IDENTIFIKACE MATERIÁLOVÝCH MODELŮ S VYUŽITÍM MODERNÍCH METOD UMĚLÉ INTELIGENCE

soutěžní práce

autor:
Anna Kučerová

odborné vedení:

Ing. Michal Mühlbauer
Prof. Ing. Zdeněk Bittnar, DrSc.

Praha, listopad 2001

1 Úvodní poznámky

Podobně jako v jiných inženýrských či vědeckých oborech, i ve stavebnictví se lze setkat s optimalizačními úlohami, které svou povahou ztěžují nebo dokonce znemožňují exaktní řešení. Často je pak nutné řešit takové úlohy metodou pokusu a omylu, jako je tomu v případě navrhování konstrukcí, nebo na základě dalších zákonitostí, které vyplývají např. z fyzikálního významu jednotlivých parametrů námi hledaného řešení, případně jiných znalostí z oblasti matematiky a statistiky. Toho se využívá například při stanovování parametrů materiálových modelů z experimentů. Uvedené postupy mají ovšem řadu nevýhod.

K využití znalostí fyzikálního významu parametrů je nutná značná zkušenost experimentátora, aby se předešlo nesmyslným odhadům jejich hodnot. Modely s jedním nebo dvěma parametry se tak zpravidla dají identifikovat „v ruce“ pomocí vhodných experimentů. Víceparametrové modely vyžadují k identifikaci numerický výpočet a často také další speciální a doplňkové zkoušky pro určení buď konkrétních hodnot některých parametrů nebo stanovení závislosti mezi některými z nich. Lemaitre a Chaboche v [7] popsali některé identifikační metody určování parametrů materiálových modelů. Identifikace modelu zahrnuje určení funkcí, které se vyskytují v modelu a nalezení numerických hodnot koeficientů zavedených těmito funkcemi. To představuje úkol, který nemá žádné dané vodítko či pravidla a v kterém zkušenosti hrají důležitou roli v propojení teorie a experimentů.

Další možností, jak identifikovat parametry materiálových modelů, je využití numerických metod. V tomto případě vycházíme ze situace, kdy známe analytické vyjádření matematického modelu a máme soubor naměřených dat, ve kterých jsou zachyceny průběhy všech proměnných modelu. Úkolem je najít hodnoty parametrů modelu tak, aby model co nejlépe vystihoval průběhy naměřených dat. Z matematického hlediska se jedná o nalezení takových parametrů, aby byl součet čtverců rozdílu naměřených hodnot a hodnot modelu minimální. Matematický postup k řešení takové úlohy ovšem předpokládá existenci minimálně první derivace funkce modelu podle jejích parametrů a celé řešení často vede k soustavě nelineárních rovnic, jejichž řešení je problematické.

Výhody genetických algoritmů

Již několik let rostou výpočtové schopnosti počítačů i jejich dostupnost pro běžného uživatele. Tato skutečnost umožňuje využívat různé stochastické algoritmy, které jsou často výpočtově náročné. V případě optimalizací se tím otevírá cesta pro uplatnění tzv. genetických algoritmů (viz např. [2, 3]) a zároveň i jejich využití při opět výpočtově náročném trénování neuronových sítí, které by mohly sloužit k přibližnému odhadu parametrů materiálových modelů z experimentů (viz [1]).

Dalším trendem poslední doby je také možnost paralelizace výpočtu, tzn. rozložení výpočtu mezi více počítačů, které jsou právě k dispozici. I tohoto faktu je možné s genetickými algoritmy dobře vyžít, neboť jejich výpočet je snadno rozdělitelný na dílčí nezávislé výpočty.

Genetické algoritmy zároveň splňují náš nejdůležitější požadavek. Při řešení jim stačí znát pouze funkční hodnotu optimalizované funkce a případně odhad mezí pro jednotlivé parametry, ve kterých se má hledané řešení nalézat. Další požadavky na optimalizovanou funkci se nekladou, funkce např. nemusí být ani diferencovatelná, ani spojitá.

Řešené úlohy

Během minulého roku bylo pomocí genetických algoritmů řešeno vedle některých matematických úloh i několik inženýrských problémů:

– optimalizace ceny železobetonového nosníku

– hledání jednotkové periodické buňky pro simulaci chování kompozitních materiálů

– analytické modelování retenční čáry

– analytické modelování závislosti deformace hornin na čase

Problému navrhování železobetonového nosníku tak, aby při dané únosnosti byla minimalizovaná jeho cena, se v [11, 12] věnuje podrobněji M. Lepš. Další úlohou bylo hledání jednotkové periodické buňky pro simulaci chování kompozitních materiálů, kterým se v [13] zabývá J. Zeman. Výsledky v této práci prezentovaného algoritmu SADE při řešení obou těchto úloh ve srovnání s dalšími genetickými algoritmy jsou prezentovány v [17]. Ve dvou dalších případech pak bylo cílem stanovit parametry materiálových modelů tak, aby modelovaná křivka co nejlépe prokládala naměřené hodnoty při experimentech. Řešení těchto dvou úloh bude podrobněji popsáno v předkládané práci.

K hlubšímu testování vlastností a schopností prezentovaného genetického algoritmu SADE byla použita sada testovacích matematických funkcí, publikovaná v [4]. Výsledky algoritmu SADE na těchto funkcích jsou prezentovány v [16]. Další dvě testovací úlohy a jejich řešení pomocí čtyř různých algoritmů, mezi nimi algoritmem SADE, jsou prezentovány v [17].

2 Problém stanovení parametrů retenční čáry

Retenční čára

Retenční čára je základní hydrostatickou charakteristikou půd a obecně porézních materiálů. Jde o funkční závislost vlhkosti na vlhkostním potenciálu čili vztah sacího tlaku zeminy a stupně jejího nasycení. Tato závislost se stanovuje experimentálně, nejčastěji laboratorně, na neporušených půdních vzorcích; výsledkem těchto experimentů je pak určitý soubor dat, který je třeba dále zpracovat. Účelem je získání analytického vyjádření tvaru retenční čáry. Podrobnější informace o retenční čáře je možné najít např. v [10].

Kvůli komplikacím, které vznikají při určování parametrů retenční čáry, je v současné době tato charakteristika považována spíše jako doplňková. Pokud by se však tyto parametry podařilo určit s dostatečnou věrohodností, mohly by zaujmout výraznou roli při zatřiďování zemin a půd. V případě provedení rozsáhlejších měření totiž tyto charakteristiky předkládají úplný popis půd daných lokalit velice jednoduše, leč účelně, a to počínaje jejich zemědělskou využitelností a konče stanovením jejich retenční vodní kapacity, což je primární charakteristika pro určení odtokových poměrů při modelování povodňových stavů.

Současný stav pokud jde o zjišťování analytického vyjádření retenční čáry

Při stanovení a interpretaci retenčních čar se objevuje závažný problém související s heterogenitou půdního prostředí a z toho dále plyne značný rozptyl retenčních čar. Dalším problémem, souvisejícím s využitím retenčních čar jakožto vstupních dat pro numerické simulační modely, je aproximace jejich průběhu. V současné době se pro tuto aproximaci používá hlavně model van Genuchtena, který aproximuje retenční čáru pomocí vztahu 1.

$$\theta_E = \frac{1}{(1+\alpha|h|^n)^m} \tag{1}$$

Řešení spočívá v optimalizaci minimálně tří parametrů (α,m,n). V případě použití této aproximace pro celý soubor měření je pak možné získat parametry, které budou již samy o sobě popisovat jednotlivé důležité půdní charakteristiky.

Vedle provedení řady podrobných měření, je nezbytné navržení vhodné metody, která by byla schopna aproximaci průběhu retenční čáry řešit. K tomuto účelu byla vyvinuta řada programů,

které využívají různé numerické metody. Z nich nejznámější je program RETC vytvořený přímo týmem van Genuchtena. U nás byla vytvořena česká obdoba tohoto softwaru UFRETC autory Valentová, Valenta.

Oba uvedené programy řeší tuto problematiku běžnými metodami matematické statistiky. Je však otázkou, jak velké množství výpočtů a s jakou přesností by bylo třeba provést, abychom získali hodnoty odpovídající skutečné fyzikální charakteristice, respektive jak dlouho by musel výpočet programu trvat, aby bylo možné s určitostí říci, že dochází ke konvergenci. Sám van Genuchten stanovil definiční obory svého vztahu: $\alpha \in \langle 10^{-3} \div 10^{-2} \rangle, m \in \langle 0 \div 1 \rangle, n \in \langle 1.5 \div 6 \rangle$. Nicméně numerická metoda, vyvinutá přímo van Genuchtenem, často vrací hodnoty, které se zcela vymykají jakékoliv fyzikální interpretaci, jako např. α dosahující hodnot o tři až čtyři řády nad horní mez tohoto intervalu. Navíc ke konvergenci dochází jen v některých speciálních případech, takže otázka věrohodnosti těchto charakteristik je velice problematická.

3 Hledání parametrů vazko-plastického modelu hornin

3.1 Lemaitreův model

V tomto oddíle se zabýváme problémem spojeným s časovým nárůstem nevratné deformace materiálu. Podobně jako pro úlohu retenční čáry i zde jde o nalezení analytického vztahu z experimentálních měření a určení jeho parametrů. V praxi se většinou nejprve provede série experimentů, ve kterých se vyskytují všechny neznámé proměnné z modelu. Úkolem je pak najít pomocí experimentálních dat hodnoty parametrů modelu tak, aby model co nejlépe vystihoval naměřená data a popisoval chování materiálu. K dobré identifikaci modelu je třeba velký počet experimentů, protože každá z proměnných se může měnit ve velkém rozsahu. Identifikace modelu tedy spočívá v určení funkcí, které se vyskytují v modelu a nalezení numerických hodnot parametrů, které definují tyto funkce. Identifikace parametrů materiálových modelů vyžaduje také velkou zkušenost v materiálovém modelování.

Pro materiály s krystalickou mřížkou, jako kovy nebo některé horniny, se pro popis vazko-plastického chování často používá modelu navrženého Lemaitrem (viz [7]), který vychází z Perzynovy teorie vazko-plasticity. Teorie vazko-plasticity popisuje tečení materiálu dotvarováním, které narozdíl od plasticity závisí na čase. U krystalických látek nárůst permanentní deformace odpovídá mikroskopickým mechanismům unvitř krystalů či zrn nebo na jejich rozhraních.

