ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ
KATEDRA ELEKTROMAGNETICKÉHO POLE

DIPLOMOVÁ PRÁCE

Hybrid Nelder-Mead a rojové optimalizace

Autor práce: **Pavel Koška**
Vedoucí práce: **Ing. Miloslav Čapek**

V Praze 19.12.2013

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra mikroelektroniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. K O Š K A Pavel**

Studijní program: Komunikace, multimédia a elektronika
Obor: Elektronika

Název tématu: **Hybrid Nelder-Mead a rojové optimalizace**

Pokyny pro vypracování:

Rekapitulujte metody PSO a Nelder-Mead. Implementujte v Matlabu vícedimenzionární a jednokriteriální Nelder-Mead metodu. Zahrňte techniky, které omezí pohyb simplexu na zvolený nadprostor. Respektujte podobné rozhraní, jako má již existující rojový optimalizátor.

Dále diskutujte rozdíly v obou přístupech a možné (známé) metody jejich propojení - horizontální i vertikální (existují-li). Jmenujte silné a slabé stránky obou algoritmů a možností propojení, vč. časové náročnosti a potenciálně vhodné množiny kriteriálních funkcí.
Druhá část práce se bude věnovat propojení obou technik za účelem získání vyšší diverzity hejna, rychlejší konvergence a omezení oscilací agentů kolem globálního minima. Otestujte vybrané postupy, které využijí PSO jako řídicí optimalizaci, během které poběží i simplexní algoritmus. Zhodnoťte součinnost metod na souboru testovacích funkcí. Jako testovací funkce volte buďto kanonické optimalizační úlohy nebo reálné inženýrské problémy.

Seznam odborné literatury:

[1] Nelder, J. A., Mead, R.: A Simplex Method for Function Minimalization, The Computer Journal, Vol. 7, No. 4, pp. 308-313, 1965.
[2] Mathews, J. H., Fink, K. D.: Numerical Methods Using Matlab, Prentice Hall, 2004
[3] Engelbrecht, A. P., Fundamentals of Computational Swarm Intelligence, Wiley, 2006.
[4] Liu, A., Yang, M.-T.: A New Hybrid Nelder-Mead Particle Swarm Optimization for Coordination Optimization of Directional Overcurrent Relays, Math. Problems in Engineering, Hindawi, Vol. 2012, ID 456047, 18 pages.

Vedoucí: **Ing. Miloslav Čapek**

Platnost zadání: 31. 8. 2014

L.S.

Prof. Ing. Miroslav Husák, CSc.
vedoucí katedry

Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 7. 2. 2013

Poděkování

Děkuji vedoucímu práce panu Ing. Miloslavu Čapkovi za trpělivé a svědomité vedení práce a především za velké množství užitečných a cenných podnětů, rad a připomínek.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

Praha, 19.12.2013 Pavel Koška

Abstract

V této práci jsou nejprve rekapitulovány optimalizačních techniky Nelder-Mead, PSO a hybridní Nelder-Mead-PSO. Byla navržena úprava hybridní Nelder-Mead-PSO metody tak, aby byla zefektivněna spolupráce obou parciálních algoritmů. Upravená hybridní Nelder-Mead-PSO metoda byla implementována v prostředí Matlab v podobě programu NM-PSOptimizer. Porovnání úspěšnosti hybridního algoritmu se samostatnými Nelder-Mead a PSO metodami ukázalo, že hybridní algoritmus dosahoval lepší úspěšnosti, než samostatné metody. Hybridní algoritmus byl dále použit pro řešení technického problému, návrhu a optimalizace čerpání ramanovského zesilovače.

Klíčová slova

Nelder-Mead simpexová metoda, rojová optimalizace, ramanovský zesilovač

Abstract

In this project are at first reviewed Nelder-Mead, PSO and hybrid Nelder-Mead-PSO optimization methods. The modification of the hybrid Nelder-Mead-PSO method was proposed to improve collaboration between both separate algorithms. The modified hybrid Nelder-Mead-PSO method was implemented as program named NM-PSOptimizer in the Matlab environment. Capabilities of all three methods were compared. It was observed, that hybrid algorithm was more successful than both separate methods. The hybrid algorithm was finally used to solve technical problem, design and optimization of Raman fiber amplifier pumping.

