrm{MPa})^{-1}$
κ	0,0077 $(\ln \mathrm{MPa})^{-1}$

Tabulka 1: Implicitní hodnoty vstupních parametrů modelu určené pro výpočtový algoritmus

2.3 Elasticita

Pro materiálový model Cam-Clay se běžně uvažuje určitá nelineárnost v elastické části přetváření, tj. elastická matice tuhosti se uvažuje jako sama závislá na napětí [5]. Takový přístup je popsán v dodatku A. V rámci této práce do našeho materiálového modelu budeme uvažovat elasticitu lineární, kde je elastická tečná matice dána vztahem [6]

$$\boldsymbol{D}_e = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}-\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}-\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}-\nu \end{pmatrix} \tag{6}$$

kde

E je elastický modul pružnosti

ν je Poissonovo číslo

Případné rozšíření implementovaného modelu o nelinearitu je v budoucnu možné bez větších zásahů do stávajících algoritmů (diskutováno v části 3.2).

2.4 Funkce plasticity

V modelu Cam-Clay je zvykem vyjadřovat hodnotu napětí pomocí dvou invariantů p a q, které reprezentují hydrostatickou, resp. deviatorickou část napětí [5]. Jejich hodnoty se určí jako

$$p = \quad -\sigma_m = -\tfrac{1}{3} I_1 \tag{7}$$

$$q = \sqrt{\tfrac{3}{2}} ||\boldsymbol{s}|| = \sqrt{3 J_2} \tag{8}$$

Vyjádření p jako záporné hodnoty středního napětí je samozřejmě vhodné z důvodu, že v zeminách se zabýváme téměř výhradně tlakovým napětím, a tedy dává smysl pro pohodlnost při tlaku vyjadřovat invariant p jako kladný.

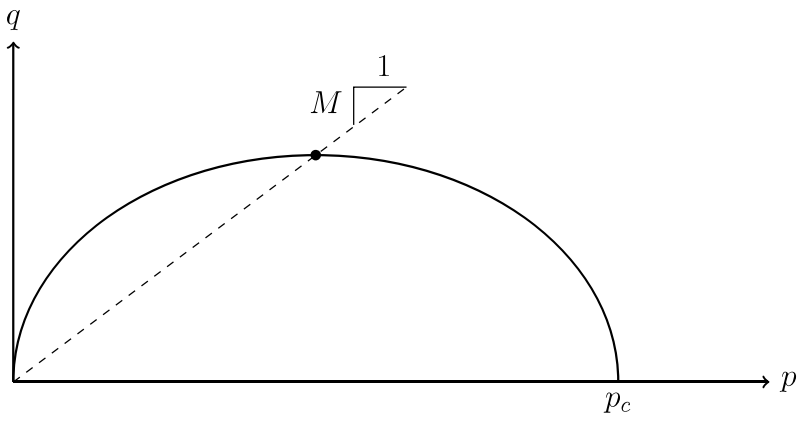
Funkce plasticity v závislosti na těchto invariantech je dána následujícím vztahem [5]:

$$f(p, q, p_c) = q^2 - M^2 p(p_c - p) \tag{9}$$

Průmětem této funkce v prostoru p-q je elipsa[2], jak je zobrazeno na obrázku 1. Poměr os elipsy je fixován neměnným parametrem M, ale její velikost se může s časem měnit v závislosti na zpevnění (změkčení), vyjádřeném prekonsolidačním tlakem p_c. Proto je funkce zapsána jako závislá na p_c [5].

Přímka čárkovaně zobrazená na obrázku 1 se nazývá *přímka kritického stavu*. Kritickým stavem se zde rozumí takový stav, kdy nedochází ke zpevnění ani k změkčení a veškerá plastická deformace je ryze deviatorická. Jak bude předvedeno v části 2.5, platí zároveň, že ve stavu napjatosti ležícím vpravo od přímky kritického stavu dochází ke zpevnění; při stavu napjatosti vlevo od přímky ke změkčení. Při všech změnách elipsy v důsledku zpevnění/změkčení její vrchol vždy zůstane na přímce kritického stavu [5].

[2] Z hlediska čistě matematického je funkcí plasticity celá elipsa, protože však, jak je zjevné ze vztahu (8), je hodnota invariantu q vždy nezáporná, postačí zobrazovat část elipsy ležící v polorovině kladné osy q



Obrázek 1: Modifikovaný model Cam-Clay: Tvar funkce plasticity v prostoru p-q

V modifikovaném modelu Cam-Clay se používá sdružený zákon plastického přetváření [5]. To znamená, že plastický gradient $\boldsymbol{g}(\boldsymbol{\sigma})$ je přímo matematicky gradientem funce plasticity, tedy její derivací podle $\boldsymbol{\sigma}$. Geometricky vyjádřeno, směr plastické deformace je v každém okamžiku kolmý na tečnu k elipse plasticity (viz obrázek 1) vedenou bodem, v němž se momentálně nachází hodnota napětí[3] [6].

Se zřetelem k jeho sdruženosti můžeme tedy přepsat zákon plastického přetváření (3) do podoby

$$\dot{\boldsymbol{\varepsilon}}_p = \dot{\lambda}\frac{\partial f}{\partial \boldsymbol{\sigma}} \tag{10}$$

Analytické vyjádření derivace funkce plasticity (9) dává následující výsledek [5]:

$$\frac{\partial f(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} = \frac{\partial f}{\partial p}\frac{\partial p}{\partial I_1}\frac{\partial I_1}{\partial \boldsymbol{\sigma}} + \frac{\partial f}{\partial q}\frac{\partial q}{\partial J_2}\frac{\partial J_2}{\partial \boldsymbol{\sigma}} \tag{11}$$

$$\frac{\partial f(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} = M^2(2p - p_c)(-\frac{1}{3})\boldsymbol{\delta} + 2q\frac{\sqrt{3}}{2\sqrt{J_2}}\boldsymbol{P}\boldsymbol{s} \tag{12}$$

$$\frac{\partial f(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} = \frac{M^2}{3}(2\sigma_m + p_c)\boldsymbol{\delta} + 3\boldsymbol{P}\boldsymbol{s} \tag{13}$$

Poznámka: Nutnost použití škálovací matice $\boldsymbol{P}$ vyplývá z použitého maticového zápisu. V případě zápisu tenzorového by se zde tato matice nevyskytovala.

Jak je vidět z výskytu matic $\boldsymbol{\delta}$ a $\boldsymbol{s}$ ve dvou polovinách vztahu (13), lze zde uplatnit deviatoricko-volumetrické rozdělení a oddělit deformaci volumetrickou (vyjádřenou střední deformací $\dot{\varepsilon}_{V,p}$) a deviatorickou $\dot{\boldsymbol{e}}_p$ do dvou nezávislých částí zákona plastického přetváření:

[3] Na obrázku 1 se pohybujeme v prostoru p-q, takže i takovýto myšlený "směr plastické deformace" není žádný reálně představitelný směr, jedná se o poměr volumetrické (vyjádřené směrem osy p) a deviatorické (vyjádřené směrem osy q) deformace.

$$\dot{\varepsilon}_{V,p} = \dot{\lambda}\frac{\partial f}{\partial \sigma_V} = \dot{\lambda}\frac{\partial f}{\partial p}\frac{\partial p}{\partial \sigma_m}\frac{\partial \sigma_m}{\partial \sigma_V} = \dot{\lambda}M^2(2p - p_c)(-1)\left(\frac{1}{3}\right) = -\dot{\lambda}\frac{M^2}{3}(2p - p_c) \tag{14}$$

$$\dot{\boldsymbol{e}}_p = \dot{\lambda}\frac{\partial f}{\partial \boldsymbol{s}} \tag{15}$$

Konečně je třeba pro potřeby algoritmu popsaného v části 3.2, respektive toho popsaného v části 3.3, analyticky vyjádřit druhé derivace funkce plasticity. Jsou dány jako:

$$\frac{\partial^2 f(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}^2} = 3\boldsymbol{S} + \frac{2}{3}M^2\boldsymbol{\Delta} \tag{16}$$

$$\frac{\partial^2 f(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}\partial p_c} = \frac{M^2}{3}\boldsymbol{\delta} \tag{17}$$

kde

$$\boldsymbol{S} = \frac{\partial(\boldsymbol{P}\boldsymbol{s})}{\partial \boldsymbol{\sigma}} = \begin{pmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix} \tag{18}$$

$$\boldsymbol{\Delta} = \frac{\partial(\sigma_m\boldsymbol{\delta})}{\partial \boldsymbol{\sigma}} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{19}$$

2.5 Zákon zpevnění

Zpevnění je v modifikovaném modelu Cam-Clay vyjádřeno změnou parametru p_c, prekonsolidačního tlaku, která matematicky vyjadřuje změnu velikosti hlavní osy elipsy plasticity (tj. protažení elipsy ve směru osy p) (viz funkce plasticity (9) a její grafické znázornění na obrázku 1) [5]. Z fyzikálního hlediska představuje hodnota p_c největší hydrostatický tlak, kterého bylo dosaženo v předchozí historii zatěžování [4].

Na obrázku 2 je v jednoduchosti znázorněna představa o závislosti hydrostatického tlaku a volumetrické deformace[4], která stojí za uvažovaným zákonem zpevnění. Při prvotním zatěžování dochází k tzv. *normální konsolidaci*, tj. komprese se řídí linií popsanou parametrem λ^*. Pokud po dosažení určitého tlaku p_c dojde k odtížení, probíhá toto

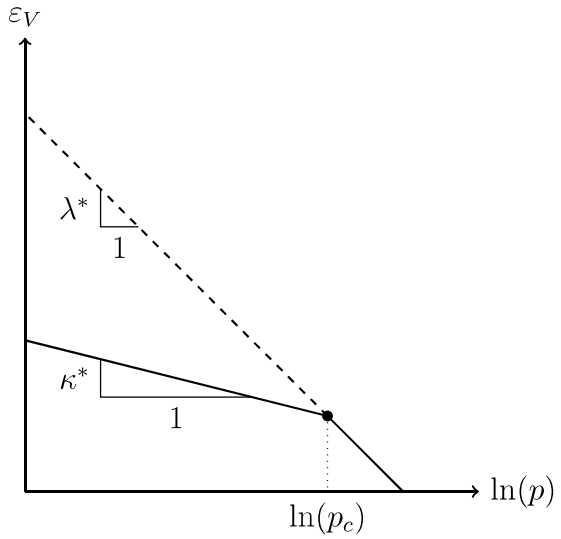
[4]Počátek tohoto grafu není v nule. Předmětem diskuze je zde konsolidace zeminy, tj. nacházíme se v oblasti, kde volumetrická deformace je záporná. Záporné je i střední napětí, tj. zde vyznačený parametr p je kladný, viz (7)

odtížení po linii popsané parametrem κ^* (*bobtnání*). Při opětovném zatěžování se deformace opět vrací po této linii bobtnání až do dosažení tlaku p_c, načež se obnoví proces normální konsolidace, jako by nikdy nebyl přerušen [4].

Parametry λ^* a κ^* vyjadřující sklony přímek v tomto grafu jsou pomocné parametry pro účely našeho zákona zpevnění. Původní idea, ve které se objevují parametry λ a κ popsané v části 2.2, popisuje závislost logaritmu hydrostatického tlaku $\ln(p)$ a pórovitosti zeminy e [4]. Pórovitost zeminy je geotechnický parametr, se kterým obecná teorie plasticity nepracuje, proto jsme místo něj použili volumetrickou deformaci. Jediná změna, kterou se toto projeví v grafu, je právě náhrada λ a κ odvozenými parametry λ^* a κ^*. Jednotlivé dvojice parametrů jsou svázány následujícími závislostmi [4]:

$$\lambda^* = \frac{\lambda}{1+e} \tag{20}$$

$$\kappa^* = \frac{\kappa}{1+e} \tag{21}$$



Obrázek 2: Odezva materiálu popsaného modifikovaným modelem Cam-Clay při zatěžování a odtěžování hydrostatickým tlakem

Abychom dostali zákon zpevnění, je naším cílem nějakým způsobem svázat vývoj prekonsolidačního tlaku p_c s vývojem volumetrické deformace. Ukazuje se, že jedinou podstatnou složkou je *plastická* volumetrická deformace. Zároveň se nám stačí zabývat normální konsolidací - prekonsolidační tlak se při bobtnání z definice nemění.

Je zjevné[5], že rozdíl mezi deformací reálně vratnou a (myšlenou) deformací, která by se vrátila, došlo-li by k dokonalé dekompresi, představuje plastickou volumetrickou defor-

[5]Z definice plastické deformace jako nevratné [6]

maci $\varepsilon_{V,p}$ [5]. Vyjádříme-li tuto myšlenku matematicky, dostáváme pro změnu plastické volumetrické deformace za časový úsek následující vztah:

$$\Delta\varepsilon_{V,p} = -\lambda^*\Delta(\ln(p)) - (-\kappa^*\Delta(\ln(p))) = -(\lambda^* - \kappa^*)\Delta(\ln(p)) \tag{22}$$

Pokud vezmeme v úvahu, že v případě normální konsolidace je vývoj okamžitého tlaku p ve výsledku totožný s vývojem prekonsolidačního tlaku p_c [5], můžeme psát, že

$$\Delta(\ln(p)) = \Delta(\ln(p_c)) = \ln((p_c)_{n+1}) - \ln((p_c)_n) = \ln\left(\frac{(p_c)_{n+1}}{(p_c)_n}\right) \tag{23}$$

Toto dosadíme do rovnice (22) a můžeme algebraicky upravovat:

$$\Delta\varepsilon_{V,p} = -(\lambda^* - \kappa^*)\ln\left(\frac{(p_c)_{n+1}}{(p_c)_n}\right) \tag{24}$$

$$-\frac{\Delta\varepsilon_{V,p}}{(\lambda^* - \kappa^*)} = \ln\left(\frac{(p_c)_{n+1}}{(p_c)_n}\right) \tag{25}$$

Připoňme vztahy (20) a (21) a nahraďme nyní pomocné parametry λ^* a κ^* z praktického hlediska vhodnějšími parametry λ a κ:

$$-\frac{1+e}{(\lambda - \kappa)}\Delta\varepsilon_{V,p} = \ln\left(\frac{(p_c)_{n+1}}{(p_c)_n}\right) \tag{26}$$

Nyní nastává čas, kdy použijeme kombinovaný parametr ϑ, jejž jsme si připravili v rovnici (5):

$$-\vartheta\Delta\varepsilon_{V,p} = \ln\left(\frac{(p_c)_{n+1}}{(p_c)_n}\right) \tag{27}$$

$$\frac{(p_c)_{n+1}}{(p_c)_n} = -\exp(\vartheta\Delta\varepsilon_{V,p}) \tag{28}$$

Drobná úprava poslední rovnice dává diskretizovanou formu zákona zpevnění pro modifikovaný model Cam-Clay:

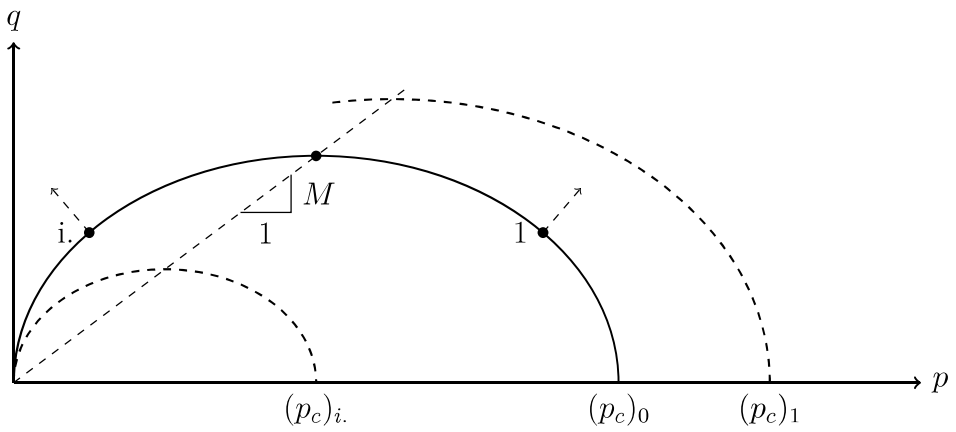
$$(p_c)_{n+1} = -(p_c)_n\exp(\vartheta\Delta\varepsilon_{V,p}) \tag{29}$$

Diferenciální forma zákona zpevnění tedy vypadá takto:

$$\dot{p}_c = -\vartheta p_c\dot{\varepsilon}_{V,p} \tag{30}$$

Vliv zpevnění/změkčení na tvar funkce plasticity je znázorněn na obrázku 3. Na počátku je prekonsolidační tlak roven $(p_c)_0$. Protože je závislý pouze na volumetrické *plastické* deformaci, po dobu, kdy je materiál v elastickém stavu, k žádným změnám nedochází. Ve chvíli, kdy se napjatost dostane do bodu označeného jako 1, dochází k plastizaci. Protože používáme sdružený zákon plastického přetváření (viz část 2.4), víme, že

"směr" plastické deformace bude kolmý na tečnu k elipse v tomto bodě (znázorněno na obrázku čárkovanou šipkou). Změna volumetrické deformace je tedy záporná[6]. Protože ϑ i p_c nabývají vždy kladných hodnot, ze zákona zpevnění (30) je zjevné, že znaménko $\dot{p}_c$ bude opačné než znaménko $\dot{\varepsilon}_{V,p}$ [5]; bude tedy docházet ke zpevnění a elipsa plasticity se zvětší. To se bude dít tak dlouho, dokud se napjatost nedostane do bodu na přímce kritického stavu, kde je tečna k elipse plasticity rovnoběžná s osou p, plastická deformace ryze deviatorická a ke zpevnění ani změkčení dále nedochází [5].



Obrázek 3: Změny elipsy plasticity v prostoru p-q při zpevnění či změkčení

Nápodobně pokud se napjatost dostane do bodu i., volumetrická plastická deformace bude stoupat, hodnota p_c se bude zmenšovat a materiál bude tedy změkčovat. Pokud tento proces bude probíhat dostatečně dlouho, opět se napjatost dostane na přímku kritického stavu a prekonsolidační tlak se zafixuje [5].

Tyto důsledky zákona zpevnění jsou konzistentní s fyzikální představou, že snižující se hodnoty plastické deformace znamenají snižující se pórovitost a tedy se zmenšuje i prostor mezi zrny jílovité zeminy, která se tak v důsledku zvýšeného tření stává pevnější, a naopak rozvolnění zeminy a zvýšení pórovitosti vede ke snížení pevnosti [5].

[6]Kladný směr osy p odpovídá záporné změně hydrostatického napětí, tj. i záporné změně volumetrické deformace

3 Použité algoritmy

Jak bude detailně popsáno v části 4, algoritmus pro plasticitní výpočet bude napsán v programovacím jazyce C++ jako modul do výpočetního softwaru pro metodu konečných prvků OOFEM [7]. Z toho vyplývají i určité požadavky na vstupy a výstupy algoritmů dané vnitřní strukturou programu.

Hlavní řešič programu OOFEM pracuje na iteračním principu [7]. Řešené konstrukci je předepisována v pseudočasových krocích jako okrajová podmínka deformace. Základním kamenem funkčnosti materiálu Cam-Clay v programu je tedy algoritmus, který pro daný materiálový bod[7] na základě požadavku řešiče z nové celkové deformace - vložené jako vstup - vypočte její rozdělení na deformaci elastickou či plastickou, stav napjatosti v materiálovém bodě a další potřebné hodnoty, jako například stav zpevnění [7].

3.1 Elastická predikce a plastická korekce

Cílem algoritmu je na základě znalosti stavu všech proměnných v předchozím zatěžovacím kroku (označovaných jako $(\cdot)_n$) a hodnoty nové celkové deformace $\boldsymbol{\varepsilon}_{n+1}$ nalézt hodnoty všech proměnných v následujícím kroku (označované jako $(\cdot)_{n+1}$) [7]. Pro úplnost poznamenejme, že zavádíme přirozě se nabízející notaci $(\Delta(\cdot))_{n+1} = (\cdot)_{n+1} - (\cdot)_n$.

Na začátku je tedy již známa celková deformace na konci kroku [7]. Z předchozích kroků přitom již známe jak dosavadní celkovou deformaci, tak její rozdělení na elastickou a plastickou část. V každém kroku se provede fáze výpočtu zvaná *elastická predikce*. Zavede se předpoklad, že veškerá nová deformace se projeví jako deformace elastická [10] [11]. Lze tedy stanovit tzv. zkušební napětí:

$$\boldsymbol{\sigma}_{n+1,tr} = \boldsymbol{D}_e(\boldsymbol{\varepsilon}_{n+1} - \boldsymbol{\varepsilon}_{p,n}) \tag{31}$$

Nyní je třeba ověřit, zda byl předpoklad správný. Protože zpevnění se projevuje pouze v závislosti na plastické deformaci (viz zákon zpevnění (30)), za tohoto předpokladu k žádnému nedojde a hodnota p_c zůstane konstantní. Pokud platí

$$f(\boldsymbol{\sigma}_{n+1,tr}, (p_c)_n) \leq 0 \tag{32}$$

předpoklad byl splněn. Uloží se

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_{n+1,tr} \tag{33}$$

$$\boldsymbol{\varepsilon}_{p,n+1} = \boldsymbol{\varepsilon}_{p,n} \tag{34}$$

[7] Takové body jsou typicky 4 na každém konečném prvku v případě 2D čtyřúhelníhové sítě, osm na každém prvku v případě 3D sítě apod. [7].

a krok skončí. V opačném případě je třeba přistoupit k tzv. *plastické korekci* neboli *návratu na plochu plasticity* [10]. Protože nyní víme, že v daném kroku musí nastat plastické přetváření, tedy

$$\Delta\boldsymbol{\varepsilon}_{p,n+1} \neq \mathbf{0} \tag{35}$$

ze zákona plastického přetváření (3) (respektive jeho diskretizované verze, ale idea je zachována) zjišťujeme že

$$(\Delta\lambda)_{n+1} \neq 0 \tag{36}$$

tedy v přímém důsledku podmínky komplementarity (4)

$$f(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) = 0 \tag{37}$$

Ze zákona plastického přetváření (3) rovněž vyplývá možnost vyjádřit, že

$$\Delta\boldsymbol{\varepsilon}_{p,n+1} = (\Delta\lambda)_{n+1}\boldsymbol{g}(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) \tag{38}$$

což umožňuje následující vyjádření pro hledanou novou hodnotu elastické deformace:

$$\boldsymbol{\varepsilon}_{e,n+1} = \boldsymbol{\varepsilon}_{n+1} - \boldsymbol{\varepsilon}_{p,n} - (\Delta\lambda)_{n+1}\boldsymbol{g}(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) \tag{39}$$

Odtud je už jen krátká cesta ke vztahu pro novou hodnotu napětí:

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{D}_e(\boldsymbol{\varepsilon}_{n+1} - \boldsymbol{\varepsilon}_{p,n} - (\Delta\lambda)_{n+1}\boldsymbol{g}(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1})) \tag{40}$$

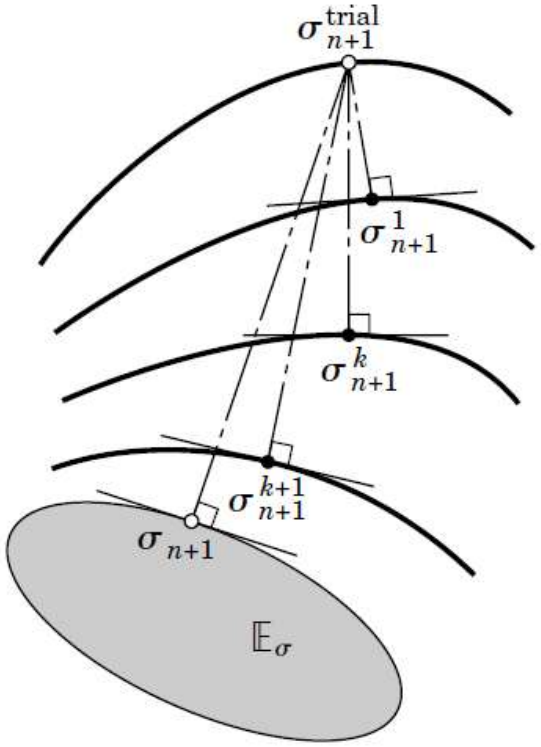
Rovnice (37) a (40) jsou základními rovnicemi pro algoritmus plastické korekce [10] [11]. Neznámými hodnotami v těchto rovnicích jsou $\boldsymbol{\sigma}_{n+1}$ a $\Delta\lambda_{n+1}$. Protože ale nyní víme, že bude docházet ke změně plastické deformace, je pravděpodobné[8] že bude docházet ke zpevnění či změkčení a bude se vyvíjet prekonsolidační tlak. Toto lze řešit dvěma způsoby: je možné vyjádřit p_c ze zákona zpevnění a dosadit do těchto dvou rovnic nebo p_c uvažovat jako další neznámou a přidat zákon zpevnění jako další rovnici. Z důvodu jednoduchosti implementace byl nakonec zvolen tento druhý způsob.