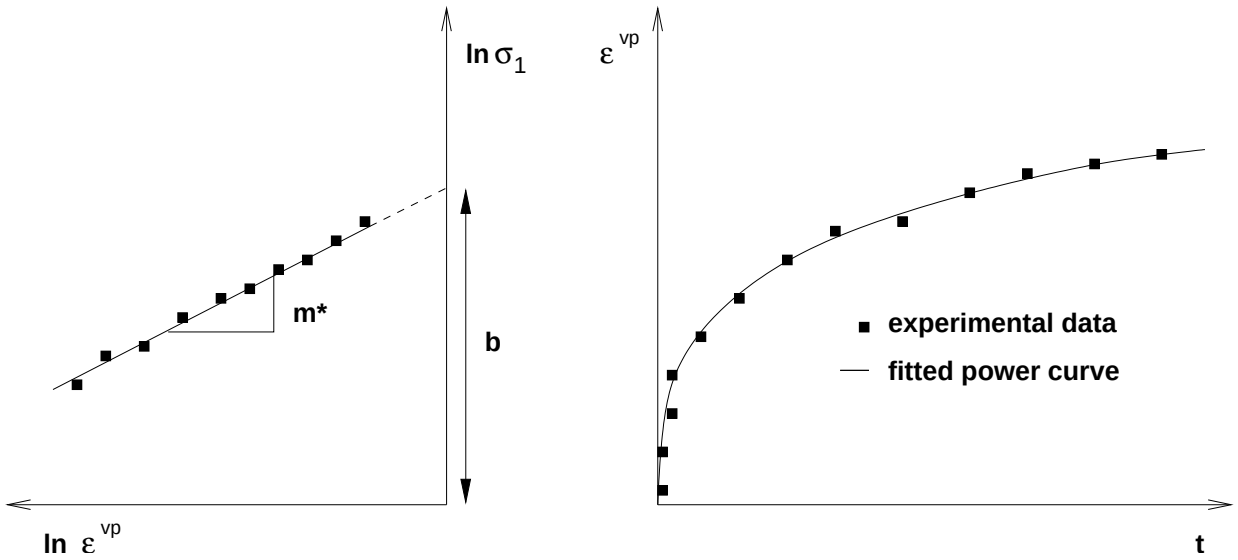
Lemaitreův model pro plastické materiály citlivé na rychlost deformace poskytuje konstitutivní vztah pro primární fázi dotvarování, kdy se rychlost vazko-plastické deformace zmenšuje. Model vychází ze základního předpokladu, že pružnou část deformace lze oddělit od nepružné a to i v jejich vyjádření v rychlostech [9]. Nepružná rychlost deformace kombinující viskózní a plastické chování se zpevněním je vyjádřena vztahem 2

$$\dot{\varepsilon}_{ij}^{vp} = A\sigma_{ij,dev}^{n}(\varepsilon^{vp})^{m} \tag{2}$$

kde
- $\dot{\varepsilon}_{ij}^{vp}$ je rychlost vazko-plastické deformace
- $\sigma_{ij,dev}$ deviátor tenzoru napětí
- ε^{vp} kumulovaná vazko-plastická deformace

V tomto modelu se vyskytují tři materiálové parametry:

- A parametr viskozity materiálu ($A > 0$)
- n exponent napětí $n \geq 1$
- m^* exponent materiálového zpevnění ($m^* = -m/n \geq 0$)



Obrázek 1: Vlevo: kvazi-statická zkouška s konstantní řízenou rychlostí deformace, Vpravo: zkouška dotvarování

Z kvasi-statických zkoušek (obr. 1), při kterých je materiál zatěžován s řízenou rychlostí deformace ($\dot{\varepsilon}^{tot} = const.$) a jejíž hodnoty jsou dostatečně malé, aby vazko-plastická deformace narůstala v reálném čase, můžeme zjistit závislost dvou parametrů z grafu vyjadřujícího vztah mezi vazko-plastickou deformací a deviátorem napětí v logaritmickém měřítku.

$$m = -m^* \, n \tag{3}$$

Parametr m^* lze odečíst z grafu jako sklon přímky obdržené lineární regresí. Další parametr A lze také vyjádřit v závislosti na n, ale odečítání funkční hodnoty na ose $\ln \sigma_1$ je nespolehlivé kvůli velkému rozptylu hodnot při malé změně sklonu křivky. Tím jsme počet stupňů volnosti zredukovali na dva.

Dva zbývající parametry se tedy určí ze zkoušky dotvarování pomocí lineární regrese na linearizovaném vztahu pro vazko-plastický model. Integrací modelu 2 obdržíme vyjádření vazko-plastické deformace v závislosti na čase, což odpovídá funkční závislosti proměnných při zkoušce dotvarování.

$$\epsilon^{vp} = \left[(1-m) A \sigma^n t \right]^{\frac{1}{1-m}} \tag{4}$$

Rovnice 4 popisující zkoušku dotvarování je mocninná funkce s proměnnou t (čas), jejíž zápis lze převést na vztah 5.

$$\varepsilon^{vp} = \alpha t^{\beta} \tag{5}$$

K identifikaci nových parametrů (α, β) se používá lineární regrese s normou nejmenších čtverců. Funkce se nejprve musí linearizovat logaritmováním obou stran na tvar

$$\ln \varepsilon^{vp} = \gamma + \beta \ln t \tag{6}$$

kde $\gamma = \ln \alpha$. Minimalizovaná funkce je pak ve tvaru

$$f = \sum_{i=1}^{p} \left(\gamma + \beta \ln t_i - \ln \varepsilon_i^{vp} \right)^2 \tag{7}$$

kde p je celkový počet experimentálních dat (ϵ_i^{vp}, t_i). Známé koeficienty γ, resp. α a β jsou pak převedeny na původní hledané parametry porovnáním rovnic 4 a 5 s využitím vztahu 3:

$$m = 1 - \frac{1}{\beta} \tag{8}$$

$$n = -\frac{m}{m^*} \tag{9}$$

$$A = \frac{\alpha^{1-m}}{1-m} \sigma^{m/m^*} \tag{10}$$

Takto provedená lineární regrese na exponenciální funkci je naprogramovaná v řadě aplikací. Podmínkou řešitelnosti vztahu 6 je ovšem $\varepsilon^{vp} > 0$, která mnohdy nebývá splněna. V některých sériích naměřených dat je totiž počáteční nárůst ε^{vp} natolik malý, že měřící přístroj, kvůli své limitované přesnosti a kolísání elektrických veličin, po určitou dobu zaznamenává nulovou nebo i zápornou hodnotu ε^{vp}. Abychom výše uvedený postup s lineární regresí mohli použít, je nutné tato "nevhodná" data opravit na hodnoty větší a blízké nule, čímž do řešení zanášíme určitou chybu, která má na řešení zásadní vliv. Stejně tak musí být splněna i podmínka $t > 0$ a proto musíme vyloučit měření v nulovém čase. Popsanými úpravami se tak výsledné řešení může od optimálního nesrovnatelně lišit.

Problém by mohl být řešen i nelineární regresí, ta je ovšem komplikovanější a přibližná, protože vyžaduje řešení nelineárních soustav rovnic (napr. iterační Newtonovu), kde se vyskytují druhé derivace minimalizovaných funkcí. Navíc je nutné získat dobrou počáteční aproximaci a to je ve více dimenzích problém.

3.2 Lemaitreův model s poškozením

Druhým řešeným příkladem je určení parametrů modelu, který je kombinací Lemaitreova vazko-plastického modelu pro primární fázi dotvarování a modelu poškozování materiálu s parametrem porušení v terciální fázi dotvarování. Model nepopisuje sekundární fázi dotvarování, kdy je rychlost vazko-plastické deformace konstantní, primární fáze přechází přímo v terciální, což pro zatížení blízká mezi porušení není daleko od skutečnosti. Model vazko-plastické deformace s porušením jako funkce času má tvar [8]

$$\varepsilon_{ij}^{vp} = \left[-\frac{M+N}{M(k-N)} \left(\frac{\sigma_{ij}}{A} \right)^{-r} \left(\frac{\sigma_{ij}}{K} \right)^{N} \left[\left(1 - (k+1) \left(\frac{\sigma_{ij}}{A} \right)^{r} t \right)^{\frac{k-N}{k+1}} - 1 \right] \right]^{\frac{M}{M+N}} \tag{11}$$

V modelu se vyskytují parametry pro vazko-pružný stav materiálu K, M, N, které mají podobný význam jako parametry z předchozího modelu. Zbývající parametry k, r, A jsou materiálové konstanty popisující porušování materiálu (viz [7]).

V tomto případě je rovnice modelu 11 opět mocninnou funkcí s proměnnou t (čas). Její matematický zápis lze převést na vztah 12.

$$\varepsilon^{vp} = \gamma((1 - \alpha t)^{\beta} - 1)^{\delta} \tag{12}$$

Ze srovnání rovnic 11 a 12 je zřejmé, že dvě z materiálových konstant K, M, N, k, r, A jsou závislé. Parametr N má stejný význam jako n v 3.1 a lze jej považovat za známý. Pro parametr A existují tabulkové hodnoty pro určité materiály anebo lze tento parametr odhadnout.

Model má tedy čtyři nezávislé parametry. V tomto případě však již není možné použít jednoduché metody jako lineární regrese k jejich určení. Problém by byl řešitelný nelineární regresí za cenu derivování funkce podle jednotlivých parametrů. Hajdu v [8] řešil identifikaci modelu parametrickou studií, ale výsledek je pouze mírně uspokojivý a pracný.

4 Použitá metoda

Již minulý rok jsme se na katedře Stavební mechaniky zabývali optimalizací železobetonového nosníku. V tomto případě se jako velmi účinná metoda ukázala tzv. diferenciální evoluce (viz [5, 6]). Tato metoda se, podobně jako o něco déle známé genetické algoritmy, inspirovala v přírodě evolučními principy: křížení genetického kódu k tvorbě nových jedinců a selekce jakožto přirozeného výběru. Tyto principy jsou však v tomto případě formulovány tak, aby nepracovaly s řetězci nul a jedniček, jak tomu bylo tradičně u genetických algoritmů, ale přímo s řetězci (resp. vektory) reálných čísel. Metoda je po algoritmické stránce velice jednoduchá, zároveň velmi spolehlivá i rychlá.