Keywords

Nelder-Mead simplex method, particle swarm optimization, Raman amplifier

Obsah

Úvod

V technické praxi při návrhu nejrůznějších systémů často vyvstává problém najít takové parametry navrhovaného systému, aby byl výsledek v nějakém smyslu co nejlepší. Takovýto problém je v podstatě optimalizační úlohou. Z formálního hlediska je optimalizační úloha úlohou nalezení minima funkce $f : \mathcal{D} \to \mathbb{R}$, zpravidla více proměnných. Funkce f se nazývá účelová funkce, nebo cenová funkce, množina $\mathcal{D}$ je nazývána parametrickým prostorem. Vzhledem k problému hledání co nejlepších parametrů systému definuje účelová funkce, co je myšleno pod pojmem nejlepší. Argumenty funkce jsou hledané parametry systému. Parametry obvykle bývají reálná čísla, množinu $\mathcal{D}$ lze pak ztotožnit s n-rozměrným reálným prostorem $\mathcal{D} = \mathbb{R}^n$. Ne vždy je potřeba hledat minimum funkce na celém $\mathbb{R}^n$. Parametry systému mívají obvykle z fyzikálního hlediska smysl pouze v určitých mezích. V takovém případě se hovoří o omezené optimalizaci a zavádí se přípustná množina Ω. Přípustná množina bývá definována sadou omezení ve tvaru rovností a nerovností:

$$\Omega = \{\mathbf{x} \in \mathbb{R}^n | h_i(\mathbf{x}) = 0 \wedge g_j(\mathbf{x}) \leq 0\}, \tag{1}$$

kde funkce $h : \mathbb{R}^n \to \mathbb{R}$ a $g : \mathbb{R}^n \to \mathbb{R}$ jsou omezující funkce. Úloha optimalizace na omezené množině má pak tvar:

$$\min_x \{f(\mathbf{x}) | x \in \Omega\}. \tag{2}$$

Účelová funkce f může mít na dané množině globální minimum, ale zároveň také několik lokálních minim [1]. Bod $\mathbf{x}_0$ je globálním minimem funkce f na množině Ω právě když platí:

$$f(\mathbf{x}_0) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega. \tag{3}$$

Bod $\mathbf{x}_0$ je lokálním minimem funkce f na množině Ω právě když existuje okolí $U \subset \Omega$ bodu $\mathbf{x}_0$ tak, že platí:

$$f(\mathbf{x}_0) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in U. \tag{4}$$

Funkce f je unimodální na množině Ω právě když má na dané množině jedno globální minimum a žádná lokální minima.

Podle charakteru funkce f lze úlohy optimalizace rozdělit na úlohy lineárního programování, úlohy kvadratického programování a úlohy nelineárního programování.

Z hlediska tohoto projektu jsou zajímavé pouze úlohy nelineárního programování s obecnou, nelineární účelovou funkcí funkcí.

Pro řešení takových úloh existuje celá řada numerických algoritmů. Všechny algoritmy prohledávají parametrický prostor více či méně inteligentním způsobem a snaží se najít minimum účelové funkce. Algoritmy můžeme rozdělit podle několika kritérií. Může jít o algoritmy globální, které jsou schopny nalézt globální minimum na dané množině, nebo algoritmy lokální, které jsou schopny nalézt pouze lokální minimum. Dále může jít o algoritmy deterministické, algoritmy stochastické (Monte Carlo), které využívají při prohledávání prostoru náhodnosti, nebo algoritmy evoluční (Genetické, PSO). Dále lze algoritmy rozdělit podle toho, jaký řád derivace účelové funkce využívají. Buď potřebují znát pro svůj běh pouze funkční hodnoty v různých bodech (Metoda bisekce, Nelder-Mead simplexová metoda), nebo potřebují znát gradient funkce (gradientní metoda), nebo dokonce matici druhých derivací (Newtonova metoda).