Řešitelnost soustavy rovnic závisí na jednom každém materiálovém modelu (a tedy tvaru funkce plasticity, zákona zpevnění atd.). V modifikovaném modelu Cam-Clay je analytické řešení nemožné - soustavu rovnic je třeba vyřešit pomocí numerických metod, kokrétně pomocí algoritmu zvaného *projekce na nejbližší bod* [11].

[8] Nelze říct že jisté, protože plastická deformace by také mohla být za určitých okolností (viz 2.5) ryze deviatorická

3.2 Projekce na nejbližší bod

Tato část popíše implicitní algoritmus provádějící projekci na nejbližší bod, tj. v postupných krocích mapující zkušební napětí na plochu plasticity. Jedná se o numerické řešení problému plastické korekce aplikací Newton-Raphsonovy metody na základní soustavu rovnic [10]. Geometrická ilustrace myšlenky algoritmu je zobrazena na obrázku 4[9].



Obrázek 4: Geometrická ilustrace algoritmu projekce na nejbližší bod (převzatá z [10])

Algoritmus je sám o sobě iterační a probíhá v krocích, tyto kroky, označované zde indexem i, jsou tedy vždy všechny vnořeny do jednoho zatěžovacího kroku označovaného proměnnou n.

Ještě předtím, než přistoupíme k odvození vlastního algoritmu, připomeňme zákon zpevnění, jehož diskretizovaná forma je zmíněna již v části 2.5 jako:

$$(p_c)_{n+1} = -(p_c)_n \exp(\vartheta(\Delta\varepsilon_{V,p})_{n+1}) \tag{41}$$

Tento vztah je teoreticky použitelný. Lze si[10] ale zjednodušit práci, pokud zákon zpevnění v jeho diferenciální formě

$$\dot{p}_c = -\vartheta p_c \dot{\varepsilon}^p_V \tag{42}$$

integrujeme numericky jako

[9] Jednotlivé kroky iterace označované zde indexem k jsou v textu označovány i.

[10] za cenu nevelké újmy na přesnosti a času konvergence

$$(p_c)_{n+1} - (p_c)_n = -\vartheta \Delta\varepsilon_{V,p}(p_c)_{n+1} \tag{43}$$

$$(p_c)_{n+1} + \vartheta \Delta\varepsilon_{V,p}(p_c)_{n+1} = (p_c)_n \tag{44}$$

K vyjádření změny volumetrické deformace použijeme nyní deviatoricko-volumetrické rozdělení zákona plastického přetváření, které bylo uvedeno ve vztahu (14), a získáváme:

$$(p_c)_{n+1} = (p_c)_n + \vartheta(\Delta\lambda)_{n+1}\frac{M^2}{3}(2p - (p_c)_{n+1})(p_c)_{n+1} \tag{45}$$

Stejně jako zákon zpevnění, i rovnici pro $\boldsymbol{\sigma}_{n+1}$ je vhodné vyjádřit ve tvaru změny. Použijeme zde změnu celkové deformace:

$$(\Delta\varepsilon)_{n+1} = \boldsymbol{D}_{n+1}^{-1}(\boldsymbol{\sigma}_{n+1} - \boldsymbol{\sigma}_n) + (\Delta\lambda)_{n+1}\frac{\partial f}{\partial \boldsymbol{\sigma}_{n+1}} \tag{46}$$

Nyní už máme i poslední rovnici, shrňme tedy soustavu rovnic, kterou máme za cíl splnit:

$$(\Delta\varepsilon)_{n+1} = \boldsymbol{D}_{n+1}^{-1}(\boldsymbol{\sigma}_{n+1} - \boldsymbol{\sigma}_n) + (\Delta\lambda)_{n+1}\frac{\partial f}{\partial \boldsymbol{\sigma}_{n+1}} \tag{47}$$

$$f(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) = 0 \tag{48}$$

$$(p_c)_{n+1} = (p_c)_n + \vartheta(\Delta\lambda)_{n+1}\frac{M^2}{3}(2p - (p_c)_{n+1})(p_c)_{n+1} \tag{49}$$

Protože napětí má šest složek, jedná se o soustavu osmi nelineárních rovnic o osmi neznámých. Neznámými jsou zde $\boldsymbol{\sigma}_{n+1}$, $(\Delta\lambda)_{n+1}$ a $(p_c)_{n+1}$. Všechny ostatní hodnoty jsou nám již na počátku iterace známé. Jak již bylo zmíněno, řešení nebude přímočaré. Namísto toho zavedeme počáteční odhady neznámých, pro něž použijeme zkušební stav z elastické predikce:

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_{n+1,tr} \tag{50}$$

$$(\Delta\lambda)_{n+1} = 0 \tag{51}$$

$$(p_c)_{n+1} = (p_c)_n \tag{52}$$

S pomocí těchto odhadů můžeme nyní převést naše tři[11] rovnice do reziduálního tvaru, a tak získáme tři vyčíslitelné hodnoty, které bude naším cílem postupně v krocích algoritmu přiblížit k nule.

[11] Rovnic a neznámých, jak bylo poznamenáno výše, je z praktického hlediska osm. Nicméně pro účely odvození tohoto algoritmu budeme mluvit o třech neznámých, z nichž jedna je sloupcová matice, a to čistě z důvodu praktičnosti a srozumitelnosti vzhledem ke třem rovnicím, které zapisujeme.

Tyto tři reziduální rovnice jsou na začátku jednoho každého $(i+1)$-ého kroku:

$$\boldsymbol{R}^i_\sigma \equiv \boldsymbol{D}^{-1}_{n+1}(\boldsymbol{\sigma}_{n+1,i} - \boldsymbol{\sigma}_n) - (\Delta\boldsymbol{\varepsilon})_{n+1} + (\Delta\lambda)_{n+1,i}\frac{\partial f}{\partial \boldsymbol{\sigma}_{n+1}} = \boldsymbol{0} \tag{53}$$

$$R^i_f \equiv f(\boldsymbol{\sigma}_{n+1,i}, (p_c)_{n+1,i}) = 0 \tag{54}$$

$$R^i_p \equiv (p_c)_{n+1,i} - \vartheta\frac{M^2}{3}(\Delta\lambda)_{n+1,i}(2p_{n+1,i} - (p_c)_{n+1,i})(p_c)_{n+1,i} - (p_c)_n = 0 \tag{55}$$

Pomocí Newtonovy metody nyní reziduály linearizujeme vzhledem ke všem třem neznámým. Jako základ je použita aktuálně známá i-tá hodnota $(\boldsymbol{R}^i_\sigma, R^i_f, R^i_p)$. Dostáváme linearizovaný problém:

$$\boldsymbol{R}^i_\sigma + \frac{\partial \boldsymbol{R}^i_\sigma}{\partial \boldsymbol{\sigma}_{n+1}}\delta\boldsymbol{\sigma} + \frac{\partial \boldsymbol{R}^i_\sigma}{\partial\Delta\lambda}\delta\Delta\lambda + \frac{\partial \boldsymbol{R}^i_\sigma}{\partial p_c}\delta p_c = 0 \tag{56}$$

$$R^i_f + \frac{\partial R^i_f}{\partial \boldsymbol{\sigma}_{n+1}}\delta\boldsymbol{\sigma} + \frac{\partial R^i_f}{\partial\Delta\lambda}\delta\Delta\lambda + \frac{\partial R^i_f}{\partial p_c}\delta p_c = 0 \tag{57}$$

$$R^i_p + \frac{\partial R^i_p}{\partial \boldsymbol{\sigma}_{n+1}\delta\boldsymbol{\sigma}} + \frac{\partial R^i_p}{\partial\Delta\lambda}\delta\Delta\lambda + \frac{\partial R^i_p}{\partial p_c}\delta p_c = 0 \tag{58}$$

Neznámými v tomto dílčím linearizovaném problému jsou teď $\delta\boldsymbol{\sigma}$, $\delta\Delta\lambda$ a δp_c. Ostatní hodnoty jsou nám buď známé (n-té hodnoty a konstanty) nebo pracujeme s jejich i-tými odhady ($(n+1)$-vé hodnoty). Pro přehlednost a srozumitelnost potlačme již v následujícím textu indexy i a tam, kde to neubere na srozumitelnosti (tj. v označení derivací a u $(\Delta\lambda)_{n+1,i}$) i indexy $n+1$.

Přepišme nyní rovnice (56), (57) a (58) do formy maticového zápisu a získejme tak vyjádření

$$\begin{bmatrix} \boldsymbol{R}^\sigma_{,\sigma} & \boldsymbol{R}^\sigma_{,f} & \boldsymbol{R}^\sigma_{,p_c} \\ \boldsymbol{R}^f_{,\sigma} & R^f_{,f} & R^f_{,p_c} \\ \boldsymbol{R}^p_{,\sigma} & R^p_{,f} & R^p_{,p_c} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\sigma} \\ \delta\Delta\lambda \\ \delta p_c \end{bmatrix} = \begin{bmatrix} -\boldsymbol{R}^i_\sigma \\ -R^i_f \\ -R^i_p \end{bmatrix} \tag{59}$$

kde

$R^x_{,y}$ je zkrácený zápis pro $\frac{\partial R_x}{\partial y}$

Hodnoty všech devíti derivací v této matici jsou známé a je možné je pro aktuální krok vyčíslit. Podotkněme ještě, že, jak bylo zmíneno v části 2.3, uvažujeme matici elastické tuhosti jako nezávislou na hodnotě napětí. Díky tomu je poslední člen ve vyjádření pro $\boldsymbol{R}^\sigma_{,\sigma}$ nulový. Pokud by matice měla na napětí být závislá, zde by se to projevilo.

Derivace jsou:

$$\boldsymbol{R}^{\sigma}_{,\sigma} = \frac{\partial \boldsymbol{R}^{i}_{\sigma}}{\partial \boldsymbol{\sigma}} = \boldsymbol{D}^{-1}_{n+1} + \Delta\lambda \frac{\partial^2 f}{\partial \boldsymbol{\sigma}^2} + \frac{\partial \boldsymbol{D}^{-1}_{n+1}}{\partial \boldsymbol{\sigma}}(\boldsymbol{\sigma}_{n+1} - \boldsymbol{\sigma}_n) \tag{60}$$

$$\boldsymbol{R}^{\sigma}_{,f} = \frac{\partial \boldsymbol{R}^{i}_{\sigma}}{\partial \Delta\lambda} = \frac{\partial f}{\partial \boldsymbol{\sigma}} \tag{61}$$

$$\boldsymbol{R}^{\sigma}_{,p_c} = \frac{\partial \boldsymbol{R}^{i}_{\sigma}}{\partial p_c} = \Delta\lambda \frac{\partial^2 f}{\partial \boldsymbol{\sigma} \partial p_c} = \Delta\lambda \frac{M^2}{3}\boldsymbol{\delta} \tag{62}$$

$$\boldsymbol{R}^{f}_{,\sigma} = \frac{\partial R^{i}_{f}}{\partial \boldsymbol{\sigma}} = \frac{\partial f}{\partial \boldsymbol{\sigma}} = \frac{M^2}{3}(2\sigma_{m,n+1} + p_c)\boldsymbol{\delta} + 3\boldsymbol{s}_{n+1} \tag{63}$$

$$R^{f}_{,f} = \frac{\partial R^{i}_{f}}{\partial \Delta\lambda} = 0 \tag{64}$$

$$R^{f}_{,p_c} = \frac{\partial R^{i}_{f}}{\partial p_c} = \frac{\partial f}{\partial p_c} = -M^2 p_{n+1} \tag{65}$$

$$\boldsymbol{R}^{p}_{,\sigma} = \frac{\partial R^{i}_{p}}{\partial \boldsymbol{\sigma}} = \frac{2}{9}\vartheta M^2 \Delta\lambda (p_c)_{n+1}\boldsymbol{\delta} \tag{66}$$

$$R^{p}_{,f} = \frac{\partial R^{i}_{p}}{\partial \Delta\lambda} = -\vartheta \frac{M^2}{3}(2p_{n+1} - (p_c)_{n+1})(p_c)_{n+1} \tag{67}$$

$$R^{p}_{,p_c} = \frac{\partial R^{i}_{p}}{\partial p_c} = 1 - 2\vartheta \frac{M^2}{3}\Delta\lambda(p_{n+1} - (p_c)_{n+1}) \tag{68}$$

Nyní, když máme číselné hodnoty matice ze vztahu (59), zbývá za jejich pomoci vyčíslit neznámé $\delta\boldsymbol{\sigma}$, $\delta\Delta\lambda$ a δp_c. Bylo by možno toto provést číselnou inverzí matice, avšak vzhledem k tomu, že se jedná o matici 8×8, jeví se jako rozumnější provést odvození přímých analytických vztahů pro jednotlivé neznámé. Převeďme nyní rovnici (58) do následující podoby:

$$\delta p_c = -\frac{1}{R^{p}_{,p_c}}\left(R^{i}_{p} + \boldsymbol{R}^{p}_{,\sigma}\delta\boldsymbol{\sigma} + R^{p}_{,f}\delta\Delta\lambda\right) \tag{69}$$

a dosaďme tento vztah do rovnice (56):

$$\boldsymbol{R}^{\sigma}_{,\sigma}\delta\boldsymbol{\sigma} + \boldsymbol{R}^{\sigma}_{,f}\delta\Delta\lambda - \frac{1}{R^{p}_{,p_c}}\boldsymbol{R}^{\sigma}_{,p_c}\left(R^{i}_{p} + \boldsymbol{R}^{p}_{,\sigma}\delta\boldsymbol{\sigma} + R^{p}_{,f}\delta\Delta\lambda\right) = -\boldsymbol{R}^{i}_{\sigma} \tag{70}$$

Úpravami získáme:

$$\delta\boldsymbol{\sigma} = -\boldsymbol{C}\left(\boldsymbol{R}^{i}_{\sigma} - \frac{R^{i}_{p}}{R^{p}_{,p_c}}\boldsymbol{R}^{\sigma}_{,p_c} + \left(\boldsymbol{R}^{\sigma}_{,f} - \frac{R^{p}_{,f}}{R^{p}_{,p_c}}\boldsymbol{R}^{\sigma}_{,p_c}\right)\delta\Delta\lambda\right) \tag{71}$$

kde

$$\boldsymbol{C} = \left(\boldsymbol{R}^{\sigma}_{,\sigma} - \frac{1}{R^{p}_{,p_c}}(\boldsymbol{R}^{\sigma}_{,p_c})^T\boldsymbol{R}^{p}_{,\sigma}\right)^{-1} \tag{72}$$

Nyní můžeme dosadit za obě neznámé δp_c and $\delta\boldsymbol{\sigma}$ do rovnice (57) pro $\delta\Delta\lambda$ a získat tak:

$$R^f_{,f}\delta\Delta\lambda = -R^i_f + (\boldsymbol{R}^\sigma_{,f})^T \boldsymbol{C}\left(\boldsymbol{R}^i_\sigma - \frac{1}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}R^i_p + \left(\boldsymbol{R}^\sigma_{,f} - \frac{1}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}R^p_f\right)\delta\Delta\lambda\right) + \\ + \frac{R^f_{,p_c}}{R^p_{,p_c}}\left(R^i_p + \boldsymbol{R}^p_{,\sigma}\delta\boldsymbol{\sigma} + R^p_{,f}\delta\Delta\lambda\right) \tag{73}$$

Ve výrazu se však stále objevuje $\delta\boldsymbol{\sigma}$ a my tedy musíme provést substituci opakovaně:

$$R^f_{,f}\delta\Delta\lambda = -R^i_f + \left(\boldsymbol{R}^\sigma_{,f} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C} \\ \left(\boldsymbol{R}^i_\sigma - \frac{1}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}R^i_p + \left(\boldsymbol{R}^\sigma_{,f} - \frac{1}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}R^p_f\right)\delta\Delta\lambda\right) + \\ + \frac{R^f_{,p_c}}{R^p_{,p_c}}\left(R^i_p + R^p_{,f}\delta\Delta\lambda\right) \tag{74}$$

Díky těmto opakovaným substitucím můžeme vytkout některé opakující se výrazy v závorkách, oddělit členy obsahující $\delta\Delta\lambda$ a vzhledem k tomu, že $\delta\Delta\lambda$ je skalár, bez obav vydělit celou rovnici vytknutým výrazem. Získáváme, jak bylo naším cílem, přímé vyjádření pro $\delta\Delta\lambda$:

$$\delta\Delta\lambda = \frac{-R^i_f + \left(\boldsymbol{R}^\sigma_{,f} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C}\left(\boldsymbol{R}^i_\sigma - \frac{R^i_p}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}\right) + \frac{R^f_{,p_c}}{R^p_{,p_c}}R^i_p}{R^f_{,f} - \left(\boldsymbol{R}^\sigma_{,f} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \frac{R^p_{,f}}{R^p_{,p_c}}\boldsymbol{R}^\sigma_{,p_c}\right) - \frac{R^f_{,p_c}}{R^p_{,p_c}}R^p_{,f}} \tag{75}$$

Po vypočtení $\delta\Delta\lambda$ pomocí vztahu (75) získáme dosazením této hodnoty do vztahu (71) i $\delta\boldsymbol{\sigma}$ a dosazením obou do (69) konečně i δp_c. Jsme na konci i-tého kroku a můžeme pomocí vypočtených rozdílů aktualizovat odhady hlavních neznámých:

$$\boldsymbol{\sigma}_{n+1,i+1} = \boldsymbol{\sigma}_{n+1,i} + \delta\boldsymbol{\sigma} \tag{76}$$

$$(\Delta\lambda)_{n+1,i+1} = (\Delta\lambda)_{n+1,i} + \delta\Delta\lambda \tag{77}$$

$$(p_c)_{n+1,i+1} = (p_c)_{n+1,i} + \delta p_c \tag{78}$$

Nyní použijeme tyto nové odhady k aktualizaci reziduálů přepočtením vztahů (53) - (55). Pokud hodnoty těchto stále nejsou nulové (nebo, v praktické aplikaci, se nepohybují v předem nastavených tolerancích), vracíme se zpět k rovnicím (56) - (58) a následuje další krok v projekci na nejbližší bod. V případě, že nulové jsou, je plastická korekce hotova a my můžeme uložit hodnoty $\boldsymbol{\sigma}_{n+1}$, $(\Delta\lambda)_{n+1,i+1}$ a $(p_c)_{n+1,i+1}$ jako konečné a dle potřeby z nich dopočítat další stavové proměnné, které nás zajímají (typicky deformace).

Je vhodné poznamenat, že algoritmus popsaný výše je vytvořen s cílem maximální obecnosti. U jiných materiálových modelů než je modifikovaný Cam-Clay by se veškeré změny omezily pouze na tvary členů v symbolické matici a symbolické pravé straně ve vztahu (59). Existuje možnost vytvoření algoritmu specificky určeného pouze pro model

Cam-Clay a využívajícího jednoduchost deviatoricko-volumetrického rozdělení v tomto modelu; nástin takového postupu je uveden v dodatku B.

3.3 Algoritmická tuhost

Pro umožnění výpočtů na úrovni konstrukce je třeba na požádání stanovit novou hodnotu pro elastoplastický tečný modul $\boldsymbol{D_{ep}}$. Tento modul vyjadřuje dle vztahu

$$\boldsymbol{D_{ep}} = \frac{\mathrm{d}\boldsymbol{\sigma}_{n+1}}{\mathrm{d}\boldsymbol{\epsilon}^{e,tr}_{n+1}} \tag{79}$$

závislost napětí na elastické deformaci v aktuálním stavu. Algoritmus pro výpočet tohoto modulu bude popsán v této kapitole.