Hlavní motivací k vývoji genetických algoritmů na naší fakultě bylo nalezení metody na trénování neuronové sítě, která by se měla používat k přibližnému odhadu parametrů materiálových modelů z experimentů. V tomto případě by se jednalo o optimalizační úlohu s vysokým počtem proměnných, které by představovaly váhy neuronové sítě. Pro posouzení vlivu velkého počtu proměnných na chování algoritmu byla vytvořena representativní jednoduchá úloha s nastavitelným počtem proměnných a na této úloze byla diferenciální evoluce testována. Nárůst její výpočtové náročnosti byl s růstem počtu proměnných nepřijatelný. Proto byla diferenciální evoluce upravena a tak vytvořen nový algoritmus SADE[1], jehož výpočtová náročnost rostla s počtem proměnných jen lineárně. Algoritmus SADE má výsledně tuto podobu:

```
void SADE ( void )
{
    FIRST_GENERATION ();
    while ( to_continue )
    {
        MUTATE ();
        LOCAL_MUTATE ();
        CROSS ();
        EVALUATE_GENERATION ();
        SELECT ();
    }
}
```

4.1 Stručně o jednotlivých funkcích

V následujícím textu je stručně rozebrán význam jednotlivých funkcí algoritmu. Podrobněji je popsán v článku [14] nebo na internetové stráce [15]. Větší pozornost je věnována několika konstantám, se kterými algoritmus pracuje. Tyto konstanty, můžeme je označit za parametry algoritmu, mají na jeho chování značný vliv. Jejich ideální nastavení se ale může pro různé řešené úlohy významně lišit. Otázka vhodné volby jejich hodnot je tedy další problém, kterým je nutné se zabývat.

[1] Simplified Atavistic Differential Evolution

FIRST_GENERATION

V této funkci volané na začátku procesu je generována populace jedinců, resp. vektorů, jejichž jednotlivé souřadnice jsou náhodná čísla v zadaných mezích. Již v této funkci vystupuje jedna konstanta, nebo-li parametr algoritmu. Je totiž nutné určit, jak veliká populace jedinců se má generovat. Při řešení několika testovacích problémů se ukázalo rozumné volit tento počet přímo úměrný počtu proměnných řešené funkce, resp. dimenzi problému. Zůstává ale parametr `pool_rate`, kterým se k získání velikosti populace přenásobí známý počet proměnných. Ve většině řešených funkcí byl algoritmus nejúspěšnější s parametrem `pool_rate = 10`. Výjimečně bylo výhodné ho zvýšit třeba až na hodnotu 30. Mimo jiné je možné tímto parametrem zpomalovat konvergenci algoritmu, což může zamezit „pádu“ do lokálního extrému.

MUTATE

Ve funkci `MUTATE` je vytvářen jistý počet nových jedinců mutací jedinců ze současné populace. Postup je tento: z populace je náhodně vybrán jedinec, kterého označíme A. Dále je vytvořen zcela nový jedinec se všemi souřadnicemi jakožto náhodnými čísly[2] ze zadaných mezí; toho označíme B. Nový jedinec, který je přidán do populace, vzniká posunutím vektoru A o náhodnou část úsečky AB směrem k vektoru B. V tomto provedení operátoru mutace se objevuje další konstanta, pomocí které je třeba určit počet jedinců, kteří se mají vytvořit právě tímto operátorem. Tuto konstantu, zvanou `radiation`, je rozumné volit od 0 do 30% počtu jedinců na počátku cyklu, nejčastěji však právě 10%. Čím větší tato konstanta je, tím větší se udržuje rozptyl jedinců v populaci a zpomaluje se konvergence. Může ovšem nastat i případ, kdy díky velikosti tohoto parametru algoritmus v průběhu výpočtu přestal konvergovat úplně.

LOCAL_MUTATE

Tento operátor vznikl jako doplňkový k rychlejšímu dohledávání řešení s vyšší přesností. Vytváří opět jistý počet nových jedinců v těsném okolí již existujících jedinců v populaci. To provede tak, že nového jedince vytvoří posunem všech souřadnic náhodně vybraného vektoru z populace o náhodně volenou nepatrnou vzdálenost. Posun je pro každou souřadnici volen zvlášť jako část z rozsahu definičního oboru pro danou proměnnou. Tato část je vybírána náhodně z intervalu $-0,0025$ až $0,0025$.

Stejně jako u předchozího operátoru MUTATE je i v tomto případě nutné určit počet jedinců, kteří se mají vytvořit. Nezbytnou konstantou, zvanou `local_radiation`, definující tento počet opět jako procento z počtu jedinců v populaci na počátku cyklu, nastanovujeme ve stejném rozmezí jako v předchozím případě, tedy na 0 až 30%, nejčastěji opět na 10%.

CROSS

Tímto operátorem je vytvořen právě takový počet nových jedinců, aby jejich celkový počet byl právě dvojnásobkem jejich počtu na počátku cyklu. Tím odpadá možná očekávaná další konstanta, která by tento počet určovala zvenčí. Zároveň je nutné tento fakt uvažovat při definovaní parametrů `radiation` a `local_radiation`, aby v průběhu cyklu nedocházelo k absolutnímu přeskakovaní tohoto operátoru, protože součet `radiation`+`local_radiation` bude větší než 1 a tak ještě před zavoláním tohoto operátoru bude počet jedinců dvojnásobný.

Nový jedinec vzniká na principu tzv. diferenciálního křížení. Z populace jsou náhodně vybráni 3 jedinci, řekněme, že vektory A, B a C. Nový jedinec D vznikne podle rovnice 13.

$$D = A + cross\_rate * (B - C) \tag{13}$$

[2]Všechna náhodná čísla jsou vždy vybírána z rovnoměrného rozdělení.

Parametr `cross_rate` má na konvergenci algoritmu ze všech největší vliv. Jeho velikost je volena v rozmezí $0,1$ až $0,5$. Čím větší je jeho hodnota, tím pomaleji algoritmus konverguje. V žádném případě ovšem všechny parametry, které mají na konvergenci algoritmu vliv, neovlivňují její průběh stejným způsobem a tudíž nejsou zaměnitelné.

EVALUATE_GENERATION

Funkce `EVALUATE_GENERATION` ohodnotí všechny nové jedince. Je nutné vždy definovat, jak ohodnocovat jedince (vektory), jejichž některá souřadnice překročila své předepsané meze. Pakliže je řešená funkce mimo tyto meze definovaná, není nutné tyto jedince jakkoli penalizovat. To nám také dává možnost najít řešení, pokud si stanovenými mezemi nejsme jisti a hledané optimum se nachází mimo ně. Pokud tomu tak ale není, je třeba se pro penalizaci rozhodnout. Metoda SADE v takovém případě používá tzv. návrat na hranici, což znamená, že souřadnici, která překročila předepsanou mez, nahradí právě touto mezí (hranicí).

SELECT

V případě tohoto operátoru je nutné zmínit, že redukuje počet všech jedinců v populaci na polovinu, to jest na počet stejný jako na začátku cyklu a to na principu podobném přirozenému výběru. Konkrétně se jedná o tzv. turnajovou selekci v obráceném smyslu. Z náhodně vybraných dvou jedinců je horší vyřazen z populace. V tomto operátoru probíhá vše náhodně a zavádění jakékoli konstanty nebylo nutné.

4.2 Nastavování parametrů

V celém algoritmu SADE jsou tedy definovány 4 konstanty, jejichž nastavení má na chování algoritmu značný vliv. V případě testování algoritmu na růst výpočtové náročnosti s růstem proměnných řešeného problému byly nastaveny takto:

```
pool_rate = 10
radiation = 5%
local_radiation = 5%
cross_rate = 0.1
```

Při použití stejného nastavení k řešení železobetonového nosníku se nepodařilo optima dosáhnout. Metodou pokusu a omylu se po nemalém množství testovacích výpočtů podařilo najít takové nastavení, se kterým metoda SADE vyřešila tuto úlohu se stoprocentní úspěšností a dokonce i v průměru rychleji než do té doby nejúspěšnější diferenciální evoluce. Jako ideální hodnoty pro parametry algoritmu se ukázaly tyto:

```
pool_rate = 25
radiation = 5%
local_radiation = 5%
cross_rate = 0.3
```

Během vývoje algoritmu byl dále algoritmus testován na různých matematických funkcích a jako ideální se ukazovaly vždy různé hodnoty parametrů. Samozřejmě bylo možné vypozorovat některé zákonitosti mezi "typem" řešené funkce a konkrétním nastavením algoritmu, ale nepodařilo se formulovat žádné definitivní pravidlo.

Značným usnadněním práce s nastavením parametrů je grafický výstup během výpočtu. Jedná se o vykreslování jedinců ve čtverci, kde jejich vodorovná a svislá poloha odpovídá relativním hodnotám dvou jejich souřadnic ve vztahu k odpovídajícím rozmezím pro tyto souřadnice.

Díky tomuto zobrazení je snadné určit, zda algoritmus nenachází řešení, protože nekonverguje, nebo nachází různé lokální extrémy, protože konverguje příliš rychle. Do jisté míry je i možné pozorovat projevy operátoru MUTATE, který generuje jedince se značným rozptylem, zatímco operátor křížení vytváří nové jedince v rámci jakéhosi shluku, který v průběhu řešení vzniká.

4.3 GATI

Cílem této práce bylo nalezení metody, pomocí které by pro každou optimalizovanou funkci bylo možné nastavit optimální parametry algoritmu SADE tak, aby jeho výpočet byl spolehlivý a pokud možno také rychlý. Dostáváme se tak před další optimalizační úlohu. Ve chvíli, kdy máme danou optimalizační úlohu pro algoritmus, objevuje se nová úloha a to hledání optimálního nastavení jeho parametrů.

Jako jedno z možných řešení se samozřejmě nabízí opět optimalizace pomocí algoritmu SADE, tentokrát v jakési druhé vrstvě. Díky prostředkům které nám nabízí objektový programovací jazyk C++ se podařilo tento dvouvrstevný algoritmus vytvořit. Datový objekt v programu, který tuto dvouvrstevnost zajišťuje je nazván GATI[3].

Jeho efektivitu byla nejdříve testována na jednoduché matematické funkci: Čebyševův polynomický problém, což je úloha, na kterou byla kdysi naprogramována diferenciální evoluce. S pomocí GATI bylo nalezeno takové nastavení parametrů SADE, že algoritmus SADE poté řešil polynomický problém nejen se stoprocentní úspěšností, ale dokonce i o něco rychleji než diferenciální evoluce.