3.3.1 Princip algoritmu

Základní podmínky (53) - (55) návratového algoritmu lze považovat za soustavu rovnic ve tvaru (viz [11], str. 238)

$$\left\{\begin{array}{c} \boldsymbol{\varepsilon}^{e}_{n+1} - \boldsymbol{\varepsilon}^{e,trial}_{n+1} + \Delta\lambda \boldsymbol{N}_{n+1} = 0 \\ f(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) = 0 \\ R_p((p_c)_{n+1}, \Delta\lambda) = 0 \end{array}\right\} \tag{80}$$

kde pro případ námi uvažovaného materiálového modelu

$\boldsymbol{\varepsilon}^{e}_{n+1}$ je hodnota vyjádřená jako $\boldsymbol{D}^{-1}_{n+1}\boldsymbol{\sigma}_{n+1}$
$\boldsymbol{\varepsilon}^{e,trial}_{n+1}$ je vyjádřeno jako $(\boldsymbol{D}^{-1}_{n+1}\boldsymbol{\sigma}_{n} + \Delta\boldsymbol{\varepsilon})$
$\boldsymbol{N}_{n+1}$ je vektor plastického toku ve tvaru $\frac{\partial f}{\partial \boldsymbol{\sigma}_{n+1}}$
$f(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1})$ je funkce plasticity analogicky k jejímu vyjádření v podobě reziduálu R_f v rovnici (54)
$R_p((p_c)_{n+1}, \Delta\lambda)$ je reziduál zpevnění podle rovnice (55)

Naším cílem je z tohoto vztahu vyjádřit závislost mezi $\mathrm{d}\boldsymbol{\sigma}_{n+1}$ a $\mathrm{d}\boldsymbol{\epsilon}^{e,tr}_{n+1}$. Jediná rovnice ze soustavy (80) obsahující $\mathrm{d}\boldsymbol{\varepsilon}^{e,tr}_{n+1}$ je rovnice první. Pokud v ní převedeme toto na pravou stranu jako neznámou, dostáváme

$$\left\{\begin{array}{c} \boldsymbol{\varepsilon}^{e}_{n+1} + \Delta\lambda \boldsymbol{N}_{n+1} \\ f(\boldsymbol{\sigma}_{n+1}, (p_c)_{n+1}) \\ R_p((p_c)_{n+1}, \Delta\lambda) \end{array}\right\} = \begin{bmatrix} \boldsymbol{\epsilon}^{e,tr}_{n+1} \\ 0 \\ 0 \end{bmatrix} \tag{81}$$

Tento problém lze linearizovat vzhledem k neznámým $\boldsymbol{\sigma}_{n+1}$, $\Delta\lambda$ a p_c obdobně způsobu, jakým byly linearizovány rovnice (53) - (55) v části 3.2, čímž byl získán symbolický maticový vztah (59). Zde se od tohoto předchozího případu odlišuje pravá strana a částečně i tvar první rovnice (postrádá $\mathrm{d}\boldsymbol{\varepsilon}^{e,tr}_{n+1}$). To ale nemá žádný vliv, vezmeme-li v úvahu, že

$$\frac{\partial}{\partial \boldsymbol{\sigma}_{n+1}} \left(\boldsymbol{D}_{n+1}^{-1} \boldsymbol{\sigma}_n + \Delta\boldsymbol{\varepsilon} \right) = 0 \tag{82}$$

$$\frac{\partial}{\partial \Delta\lambda} \left(\boldsymbol{D}_{n+1}^{-1} \boldsymbol{\sigma}_n + \Delta\boldsymbol{\varepsilon} \right) = 0 \tag{83}$$

$$\frac{\partial}{\partial p_c} \left(\boldsymbol{D}_{n+1}^{-1} \boldsymbol{\sigma}_n + \Delta\boldsymbol{\varepsilon} \right) = 0 \tag{84}$$

a tedy derivace první rovnice podle těchto neznámých zůstanou shodné se vztahy (60), (61) a (62). Můžeme tedy použít celou symbolickou matici ze vztahu (59) a dostáváme vyjádření

$$\begin{bmatrix} \boldsymbol{R}_{,\sigma}^{\sigma} & \boldsymbol{R}_{,f}^{\sigma} & \boldsymbol{R}_{,p_c}^{\sigma} \\ \boldsymbol{R}_{,\sigma}^{f} & R_{,f}^{f} & R_{,p_c}^{f} \\ \boldsymbol{R}_{,\sigma}^{p} & R_{,f}^{p} & R_{,p_c}^{p} \end{bmatrix} \begin{bmatrix} \mathrm{d}\boldsymbol{\sigma} \\ \mathrm{d}\Delta\lambda \\ \mathrm{d}p_c \end{bmatrix} = \begin{bmatrix} \mathrm{d}\boldsymbol{\varepsilon} \\ 0 \\ 0 \end{bmatrix} \tag{85}$$

Nyní nastupuje hlavní myšlenka algoritmu, kdy provedeme inverzi matice ve vztahu (85) tak, abychom dostali vztah [11]

$$\begin{bmatrix} \mathrm{d}\boldsymbol{\sigma} \\ \mathrm{d}\Delta\lambda \\ \mathrm{d}p_c \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_{11} & \boldsymbol{D}_{12} & \boldsymbol{D}_{13} \\ \boldsymbol{D}_{21} & D_{22} & D_{23} \\ \boldsymbol{D}_{31} & D_{32} & D_{33} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\varepsilon} \\ 0 \\ 0 \end{bmatrix} \tag{86}$$

Pokud provedeme vynásobení prvního řádku v tomto vztahu, je na první pohled vidět, že

$$\mathrm{d}\boldsymbol{\sigma} = \boldsymbol{D}_{11} \mathrm{d}\boldsymbol{\varepsilon} \tag{87}$$

a proto tedy

$$\boldsymbol{D}_{11} = \boldsymbol{D}_{ep} \tag{88}$$

Ostatní prvky invertované matice, třebaže některé z nich vyjadřují určité závislosti, jsou pro naše účely nepotřebné [11].

3.3.2 Výpočetní vztah pro algoritmickou tuhost

Následuje problém, jak se dostat od vztahu (85) ke vztahu (86). Z rozměrů skalárů a tenzorů v matici, stejně jako z rozměrů neznámých (šest členů v matici napětí a dvě skalární hodnoty) vyplývá, že bychom se po číselném vyjádření zabývali maticí o rozměru 8×8. Teoreticky by bylo možno tuto matici číselně invertovat a poté vyjmout z výsledku prvních 6×6 prvků jako námi hledanou matici $\boldsymbol{D}_{ep}$. Toto by ale bylo značně (a zejména nepotřebně) výpočetně náročné. Lze odvodit výpočetně jednodušší vztah pro přímé vyjádření $\boldsymbol{D}_{ep}$, a to bude předmětem této části.

Rozepišme soustavu rovnic vyjádřenou vztahem (85) a mějme přitom na paměti, že dle (64) vypadne člen $R^f_{,f}$. Získáváme následující tři rovnice:

$$\boldsymbol{R}^{\sigma}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + \boldsymbol{R}^{\sigma}_{,f}\mathrm{d}\lambda + \boldsymbol{R}^{\sigma}_{,p_c}\mathrm{d}p_c = \mathrm{d}\boldsymbol{\varepsilon}^{tr} \tag{89}$$

$$\boldsymbol{R}^{f}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + R^{f}_{,p_c}\mathrm{d}p_c = 0 \tag{90}$$

$$\boldsymbol{R}^{p}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + R^{p}_{,f}\mathrm{d}\lambda + R^{p}_{,p_c}\mathrm{d}p_c = 0 \tag{91}$$

Poslední rovnici lze převést na tvar

$$\mathrm{d}p_c = -\frac{1}{R^{p}_{,p_c}}\left(\boldsymbol{R}^{p}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + R^{p}_{,f}\mathrm{d}\lambda\right) \tag{92}$$

který následně můžeme dosadit do rovnice první:

$$\boldsymbol{R}^{\sigma}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + \boldsymbol{R}^{\sigma}_{,f}\mathrm{d}\lambda + \boldsymbol{R}^{\sigma}_{,p_c}\mathrm{d}p_c = \mathrm{d}\boldsymbol{\varepsilon}^{tr} \tag{93}$$

$$\boldsymbol{R}^{\sigma}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + \boldsymbol{R}^{\sigma}_{,f}\mathrm{d}\lambda - \boldsymbol{R}^{\sigma}_{,p_c}\frac{1}{R^{p}_{,p_c}}\left(\boldsymbol{R}^{p}_{,\sigma}\mathrm{d}\boldsymbol{\sigma} + R^{p}_{,f}\mathrm{d}\lambda\right) = \mathrm{d}\boldsymbol{\varepsilon}^{tr} \tag{94}$$

To nám dává přímé vyjádření pro $\mathrm{d}\boldsymbol{\sigma}$:

$$\mathrm{d}\boldsymbol{\sigma} = \boldsymbol{C}\left(\mathrm{d}\boldsymbol{\varepsilon}^{tr} - \left(\boldsymbol{R}^{\sigma}_{,f} - \boldsymbol{R}^{\sigma}_{,p_c}\frac{R^{p}_{,f}}{R^{p}_{,p_c}}\right)\mathrm{d}\lambda\right) \tag{95}$$

kde

$$\boldsymbol{C} = \left(\boldsymbol{R}^{\sigma}_{,\sigma} - \frac{1}{R^{p}_{,p_c}}\boldsymbol{R}^{\sigma}_{,p_c}(\boldsymbol{R}^{p}_{,\sigma})^T\right)^{-1} \tag{96}$$

Povšimněme si ,že až doposud se vyjádření velmi podobají odvozování přímých vztahů pro výpočet návratu na plochu plasticity v části 3.2. Zde jsou vztahy pouze o něco jednodušší v důsledku absence dvou pravých stran. Pomocná matice $\boldsymbol{C}$ podle vyjádření (96) se shoduje se shodně označenou maticí ve vyjádření (72).

Zbyla nám druhá rovnice. Pokud do ní dosadíme za $\mathrm{d}p_c$ a $\mathrm{d}\boldsymbol{\sigma}$, získáváme vztah

$$\begin{aligned}(\boldsymbol{R}^{f}_{,\sigma})^T\boldsymbol{C}\left(\mathrm{d}\boldsymbol{\varepsilon}^{tr} - \left(\boldsymbol{R}^{\sigma}_{,f} - \boldsymbol{R}^{\sigma}_{,p_c}\frac{R^{p}_{,f}}{R^{p}_{,p_c}}\right)\mathrm{d}\lambda\right) - \\ -\frac{R^{f}_{,p_c}}{R^{p}_{,p_c}}\left((\boldsymbol{R}^{p}_{,\sigma})^T\boldsymbol{C}\left(\mathrm{d}\boldsymbol{\varepsilon}^{tr} - \left(\boldsymbol{R}^{\sigma}_{,f} - \boldsymbol{R}^{\sigma}_{,p_c}\frac{R^{p}_{,f}}{R^{p}_{,p_c}}\right)\mathrm{d}\lambda\right) + R^{p}_{,f}\mathrm{d}\lambda\right) = 0\end{aligned} \tag{97}$$

který po vytknutí opakujících se členů dává následující součin:

$$\begin{aligned}\left(-\left(\boldsymbol{R}^{f}_{,\sigma} - \frac{R^{f}_{,p_c}}{R^{p}_{,p_c}}\boldsymbol{R}^{p}_{,\sigma}\right)^T\boldsymbol{C}\left(\boldsymbol{R}^{\sigma}_{,f} - \boldsymbol{R}^{\sigma}_{,p_c}\frac{R^{p}_{,f}}{R^{p}_{,p_c}}\right) - \frac{R^{f}_{,p_c}}{R^{p}_{,p_c}}R^{p}_{,f}\right)\mathrm{d}\lambda = \\ = -\left(\boldsymbol{R}^{f}_{,\sigma} - \frac{R^{f}_{,p_c}}{R^{p}_{,p_c}}\boldsymbol{R}^{p}_{,\sigma}\right)^T\boldsymbol{C}\mathrm{d}\boldsymbol{\varepsilon}^{tr}\end{aligned} \tag{98}$$

Vyjádřit $\mathrm{d}\lambda$ už zde není problém:

$$\mathrm{d}\lambda = \frac{\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C}\mathrm{d}\boldsymbol{\varepsilon}^{tr}}{\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \boldsymbol{R}^\sigma_{,p_c}\frac{R^p_{,f}}{R^p_{,p_c}}\right) + \frac{R^f_{,p_c}}{R^p_{,p_c}}R^p_{,f}} \tag{99}$$

Vraťme se nyní k rovnici (95) a dosaďme do ní toto vyjádření. Úpravou a roznásobením získáváme přímý vztah mezi $\mathrm{d}\boldsymbol{\sigma}$ a $\mathrm{d}\boldsymbol{\varepsilon}$, což bylo cílem celého algoritmu.

$$\mathrm{d}\boldsymbol{\sigma} = \left(\boldsymbol{C} - \frac{\left[\boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \boldsymbol{R}^\sigma_{,p_c}\frac{R^p_{,f}}{R^p_{,p_c}}\right)\right]\left[\boldsymbol{C}\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)\right]^T}{\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)^T \boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \boldsymbol{R}^\sigma_{,p_c}\frac{R^p_{,f}}{R^p_{,p_c}}\right) + \frac{R^f_{,p_c}}{R^p_{,p_c}}R^p_{,f}}\right)\mathrm{d}\boldsymbol{\varepsilon}^{tr} \tag{100}$$

Můžeme tedy tvrdit, že elastoplastická tečná matice algoritmické tuhosti se vypočítá dle vzorce

$$\boldsymbol{D}_{ep} = \boldsymbol{C} - \frac{\left[\boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \boldsymbol{R}^\sigma_{,p_c}\frac{R^p_{,f}}{R^p_{,p_c}}\right)\right]\left[\boldsymbol{C}\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)\right]^T}{\left(\boldsymbol{R}^f_{,\sigma} - \frac{R^f_{,p_c}}{R^p_{,p_c}}\boldsymbol{R}^p_{,\sigma}\right)\boldsymbol{C}\left(\boldsymbol{R}^\sigma_{,f} - \boldsymbol{R}^\sigma_{,p_c}\frac{R^p_{,f}}{R^p_{,p_c}}\right) + \frac{R^f_{,p_c}}{R^p_{,p_c}}R^p_{,f}} \tag{101}$$

Lze si povšimnout, že tato matice je nesymetrická. Na vině je zde zákon zpevnění v námi uvažovaném modelu. Pokud bychom uvažovali dokonalou plasticitu a fakt, že

$$\boldsymbol{R}^\sigma_{,f} = \boldsymbol{R}^f_{,\sigma} = \frac{\partial f}{\partial \boldsymbol{\sigma}_{n+1}} \tag{102}$$

matice tuhosti se nám zredukuje na

$$\boldsymbol{D}_{ep} = \boldsymbol{C} - \frac{[\boldsymbol{C}\left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)][\boldsymbol{C}\frac{\partial f}{\partial \boldsymbol{\sigma}}]^T}{\frac{\partial f}{\partial \boldsymbol{\sigma}}\boldsymbol{C}\frac{\partial f}{\partial \boldsymbol{\sigma}}} \tag{103}$$

což už je dle očekávání symetrické vyjádření.

4 Implementace

Tato kapitola popíše software OOFEM, principy, na nichž pracuje, relevantní části jeho vnitřní struktury a uvede příklady kódu, kterým jsou implementovány algoritmy popsané v kapitole 3.

4.1 Software OOFEM

OOFEM[12] je univerzální software pro metodu konečných prvků vyvíjený na Fakultě stavební ČVUT. Je psán v programovacím jazyce C++ a s důrazem na objektově orientované programování, které tento jazyk umožňuje [7] [13]. Cílem tohoto přístupu je velká modularita a možnost jednoduché implementace nových funkcí, jak je také předmětem této práce.

4.1.1 Objektová struktura

Principem objektově orientovaného programování je myšlenka striktní organizace proměnných a funkcí do logických celků, *objektů* neboli *tříd*. Do proměnných se potom dají ukládat jednotlivé instance těchto objektů, které reprezentují celý soubor uvnitř uložených hodnot, čímž získávají řadu vlastností a funkcí [13].

Příkladem takového přístupu konkrétně pro OOFEM je způsob implementace jednotlivých materiálových modelů. Ty jsou napsány s přísnou aplikací principu dědičnosti - tj. existuje řada tříd, z nichž každá je rozšířením (*potomkem*) té předchozí, má veškeré její funkce a vnitřní parametry, k nimž přidává své, nové [13]. Konkrétně třída `CamClayMat`, která je vlastně jediným produktem této práce, je potomkem třídy `StructuralMaterial`, která je sama potomkem `Material`, a ta potomkem základní třídy `FEMComponent` [7].

Předností tohoto přístupu v objektově orientovaném programování je následná obecnost zacházení s instancemi takovýchto tříd, kdy jakýkoli potomek zaručeně má všechny vlastnosti kteréhokoli ze svých předků a lze s ním bez jakékoli újmy zacházet jako s tímto předkem, a to bez ohledu na to, který z možná celé řady potomků to ve skutečnosti je [13]. Předvedeno opět na příkladu, pokud má některá část programu OOFEM získat instanci materiálu a přiřadit ji určitému konečnému prvku, zachází s tímto objektem jako s objektem třídy `Material`, aniž by se zajímala, zda to ve skutečnosti je `CamClayMat` nebo například `MisesMat`. Když byl ten samý objekt vytvářen, bylo s ním dokonce zacházeno pouze jako s třídou `FEMComponent`, zastřešujícím typem, který obsahuje jen málo navíc než vnitřní pořadové číslo a objekt typu `Domain` sloužící ke komunikaci s ostatními objekty uvnitř programu [7].

[12]Zkratka pro *Object Oriented Finite Element Method Solver*, tj. Objektově orientovaný řešič pro metodu konečných prvků [7]

4.1.2 Práce s programem

Zadání úlohy do programu OOFEM se provádí pomocí speciálního vstupního souboru (soubor typu `.in`). Jedná se o textový skript, který udává všechny potřebné parametry pro výpočet v určité standardizované podobě, pro kterou je ve vnitřní struktuře programu připraven algoritmus, jenž ji interpretuje a získaná data předává ostatním modulům k využití [7]. Příklad takového vstupního souboru je uveden ve výpisu 1.

```
1  camclay.out
2  test of cam clay material model
3  NonLinearStatic nsteps 5 renumber 1 nmodules 2 controllmode 1
       rtolv 1.e-6 stiffMode 0 manrmsteps 1 maxiter 300
     initialGuess 0
4  vtkxml tstep_all domain_all primvars 1 1 vars 2 1 52 cellVars
       2 1 52
5  matlab tstep_all domain_all data 1 reactionforces 1
     dofmanlist 2 189 226
6  domain 2dPlaneStress
7  OutputManager tstep_all dofman_all element_all
8  ndofman 4 nelem 1 ncrosssect 1 nmat 1 nbc 3 nic 0 nltf 2
9  node 1 coords 2 0.000000 0.000000 bc 2 1 1
10 node 2 coords 2 1.000000 0.000000 bc 2 2 1
11 node 3 coords 2 1.000000 1.000000 bc 2 2 3
12 node 4 coords 2 0.000000 1.000000 bc 2 1 3
13 Quad1PlaneStrain 1 nodes 4 1 2 3 4 mat 1 crossSect 1
14 SimpleCS 1 thick 1.0
15 Camclay 1 d 0.  E 20.0e3 n 0.0 tAlpha 0. m 10 pc 10 voidratio
       -1.0 maxiter 100
16 BoundaryCondition 1 loadTimeFunction 1 PrescribedValue 0.0
17 BoundaryCondition 2 loadTimeFunction 2 PrescribedValue 0.0005
18 BoundaryCondition 3 loadTimeFunction 2 PrescribedValue
     -0.0009
19 ConstantFunction 1 f(t) 1.0
20 PiecewiseLinFunction 2 nPoints 2 t 2 0. 2000. f(t) 2 0. 2000.
```

Výpis 1: Příklad vstupního souboru softwaru OOFEM

Naproti tomu výstupy z programu OOFEM jsou zapisovány do souborů `.out` ve standardizované formě. Do takového souboru, jehož příklad je ve výpisu 2, jsou ukládány výsledky v čisté a přehledné formě. Kromě toho je možné vygenerovat ještě další soubory, které pak slouží k vizualizaci výsledků v externím software, jako je například *ParaView* [1].

Protože program je spouštěn z příkazového řádku operačního systému, dalším

výstupem je přímo tento příkazový řádek, do nějž se zapisují chyby, varování a informace o běhu programu. To je výhodné zejména u řešení složitých úloh, jejichž výpočet běží po nějakou dobu. Těchto výpisů bylo využito například při měření rychlosti konvergence v části 5.2.

```
1          R E A C T I O N S   O U T P U T:
2          -------------------------------
3
4
5          Node         1 iDof   1 reaction  -1.0247e+01     [bc-id:
               1]
6          Node         1 iDof   2 reaction   1.8201e+01     [bc-id:
               1]
7          Node         2 iDof   1 reaction   1.0247e+01     [bc-id:
               2]
8          Node         2 iDof   2 reaction   1.8201e+01     [bc-id:
               1]
9          Node         3 iDof   1 reaction   1.0247e+01     [bc-id:
               2]
10         Node         3 iDof   2 reaction  -1.8201e+01     [bc-id:
               3]
11         Node         4 iDof   1 reaction  -1.0247e+01     [bc-id:
               1]
12         Node         4 iDof   2 reaction  -1.8201e+01     [bc-id:
               3]
13
14 User time consumed by solution step 5: 0.000 [s]
15
16
17 Finishing analysis on: Wed May 23 21:33:00 2018
18
19 Real time consumed: 000h:00m:00s
20 User time consumed: 000h:00m:00s
```

Výpis 2: Neúplný příklad výstupního souboru softwaru OOFEM

4.2 Popis nového kódu

Jak vyplývá z objektové struktury programu OOFEM a s tím souvisejících skutečností uvedených v části 4.1.1, vytvoření modulu pro nový materiál se omezuje pouze na vytvoření jedné nové třídy, zde konkrétně `CamClayMat`, přičemž její kompatibilita se zbytkem programu se zajistí jejím deklarováním jako potomka třídy `StructuralMaterial`, se kterou umí zbytek programu zacházet. Změnou oproti třídám popisujícím ostatní materiály prošly pouze relevantní části, tj. zejména implementace algortmů popsaných v kapitole 3, které budou podrobně popsány v následujících odstavcích.