Je však nutné si uvědomit následující. Označíme n počet vyhodnocení optimalizované funkce potřebných k tomu, aby algoritmus SADE nalezl její optimum. Stejně tak je nutné m-krát zavolat výpočet algoritmu SADE, než je pomocí jeho samého nalezeno jeho optimální nastavení. Než tedy tyto optimální parametry nalezneme stoupne počet vyhodnocení řešené optimalizační úlohy na $n.m$. Vzhledem k jednoduchosti Čebyševova polynomického problému to nečinilo nijak velký problém. V případě optimalizace funkce nejmenších čtverců, jak tomu je v případě hledání parametrů retenční čáry nebo Lemaitrova modelu, trvá výpočet podstatně déle. Jedná se totiž o vyhodnocování funkce nejmenších čtverců ze souboru dat řádově o velikosti kolem 10000 bodů. V takovém případě se pak metoda GATI nedala z časových důvodu použít.

4.4 CERAF

Jiný pohled na problém nastavování parametrů získáme v prostředí inženýrské praxe. Často je totiž důležitější najít rychle dobré řešení optimalizačního problému i za cenu, že toto řešení nebude absolutním globálním optimem. Takový přístup odsouvá problém nastavování parametrů algoritmu do pozadí, neboť např. s nastavením:

```
pool_rate = 10
radiation = 10%
local_radiation = 10%
cross_rate = 0.3
```

je možné téměř vždy k nějakému dobrému řešení dospět. Často se však bude jednat o lokální extrém. Pokud nám jen tento jediný výsledek nebude stačit, je možné výpočet spustit několikrát a sledovat, zda algoritmus dospívá stále ke stejnému řešení, nebo jestli je nerozhodný a vlivem náhody nabízí pokaždé jiné řešení. Je ale také možné, že nabízí stále stejné řešení, ale i přesto je toto řešení lokálním extrémem. Zkrátka, několikanásobným spuštěním se o funkci něco dozvíme, ale jakékoli závěry budou stále velice nespolehlivé. Bylo by tedy dobré, kdyby se po nalezení nějakého extrému algoritmus sám snažil hledat pokud možno řešení jiné a nevracel se stále

[3]Genetic Algorithm Tunning Itself

ke stejnému. Výsledně by pak nabídl několik řešení, přičemž výběr nejlepšího z nich by už závisel na nás.

Realizací této myšlenky je zavedení technologie CERAF[4], neboli metody radioaktivních center, do algoritmu SADE. Její princip je následující: předpokládá se, že algoritmus najde nějaké řešení a pokud to bude nějaký extrém, nebude se po několik následujících generací jeho hodnota zlepšovat. Nejlepší nalezená hodnota se tedy stabilizuje. V tuto chvíli bylo nutné zavést konstantu, podle které se algoritmus rozhodne, že byl nalezen extrém. Konstanta `quiet_generation` definuje maximální počet generací, během nichž funkční hodnota nejlepšího řešení nestoupne o víc než je rozlišovací přesnost algoritmu pro funkční hodnotu řešeného problému. Tuto přesnost je nutné definovat pro každou funkci; přitom postačí její hodnotu jen přibližně odhadnout. Měla by však odrážet přesnost, s jakou chceme řešení nalézt. Algoritmus označí extrém ve chvíli, kdy takto definovaný počet generací překročí počet definovaný konstantou `quiet_generation`.

Po označení extrému následuje zánik současné populace a vytvoření nové počáteční populace, která je opět zcela náhodná. V tuto chvíli je nutné nějak zařídit, aby se algoritmus vyhýbal již nalezenému extrému a snažil se najít jiný, resp. aby jedince v blízkosti nalezeného řešení nějak penalizoval. Zde vyvstalo hned několik obtížných otázek.

1. Jak definovat blízké okolí nalezeného řešení.

2. Jak jedince v tomto okolí penalizovat.

O tom, jaká byla zvolena penalizace, napovídá sám název metody. Jedná se o představu nalezeného extrému jako jakéhosi radioaktivního centra. Jedinec, který se dostane do jeho blízkosti podléhá mutaci, resp. je nahrazen zcela nově vytvořeným náhodným jedincem.

Mnohem horší byla otázka definice blízkého okolí. Ukázalo se, že pevné stanovení nějaké hranice často vede k vytvoření nových "virtuálních" extrémů. Pevná hranice kolem nalezeného extrému může také pojmout i zcela jiný extrém a je velice těžké rozhodnout, jak velikou oblast by tedy měla zahrnovat. Daleko rozumnější se ukázalo stanovit oblast zpočátku dostatečně velikou a v průběhu dalšího výpočtu ji případně zmenšovat. Tato její neutralizace by měla být závislá na počtu jedinců, kteří se do této oblasti dostávají. Je zde však nutná další konstanta, která bude definovat konkrétní míru neutralizace. Tato konstanta byla označena `deact_rate`.

Výsledně má radioaktivní oblast tvar elipsoidu, jehož průměr ve směru každé proměnné je roven polovině definičního oboru dané proměnné. V případě pádu nějakého jedince do této oblasti se její průměr sníží na `deact_rate`-násobek. Po několika testech bylo rozhodnuto pro ještě jednu úpravu. Totiž, že jedinci, kteří vzniknou operací mutace nebo lokální mutace, by na zmenšování radioaktivní zóny neměli mít vliv. Jejich pád do zóny je totiž čistě náhodný a nijak nereprezentuje směr, kterým algoritmus konverguje. Naproti tomu, pokud do této zóny padají jedinci při operaci křížení, znamená to, že algoritmus považuje tuto oblast za zajímavější než jiné navzdory penalizaci a tím dává najevo, že extrém, který již našel, je s největší pravděpodobností skutečným optimem anebo že optimum se někde v této penalizované oblasti nachází.

Nové konstanty byly pro všechny výpočty stanoveny pevně a není nutné je pro různé funkce jakkoli upravovat. Tato metoda nevykazuje rozdílné chování u různých řešených funkcí. Konstantu `quiet_generation` se mi zdálo vhodné uvést do závislosti na dimenzi řešeného problému a zvolené velikosti populace. Velikost populace je ale také na dimenzi problému závislá. Výsledně se tedy hodnota `quiet_generation` vždy na začátku cyklu vypočítá podle vztahu:

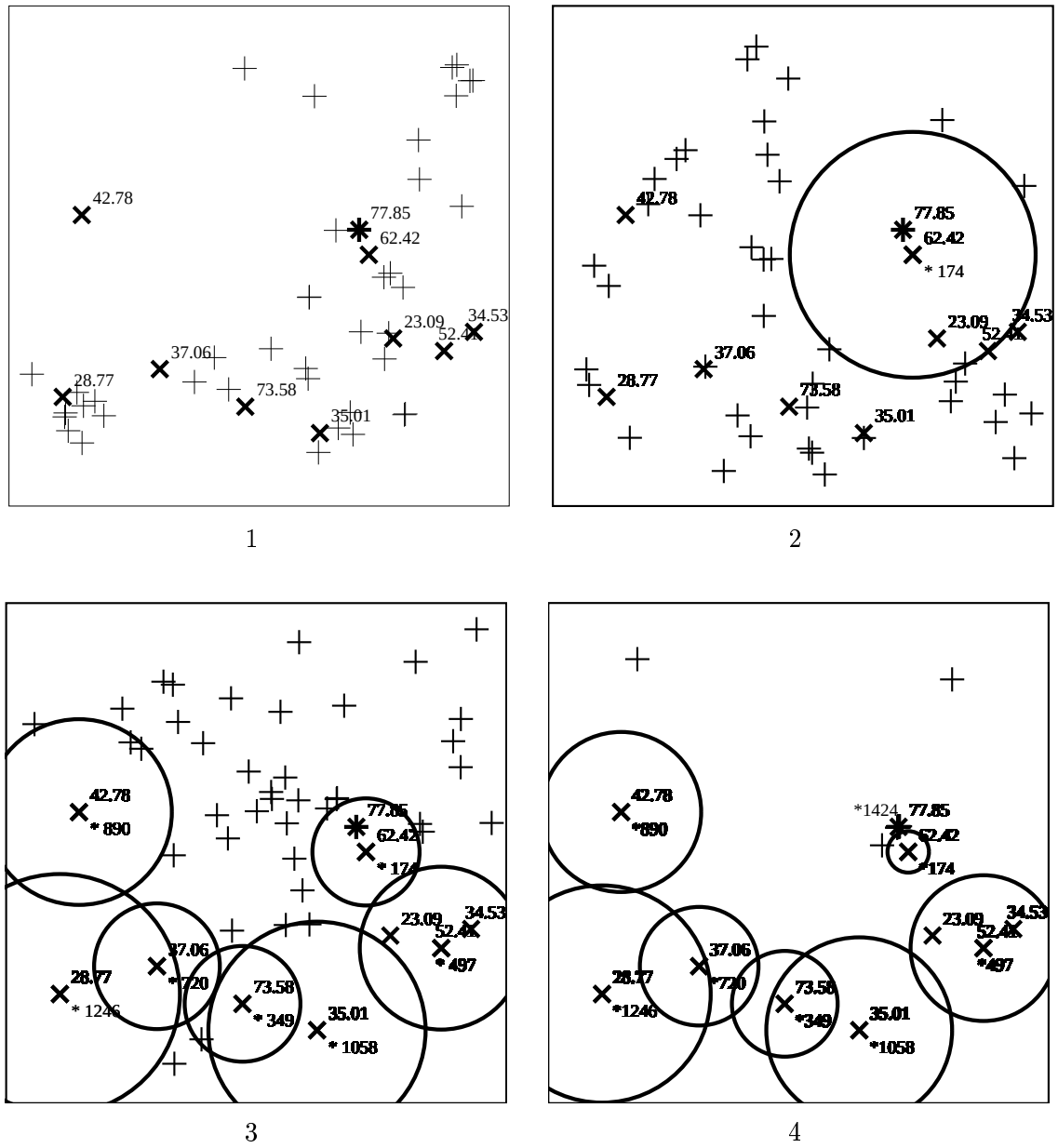
$$quiet\_generation = \frac{quiet}{pool\_rate}.$$

Hodnoty výsledných dvou parametrů byly stanoveny takto:

[4]CentrE Radio-ActiF

```
deact_rate = 0.995
quiet = 3400
```

Přesto je nutné v některých případech konstantu quiet změnit a to v souvislosti s přesností, s jakou hledáme řešení. Problém nastává, pokud je požadována vysoká přesnost řešení a hodnota parametru `quiet` je nízká. Dejme tomu, že algoritmus konverguje ke správnému řešení, ale konvergence je pomalá. Metoda CERAF pak může označit extrém předčasně a optimum nebude nalezeno. Tento problém je ale snadno odhalitelný tím, že výpočet jednou spustíme bez metody CERAF. Ukáže se, že pak algoritmus najde přibližně stejné řešení, ale s vyšší přesností. Nevýhodou nastavení vysoké hodnoty parametru `quiet` je pouze prodloužení doby výpočtu.