4.2.1 Načítání parametrů

Jak bylo popsáno v části 4.1.2, úloha do programu OOFEM je zadána pomocí vstupního souboru. Data z tohoto vstupního souboru, která jsou relevantní pro výpočty s modelem Cam-Clay, je třeba načíst do nového kódu. Způsob, jakým je toto provedeno ve funkci `initializeFrom()`, je uveden ve výpisu 3.

```
1  CamClayMat :: initializeFrom(InputRecord *ir) {
2      [...]
3
4      // parameter describing the ration of the minor to major
           axis of the yield function
5      IR_GIVE_FIELD(ir, M, _IFT_CamClayMat_M);
6          // initial preconsolidation pressure
7      IR_GIVE_FIELD(ir, pc0, _IFT_CamClayMat_pc);
8
9
10     e = 0.2;
11     IR_GIVE_OPTIONAL_FIELD(ir, e, _IFT_CamClayMat_e);
12
13     consolidationIndex = 0.066;
14     IR_GIVE_OPTIONAL_FIELD(ir, consolidationIndex,
           _IFT_CamClayMat_consolidationIndex);
15
16     swellingIndex = 0.0077;
17     IR_GIVE_OPTIONAL_FIELD(ir, swellingIndex,
           _IFT_CamClayMat_swellingIndex);
18
19     //combined hardening parameter
20     theta = (1 + e) / (consolidationIndex - swellingIndex);
21     yieldTolerance = 1.e-9;
```

```
22      IR_GIVE_OPTIONAL_FIELD(ir, yieldTolerance,
            _IFT_CamClayMat_yieldTol);
23      plasticStrainTolerance = 1.e-9;
24      IR_GIVE_OPTIONAL_FIELD(ir, plasticStrainTolerance,
            _IFT_CamClayMat_plStrainTol);
25      pcTolerance = 1.e-9;
26      IR_GIVE_OPTIONAL_FIELD(ir, pcTolerance,
            _IFT_CamClayMat_pcTol);
27      maxIter = 200;
28      IR_GIVE_OPTIONAL_FIELD(ir, maxIter,
            _IFT_CamClayMat_maxIter);
29      return IRRT_OK;
30  }
```

Výpis 3: Část funkce initializeFrom() načítající data pro model

Povšimněme si, že načítání je velmi zjednodušeno pomocí maker vytvořených v programu OOFEM přímo pro tento účel, jako jsou `IR_GIVE_FIELD` a `IR_GIVE_OPTIONAL_FIELD`. Rozdíl mezi nimi je samozřejmě ten, že program nebude vyžadovat, aby hodnota, která je získávána makrem `IR_GIVE_OPTIONAL_FIELD`, ve vstupním souboru byla. Naopak tu, kterou získáváme pomocí `IR_GIVE_FIELD` je pro běh programu zadat nutné. Proto jsou u nepovinných parametrů před jejich načtením stanoveny implicitní hodnoty, které odpovídají hodnotám uvedeným v části 2.2 v tabulce 1. V uvedeném úseku kódu si také lze povšimnout výpočtu materiálového parametru ϑ v souladu s rovnicí (5).

Tolerance	Hodnota
Maximum iterací	200
Tolerance pro $\boldsymbol{R}_\sigma^i$	1e-9
Tolerance pro R_f^i	1e-9
Tolerance pro R_p^i	1e-9

Tabulka 2: Implicitní hodnoty tolerancí iteračního algoritmu pro návrat na plochu plasticity

Kromě parametrů modelu, o nichž se zmiňuje část 2.2, umožňuje program zadat i hodnoty vnitřních tolerancí pro iterační algoritmus návratu na plochu plasticity. Jedná se o maximální počet iterací a o tolerance jednotlivých kontrolovaných reziduálů. Hodnoty

těchto tolerancí jsou shrnuty v tabulce 2.

4.2.2 Návrat na plochu plasticity

Celý mechanismus elastické predikce a plastické korekce je obsažen ve funkci `performPlasticityReturn()`. Tato funkce je volána z funkce `giveRealStressVector_3d()`, která je vlastní všem materiálovým modelům založeným na třídě `StructuralMaterial`. Jejími argumenty jsou:

- `GaussPoint *gp` - materiálový bod, v němž má probíhat výpočet
- `const FloatArray &totalStrain` - nová hodnota celkové deformace $\boldsymbol{\varepsilon}_{n+1}$
- `TimeStep *tStep` - časový krok

Počátek funkce `performPlasticityReturn()`, zabývající se elastickou predikcí, tj. stanovením hodnoty zkušebního napětí dle rovnice (31), je uveden ve výpisu 4.

```
1  CamClayMatStatus *status = static_cast<CamClayMatStatus * >(
       this->giveStatus(gp));
2  double pc = status->givePreconsolidationPressure();
3  FloatArray plasticStrain;
4  FloatArray fullStress;
5  // get the initial plastic strain and initial kappa from the
       status
6  plasticStrain = status->givePlasticStrain();
7  // elastic predictor
8  FloatArray elStrain = totalStrain;
9  elStrain.subtract(plasticStrain);
10 FloatArray elStrainDev;
11 double elStrainVol;
12 elStrainVol = computeDeviatoricVolumetricSplit(elStrainDev,
       elStrain);
13 FloatArray trialStressDev;
14 applyDeviatoricElasticStiffness(trialStressDev, elStrainDev,
       G);
15 double trialStressVol = 3 * K * elStrainVol;
16 // check the yield condition at the trial state
17 double trialQ = sqrt(3./ 2) * computeStressNorm(
       trialStressDev);
18 //because q = sqrt(3/2)*||s||
19 double trialP = -trialStressVol;
```

```
20 //because p= -sigma_m
21 double yieldValue = trialQ*trialQ - M*M*trialP*(pc-trialP);
22 double lambda = status->givePlasticMultiplier();
23 if (yieldValue > 0.) {
```

Výpis 4: Elastická predikce ve funkci performPlasticityReturn()

Elastická predikce je zde prováděna pomocí deviatoricko-volumetrického rozdělení, využívajíc přitom původní, již existující funkce, jako je například `computeDeviatoricVolumetricSplit()`. Za povšimnutí dále stojí získávání původních, n-tých hodnot veličin z objektu `CamClayMatStatus`, který sám byl získán od materiálového bodu, ve kterém výpočet probíhá.

Na konci výpisu 4 je podmínka, v jejímž těle se nachází celý algoritmus pro plastickou korekci. Ten začíná dalším získáváním a inicializací parametrů, jako je například matice tuhosti. Poté následuje kód uvedený ve výpisu 5, tedy úvodní výpočet reziduálů dle rovnic (53) až (55) a počátek iteračního cyklu `do{...}while();`.

```
1  //retrieving elasticity matrix
2  FloatMatrix dE;
3  give3dMaterialStiffnessMatrix(dE, MatResponseMode::
       ElasticStiffness, gp, tStep); //for linear elasticity
4  FloatMatrix inverseDe;
5  inverseDe.beInverseOf(dE);
6
7  [...]
8
9  //initial computation of residuals
10 //Ri_sigma
11 FloatArray rISigma;
12 FloatArray sigmaDifference;
13 sigmaDifference.add(lastStress);
14 sigmaDifference.times(-1.0);
15 sigmaDifference.add(nextStress);
16 givePlasticGradientAtStress(rISigma, nextStress, nextPc);
17 rISigma.times(deltaLambda);
18 rISigma.add(strainDifference);
19 FloatArray elasticDeformationDifference;
20 elasticDeformationDifference.beProductOf(inverseDe,
       sigmaDifference);
21 rISigma.add(elasticDeformationDifference);
22
```

```
23 //Ri_f
24 double rIF = giveYieldValueAtStress(nextStress, nextPc);
25
26 //Ri_p
27 double nextP = (-1. / 3.)*(nextStress.at(1) + nextStress.at
       (2) + nextStress.at(3));
28 double rIP = nextPc - theta*M*M/3*deltaLambda*(2 * nextP -
       nextPc)*nextPc - pc;
29
30 do {
```

Výpis 5: Počátek plastické korekce ve funkci performPlasticityReturn()

Ve výpisu 5 se také objevuje první případ využití v programu OOFEM existujících tříd `FloatArray` a `FloatMatrix`, které slouží k číselné reprezentaci vektorů a matic a provádění operací s nimi. Možnost využití těchto tříd byla zásadním ulehčením pro psaní zde popisovaného kódu.

Principem iteračního cyklu plastické korekce, jak je popsáno v části 3.2, je opakované řešení soustavy rovnic (59) a aktualizace odhadů neznámých $\boldsymbol{\sigma}_{n+1}$, $(\Delta\lambda)_{n+1}$ a $(p_c)_{n+1}$, reprezentovaných zde v kódu proměnnými `nextStress`, `deltaLambda` a `nextPc`, v tomto pořadí. To spočívá ve vyčíslení devíti derivací dle vztahů (60) až (68) a získání změn jednotlivých neznámých $\delta\Delta\lambda$, $\delta\boldsymbol{\sigma}$ a δp_c podle rovnic (75), (71) a (69). To je sice zdlouhavý, nicméně v zásadě přímočarý a nepříliš zajímavý výpočet, proto zde nebude uveden. Soustřeďme se raději na konec funkce `performPlasticityReturn()`, uvedený ve výpisu 6.

```
1           //4) update stress, pc, deltaLambda,
2           nextStress.add(deltaSigma);
3           nextPc += deltaPc;
4           deltaLambda += deltaDeltaLambda;
5
6           //5) update residuals
7           [...]
8
9           //6) check for convergence
10          iterationNumber++;
11          double strainError = sqrt(rISigma.dotProduct(rISigma)
                );
12          convergence = (fabs(rIF) < yieldTolerance &&
                strainError < plasticStrainTolerance && fabs(rIP)
                < pcTolerance);
```

```
13          } while (iterationNumber < maxIter && !convergence);
14
15          //convergence reached
16          //updating all values to values reached by algorithm
17          pc = nextPc;
18          fullStress = nextStress;
19
20          FloatArray plasticStrainIncrement;
21          givePlasticGradientAtStress(plasticStrainIncrement,
                fullStress, pc);
22          plasticStrainIncrement.times(deltaLambda);
23          plasticStrain.add(plasticStrainIncrement);
24
25          lambda += deltaLambda;
26          if(!convergence) {
27                      this->giveDomain()->giveEngngModel()->
                            setAnalysisCrash(true);
28                      OOFEM_WARNING("Local equlibrium of CPP
                            algorithm not reached in %d iterations,
                            Element number %d, gp %d, continuing",
                            iterationNumber, gp->giveElement()->
                            giveNumber(), gp->giveNumber() );
29          }
```

Výpis 6: Závěr plastické korekce ve funkci performPlasticityReturn()

Opětovný výpočet reziduálů prováděný na konci kroku byl z výpisu 6 vynechán, je shodný s výpočtem, který se už objevuje ve výpisu 5.

Mimo jiné poskytuje výpis 6 vhled do postupu, kterým se kontroluje konvergence algoritmu. Hodnoty reziduálů jsou zkontrolovány proti přednastaveným tolerancím a počet iterací proti maximálnímu počtu iterací. Pokud v maximálním počtu iterací nedošlo ke konvergenci, program uvědomí hlavní řešič, že došlo k lokálnímu problému, a pomocí přednastaveného makra ohlásí varování.

Po dokončení algoritmu plastické korekce zbývá ve funkci `performPlasticityReturn()` už jen uložení výsledků výpočtu. To je provedeno kódem na výpisu 7. Povšimněme si, že výsledky se ukládají opět pomocí objektu typu `CamClayMatStatus`, ale že se neuloží přímo jako trvalé, ale pouze do kolonek pro hodnoty dočasné. Důvodem je, že nevíme, zda s těmito hodnotami globální algoritmus zkonverguje. Hodnoty se později načtou pro výpočet algoritmické tuhosti, a ve chvíli, kdy globální algoritmus dosáhne konvergence,

zavolá funkci `MaterialStatus.updateYourself()`, která hodnoty dočasné přepíše na trvalé.

```
1 // store the stress in status
2 status->letTempStressVectorBe(fullStress);
3 // store the plastic strain, preconsolidation stress,
4 // and plastic multiplier
5 status->letTempPlasticStrainBe(plasticStrain);
6 status->setTempPreconsolidationPressure(pc);
7 status->setTempPlasticMultiplier(lambda);
```

Výpis 7: Konečné řádky funkce performPlasticityReturn()

4.2.3 Algoritmická tuhost

Algoritmická tuhost je v programu řešena funkcí `give3dMaterialStiffnessMatrix()`. Tato funkce je opět, stejně jako `performPlasticityReturn()`, volána ostatními částmi programu externě, a má i podobné argumenty:

- `FloatMatrix &answer` - objekt pro uložení výsledku výpočtu
- `MatResponseMode mode` - přepínač určující požadovaný typ matice
- `GaussPoint *gp` - materiálový bod, v němž má probíhat výpočet
- `TimeStep *tStep` - časový krok

Počátek funkce je uveden ve výpisu 8. Prvním krokem je stanovení běžné elastické matice tuhosti, k čemuž zde slouží existující funkce objektu `linearElasticMaterial`. Pokud není přepínač `MatResponseMode` nastaven pro tečnou algoritmickou tuhost, tato matice je použita i jako návratová hodnota. Použití funkce `give3dMaterialStiffnessMatrix()` v tomto režimu je možné vidět i v kódu návratu na plochu plasticity, a to ve výpisu 5. Pokud však je tečná tuhost požadována, elastická matice je použita pouze jako základ pro výpočet dle algoritmu popsaného v části 3.3. Pro to potřebujeme hodnoty neznámých vypočtených v naposled počítaném zatěžovacím kroku - ty dostaneme jako dočasné proměnné ze statusu, kam jsme je na konci plastické korekce uložili.

```
1 // start from the elastic stiffness
2 FloatMatrix dE;
3 this->linearElasticMaterial.give3dMaterialStiffnessMatrix (dE
      , mode, gp, tStep);
4 if ( mode != TangentStiffness ) {
5     answer = dE;
```

```
6       return;
7   }
8
9   CamClayMatStatus *status = static_cast< CamClayMatStatus * >
        ( this->giveStatus(gp) );
10  double tempLambda = status->giveTempPlasticMultiplier();
11  if (tempLambda - status->givePlasticMultiplier > 0) {
12      //need to assemble the tangent matrix
13
14      //retrieving important parametres
15
16      FloatMatrix inverseDe;
17      inverseDe.beInverseOf(dE);
18
19      double deltaLambda = tempLambda - status->
            givePlasticMultiplier();
20
21      FloatArray lastStress;
22      lastStress = status->giveStressVector();
23
24      FloatArray nextStress;
25      nextStress = status->giveTempStressVector();
26
27      double nextPc = status->giveTempPreconsolidationPressure
            ();
28
29      double pc = status->givePreconsolidationPressure();
30
31      //elements of matrix are the same as in plasticity return
32      //therefore the code is too
```

Výpis 8: Počátek funkce give3dMaterialStiffnessMatrix()

Jak je uvedeno v komentáři na posledních dvou řádcích výpisu 8 a bohatě diskutováno v části 3.3, postup výpočtu zde a v algoritmu plastické korekce se značně podobá. Uvedeme zde už tedy jen jeden výpis, a to obsahující konečné sestavení algoritmické tečné matice tuhosti $\boldsymbol{D}_{ep}$, jejímž uložení do odpovědního objektu se funkce `give3dMaterialStiffnessMatrix()` ukončuje.

```
1       //finalising fraction
2       FloatMatrix fractionResult;
```

```
3         fractionResult.add(upper);
4         fractionResult.times((1 / lower));
5
6         //finalising Dep
7         FloatMatrix dEp;
8         dEp.add(cMatrix);
9         dEp.subtract(fractionResult);
10
11        answer = dEp;
12
13      }
14      else {
15        //no change in plastic multiplier -> no plastic step
              occured -> no need to compute tangent matrix
16        answer = dE;
17      }
18  }
```

Výpis 9: Závěr funkce give3dMaterialStiffnessMatrix()

4.2.4 Další funkce

Pro zjednodušení některých částí algoritmu vznikly i další funkce, zabývající se vesměs banálními a často opakovanými úkony, jako je například výpočet hodnoty funkce plasticity (9) pro dané napětí. Jsou to funkce `giveYieldValueAtStress()`, `givePlasticityGradientAtStress()` a `giveYieldFunctionDoubleDerivative()`.

5 Testování programu

V této kapitole budeme program OOFEM obsahující námi vytvořený modul pro modifikovaný model Cam-Clay používat pro řešení úloh o postupně se zvyšující úrovni složitosti, abychom předvedli jednak správnost práce algoritmů a jednak i možnosti, které nám OOFEM s implemetovaným modelem Cam-Clay dává.

5.1 Krychlový vzorek ze třech stran fixovaný (bez uvážení zpevnění)

Hlavním cílem tohoto jednoduchého příkladu je demonstrace správnosti algoritmu z hlediska dosažené plochy plasticity. Aby bylo z dosaženého výsledku lépe vidět, zda se napětí pohybuje v očekávaných hodnotách, zanedbáme zpevnění a zafixujeme tak plochu plasticity v jednotném tvaru. To se provede v zadání úlohy prostým vynulování parametru zpevnění ϑ pomocí nastavení hodnoty pórovitosti na $e = -1$ (srovnejme (5)). Dle zákona zpevnění (30) potom bude vždy platit, že

$$(p_c)_{n+1} = (p_c)_n \tag{104}$$

a prekonsolidační tlak zůstane na původní hodnotě, která byla zadána jako parametr $(p_c)_0$.

Vstupní soubor pro tento příklad je uveden ve výpisu 10.

```
1  camclayBrick.out
2  test of cam clay material model
3  NonLinearStatic nsteps 5 renumber 1 nmodules 2 controllmode 1
       rtolv 1.e-6 stiffMode 0 manrmsteps 1 maxiter 300
     initialGuess 0
4  vtkxml tstep_all domain_all primvars 1 1 vars 2 1 52 cellVars
       2 1 52
5  matlab tstep_all domain_all data 1 reactionforces 1
     dofmanlist 2 189 226
6  domain 3d
7  OutputManager tstep_all dofman_all element_all
8  ndofman 8 nelem 1 ncrosssect 1 nmat 1 nbc 4 nic 0 nltf 2
9  node 1 coords 3 0.000000 0.000000 0.000000 bc 3 1 1 1
10 node 2 coords 3 1.000000 0.000000 0.000000 bc 3 2 1 1
11 node 3 coords 3 1.000000 1.000000 0.000000 bc 3 2 3 1
12 node 4 coords 3 0.000000 1.000000 0.000000 bc 3 1 3 1
13 node 5 coords 3 0.000000 0.000000 1.000000 bc 3 1 1 4
14 node 6 coords 3 1.000000 0.000000 1.000000 bc 3 2 1 4
15 node 7 coords 3 1.000000 1.000000 1.000000 bc 3 2 3 4
```

```
16 node 8 coords 3 0.000000 1.000000 1.000000 bc 3 1 3 4
17 LSpace 1 nodes 8 1 2 3 4 5 6 7 8 mat 1 crossSect 1
18 SimpleCS 1 thick 1.0
19 Camclay 1 d 0.  E 20.0e3 n 0.0 tAlpha 0. m 1.2 pc 0.1
      voidratio -1.0
20 BoundaryCondition 1 loadTimeFunction 1 PrescribedValue 0.0
21 BoundaryCondition 2 loadTimeFunction 2 PrescribedValue -0.001
22 BoundaryCondition 3 loadTimeFunction 2 PrescribedValue -0.001
23 BoundaryCondition 4 loadTimeFunction 2 PrescribedValue -0.001
24 ConstantFunction 1 f(t) 1.0
25 PiecewiseLinFunction 2 nPoints 2 t 2 0. 2000. f(t) 2 0. 2000.
```

Výpis 10: Fixovaná krychle: Vstupní soubor

Základní parametry modelu použité pro tento příklad (u parametrů κ a λ, které se díky zanedbání zpevnění neprojeví, byla ponechána implicitní hodnota) jsou uvedeny v tabulce 3.

Parametr	Hodnota
M	1,2
$(p_c)_0$	100 kPa

Tabulka 3: Fixovaná krychle: Parametry modelu

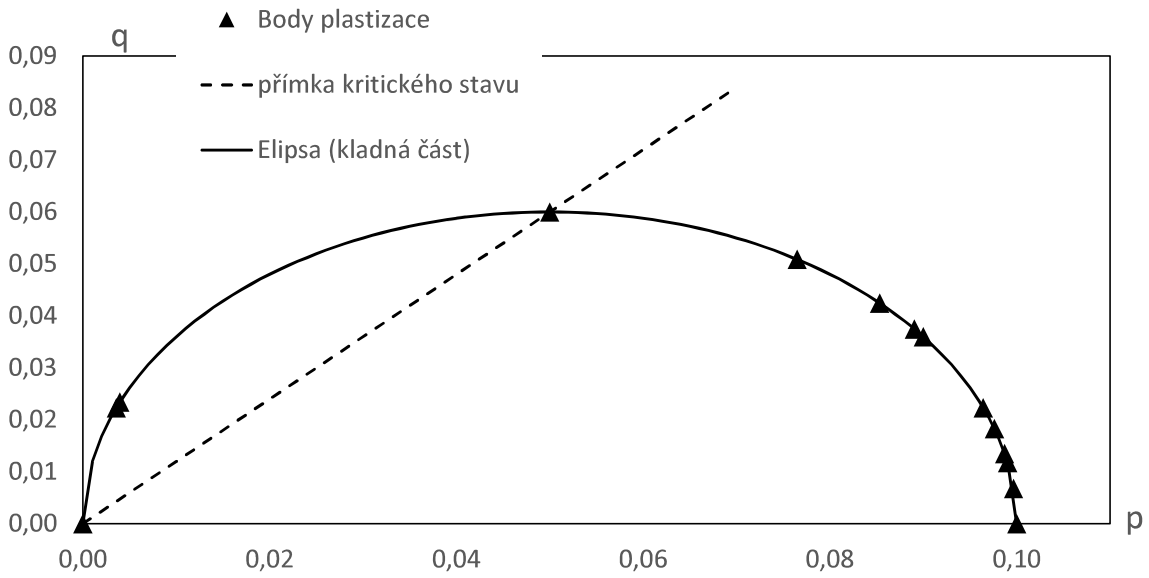
Příklad se sestává z jednoho krychlového konečného prvku, který je ve včech třech směrech vždy z jedné strany zafixován. Jak je vidět na řádcích 21-23 výpisu 10, zatížení vzorku je prováděno předepisovanou deformací. Provádí se 4 zatěžovací kroky, v nichž se deformace v jednotlivých směrech vždy navýší o stanovené hodnoty. Výpočet byl proveden opakovaně, přičemž poměry předepisovaných změn deformace v těchto směrech byly vždy odlišné. Díky tomu se při plastizaci dostal stav napjatosti vždy do jiného bodu v prostoru p-q. Pokud program pracuje správně, musí však tyto body vždy ležet na elipse plasticity dané parametry v tabulce 3. Výsledná napětí při pátém zatěžovacím kroku jsou shrnuta v tabulce 4. Jejich grafické znázornění je na obrázku 5.