Obrázek 2: Výpočet s využitím metody CERAF

Na obrázku 2 je zachycen průběh výpočtu s využitím metody CERAF. Pro vystižení vlivu této metody na chování algoritmu a průběh výpočtu byla k optimalizaci zvolena transparentní matematická funkce dvou proměnných, která má deset předem známých lokálních extrémů. Její předpis je dán výrazem:

$$f(x) = \sum_{i=1}^{10} y_i \left(\frac{\pi}{2} + \arctan \|\mathbf{x} - \mathbf{x_i}\| \right).$$

Souřadnice lokálních extrémů, které představují vektory x_i, jsou stejně jako další konstanty y_i zvoleny náhodně z těchto intervalů:

$$\mathbf{x_i} \in \langle -100,0 \div 100,0 \rangle \times \langle -100,0 \div 100,0 \rangle,$$

$$y_i \in \langle 10,0 \div 50,0 \rangle.$$

Ve všech čtyřech oknech obrázku jsou všechny lokální extrémy funkce označeny symboly $\times$, u kterých je vždy vpravo nahoře uvedena funkční hodnota extrému. Globální extrém je pak označen překrývájícími se symboly $\times$ a $+$.

V levém horním okně obrázku je zachyceno rozptýlení počáteční generace algoritmu. Symboly $+$ jsou vyznačeny jedinci v zobrazené generaci. V pravém horním okně je zachycena generace jedinců právě poté, co byl nalezen a označen první lokální extrém. U tohoto extrému je vpravo dole uveden sybmol * a vedle něho číslo generace, ve které byl extrém označen. Radioaktivní zóna je vyznačena kruhem se středem v tomto extrému. Rozptýlení jedinců v generaci, kdy je označeno už sedm lokálních extrémů s velikostí radioaktivních zón jednotlivých extrémů je zobrazeno v levém dolním okně. V posledním okně je pak zachycen stav radioaktivních zón po nalezení globálního extrému.

Toto konkrétní rozmístění lokálních extrémů bylo zvoleno úmyslně, neboť tu nastal případ, kdy se globální extrém nachází vně ostatních lokálních extrémů a zároveň je v jeho blízkosti jiný extrém. V podobných situacích, pro algoritmus nejvýce nepříznivých, metoda SADE, stejně jako jiné genetické či evoluční algoritmy, často konverguje k lokálnímu extrému v blízkosti globálního. S využitím metody CERAF je tento extrém označen a v jeho okolí je vytvořena radioaktivní zóna. Ta pak ovšem pokrývá i hledaný globální extrém. Algoritmus proto nejprve nachází ostatní lokální extrémy vně radioaktivních zón. Ve chvíli, kdy radioaktivní zóna pokrývá více než polovinu definičního oboru algoritmus začne nakonec konvergovat směrem ke globálnímu extrému, který posléze nalezne.

Tímto příkladem mělo být prezentováno chování algoritmu při řešení velice obtížné funkce. Algoritmu se podařilo objevit osm extrémů, přičemž do každého z nich konvergoval právě jednou a proto nám po relativně krátké době výpočtu nabídl značné množství informací o řešené funkci. Je tedy velice efektivní. Zároveň je velice důležité, že vznik radioaktivního centra v blízkosti globálního extrému algoritmu nezabrání, aby globální extrém nalezl. Pro genetický algoritmus už proto není „pád“ do lokálního extrému osudný.

V případě řešení obou inženýrských úloh bylo metody CERAF použito především k získání větší spolehlivosti výpočtů.

5 Optimalizace paramentrů retenční čáry

Jak už bylo zmíněno v úvodu, v současné době se pro analytické vyjádření průběhu retenční čáry používá nejčastěji model podle van Genuchtena, který definuje vztah 1. Jeho velkou výhodou je určitý fyzikální význam použitých konstant α,m a n. Má-li však křivka retenční čáry optimálním způsobem prokládat naměřené hodnoty a zároveň má-li být možné hodnoty těchto konstant rozumně fyzikálně interpretovat, bývá s využitím známých programů RETC nebo UFRETC nalezení jejich hodnot problematické. Cílem této práce bylo zjistit, zda by bylo možné nepříliš úspěšné programy RETC či UFRETC nahradit genetickým algoritmem, konkrétně algoritmem SADE.

Již na první pohled má SADE jednu podstatnou výhodu. Programy RETC a UFRETC vracejí v některých případech hodnoty, které o několik řádů přesahují meze stanovené pro hledané parametry samotným van Genuchtenem a tak není možné je fyzikálně interpretovat. Algoritmus SADE lze však snadno nastavit tak, aby hledal řešení jen v zadaných mezích. Toho se docílí tím, že v průběhu výpočtu jsou jednotlivá navržená řešení mimo daný definiční obor penalizována.

Ke konkrétním výpočtům s využitím algoritmu SADE byla použita data ze 30 měření. Ve všech případech byla tato měření prováděna na zeminách z rekultivovaných výsypek na Mostecku. Meze definičního oboru pro jednotlivé parametry byly nastaveny podle van Genuchtena, tzn.

$$\alpha \in \langle 10^{-3} \div 10^{-2} \rangle,$$
$$m \in \langle 0 \div 1 \rangle,$$
$$n \in \langle 1.5 \div 6 \rangle.$$

Optimalizovanou funkcí byla funkce součtu nejmenších čtverců rozdílu naměřených hodnot a hodnot modelu. Protože optimalizovaná funkce přímo závisí na naměřených hodnotách, má pro každý vzorek zcela jiný průběh. V takovém případě je celkem zbytečné hledat optimální nastavení algoritmu, protože by se pro jednotlivé vzorky také výrazně lišilo. Parametry algoritmu byly nastaveny pro všechny výpočty stejně na tyto hodnoty:

```
pool_rate = 10
radiation = 10%
local_radiation = 10%
cross_rate = 0.3
```

Po jednom spuštění není teoreticky jisté, zda bylo nalezeno optimální řešení. Aby se spolehlivost jednotlivých výpočtů zvýšila, byl použit algoritmus SADE rozšířený o metodu CERAF. V takovém případě už nalezený výsledek nabývá jisté důvěryhodnosti, ale v žádné případě není možné posoudit dobu výpočtu, což je vedle spolehlivosti dané optimalizační metody její další důležitá charakteristika. Vzhledem k tomu, že se jedná o stochastickou metodu, potřebuje SADE pokaždé různý počet vyhodnocení řešené funkce. Z těchto důvodů byl pro každý vzorek spuštěn výpočet stokrát. Z výsledků pak bylo hodnoceno, zda algoritmus vždy dospěl ke stejné optimální hodnotě a kolik k tomu potřeboval vyhodnocení řešené funkce.

Jelikož algoritmus nacházel opakovaně stejné řešení s přesností 10^{-10}, dá se předpokládat, že se jedná o skutečné optimum a algoritmus je možné považovat za spolehlivý. V tabulce 1 jsou pro každý vzorek, reprezentovaný identifikačním číslem, uvedeny nalezené optimální parametry, výsledná chyba nejmenších čtverců a dále průměrný počet vyhodnocení řešené funkce potřebný k nalezení optima.

Pokud se pozorněji podíváme na hodnoty jednotlivých parametrů, zjistíme, že pro parametr α a n je často optimální hodnotou jedna z mezí definovaná pro daný parametr. Můžeme se proto domnívat, že pokud by byly stanovené meze širší, bylo by jiné i nalezené optimum, resp. že skutečné optimální řešení leží vně těchto mezí. Aby se tato hypotéza potvrdila, byl spuštěn

výpočet znovu pro širší meze. Protože se nejčastěji vyskytla mezní hodnota jako optimální pro parametr α byl nový výpočet spuštěn s upravenými mezemi jen pro tento parametr a to pro $\alpha \in \langle 10^{-5} \div 10^{0} \rangle$.

ID	α	**n**	**m**	**SNČ**	**poč.vyhodn.**
1004	0.001	1.5	0.092649	0.0044052533	3054.6
101	0.001	1.5	0.077659	0.0024399199	3177.0
109	0.01	1.6585	0.044584	0.0043729931	11149.2
135	0.01	3.7048	0.012254	0.0020460438	68646.6
1376	0.002853	6.0	0.0098269	0.0004754130	664008.0
1430	0.01	1.7983	0.05395	0.0041461212	16632.3
163	0.001	1.5	0.053664	0.0019486609	4177.2
1673	0.001	1.5	0.031901	0.0002420876	7046.7
2	0.01	6.0	0.011653	0.0032027058	13677.6
201	0.001	1.5	0.041208	0.0055732120	3670.5
212	0.01	2.2672	0.024854	0.0041564802	17256.0
23	0.01	2.7203	0.018655	0.0039690931	25704.3
230	0.001	1.5	0.034343	0.0004522415	4296.0
234	0.001	1.6699	0.041295	0.0006061593	27905.7
237	0.01	2.3144	0.025527	0.0035966123	12668.1
24	0.002379	1.5	0.044042	0.0006910113	19759.5
240	0.0087062	1.5	0.048041	0.0030163443	21141.6
3	0.01	2.4833	0.030026	0.0036799656	19036.5
318	0.0079274	1.5	0.088757	0.0086601021	6316.8
377	0.01	1.5	0.038455	0.0019689478	12617.1
402	0.01	1.6342	0.055964	0.0041880017	11549.4
447	0.01	3.9195	0.013573	0.0044487432	174926.1
469	0.0044861	1.5	0.08843	0.0061487183	13586.7
51	0.0045798	1.5	0.033281	0.0017438710	11768.7
57	0.001	1.7057	0.046368	0.0025499753	15504.0
61	0.001	1.5	0.063584	0.0013697709	4036.8
64	0.0083679	1.5	0.057667	0.0033036833	11622.3
87	0.01	6.0	0.011648	0.0046442482	11538.6
91	0.01	1.5909	0.070959	0.0061854598	10885.5
94	0.0055516	1.5	0.052618	0.0028140504	11580.0

Tabulka 1: Hodnoty parametrů retenční čáry nalezené algoritmem SADE v mezích podle van Genuchtena

Při všech výpočtech se také ukázalo, že ze sady řešení, které algoritmus nacházel během jednoho výpočtu díky metodě CERAF, bylo výsledné optimum vždy nalezeno jako první. To znamená, že všechny funkce mají možná nějaké lokální extrémy, ale metoda SADE sama o sobě je schopná je překonávat a nacházet sama globální optimum. Jakékoli používání metody CERAF je tedy při dalších výpočtech zbytečné. Zároveň se tak vyhneme problému se správným nastavením parametru `quiet`, neboť požadovaná přestnost těchto výpočtů je relativně vysoká a v některých případech by pak mohlo docházet k předčasnému označení extrému a tak nenalezení optimální hodnoty.