Ukazuje se, že napětí zcela dle očekávání kopíruje na obrázku 5 vynesenou plochu plasticity získanou z parametrů modelu. První test programu dopadl tedy úspěšně.

Je zjevné, že touto úlohou nebyla nijak otestována funkčnost zpevnění, ale díky vlastnostem úlohy (všestranné podepření či předepsání deformace) nebyla aplikována ani algoritmická tuhost. Funkčnost tohoto algoritmu, stejně jako jeho případný vliv na rychlost konvergence úlohy, je stále třeba otestovat, k čemuž dobře poslouží následující úloha.

$\Delta\varepsilon_x$	$\Delta\varepsilon_y$	$\Delta\varepsilon_z$	σ_x [MPa]	σ_y [MPa]	σ_z [MPa]	p [MPa]	q [MPa]
-0,001	-0,001	-0.001	-0,1000	-0,1000	-0,1000	0,1000	0,0000
-0,001	-0,001	-0.002	-0,0951	-0,0951	-0,1069	0,0990	0,0118
-0,001	-0,002	-0.001	-0,0951	-0,1069	-0,0951	0,0990	0,0118
-0,001	-0,002	-0.003	-0,0909	-0,0987	-0,1065	0,0987	0,0135
-0,002	-0,003	-0.002	-0,0974	-0,1042	-0,0974	0,0997	0,0068
-0,003	-0,001	-0.001	-0,1098	-0,0915	-0,0915	0,0976	0,0183
-0,001	0,000	-0.001	-0,1039	-0,0816	-0,1039	0,0964	0,0223
-0,001	0,001	-0.001	-0,0935	-0,0426	-0,0935	0,0765	0,0509
0,003	0,002	0.000	0,0078	-0,0010	-0,0187	0,0040	0,0234
0,001	0,001	0.000	0,0039	0,0039	-0,0184	0,0036	0,0223
-0,001	0,000	0.000	-0,1140	-0,0765	-0,0765	0,0890	0,0375
-0,008	-0,007	0.005	-0,1011	-0,0977	-0,0571	0,0853	0,0425
-0,005	0,002	-0.004	-0,1046	-0,0662	-0,0992	0,0900	0,0360
-0,002	0,001	0.001	-0,0900	-0,0300	-0,0300	0,0500	0,0600
0,001	0,001	0,001	0,0000	0,0000	0,0000	0,0000	0,0000

Tabulka 4: Fixovaná krychle: Napětí při různých poměrech deformace



Obrázek 5: Fixovaná krychle: Napětí při různých poměrech deformace v prostoru p-q

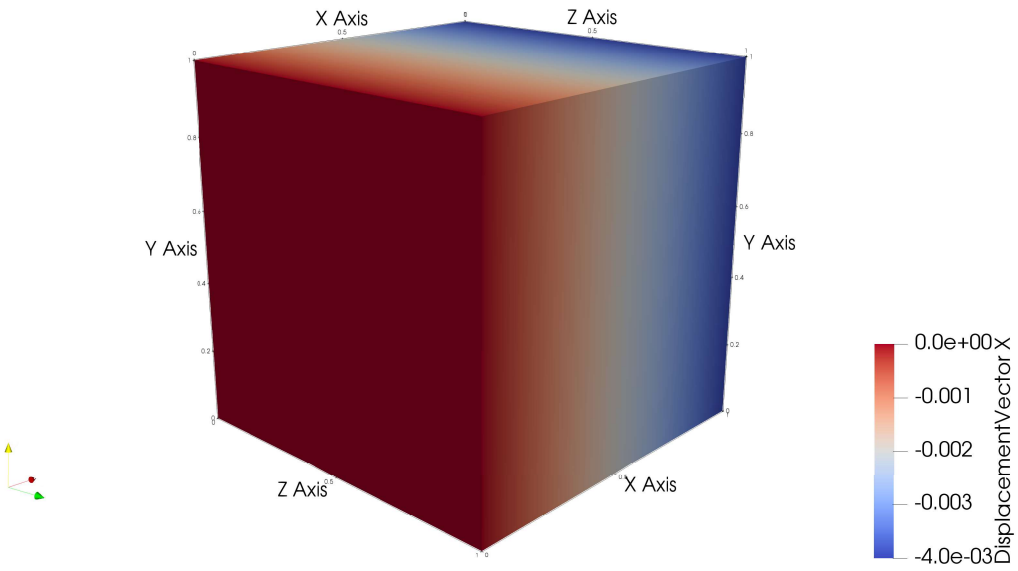
5.2 Krychlový vzorek jednoose namáhaný

Mějme nyní stejný vzorek o jednom konečném prvku jako v první úloze. Je již uvažováno zpevnění. Parametry materiálového modelu pro tuto úlohu jsou shrnuty v tabulce 5.

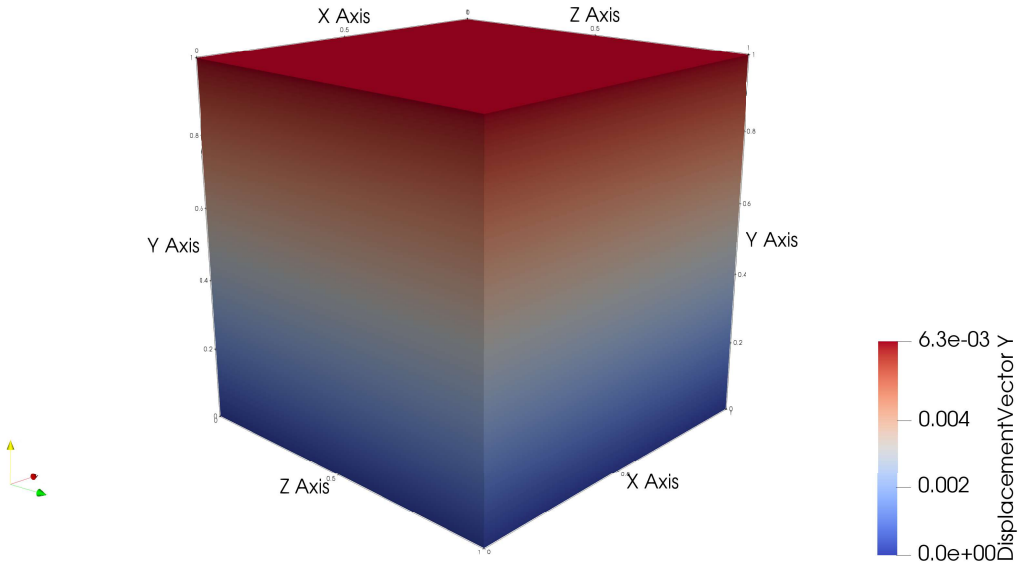
Parametr	Hodnota
M	1,2
$(p_c)_0$	100 kPa
e	0,2
λ	0,066 $(\ln \mathrm{MPa})^{-1}$
κ	0.0077 $(\ln \mathrm{MPa})^{-1}$

Tabulka 5: Jednoose namáhaná krychle: Parametry modelu

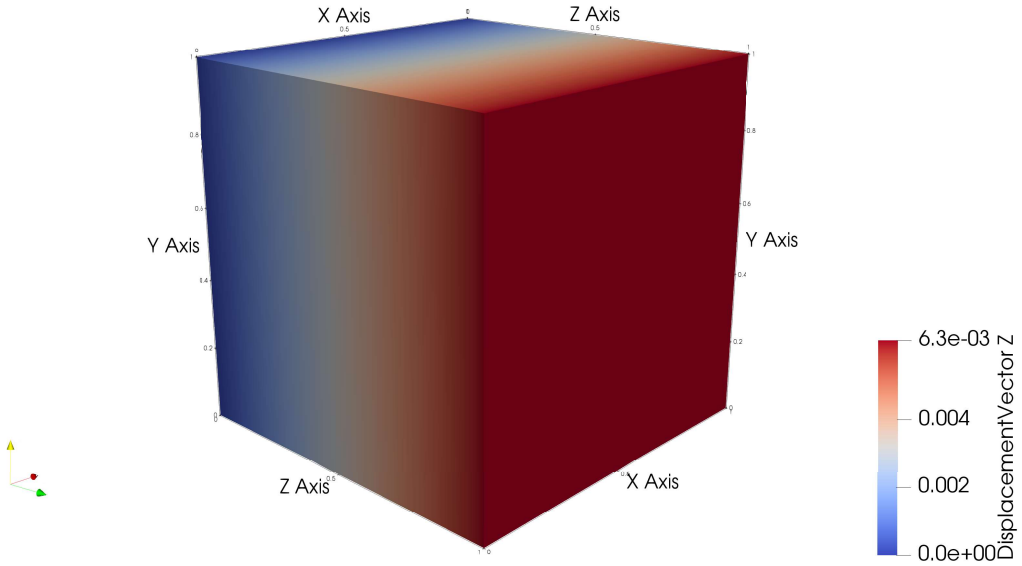
Okrajové podmínky úlohy zůstávají podobné jako v předchozím případě, nyní však zatěžování deformací probíhá pouze podél osy x. Lze tedy očekávat, že odezvou vzorku bude stlačení ve směru osy x a roztažení ve směru os ostatních. Opět provádíme výpočet ve 4 zatěžovacích krocích. Obrázky 6 až 9, na nichž je stav deformace vzorku na konci posledního kroku, ukazují, že výsledek odpovídá předpokladu.



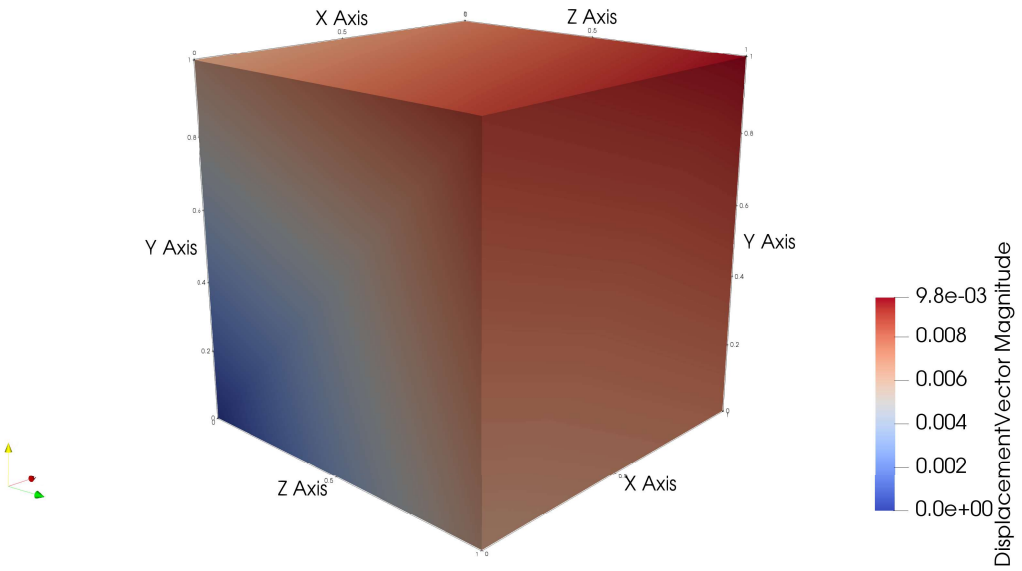
Obrázek 6: Jednoose namáhaná krychle: Deformace ve směru x



Obrázek 7: Jednoose namáhaná krychle: Deformace ve směru y



Obrázek 8: Jednoose namáhaná krychle: Deformace ve směru z



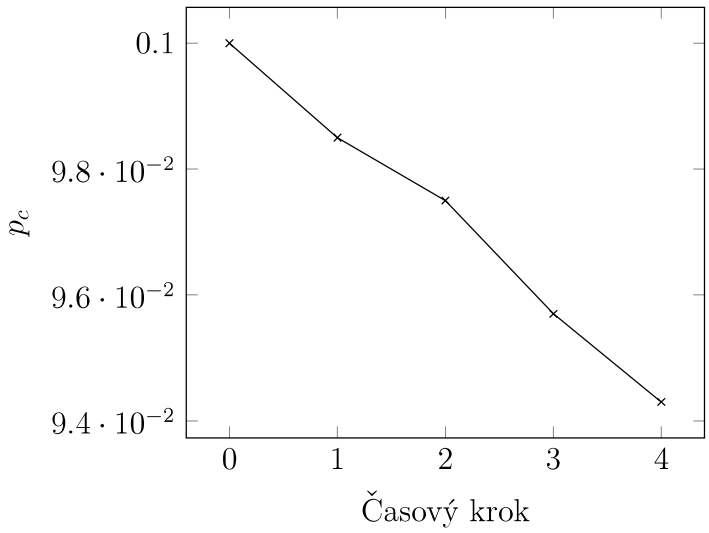
Obrázek 9: Jednoose namáhaná krychle: Norma deformace

Mějme na paměti, že stále vždy jedna strana vzorku v každém směru je fixována, a proto se tedy na ní objevuje nulová deformace.

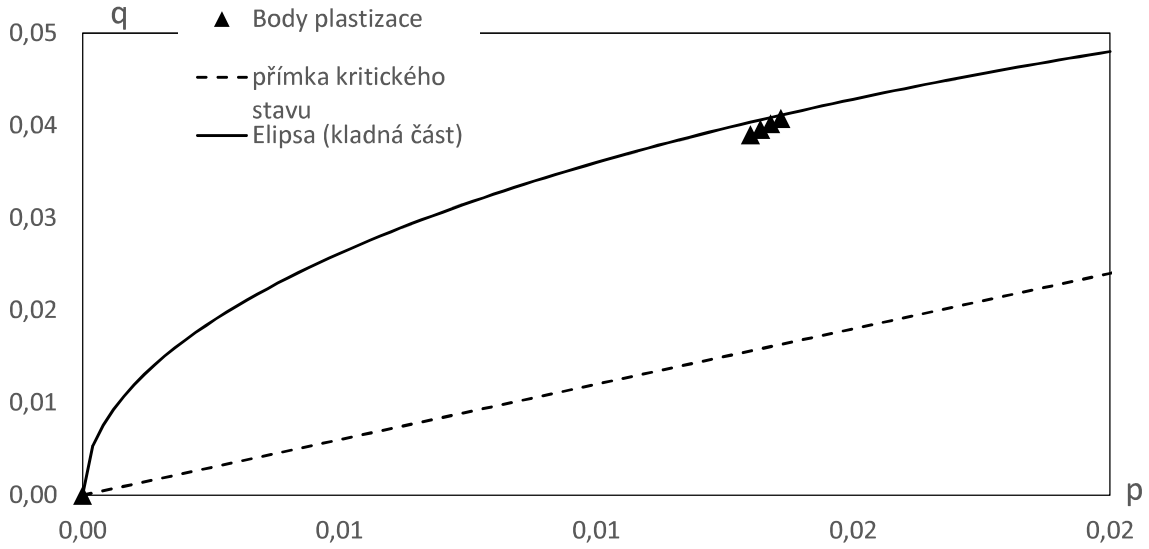
Nyní se podívejme blíže na zpevnění. Protože deformace je postupně vnášena ve čtyřech krocích, průběh zpevnění lze dobře odpozorovat z hodnot dosaženého napětí. Vezměme pro příklad napětí v materiálovém bodu 8 a sledujme jeho vývoj pro všechny kroky. Protože známe parametr M, můžeme z funkce plasticity (9) vypočítat vždy aktuální hodnotu prekonsolidačního tlaku p_c. Výpočet je shrnut v tabulce 6 a vývoj p_c je vykreslen na obrázku 10. Jak je vidět, materiál změkčuje. Na obrázku 11 je znázorněno, kde se pohybují jednotlivé body napjatosti v časových krocích vzhledem k počáteční elipse plasticity. Jejich posun, potvrzující změkčování, je v souladu s představou uvedenou v části 2.5, a to sice že při zvyšování volumetrické plastické deformace se snižuje prekonsolidační tlak.

Čas	σ_x [MPa]	σ_y [MPa]	σ_z [MPa]	p [MPa]	q [MPa]	p_c [MPa]
0	0,0000	0,0000	0,0000	0,0000	0,0000	0,1000
1	-0,0408	0,0000	0,0000	0,0136	0,0408	0,0985
2	-0,0402	0,0000	0,0000	0,0134	0,0402	0,0971
3	-0,0396	0,0000	0,0000	0,0132	0,0396	0,0957
4	-0,0390	0,0000	0,0000	0,0130	0,0390	0,0943

Tabulka 6: Jednoose namáhaná krychle: Vývoj prekonsolidačního tlaku



Obrázek 10: Jednoose namáhaná krychle: Vývoj prekonsolidačního tlaku



Obrázek 11: Jednoose namáhaná krychle: Vývoj napětí v časových krocích vzhledem k počáteční elipse plasticity

Ke zkoumání rychlosti konvergence použijeme výstup z příkazového řádku vygenerovaný za běhu programu a uvedený ve výpisu 11. Jedná se o průběh posledního zatěžovacího

kroku.

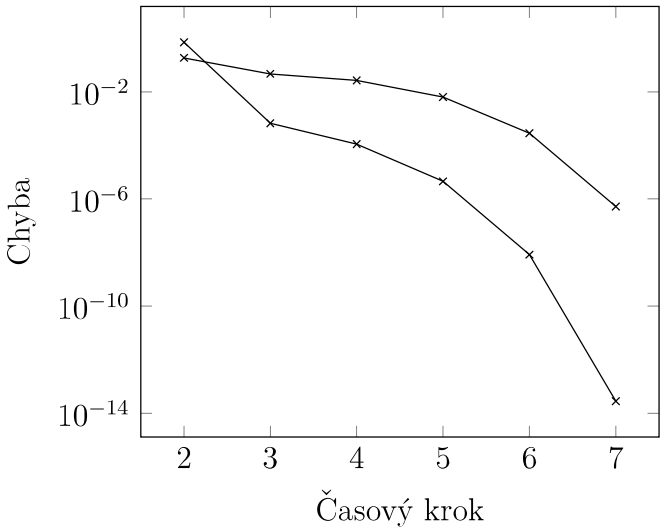
```
1 Solving        [step number      5.0, time = 4.000000e+000]
2
3 NRSolver: Iteration ForceError DisplError
4 ------------------------------------------------------------------------------

5 NRSolver: 1        D_v:  7.071e-001  0.000e+000  D_w:  7.071e
      -001  0.000e+000
6 NRSolver: 2        D_v:  7.071e-001  1.859e-001  D_w:  7.071e
      -001  1.859e-001
7 NRSolver: 3        D_v: *6.730e-004  4.693e-002  D_w: *6.730e
      -004  4.693e-002
8 NRSolver: 4        D_v: *1.118e-004  2.698e-002  D_w: *1.118e
      -004  2.698e-002
9 NRSolver: 5        D_v: *4.563e-006  6.440e-003  D_w: *4.563e
      -006  6.440e-003
10 NRSolver: 6       D_v: *8.358e-009  2.857e-004  D_w: *8.358e
      -009  2.857e-004
11 NRSolver: 7       D_v: *2.816e-014  5.252e-007  D_w: *2.816e
      -014  5.252e-007
12 Equilibrium reached at load level = 1.000000 in 7 iterations
13 Reseting load level
14 EngngModel info: user time consumed by solution step 5: 0.00s
```

Výpis 11: Jednoose namáhaná krychle: Výstup z příkazového řádku

Průběh chyb v hodnotách napětí i deformace je shrnut v tabulce 7 a zobrazen na obrázku 12. Jak je vidět, algoritmus konverguje přibližně kvadraticky, což je vzhledem k použitým algoritmům očekávané a uspokojivé.

Krok	Chyba σ	Chyba ε
1	7.071e-001	0.000e-000
2	7.071e-001	1.859e-001
3	6.730e-004	4.693e-002
4	1.118e-004	2.698e-002
5	4.563e-006	6.440e-003
6	8.358e-009	2.857e-004
7	2.816e-014	5.252e-007

Tabulka 7: Jednoose namáhaná krychle: Vývoj chyby v posledním zatěžovacím kroku



Obrázek 12: Jednoose namáhaná krychle: Vývoj chyby v posledním zatěžovacím kroku

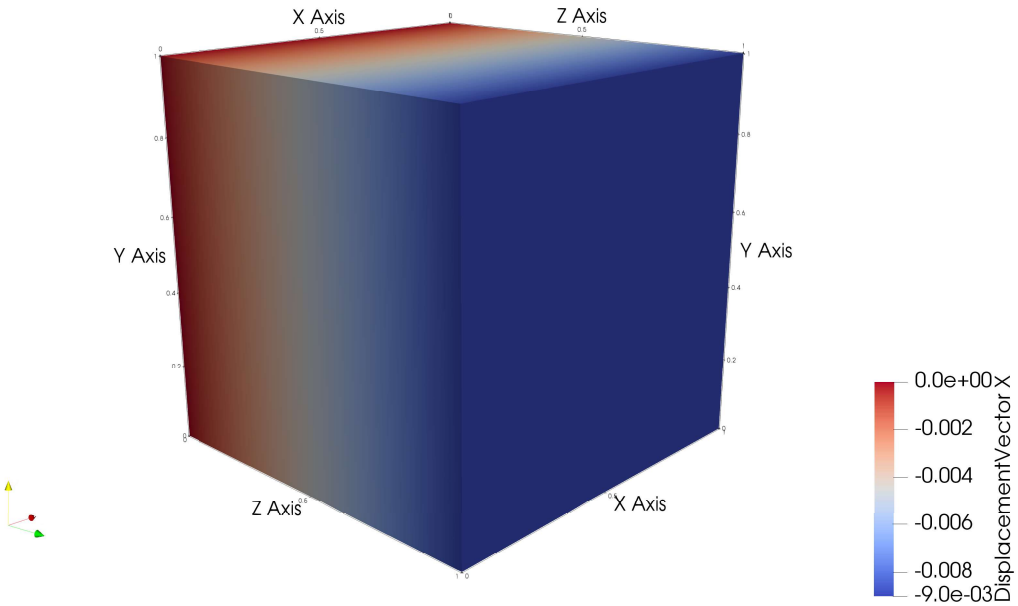
5.3 Krychlový vzorek namáhaný obecným zatížením

Vezměme potřetí náš krychlový vzorek a zatižme ho nyní obecným zatížením ve všech směrech. Parametry modelu zůstávají stále stejné, pro pořádek jsou shrnuty v tabulce 8.

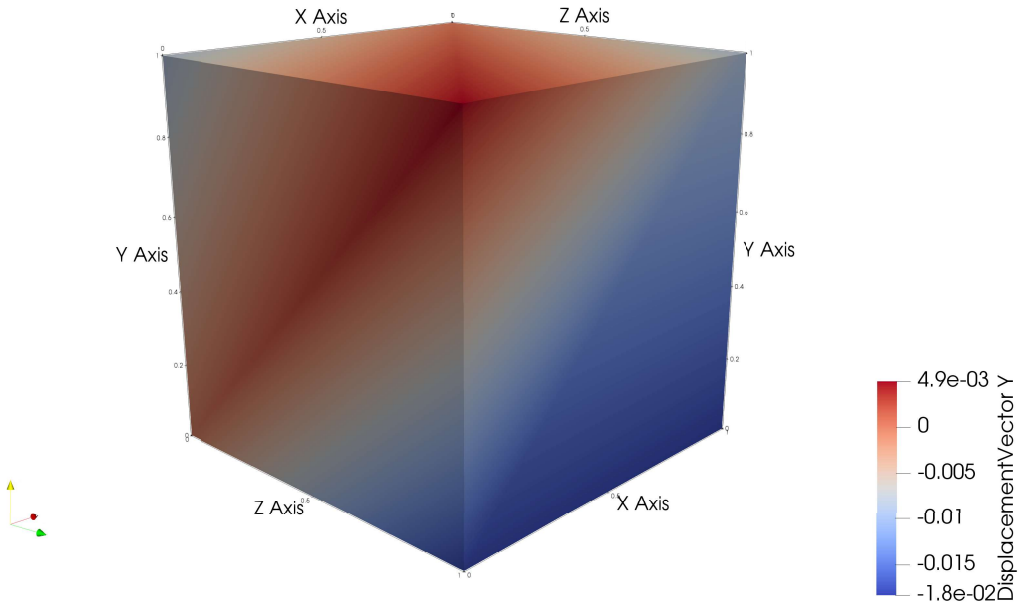
Již neplatí, že by tři strany krychle byly zcela upevněny. Upevněny jsou jen některé body v některých směrech tak, aby úloha nebyla staticky přeurčitá. Vnášeny jsou deformace ve všech třech směrech. Výpočet navíc pro získání výraznějších výsledků probíhá v 10, nikoli 5 krocích. Výsledné posuny jsou zobrazeny na obrázcích 13 až 16. Na obrázku 17 je navíc pro ujasnění vykreslen tvar deformace.

Parametr	Hodnota
M	1,2
$(p_c)_0$	100 kPa
e	0,2
λ	0,066 $(\ln \mathrm{MPa})^{-1}$
κ	0.0077 $(\ln \mathrm{MPa})^{-1}$

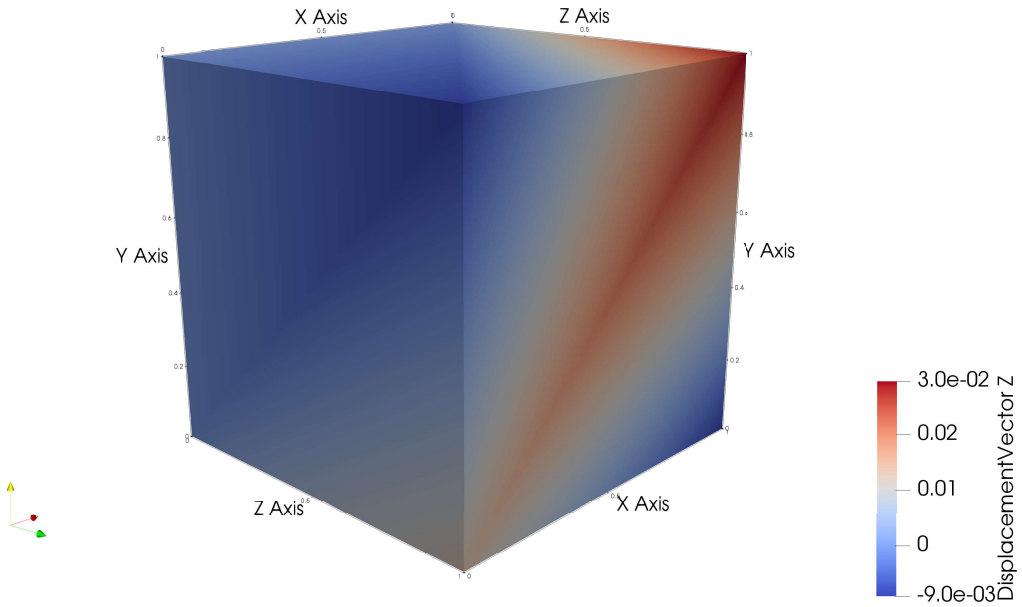
Tabulka 8: Obecně namáhaná krychle: Parametry modelu



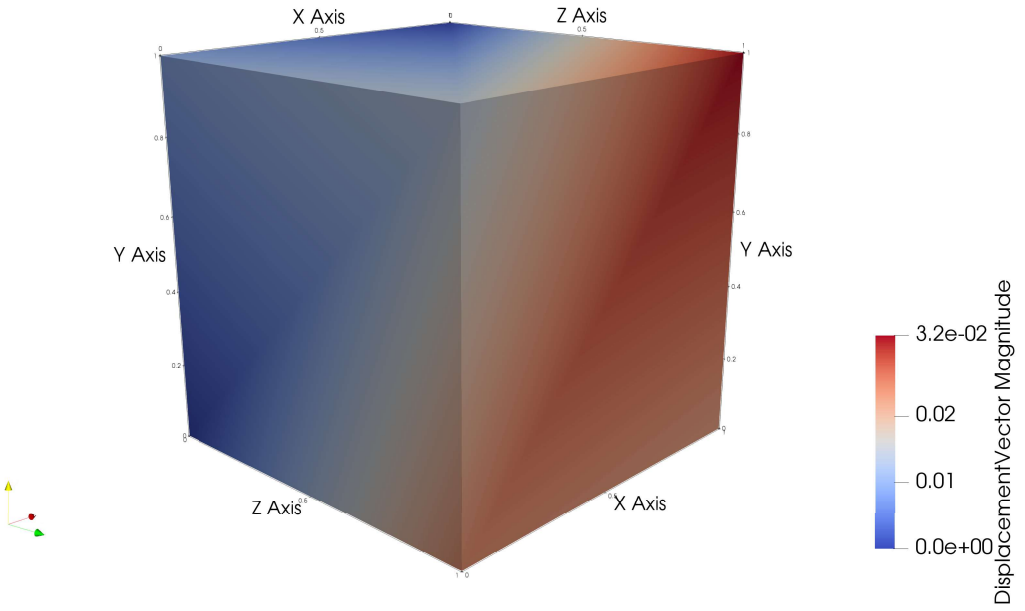
Obrázek 13: Obecně namáhaná krychle: Deformace ve směru x



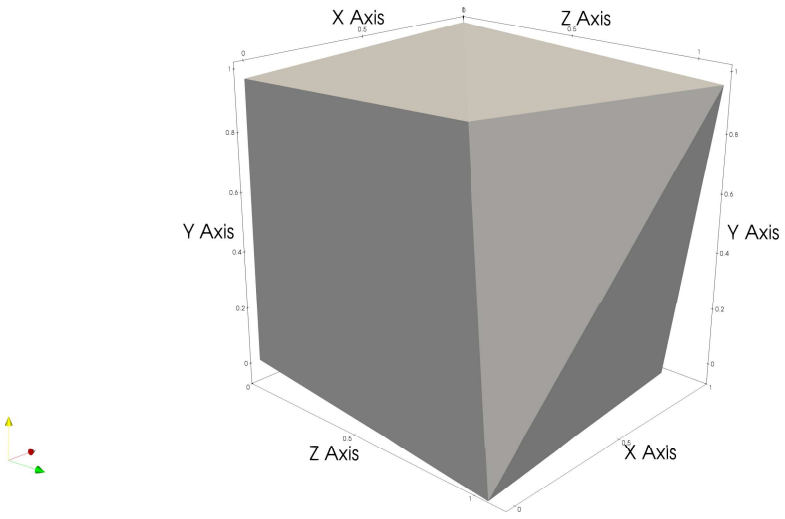
Obrázek 14: Obecně namáhaná krychle: Deformace ve směru y



Obrázek 15: Obecně namáhaná krychle: Deformace ve směru z



Obrázek 16: Obecně namáhaná krychle: Norma deformace



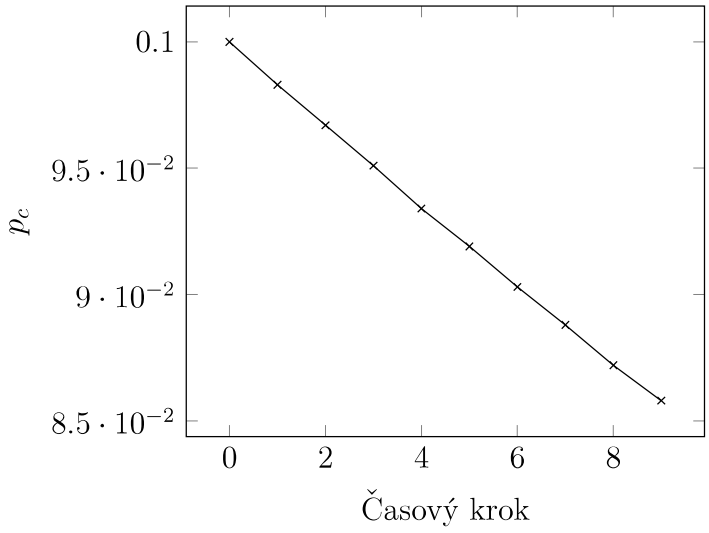
Obrázek 17: Obecně namáhaná krychle: Tvar deformace

Využijme nyní toho, že máme 10 zatěžovacích kroků a dostaneme díky tomu více dat, a proveďme podobnou analýzu vývoje zpevnění jako v minulém kroku. Výpočet prekonsolidačního tlaku v jednotlivých časech podle dosažených napětí (nyní jsou uvedena i napětí smyková, která v předchozích úlohách nevznikala) je zanesen v tabulce 9. Na obrázku 18 je zobrazen vývoj prekonsolidačního tlaku p_c: opět dochází ke změkčení. To je

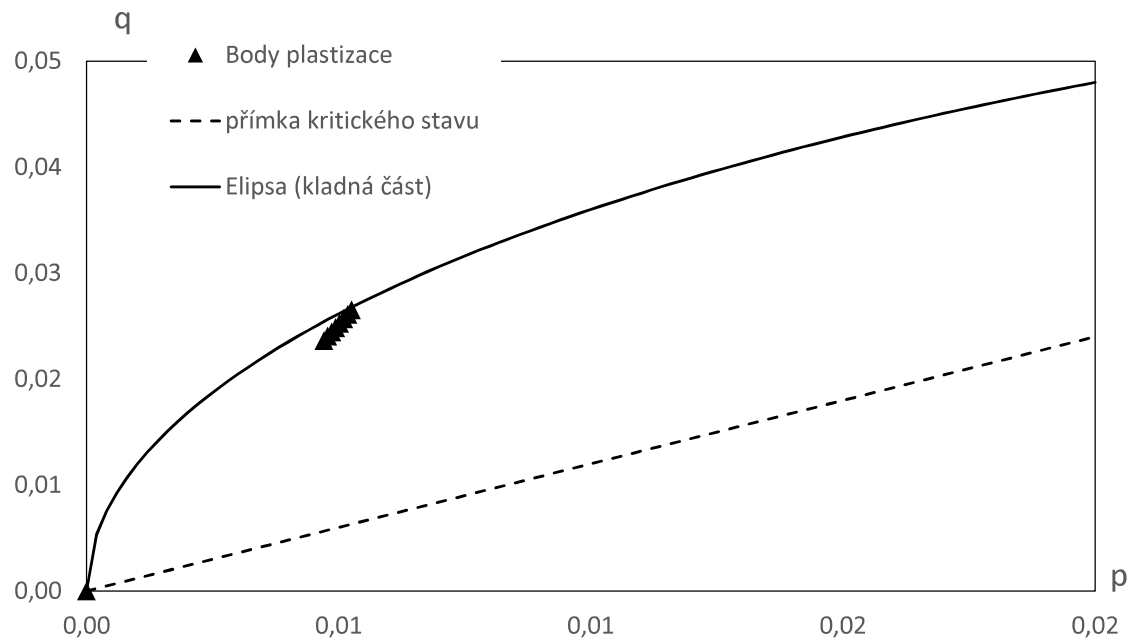
dokumentováno i vývojem napětí zobrazeným na obrázku 19.

Čas	σ_x	σ_y	σ_z	τ_{yz}	τ_{xz}	τ_{xy}	p	q	p_c
0	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,1000
1	-0,0193	-0,0067	0,0103	-0,0002	-0,0034	0,0017	0,0053	0,0265	0,0983
2	-0,0190	-0,0067	0,0101	-0,0002	-0,0033	0,0016	0,0052	0,0261	0,0967
3	-0,0187	-0,0066	0,0100	-0,0002	-0,0033	0,0016	0,0051	0,0257	0,0951
4	-0,0184	-0,0065	0,0098	-0,0002	-0,0032	0,0016	0,0050	0,0253	0,0934
5	-0,0181	-0,0064	0,0097	-0,0002	-0,0031	0,0015	0,0049	0,0249	0,0919
6	-0,0178	-0,0064	0,0095	-0,0001	-0,0030	0,0015	0,0049	0,0245	0,0903
7	-0,0175	-0,0063	0,0094	-0,0001	-0,0029	0,0015	0,0048	0,0241	0,0888
8	-0,0172	-0,0062	0,0093	-0,0001	-0,0029	0,0014	0,0047	0,0237	0,0872
9	-0,0169	-0,0061	0,0091	-0,0001	-0,0028	0,0014	0,0046	0,0233	0,0858

Tabulka 9: Obecně namáhaná krychle: Vývoj prekonsolidačního tlaku (napětí v [MPa])



Obrázek 18: Obecně namáhaná krychle: Vývoj prekonsolidačního tlaku

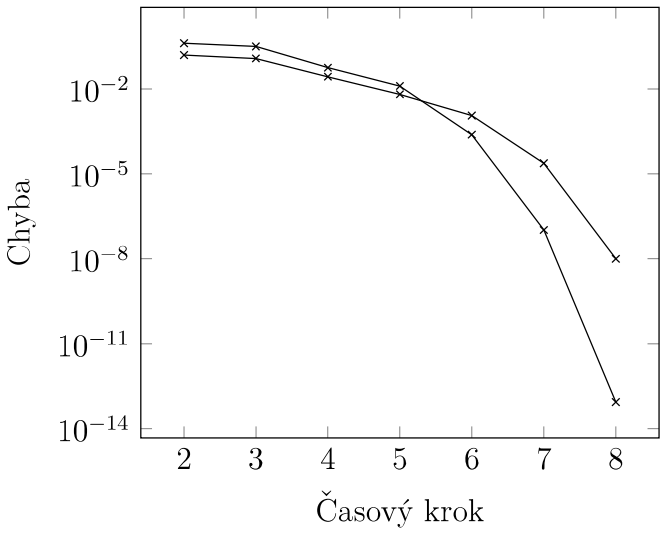


Obrázek 19: Obecně namáhaná krychle: Vývoj napětí v časových krocích vzhledem k počáteční elipse plasticity

I v případě této úlohy je konvergence kvadratická. Předvedeno je to v tabulce 10 a na obrázku 20. Data pocházejí opět z příkazového řádku a jedná se o konvergenci pro poslední zatěžovací krok.

Krok	Chyba σ	Chyba ε
1	6,464e-001	0,000e-000
2	4,102e-001	1,561e-001
3	3,153e-001	1,190e-001
4	1,741e-001	5,636e-002
5	1,200e-002	1,265e-002
6	2,411e-004	1,146e-003
7	1,035e-007	2,345e-005
8	8,678e-014	9,985e-009

Tabulka 10: Obecně namáhaná krychle: Vývoj chyby v posledním zatěžovacím kroku



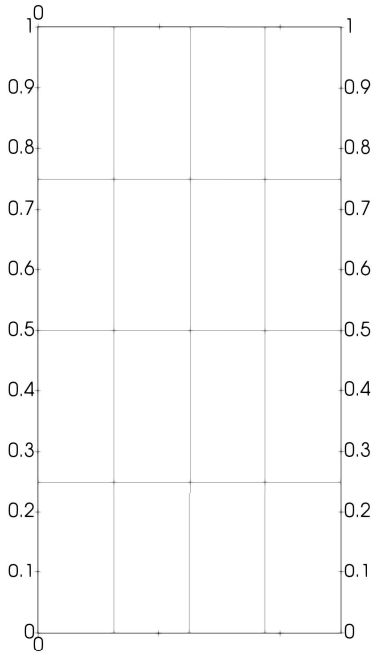
Obrázek 20: Obecně namáhaná krychle: Vývoj chyby v posledním zatěžovacím kroku

5.4 Odvodněná triaxiální zkouška

Nyní opustíme úlohy pouze o jednom konečném prvku a pokusíme se nasimulovat odvodněnou triaxiální zkoušku pomocí 2D sítě o 16 rotačně symetrických prvcích. Prezentovaná úloha je významně inspirována podobným testem představeným v [3].

Triaxiální zkoušky se běžně provádějí na válcových vzorcích, což je pro simulaci pomocí 2D sítě poněkud problematické. Proto použijeme podobně jako v [3] prvky, které jsou jsou rotačně symetrické - daná úloha tedy ve výsledku skutečně představuje válec.

Uvažovaný výsek vzorku zeminy je obdélník v rovině xy o rozměrech $0,5 \times 1.0\,\mathrm{m}$. Je diskretizován šestnácti obdélníkovými prvky v síti 4×4. Popsaná síť je zobrazena na obrázku 21. Použité parametry modelu Cam-Clay shrnuje tabulka 11.



Obrázek 21: Triaxiální zkouška: Síť

Parametr	Hodnota
M	1
$(p_c)_0$	50 kPa
e	0,2
λ	0,066 $(\ln \mathrm{MPa})^{-1}$
κ	0.0077 $(\ln \mathrm{MPa})^{-1}$

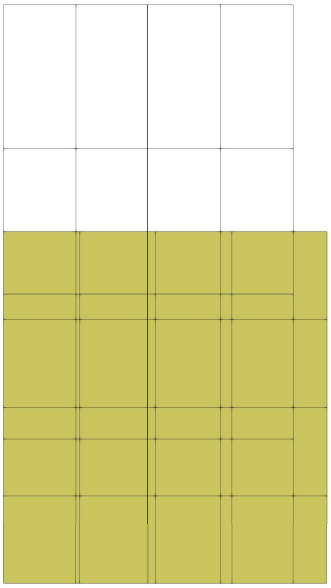
Tabulka 11: Triaxiální zkouška: Parametry modelu

Principem triaxiální zkoušky je zatěžování vzorku zeminy jednoosým tlakem za současného zatěžování tlakem hydrostatickým, což simuluje podmínky, v jakých se materiál běžně vyskytuje v praxi. Zde uvažovaná triaxiální zkouška je odvodněná, což ve výsledku znamená, že odpadají problémy s pórovým tlakem vody a totální napětí se rovnají efektivním, které jsme schopni pomocí našeho materiálového modelu vypočítat [12].

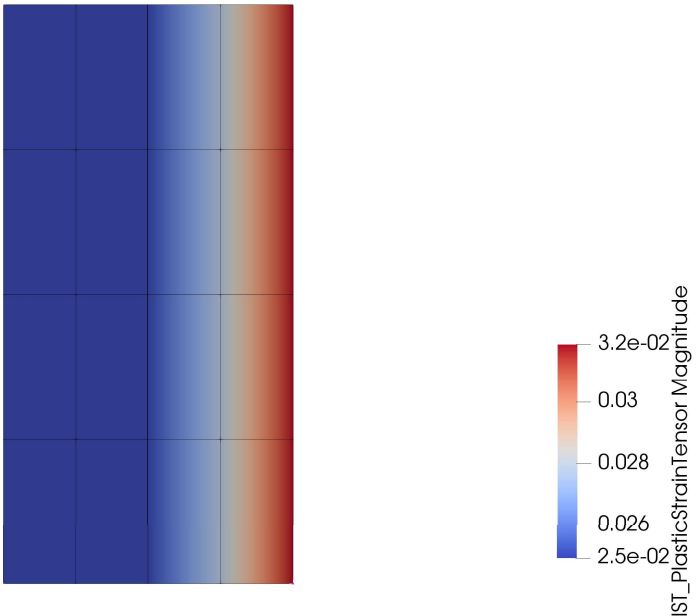
Podmínky triaxiální zkoušky jsou simulovány nejprve zatížením vzorku příčným tlakem, který je nadále držen konstantní, načež je předepisována deformace horního okraje ve svislém směru. Výpočet probíhá v 50 krocích.

Na obrázku 22 je zobrazen v šestnáctinásobném zvětšení výsledný tvar. Jak je vidět, došlo navzdory působícímu hydrostatickému tlaku k příčnému roztažení vzorku. Na obrázku 23 je zobrazena norma plastické deformace $||\boldsymbol{\varepsilon}_p||$ na konci kroku. Poznamenejme,

že hodnoty se všude pohybují v podobném řádu. Menší hodnota v levé části vzorku je dána tím, že je z této strany je vzorek pevně uchycen.



Obrázek 22: Triaxiální zkouška: Výsledný tvar (deformace 16× přehnána)



Obrázek 23: Triaxiální zkouška: Norma plastické deformace

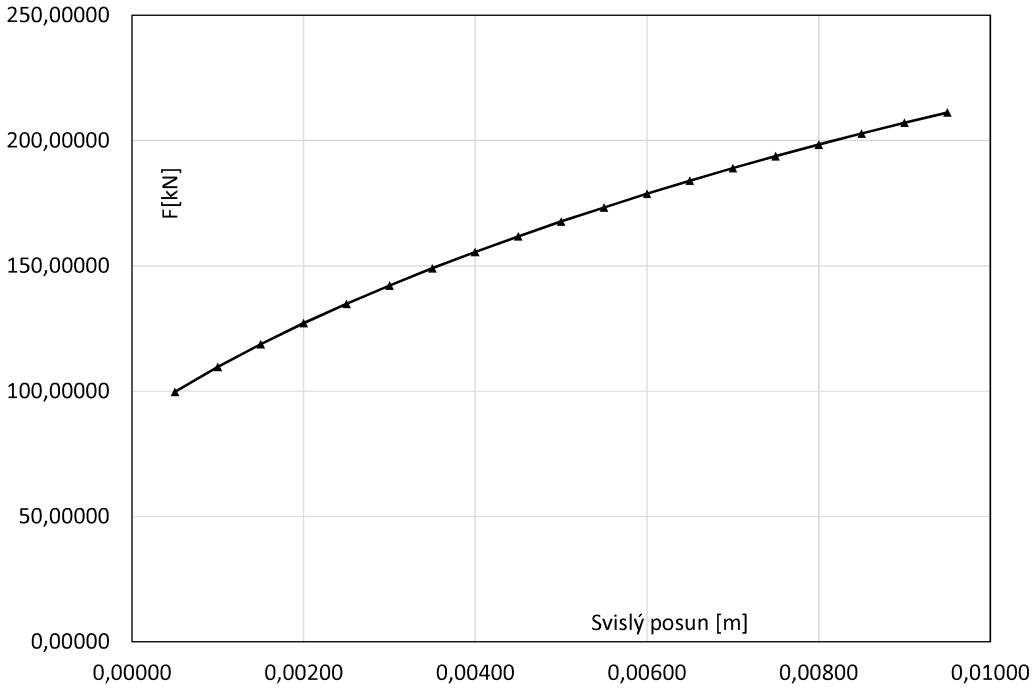
Výsledky zobrazené na těchto obrázcích byly dosaženy za příčného tlaku 10 kPa. Podívejme se nyní na závislost svislé deformace vzorku na osově působící síle[13], což nám dá určitou globální představu o zpevnění či změkčení materiálu.