Ostatní nastavení jak řešené funkce, tak algoritmu zůstalo stejné. Výpočet byl opět spuštěn

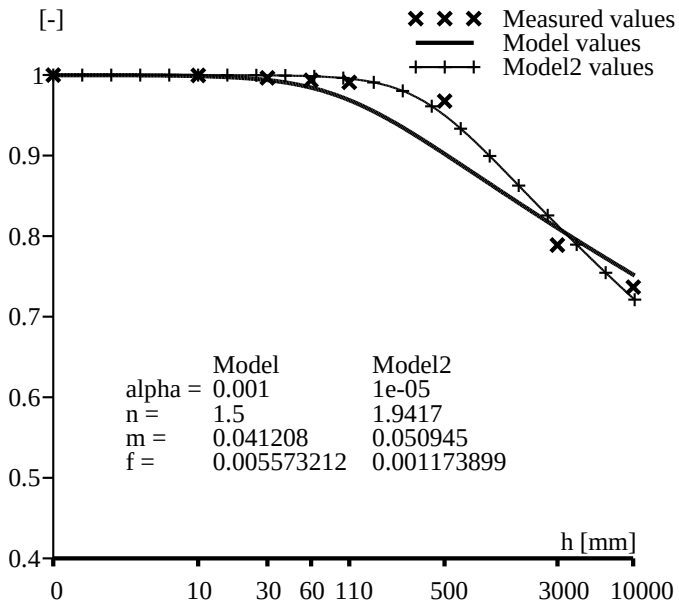
stokrát pro každý vzorek. V tabulce 2 jsou stejně jako v předchozím případě pro všechny vzorky označené identifikačním číslem uvedeny optimální hodnoty parametrů α, m, n, výsledná hodnota funkce nejmenších čtverců a také průměrný počet vyhodnocení funkce.

ID	α	**n**	**m**	**SNČ**	**poč.vyhodn.**
1004	1e-05	2.5833	0.052225	0.0027405636	12516.0
101	0.00073163	1.5	0.082894	0.0022271270	14148.5
109	0.017756	1.5	0.048182	0.0040385908	9497.9
135	0.11971	1.5	0.031269	0.0017790049	49594.3
1376	0.0028528	6.0	0.009827	0.0004754130	204166.7
1430	0.02079	1.5	0.064815	0.0038982623	10786.8
163	0.00049086	1.5	0.062194	0.0014140334	12990.9
1673	0.00069633	1.5	0.034244	0.0001860774	23155.4
2	0.60014	1.5	0.043006	0.0022976250	273177.2
201	1e-05	1.9417	0.050945	0.0011738990	10788.4
212	0.041634	1.5	0.038218	0.0036025217	16327.0
23	0.060091	1.5	0.035097	0.0033869796	21891.5
230	1e-05	2.4617	0.021575	0.0001939481	28637.6
234	7.8167e-05	2.3497	0.028624	0.0005320695	31439.2
237	0.043955	1.5	0.040115	0.0029973019	16303.8
24	0.002379	1.5	0.044042	0.0006910113	21131.2
240	0.0087062	1.5	0.048041	0.0030163443	12861.2
3	0.050868	1.5	0.051047	0.0030077200	18950.8
318	0.0079274	1.5	0.088757	0.0086601021	7173.2
377	0.012596	1.5	0.037064	0.0019163657	8575.3
402	0.016716	1.5	0.059531	0.0037842206	7709.7
447	0.14598	1.5	0.036163	0.0043445990	83219.1
469	0.0044861	1.5	0.08843	0.0061487183	14454.9
51	0.0045798	1.5	0.033281	0.0017438710	13163.9
57	1e-05	2.909	0.026722	0.0021989645	27893.3
61	0.00085296	1.5	0.065656	0.0013307301	14823.5
64	0.0083679	1.5	0.057667	0.0033036833	7331.7
87	0.96166	1.5	0.040706	0.0026556786	125622.2
91	0.015023	1.5	0.073308	0.0057922979	6684.6
94	0.0055516	1.5	0.052618	0.0028140504	12006.1

Tabulka 2: Hodnoty parametrů retenční čáry nalezené algoritmem SADE v rozšířených mezích

Při srovnání hodnot v tabulkách 1 a 2 zjistíme, že pro většinu vzorků bylo v rozšířených mezích skutečně nalezeno kvalitnější řešení. Největší rozdíl je pro vzorek číslo 201, kde je hodnota funkce součtu nejmenších čtverců řešení v rozšířených mezích pětkrát nižší, než pro řešení v mezích podle van Genuchtena. Aby bylo možné lépe posoudit význam takového rozdílu, jsou v obrázku 3 vykreslena data z měření a průběh retenčních čar pro obě nalezená řešení. Retenční čára s parametry v mezích podle van Genuchtena je označena jako model, retenční čára s parametry z rozšířených mezí je označena jako model2.

Z obrázku je zřejmé, že řešení nalezené v rozšířených mezích podstatně lépe vystihuje průběh naměřených dat, je ale otázkou, zda mají takové hodnoty parametrů retenční čáry fyzikální opodstatnění. Volba správných mezích už ale není problém optimalizace samotné. Předložené



Obrázek 3: Průběh retenčních čar, jejichž paramtry byly hledané v různých mezích

výpočty měly pouze ukázat, že algoritmus je schopen se změnám mezí přízpůsobit a být dobrým nástrojem k nalezení optimálních hodnot. Jediné, co se v chování algoritmu mění zároveň se změnou mezí, je délka výpočtu, která s šířkou definičního oboru stoupá, ale tento nárůst není nijak razantní. Pro představu o časové náročnosti jednoho výpočtu parametrů retenční čáry - výpočet s miliónem vyhodnocení funkce nejmenších čtverců ze souboru dat o osmi bodech trvá přibližně jednu minutu na počítači s procesorem Xeon 550MHz a 1GB operační paměti.

6 Optimalizace parametrů vazkoplastického modelu hornin

6.1 Optimalizace parametrů Lemaitrova modelu

Stejně jako v předchozím případě retenční čáry se i zde jedná o nalezení takových parametrů určitého modelu tak, aby jeho průběh co nejlépe prokládal naměřená data, kterými jsou v předkládaném příkladě hodnoty poměrné vazko-plastické deformace horniny anhydritu v závislosti na čase. Lemaitreův model, který se v tomto případě k modelování měřených dat používá a byl použit i při řešení pomocí algoritmu SADE je definován vztahem 4. Parametry modelu se pro materály s krystalickou mřížkou za teploty 20^{o} pohybují v přibližných mezích [7]:

$$A \in \langle 10^{-30} \div 10^{-15} \rangle,$$

$$n \in \langle 1 \div 20 \rangle,$$

$$m \in \langle -5 \div 0 \rangle.$$

Hodnoty napětí σ jsou známy ze zkoušek dotvarování a pohybují se v intervalu $\sigma \in \langle 1 \div 65 \rangle$ MPa. Už v úvodu bylo ovšem zmíněno, že parametry tohoto modelu jsou závislé. Proto byl

model přepsán do rovnice 14, která obsahuje již jen dva na sobě nezávislé parametry a ze které je možné vztahy pro parametry Lemaitreova modelu snadno vyjádřit.

$$\varepsilon^{vp} = \left(\frac{a}{b}t\right)^b \tag{14}$$

$$a = A\sigma^n \tag{15}$$

$$b = \frac{1}{1-m} \tag{16}$$

Abychom mohli parametry a a b optimalizovat algoritmem SADE bylo nutné odhadnout intervaly jejich hodnot. Ty je možné vypočítat dosazením krajních hodnot pro Lemaitreovy parametry do vztahů 15 a 16. Tím jsme dostali následující intervaly:

$$a \in \langle 10^{-30} \div 10^{-1} \rangle,$$

$$b \in \langle 0,1 \div 1 \rangle.$$

Vzhledem k velkému řádovému rozdílu mezí pro parametr a je nutné si uvědomit, že algoritmus SADE vybírá náhodné hodnoty pro optimalizované parametry z rovnoměrného rozdělení z mezí pro daný parametr. V tomto případě je však žádoucí, aby byl z rovnoměrného rozdělení vybírán exponent, jehož umocněním čísla 10 získáme hodnotu a. Proto byl zaveden další parametr c vyjadřující hodnotu právě tohoto exponentu podle vztahu $a = 10^c$. Parametr c je pak z intervalu:

$$c \in \langle -30 \div -1 \rangle$$

a výsledný optimalizovaný model popisuje rovnice 17.

$$\varepsilon^{vp} = \left(\frac{10^c}{b}t\right)^b \tag{17}$$

Abychom pro Lemaitreovy parametry získali konkrétní hodnoty, bylo nutné převést nalezené parametry na původní. Hodnoty parametrů pak získáme postupným dosazováním vypočtených hodnot do vztahů 18, 9 a 19.

$$m = \frac{b-1}{b} \tag{18}$$

$$A = \frac{10^c}{\sigma^n} \tag{19}$$

Optimalizovanou funcí byla funkce součtu čtverců rozdílu naměřených hodnot a hodnot modelu, která je dána předpisem 20,

$$f = \sum_{i=1}^{p} \left[\left(\frac{10^c}{b}t_i\right)^b - \varepsilon_i^{vp} \right]^2 \tag{20}$$

kde p je celkový počet experimentálních dat (t_i, ϵ_i^{vp}).

Ke konkrétním výpočtům byla použita data z měření vazko-plastické deformace horniny anhydritu při zkoušce dotvarování v jednoosé napjatosti.