V tabulce 12 jsou vyneseny síly potřebné k vnesení různých velikostí deformace při příčném tlaku 10 kPa. Na obrázku 24 jsou tato data vizualizována. Dochází k mírnému změkčení.

ε_y	F	ΔF
0,0005	99,6951	-
0,0010	109,7235	10,0283
0,0015	118,7839	9,0605
0,0020	127,1191	8,3351
0,0025	134,8776	7,7585
0,0030	142,1588	7,2812
0,0035	149,0345	6,8758
0,0040	155,5563	6,5218
0,0045	161,7651	6,2088
0,0050	167,6899	5,9249
0,0055	173,3578	5,6679
0,0060	178,7880	5,4302
0,0065	183,9909	5,2029
0,0070	188,9945	5,0036
0,0075	193,7928	4,7983
0,0080	198,4111	4,6183
0,0085	202,8403	4,4291
0,0090	207,0985	4,2582
0,0095	211,1892	4,0907

Tabulka 12: Triaxiální zkouška: Závislost svislého posunu a působící síly

[13]Tato síla není přímo zadána, protože model je ve skutečnosti zatěžován deformací. Byla získána zpětně z reakcí v jednotlivých bodech na horním okraji sítě.



Obrázek 24: Triaxiální zkouška: Závislost svislého posunu a působící síly

5.5 Svah

Jednou z nejtypičtějších praktických úloh mechaniky zemin je posouzení stability svahu, a podobný posudek bude i poslední úloha, kterou v rámci této práce spočítáme.

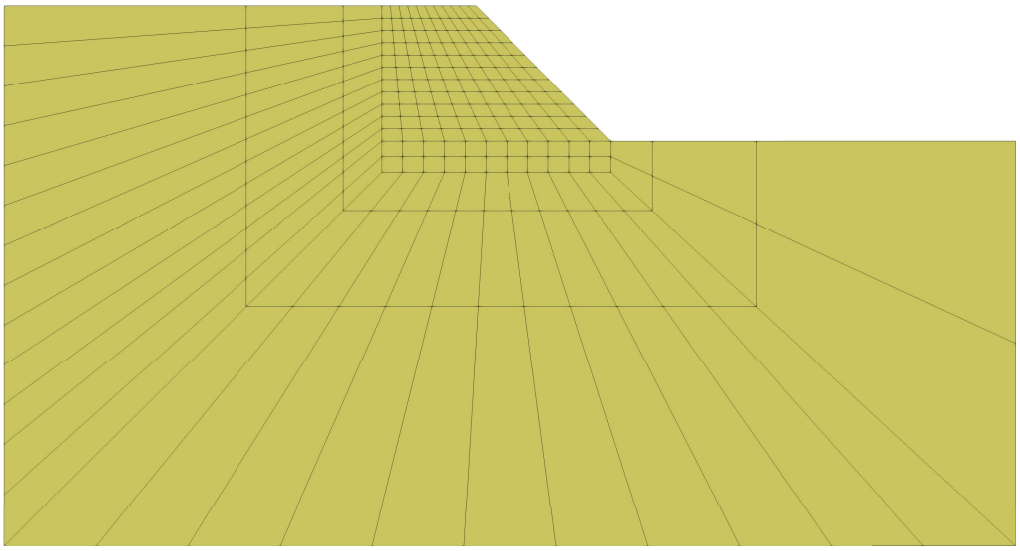
Mějme svah z jílovité zeminy popsané modifikovaným modelem Cam-Clay. Parametry tohoto modelu jsou uvedeny v tabulce 13. Svah je reprezentován v 2D prostředí nepravidelnou sítí 221 čtyřúhelníkových konečných prvků. Je simulováno přitížení horní hrany svahu pomocí předpisu deformace na části terénu nad svahem. Výpočet probíhá v 400 krocích. Očekáváme, že se vytvoří smyková plocha přibližně kruhového tvaru, podle které se bude zemina deformovat [12].

Parametr	Hodnota
M	0,8
$(p_c)_0$	50 kPa
e	0,2
λ	0,066 $(\ln \mathrm{MPa})^{-1}$
κ	0.0077 $(\ln \mathrm{MPa})^{-1}$

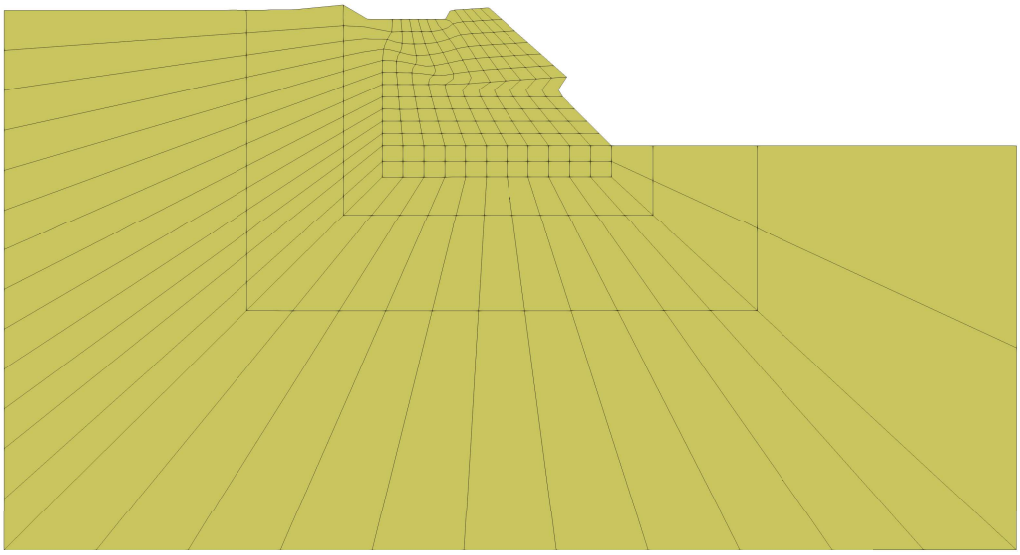
Tabulka 13: Svah: Parametry modelu

Na obrázku 25 je zobrazena síť konečných prvků, která byla použita k popisu úlohy. Obrázky 26 až 37 zobrazují výsledky úlohy. Pro 100., 200., 300. a 400. krok je vždy zobrazen deformovaný tvar (deformace 2× přehnány), norma tenzoru plastické deformace a norma tenzoru napětí.

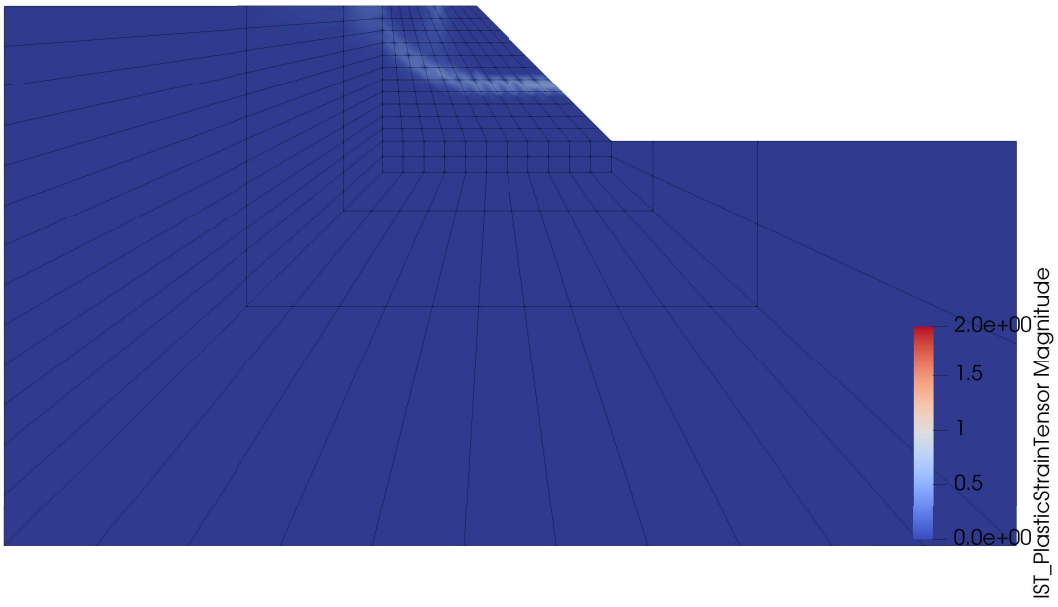
Z dosažených výsledků lze usuzovat, že výpočet proběhl dle předpokladů: vypočtená odezva svahu odpovídá teoretické představě o jeho chování pod daným zatížením. Zejména na obrázcích 35 a 36 je dobře vidět, že skutečně vznikla očekávaná smyková plocha v téměř kruhovém tvaru.



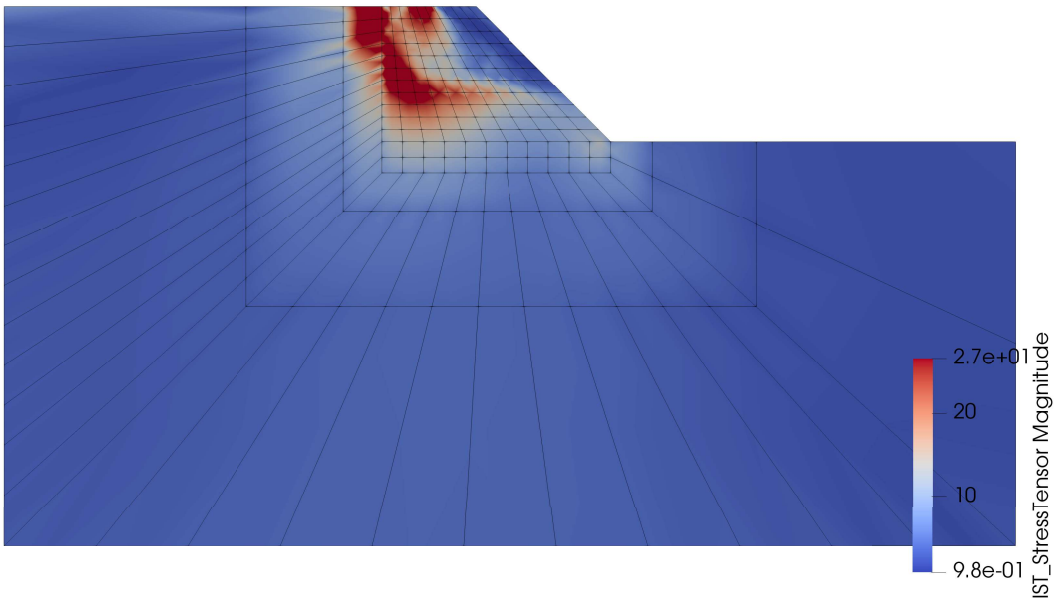
Obrázek 25: Svah: Síť



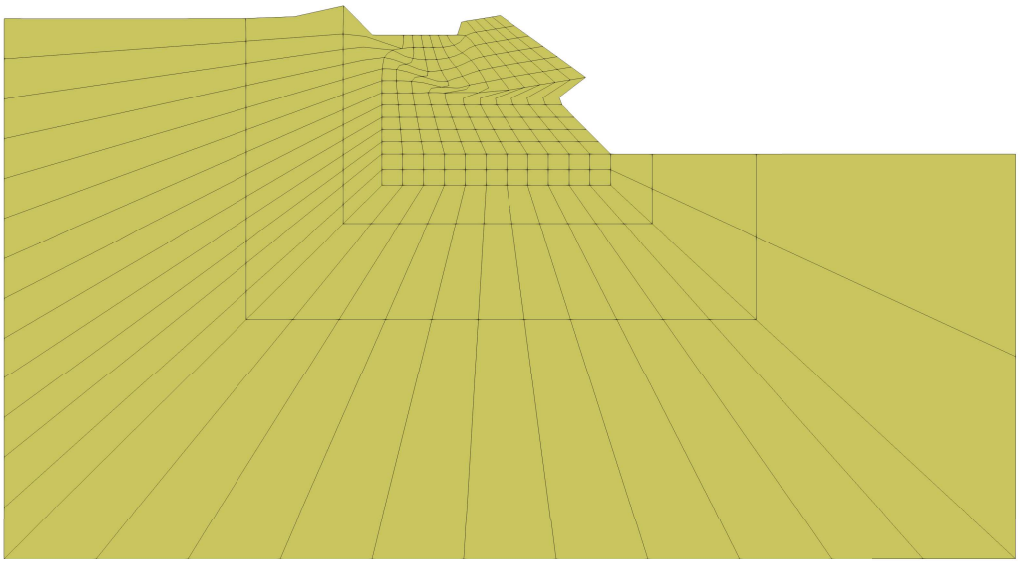
Obrázek 26: Svah: Krok 100: Tvar deformace



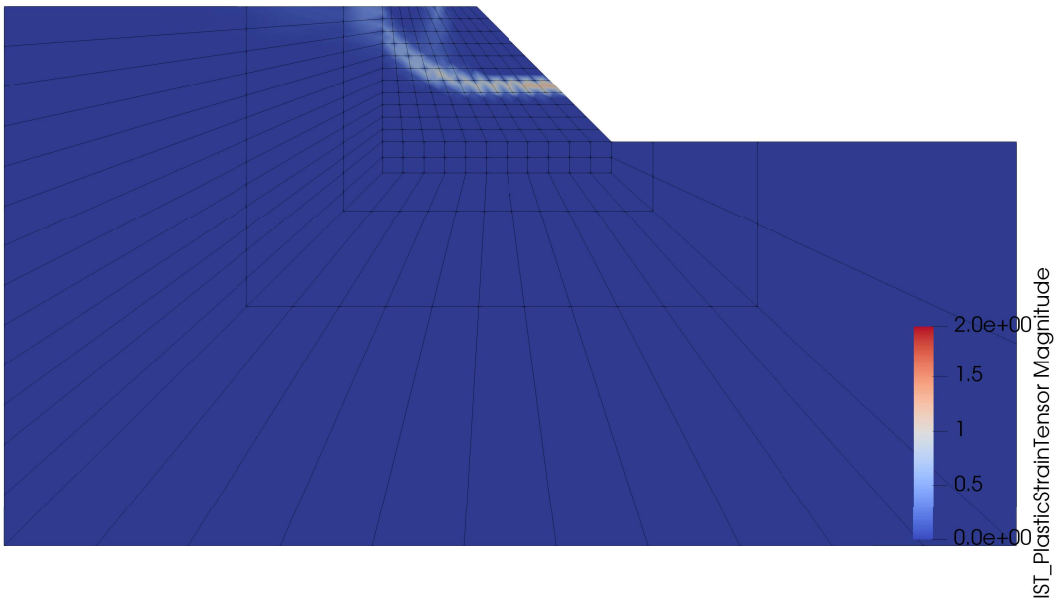
Obrázek 27: Svah: Krok 100: Norma plastické deformace



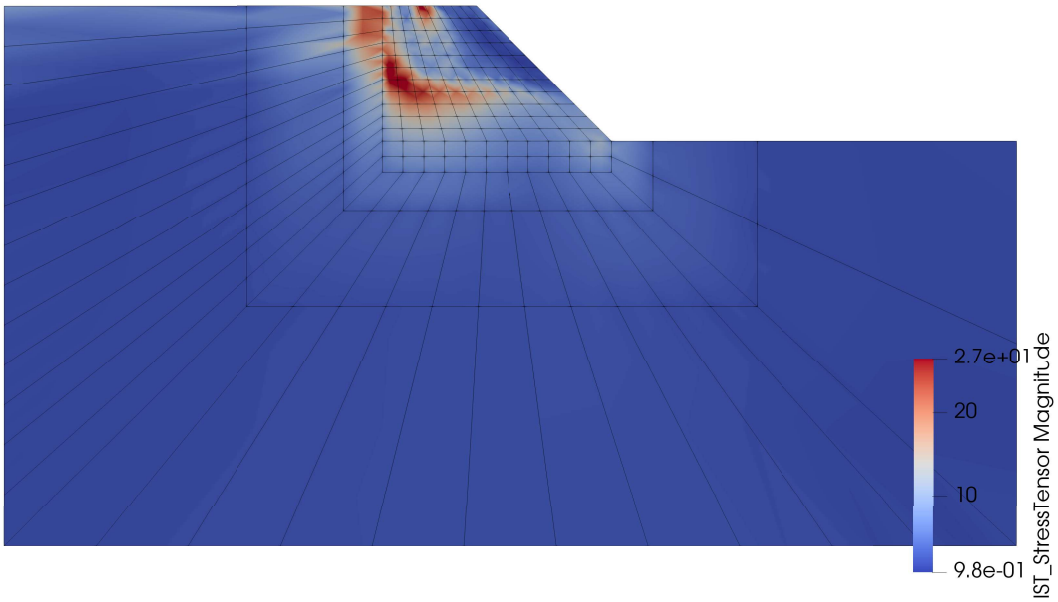
Obrázek 28: Svah: Krok 100: Norma napětí



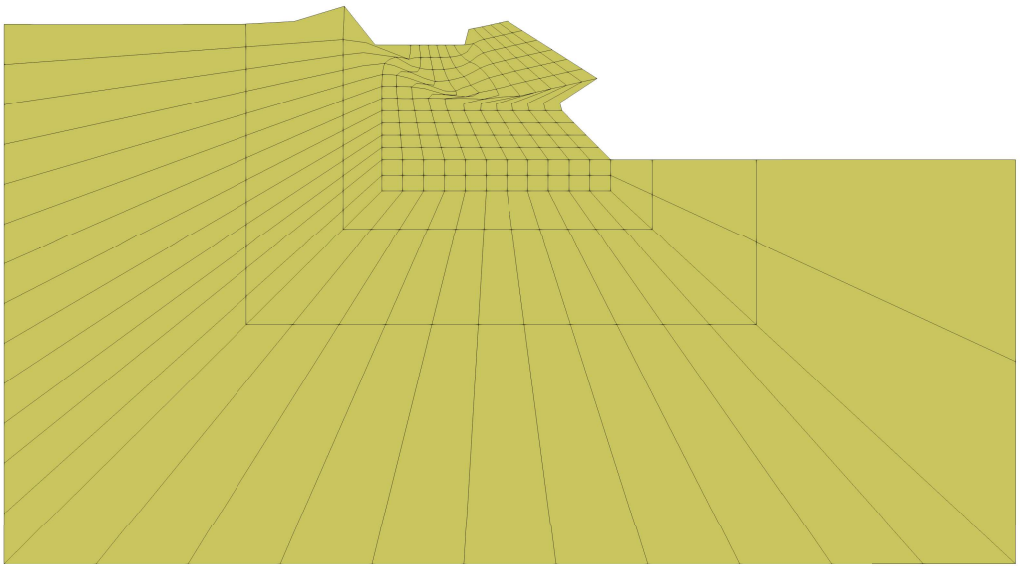
Obrázek 29: Svah: Krok 200: Tvar deformace



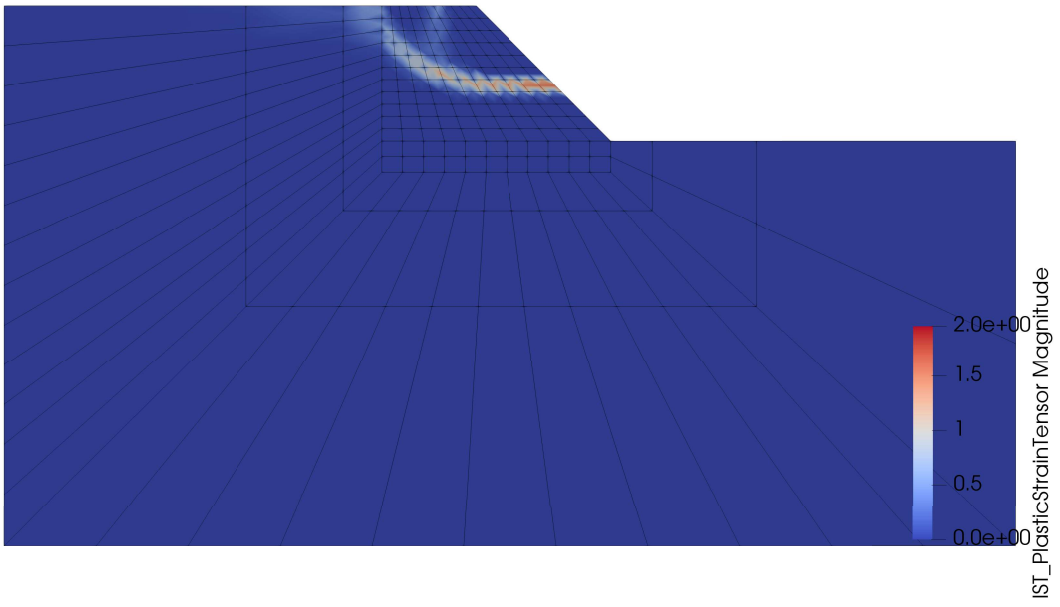
Obrázek 30: Svah: Krok 200: Norma plastické deformace



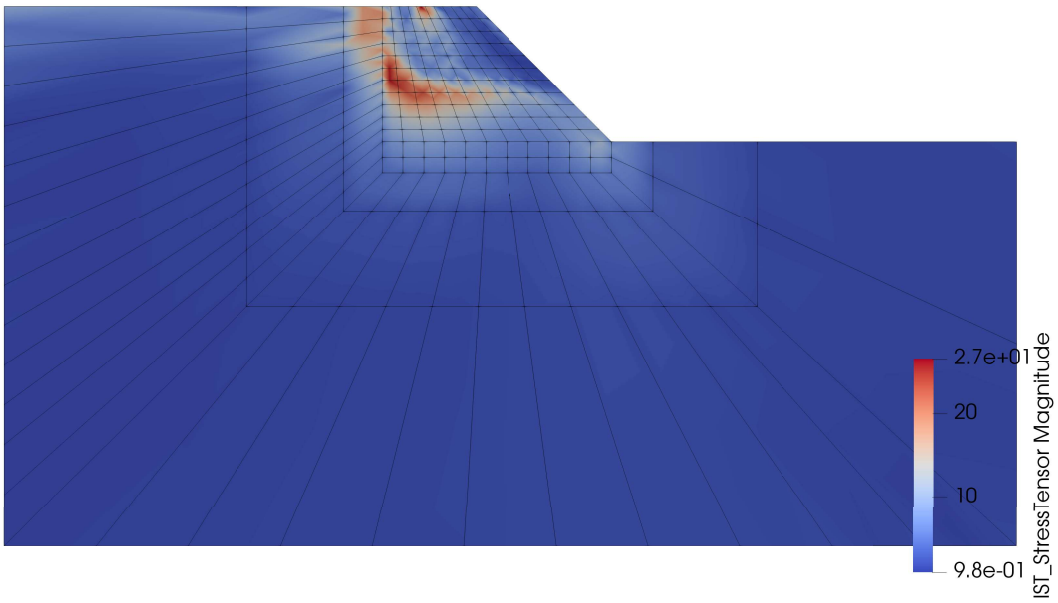
Obrázek 31: Svah: Krok 200: Norma napětí



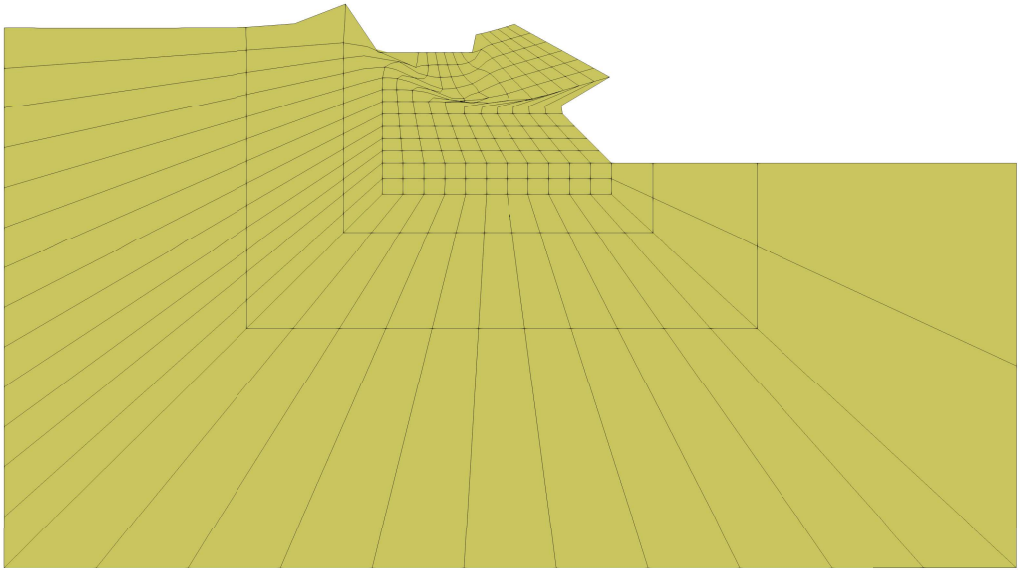
Obrázek 32: Svah: Krok 300: Tvar deformace



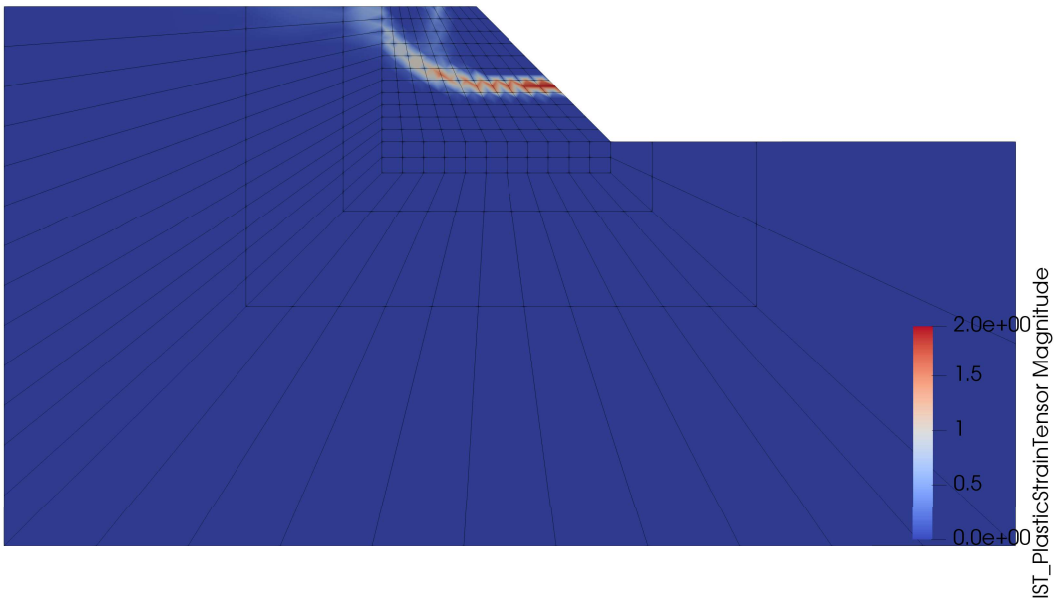
Obrázek 33: Svah: Krok 300: Norma plastické deformace



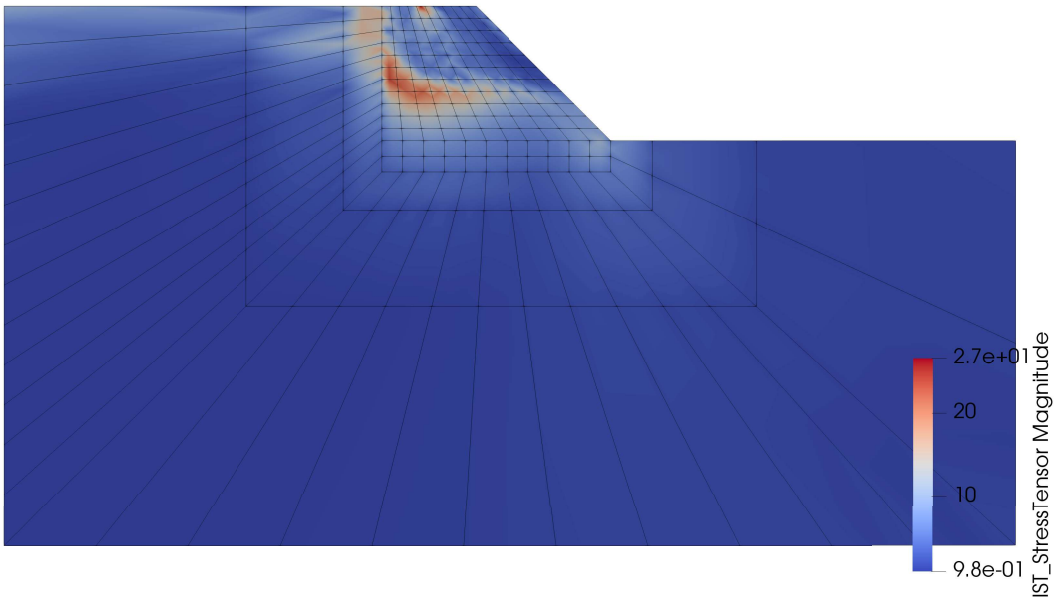
Obrázek 34: Svah: Krok 300: Norma napětí



Obrázek 35: Svah: Krok 400: Tvar deformace



Obrázek 36: Svah: Krok 400: Norma plastické deformace



Obrázek 37: Svah: Krok 400: Norma napětí

6 Závěr

Cílem této práce byla implementace modifikovaného modelu Cam-Clay do programu OOFEM. Odvození algoritmů provedené v kapitole 3 a následná praktická implementace těchto algoritmů popsaná v kapitole 4 vedly ke zdárnému cíli, jak bylo detailně předvedeno a bohatě dokumentováno na úlohách počítaných v kapitole 5. Lze konstatovat, že modul `CamClayMat` v programu OOFEM pracuje dle očekávání. To neznamená, že zde není prostor ke zlepšení; vytvořený kód je zajisté třeba dále testovat, ladit a drobně upravovat. Model také dosud neobsahuje nelineární elasticitu. Její dodatečná implementace by však neměla být složitá a hlavně by neměla zásadně zasáhnout do struktury v současné době implementovaných algoritmů, jak bylo diskutováno v částech 2.3 a 3.2. Přes tyto drobné nedostatky lze tvrdit, že cíle práce se podařilo zdárně splnit.

Dodatky

A Nelineární elasticita

V tomto dodatku stručně popíšeme myšlenku nelineární elasticity, ktará se běžně pojí s modifikovaným modelem Cam-Clay, třebaže my ji v naší implementaci nepoužívame.

Hookeův zákon je třeba uvažovat v přírůstkovém tvaru:

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{D}_e \dot{\boldsymbol{\varepsilon}}_e = K \hat{\boldsymbol{D}}_e \dot{\boldsymbol{\varepsilon}}_e \tag{105}$$

kde

$$\boldsymbol{D}_e = K \boldsymbol{\delta}\boldsymbol{\delta}^T + 2G \left(\boldsymbol{I} - \frac{1}{3} \boldsymbol{\delta}\boldsymbol{\delta}^T \right) \tag{106}$$

$$\hat{\boldsymbol{D}}_e = \boldsymbol{\delta}\boldsymbol{\delta}^T + \frac{3(1-2\mu)}{2(1+\mu)} \left(\boldsymbol{I} - \frac{1}{3} \boldsymbol{\delta}\boldsymbol{\delta}^T \right) \tag{107}$$

Pro vyjádření elastického objemového tečného modulu K lze použít podobnou ideu, jaká byla použita při odvození zákona zpevnění (srovnejme zejména obrázek 2) [5]. Dostáváme závislost modulu na indexu bobtnání κ:

$$K = \frac{1+e}{\kappa} p \tag{108}$$

To pramení z rozšíření myšlenky, představené v části 2.5, že procesy, probíhající po linii bobtnání a tudíž vratné, představují elastickou deformaci, zatímco zbytek deformace je plastický [5].

Nyní z tohoto vztahu vyjadřeme vývoj tlaku, tj. invariantu p. Dostáváme diferenciální vyjádření

$$\dot{p} = K \dot{\varepsilon}_{e,V} = \frac{1+e}{\kappa} p \dot{\varepsilon}_{e,V} \tag{109}$$

které dále vede, analogicky k odvození zákona zpevnění, na diskretizovanou formu

$$p_{n+1} = -p_n \exp\left(\frac{1+e}{\kappa} \Delta\varepsilon_{e,V} \right) \tag{110}$$

Dalším zásadním předpokladem je, že i přes proměnlivost matice tuhosti jako takové, Poissonovo číslo zůstává konstantní [5], což vede k závěru, že

$$G = \frac{3K(1-2\nu)}{2(1+\nu)} \tag{111}$$

Poznamenejme, že takový model je *hypoelastický*, nikoli *hyperelastický*, tj, nedá se odvodit z elastického potenciálu [5].

Integrace rovnice (105) vede k diskretizovaným vyjádřením

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_n + \bar{K}\hat{\boldsymbol{D}}^e \Delta\boldsymbol{\varepsilon}^e \tag{112}$$

$$p_{n+1} = p_n + \bar{K}\Delta\varepsilon_V^e \tag{113}$$

kde průměrný objemový modul $\bar{K}$ je dán jako

$$\bar{K} = \frac{\Delta p}{\Delta\varepsilon_V^e} \tag{114}$$

Nyní můžeme porovnat rovnice (113) a (110). Získáváme jedno z následujících vyjádření, a to v závislosti na tom, zda jsme použili integraci analytickou nebo numerickou:

$$\bar{K} = \frac{p_n}{\Delta\varepsilon_V^e}\left[\exp\left(\frac{1+e}{\kappa^*\Delta\varepsilon_V^e}\right)\right] \tag{115}$$

$$\bar{K} = K_{n+\theta} = K((1-\theta)p_n) + K(\theta p_{n+1}) \tag{116}$$

Na závěr ještě vyjádříme tečnou elastickou tuhost $\boldsymbol{D}_t^e$. Lze to provést přímo z její definice:

$$\boldsymbol{D}_t^e = \frac{\partial\boldsymbol{\sigma}_{n+1}}{\partial\boldsymbol{\varepsilon}_{n+1}^e} = \frac{\partial}{\partial\boldsymbol{\varepsilon}_{n+1}^e}\left(\bar{K}\hat{\boldsymbol{D}}^e : \Delta\boldsymbol{\varepsilon}^e\right) = \bar{K}\hat{\boldsymbol{D}}^e + \hat{\boldsymbol{D}}^e : \Delta\boldsymbol{\varepsilon}^e\left(\frac{\partial\bar{K}}{\partial\boldsymbol{\varepsilon}_{n+1}^e}\right)^T \tag{117}$$

B Nástin alternativního algoritmu pro projekci na nejbližší bod založeného na deviatoricko-volumetrickém rozdělení

Mějme deviatoricko-volumetricky rozdělenou diskretizovanou formu zobecněného Hookeova zákona ve tvaru

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_n + K_{n+\alpha}\Delta\varepsilon_V^e + G_{n+\alpha}\Delta\boldsymbol{e}_e \tag{118}$$

a povšimněme si možnosti deviatoricko-volumetrického rozdělení tenzoru napětí na

$$\boldsymbol{\sigma}_n = p_n\boldsymbol{\delta} + \boldsymbol{s}_n \qquad \boldsymbol{\sigma}_{n+1} = p_{n+1}\boldsymbol{\delta} + \boldsymbol{s}_{n+1} \tag{119}$$

Kombinace těchto dvou vztahů dává

$$p_{n+1} = p_n + K_{n+\theta}(\Delta\varepsilon_V - \Delta\varepsilon_{V,p}) \qquad \boldsymbol{s}_{n+1} = \boldsymbol{s}_n + G_{n+\theta}(\Delta\boldsymbol{e}_{n+1} - \Delta\boldsymbol{e}_{p,n+1}) \tag{120}$$

Zákon zpevnění budeme uvažovat v jeho diskretizované analyticky integrované podobě, tj.

$$(p_c)_{n+1} = -(p_c)_n \exp(\vartheta \Delta\varepsilon_{p,V}) \tag{121}$$

Přistupme nyní k elastické predikci. Zkušební napětí lze definovat jako

$$\boldsymbol{\sigma}_{n+1,tr} = \boldsymbol{\sigma}_n + \boldsymbol{D}_{e,n}\Delta\boldsymbol{\varepsilon}_{e,n+1} \tag{122}$$

což při kombinaci s předchozími vztahy a s definicemi invariantů p a q podle (7) a (8) dává následující vztahy:

$$p_{n+1} = p_{tr} - K_{n+\theta}\Delta\varepsilon_{V,p} = p^{tr} - K_{n+\theta}\Delta\lambda(2p_{n+1} - (p_c)_{n+1}) \tag{123}$$

$$q_{n+1} = \sqrt{\frac{3}{2}}\|\boldsymbol{s}_{tr} - 2G_{n+\theta}\Delta\boldsymbol{e}_p\| \tag{124}$$

$$\boldsymbol{s}_{n+1} = \boldsymbol{s}_{tr} - 2G_{n+\theta}\Delta\boldsymbol{e}^p \qquad \Delta\boldsymbol{e}_p = \frac{3}{2}\Delta\lambda\frac{\partial f}{\partial q} = \sqrt{\frac{3}{2}}\Delta\lambda\frac{\boldsymbol{s}_{n+1}}{\|\boldsymbol{s}_{n+1}\|} \tag{125}$$

Pokusme se nyní vyjádřit $\boldsymbol{s}_{tr}$:

$$\boldsymbol{s}_{n+1} = \boldsymbol{s}_{tr} - 2G_{n+\theta}\sqrt{\frac{3}{2}}\Delta\lambda\frac{\boldsymbol{s}_{n+1}}{\|\boldsymbol{s}_{n+1}\|} \tag{126}$$

$$\boldsymbol{s}_{tr} = \boldsymbol{s}_{n+1} + \sqrt{6}G_{n+\theta}\Delta\lambda\frac{\boldsymbol{s}_{n+1}}{\|\boldsymbol{s}_{n+1}\|} = \left(1 + \frac{\sqrt{6}G_{n+\theta}\Delta\lambda}{\|\boldsymbol{s}_{n+1}\|}\right)\boldsymbol{s}_{n+1} \tag{127}$$

Dostáváme se ke kolinearitě zkušebního a skutečného nového deviatorického napětí a změny plastické deviatorické deformace:

$$\frac{\boldsymbol{s}_{tr}}{\|\boldsymbol{s}_{tr}\|} = \frac{\boldsymbol{s}_{n+1}}{\|\boldsymbol{s}_{n+1}\|} = \frac{\Delta\boldsymbol{e}_p}{\|\Delta\boldsymbol{e}_p\|} \tag{128}$$

a můžeme tedy rozdělit normu ve vztahu (124) a vyjádřit q jako

$$q_{n+1} = \sqrt{\frac{3}{2}}\|\boldsymbol{s}_{tr}\| - 2G_{n+\theta}\sqrt{\frac{3}{2}}\|\Delta\boldsymbol{e}_p\| \tag{129}$$

Výsledkem našeho snažení budou vztahy pro vývoj invariantů napětí p a q:

$$p_{n+1} = \frac{1}{3}\mathrm{tr}[\boldsymbol{\sigma}_{n+1}] = p_{tr} - K\Delta\lambda\frac{\partial f}{\partial p_{n+1}} = p_{tr} - K\Delta\lambda(2p_{n+1} - (p_c)_{n+1}) \tag{130}$$

$$q_{n+1} = q_{tr} - \sqrt{6}G_{n+\theta}\Delta\lambda\frac{\partial f}{\partial q_{n+1}} = \frac{q_{tr}M^2}{M^2 + 6G_{n+\theta}\Delta\lambda} \tag{131}$$

Výsledkem všeho dosavadního postupu je vektor reziduálů ve tvaru $\boldsymbol{R}(\boldsymbol{v}) = (r_1, r_2, r_3, r_4)$, kde neznámé jsou dané jako $\boldsymbol{v} = (p, q, p_c, \Delta\lambda)$. Jednotlivé reziduály vyjádřeme jako

$$r_1 = (1 + 2K_{n+1}\Delta\lambda)p_{n+1} - K_{n+1}\Delta\lambda(p_c)_{n+1} - p_{tr} \tag{132}$$

$$r_2 = q_{n+1} - \frac{q_{tr}M^2}{M^2 + 6G_{n+1}\Delta\lambda} \tag{133}$$

$$r_3 = (p_c)_{n+1} - (p_c)_n \exp(\vartheta\Delta\lambda(2p_{n+1} - (p_c)_{n+1})) \tag{134}$$

$$r_4 = \frac{q_{n+1}^2}{M^2} + p_{n+1}(p_{n+1} - (p_c)_{n+1}) \tag{135}$$

Tato soustava čtyřech rovnic o čtyřech neznámých může být řešena Newtonovou metodou. Jako počáteční hodnoty odhadů neznámých jsou použity, dle očekávání

$$\boldsymbol{v}_0 = (p_t r, q_t r, (p_c)_n, 0) \tag{136}$$

Zásadní výhodou tohoto přístupu oproti přístupu představenému v kapitole 3.2 je redukce počtu rovnic na čtyři neliniární rovnice o čtyřech skalárních neznámých. Není třeba počítat ani žádné maticové inverze. Viz [2].

Seznam obrázků

Seznam tabulek

Seznam výpisů

Reference

[1] AYACHIT, Utkarsh. *The ParaView Guide: A Parallel Visualization Application.* Kitware, 2015. ISBN 978-1930934306.

[2] BORJA, Ronaldo I. a Sueng R. LEE. Cam-Clay plasticity, Part 1: Implicit integration of elasto-plastic constitutive relations. *Computer Methods in Applied Mechanics and Engineering.* 1990, **78**(1), 49-72. ISSN 0045-7825.

[3] DEVI, Dipika. *Object Oriented Non-Linear Finite Element Analysis Framework For Implementing Modified Cam Clay Model.* Guwahati, 2011. Doctoral thesis. Department of Civil Engineering, Indian Institute of Technology Guwahati.

[4] FINE. *GEO5: Uživatelská příručka.* Edice 2018. Praha, c2018.

[5] JIRÁSEK, Milan. a Z. P. BAŽANT. *Inelastic analysis of structures.* New York: Wiley, c2002. ISBN 978-0-471-98716-1.

[6] JIRÁSEK, Milan a Jan ZEMAN. *Přetváření a porušování materiálů: dotvarování, plasticita, lom a poškození.* 2. vyd. Praha: České vysoké učení technické, 2012. ISBN 978-80-01-05064-4.

[7] PATZÁK, Bořek. OOFEM - an object-oriented simulation tool for advanced modeling of materials and structures. *Acta Polytechnica.* 2012, **52**(6), 59–66. ISSN 1210-2709.

[8] POTTS, David M. a Lidija. ZDRAVKOVIĆ. *Finite element analysis in geotechnical engineering: theory.* Reston, VA: Distributed by ASCE Press, 1999. ISBN 0727727532.

[9] SCHOFIELD, Andrew N. a Charles Peter WROTH. Keneth Harry Roscoe, 1914-1970: Obituary. *Geotechnique.* 1970, **20**(2), 123-126. ISSN 0016-8505.

[10] SIMO, J. C. a Thomas J. R. HUGHES. *Computational inelasticity.* New York: Springer, c1998. ISBN 0-387-97520-9.

[11] de SOUZA NETO, E. A., Djordje. PERIĆ a D. R. J. OWEN. *Computational methods for plasticity: theory and applications.* Chichester, West Sussex, UK: Wiley, 2008. ISBN 978-0-470-69452-7.

[12] TERZAGHI, Karl, Ralph B. PECK a Gholamreza MESRI. *Soil mechanics in engineering practice.* 3rd ed. New York: Wiley, c1996. ISBN 0-471-08658-4.

[13] VIRIUS, Miroslav. *Programování v C++: od základů k profesionálnímu použití.* Praha: Grada Publishing, 2018. ISBN 978-80-271-0502-1.