Algoritmus SADE byl opět použit se standartním nastavením parametrů:

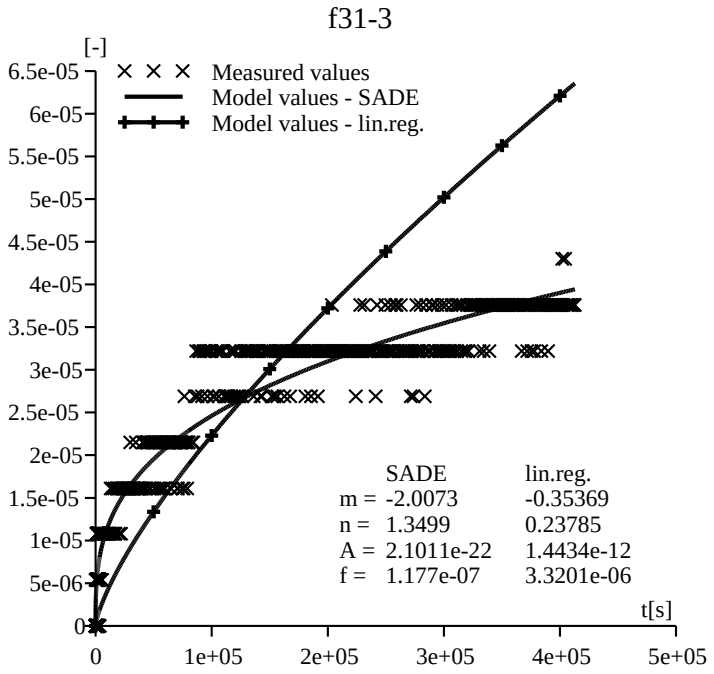
```
pool_rate = 10
radiation = 10%
local_radiation = 10%
cross_rate = 0.3
```

vzorek	σ	SNČ	m	n	A
f31-1	38.49	1.4982e-7	-6.4728	4.3260	5.7059e-46
f31-2	49.19	1.2336e-7	-5.3390	3.5905	4.8904e-40
f31-3	54.00	1.1770e-7	-2.0073	1.3499	2.1011e-22
f31-4	58.96	6.8516e-8	-1.6006	1.0764	1.8813e-19
f31-5	61.44	8.3854e-8	-1.2899	0.86745	3.7984e-18

Tabulka 3: Hodnoty parametrů Lemaitrova modelu nalezené algoritmem SADE

vzorek	σ	SNČ	m	n	A
f31-1	38.49	2.6717e-7	-4.2815	2.8793	4.5646e-34
f31-2	49.19	3.1674e-7	-4.4268	2.9770	5.6276e-35
f31-3	54.00	3.3201e-6	-0.3537	0.2379	1.4434e-22
f31-4	58.96	4.4970e-7	-0.0692	-0.0465	1.7610e-9
f31-5	61.44	2.0863e-6	-0.2148	-0.1445	4.0806e-9

Tabulka 4: Hodnoty parametrů Lemaitreova modelu nalezené lineární regresí



Obrázek 4: Srovnání průběhu Lemaitrova modelu s parametry získanými lineární regresí a algoritmem SADE

Protože k jednomu vzorku bylo vždy k dispozici řádově 10000 naměřených dat, trval výpočet s 20000 ohodnoceními cílové funkce nejmenších čtverců vždy několik minut a proto byl v tomto případě spuštěn výpočet pro každý vzorek jen jednou, nicméně s využitím metody CERAF. Jednotlivé extrémy byly dohledávány s přesností 10^{-19}.

V tabulce 3 je pro každý vzorek uvedena hladina napětí σ při zkoušce dotvarování, hodnota cílové funkce součtu nejmenších čtverů (SNČ) a nalezené hodnoty parametrů m, n a A. V tabulce 4 jsou stejným způsobem uvedeny výsledky výpočtu lineární regresí popsanou v kapitole 3.1., přičemž hodnoty vazko-plasticé deformace menší nebo rovny nule byly pro výpočet lineární regresí nahrazeny ve vstupním souboru hodnotou 10^{-9} z důvodů uvedených na straně 6.

Srovnáním obou tabulek zjistíme, že chyba nejmenších čtverců je pro genetický algoritmus vždy menší než pro linární regresi. To je možné vysvětlit důvody uvedenými v oddíle 3.1. Pro vzorek f31-3 je mezi výsledky obou metod největší rozdíl. Abychom měli lepší představu o významu takového rozdílu, je průběh obou navržených řešení vykreslen v obrázku 4.

Z obrázku 4 je zřejmé, že v tomto případě se průběh modelu s parametry získanými lineární regresí díky úpravě dat značně liší od průběhu měřených dat, zatímco model s parametry získanými algoritmem SADE měřená data dobře vystihuje.

Je ale nutné zmínit následující: hustota naměřených dat není šude stejná. Na začátku měření, kdy je nárůst deformace největší, byla data aznamenávána po kratších časových intervalech než v momentě, kdy se nárůst ustálil. uvedených příkladech je podíl hustoty dat ze začátku měření a z jeho konce přibližně 60. Tím dochází k přílišnému ovlivnění růběhu prokládané křivky velkou hustotou dat na začátku měření. Aby data ze začátku i konce měření měla stejnou váhu, je třeba, aby měla stejnou hustotu. Proto byla následně data na počátku měření vytříděna, aby bylo dosaženo rovnoměrného rozložení hustoty dat po celém vstupním souboru. S takto upravenými daty byly znovu spočítány hodnoty arametrů modelu jak lineární regresí, tak genetickým algoritmem SADE. Výsledky těchto výpočtů jsou zaznamenány v tabulkách 5 a 6.

vzorek	σ	**SNČ**	**m**	**n**	**A**
f31-1	38.49	7.6489e-8	-7.5336	5.0663	5.0442e-52
f31-2	49.19	5.9722e-8	-6.5024	4.3729	2.0882e-46
f31-3	54.00	6.9884e-8	-2.6341	1.7715	5.0528e-26
f31-4	58.96	2.1127e-8	-2.2490	1.5125	3.3952e-23
f31-5	61.44	4.8488e-8	-1.5345	1.0319	1.4189e-19

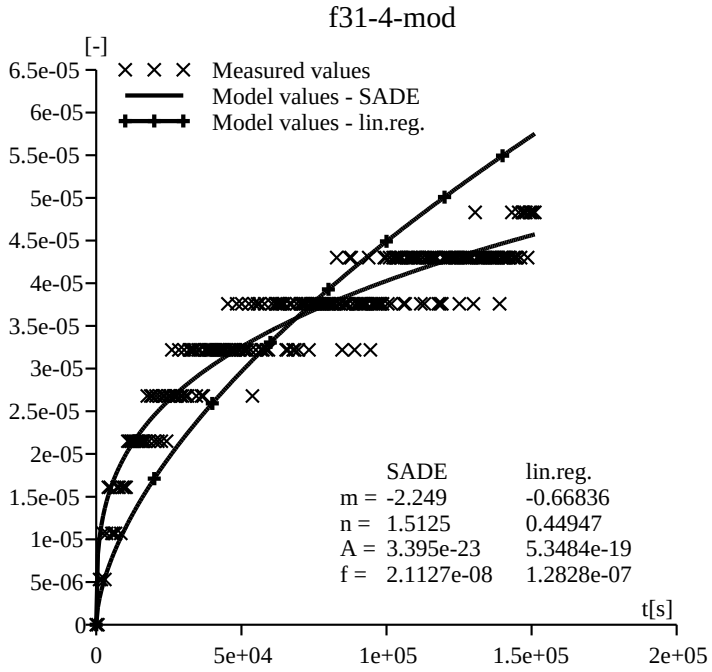
Tabulka 5: Hodnoty parametrů Lemaitrova modelu nalezené algoritmem SADE na tříděných datech

vzorek	σ	**SNČ**	**m**	**n**	**A**
f31-1	38.49	8.1084e-8	-6.3105	4.2438	2.5605e-45
f31-2	49.19	6.8180e-8	-5.3295	3.5840	5.5503e-40
f31-3	54.00	8.0485e-8	-1.8554	1.2478	1.6579e-21
f31-4	58.96	1.2828e-7	-0.6684	0.4495	5.3484e-19
f31-5	61.44	5.3366e-8	-1.2746	0.8572	4.7689e-18

Tabulka 6: Hodnoty parametrů Lemaitrova modelu získané lineární regresí na tříděných datech

Je logické, že křivky modelu navrženého lineární regresí tentokrát podstatně lépe prokládají měřená data. Je to způsobeno také tím, že tříděním byl vyloučen velký počet měření s nulovou

nebo zápornou hodnotou deformace z počátku měření a tak úpravou zbylých "nevyhovujících" dat už byla do výpočtu zanesena podstatně menší chyba. Přesto např. pro vzorek f31-4 navrhl algoritmus SADE výrazně lepší řešení než lineární regrese. Průběh křivek modelů navržených oběma metodami je možné porovnat na obrázku 5.



Obrázek 5: Srovnání průběhu Lemaitrova modelu s parametry získanými lineární regresí a algoritmem SADE

6.2 Optimalizace parametrů Lemaitreova modelu s poškozením

V případě Lemaitreova modelu s poškozením je předpis popisující průběh naměřených dat podstatně složitější, než je tomu u retenční čáry nebo jednoduchého Lemaitreova modelu bez poškození. Funkční předpis tohoto modelu popisuje rovnice 11.

Již v úvodu bylo zmíněno, že daný model má pouze 4 nezávislé parametry a proto určujeme hodnoty parametrů A a N z jiných zkoušek nebo odhadem. Dále bylo nutné definovat vztahy mezi novými parametry jednoduchého matematického modelu daného rovnicí 12 a původními parametry Lemaitreova modelu s poškozením, což je v tomto případě velice pracné. Při optimalizaci algoritmem SADE stačí předem určit hodnoty parametrů A a N a v průběhu výpočtu stejně jako v případě konstanty σ jen dosazovat odpovídající hodnotu. Dále je známá hodnota parametru matematického modelu α, neboť výraz $\frac{1}{\alpha}$ představuje čas kolapsu materiálu, resp. čas poslední naměřené hodnoty, která je vždy na začátku výpočtu známa. Po dosazení odpovídajícího výrazu z Lemaitreova modelu za parametr α, dostaneme vztah mezi parametry A, k a r, daný rovnicí 21; t_p je čas kolapsu materiálu.

$$(k+1)\left(\frac{\sigma}{A}\right)^r = \frac{1}{t_p} \tag{21}$$

Pří výpočtu pak neznámý parametr r vždy nahradíme výrazem 22, kde bude neznámý už

jen hledaný parametr k.

$$r = -\frac{ln[t_p(k+1)]}{ln\frac{\sigma}{A}} \tag{22}$$

Optimalizovanou funkcí je pak opět cílová funkce součtu čtverců rozdílu hodnot modelu a naměřených hodnot, tentokrát se třemi neznámými parametry M, K a k. Intervaly možných hodnot těchto parametrů byly s ohledem na definiční obor funkce 11 odhadnuty takto:

$$M \in \langle 1 \div 30 \rangle$$

$$K \in \langle 10^0 \div 10^{30} \rangle$$

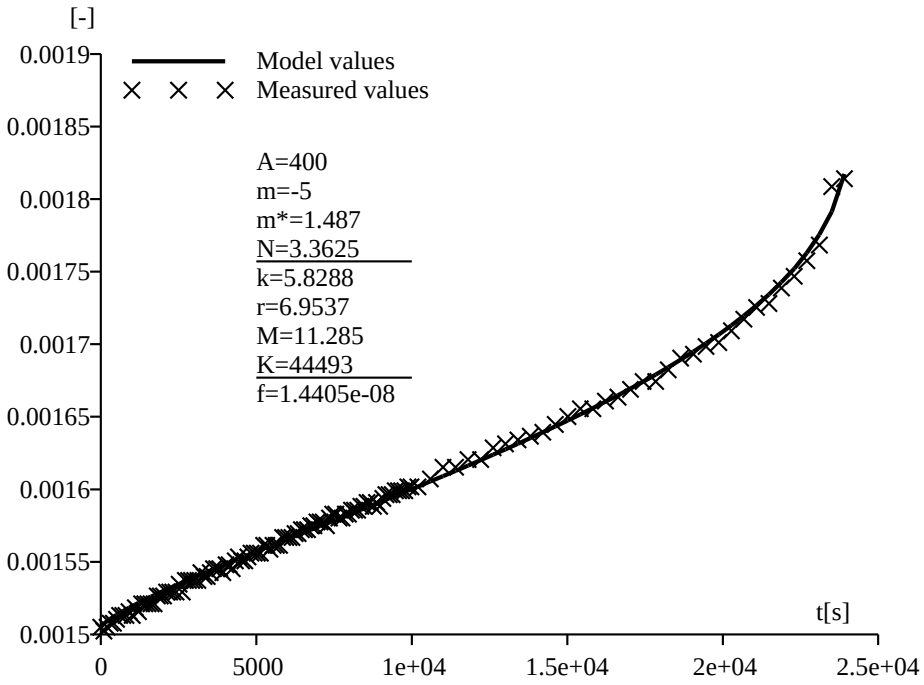
$$k \in \langle N+1 \div 30 \rangle$$

Parametr K může opět nabývat hodnot se značným řádovým rozptylem, proto byl zaveden nový parametr K'. Hodnotu parametru K pak vypočítáme podle vztahu: $K = 10^{K'}$, přičemž parametr K' nabývá hodnot z intervalu $\langle 0 \div 30 \rangle$.

Pro výpočet byla použita data ze zkoušky dotvarování anhydritu při jednoosé napjatosti $\sigma_1 = 71.21$ MPa. Parametry algoritmu byly nastaveny standartním způsebem na hodnoty:

```
pool_rate = 10
radiation = 10%
local_radiation = 10%
cross_rate = 0.3
```

Výpočet byl spuštěn s využitím metody CERAF a trval několik minut. Nalezené hodnoty parametrů Lemaitrova modelu, průběh naměřených hodnot i hodnot modelu jsou zaznamenány v obrázku 6.



Obrázek 6: Průběh Lemaitrova modelu s poškozením

7 Závěr

V této práci byly předvedeny dva inženýrské optimalizační problémy, které se tradičními postupy vycházejícími ze znalostí z oblasti matematiky či statistiky, popřípadě jiných vědních oborů, nepodařilo uspokojivě vyřešit. S rychlým vývojem výpočetní techniky se ale otevírá nová cesta řešení podobných problémů s vyžitím výpočtově náročnějších stochastických postupů. Jedním z nich je zde prezentovaný genetický algoritmus SADE.

Protože jsou genetické algoritmy poměrně mladá a rozvíjející se metoda, naráží jejich využití v praxi na některé problémy. Jedním z nich je jistá nespolehlivost v nalezení skutečného globálního extrému, zapříčiněná častým „uvíznutím“ v lokálním extrému, neboli tzv. předčasná konvergence algoritmu. V minulosti bylo navrženo několik možných řešení posledně zmíněného problému a většina z nich v různé míře předčasné konvergenci brání. Jiný možný přístup k tomuto problému představuje využití metody CERAF, která se nesnaží zpomalovat konvergenci algoritmu, ale brání mu v konvergenci do stále stejných, již nalezených lokálních extrémů, jak je podrobněji popsáno v kapitole 4.4. V důsledku toho nám algoritmus nabízí větší množství informací o řešené funkci a dává možnost uživateli posoudit kvalitu výsledného řešení.

Dalším problémem genetických algoritmů je využívání různého počtu konstant, které ovlivňují jejich chování. Algoritmus SADE např. pracuje se čtyřmi konstantami. V této práci je proto popsáno konkrétní nastavení, se kterým je možné vyřešit široké spektrum úloh. Vhodná změna parametrů může posloužit ke zrychlení výpočtu, ale není nezbytně nutná pro nalezení optima. Například v případě problému retenční čáry bylo řešeno třicet různých funkcí, přičemž všechny byly řešeny se stejným nastavením algoritmu SADE. Po různě dlouhých výpočtech byly všechny funkce vyřešeny.

V případě hledání parametrů retenční čáry byl na použitý algoritmus kladen další nárok a to nalezení takových hodnot parametrů, aby nebyly překročeny meze fyzikální interpretace. Tento nárok byl použitím algoritmu SADE snadno splněn, neboť algoritmus vychází při hledání optimálního řešení z náhodných řešení, generovaných v zadaných mezích. V mezích doporučených van Genuchtenem bylo pro všech třicet vzorků nalezeno optimální řešení s požadovanou přesností. Z výsledků je ovšem zřejmé, že nalezené řešení je přímo determinováno zadanými mezemi parametrů. Je tedy na uživateli, aby tyto meze zvolil pro řešený vzorek zeminy vhodným způsobem. Výpočty prokázaly, že algoritmus SADE je dobrým nástrojem k nalezení řešení bez ohledu na šířku zvolených mezí hledaných hodnot parametrů.

Při hledání parametrů Lemaitreova modelu bylo možné srovnat řešení nalezená algoritmem SADE a lineární regresí. Ačkoli lineární regrese pracuje s upravenými daty, byly tyto změny pro několik konkrétních vzorků natolik malé, že výsledné řešení lineární regresí a algoritmem SADE se příliš nelišila. Řešení lineární regresí pak posloužilo k ověření, že algoritmus SADE nalezl globální extrém a nebylo nutné provádět výpočet pro jednotlivé vzorky vícekrát. Z výsledků uvedených v kapitole 6.1 vyplývá, že algoritmu SADE se podařilo pro všechny vzorky najít optimální hodnoty parametrů modelu a to se značnou přesností, ačkoli se jedná o přibližnou metodu. Zároveň pro všechny vzorky řešení algoritmu SADE lépe vystihovalo naměřená data oproti řešení lineární regresí, neboť ve všech případech bylo nutné data pro lineární regresi v různé míře upravit, protože nevyhovovala definičnímu oboru nebo oboru hodnot linearizované funkce. V případě řešení Lemaitrova modelu s poškozením bylo posouzení výsledku optimalizace ponecháno na čtenáři z prezentovaných výsledků.

Z výsledků této práce vyplývá, že algoritmus SADE je pro obě uvedené úlohy dobrým nástrojem k jejich řešení a není důvod ho nevyužít v praxi.

Poděkování

Tato práce je podporována Ministerstvem školství, mláděže a tělovýchovy pod čislem projektu MŠMT 210000003.

Reference

[1] G. Yagawa and H. Okuda, *Neural networks in computational mechanics*, CIMNE, 1996

[2] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, 1989

[3] Z. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*, Springer-Verlag, 1992

[4] J. Andre and P. Siarry and T. Dognon, *An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization*, Advances in Engineeritng Software, Elsevier, 32, 49-60, 2001

[5] R. Storn, *On the usage of differential evolution for function optimization*, NAPHIS, 1996

[6] Homepage of Differential Evolution
`http://www.icsi.berkeley.edu/~storn/code.html`

[7] J. Lemaitre and J.-L. Chaboche *Mechanics of solid materials*, Cambridge University Press, USA, 2000

[8] A. Hajdu, *Modèle viscoplastique endommageable de Lemaitre et son application dans Castem 2000*, PhD thesis, 3S Laboratory, Universite Joseph Fourrier, 2001.

[9] E. Boidy, F. Pellet, M. Boulon, *Numerical modelling of deep tunnels including time-dependent behavior*, 2001, submitted.

[10] M. Kutílek and V. Kuráž and M. Císlerová, *Retenční čáry půdní vlhkosti*, Hydropedologie 10, 102-107, 2000

[11] M. Lepš and M. Šejnoha, *New approach to optimization of reinforced concrete beams*, Computional Concrete Structures Technology, CIVIL-COMP, 2000

[12] K. Matouš and M. Lepš and J. Zeman and M. Šejnoha, *Applying genetic algorithms to selected topics commonly encountered in engineering practice*, Computer Methods in Applied Mechanics and Engineering, 190, 13-14, 1629-1650, 2000

[13] J. Zeman and M. Šejnoha, *Numerical evaluation of effective elastic properties of graphite fiber tow impregnated by polymer matrix*, Journal of the Mechanics and Physics of Solids, 49, 1, 69-90, 2001

[14] O. Hrstka and A. Kučerová, *Search for optimization method on multidimensional real domains*, Contributions to Mechanics of Materials and Structures, CTU Reports, 4, 87-104, 2000

[15] Homepage of SADE
`http://klobouk.fsv.cvut.cz/~ondra/sade.html`

[16] O. Hrstka and A. Kučerová, *Improvemets of the different types of binary and real coded genetic algorithms preventing the premature convergence*, submitted to Advances in Engineering Software, 2001

[17] O. Hrstka and A. Kučerová and M. Lepš and J. Zeman, *A competitiv comparison of diferent types of evolutionary algorithms*, Proceedings of The Sixth International Conference on The Application of Artificial Intelligence to Civil and Structural Engineering, CIVIL-COMP Press, 2001