Hlavním cílem práce je implementace hybridního Nelder-Mead-PSO algoritmu, který má za úkol spojit vlastnosti globálního evolučního PSO algoritmu a velmi efektivní lokální Nelder-Mead simplexové metody.

První kapitola je věnována Nelder-Mead simplexové metodě, jejím vlastnostem a aspektům její implementace. Ve druhé kapitole je rozebrána PSO metoda a její základní vlastnosti. Třetí kapitola popisuje a analyzuje hybridní Nelder-Mead-PSO algoritmus, dále jsou navrženy jeho úpravy pro dosažení lepší součinnosti obou metod. Čtvrtá kapitola popisuje program NM-PSOptimizer, který je implementací hybridní metody. Pátá kapitola představuje testovací funkce použité pro porovnání jednotlivých metod. V šesté kapitole jsou prezentovány a diskutovány výsledky porovnání optimalizačních metod. V sedmé a poslední kapitole je uveden praktický příklad použití vytvořeného programu pro optimalizaci čerpání ramanovského zesilovače.

Kapitola 1

Nelder-Mead simplexová metoda

Nelder Mead simplexová metoda se řadí do třídy algoritmů založených na přímém prohledávání stavového prostoru. Velkou výhodou algoritmu je, že nevyžaduje vyhodnocení derivací účelové funkce, jak je tomu v případě gradientních metod. Díky této vlastnosti umožňuje zpracovávat nespojité funkce a funkce, u nichž je vyčíslení derivací obtížné či neúměrně drahé. Metoda byla navržena Nelderem a Meadem roku 1965 [2].

Algoritmus metody je založen na postupných modifikacích k-simplexu, k-simplex je konvexní obal $k+1$ afinně nezávislých bodů [3]. V každém kroku je prostřednictvím jednoduchých transformací simplexu hledán bod s nižší hodnotou účelové funkce, než je současná nejlepší hodnota mezi vrcholy simplexu.

1.1 Algoritmus Nelder-Mead simplexové metody

Prvním krokem každé iterace je setřídění bodů simplexu podle velikosti. Označme $\mathbf{x}_1, ..., \mathbf{x}_{n+1}$ vrcholy simplexu. Po setřídění platí:

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq ... \leq f(\mathbf{x}_{n+1}), \tag{1.1}$$

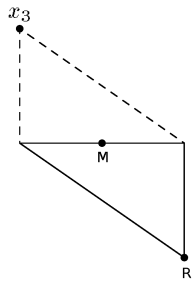
kde $f(\mathbf{x}_i)$ jsou hodnoty účelové funkce ve vrcholech simplexu.

Druhým krokem je reflexe nejhoršího bodu $\mathbf{x}_{n+1}$ proti těžišti protější stěny,

$$\mathbf{M} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}_k, \tag{1.2}$$

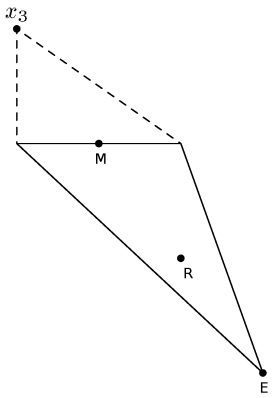
$$\mathbf{R} = \mathbf{M} + \rho\,(\mathbf{M} - \mathbf{x}_{n+1}). \tag{1.3}$$

Bod $\mathbf{M}$ reprezentuje těžiště stěny protilehlé bodu $\mathbf{x}_{n+1}$, bod $\mathbf{R}$ je pak reflexí tohoto bodu. Parametr ρ umožňuje ovlivnit délku reflexe, tedy zda bude bod R ležet ve



Obrázek 1.1: Operace reflexe.

stejné vzdálenosti od bodu $\mathbf{M}$ jako bod $\mathbf{x}_{n+1}$, nebo blíže či dále. Změna tohoto parametru může ovlivnit rychlost konvergence metody. Standardně se volí $\rho = 1$. Pokud platí $f(\mathbf{x}_1) \leq f(\mathbf{R}) < f(\mathbf{x}_n)$, byla reflexe úspěšná, bod $\mathbf{x}_{n+1}$ je nahrazen bodem R a iterace končí.



Obrázek 1.2: Operace expanze.

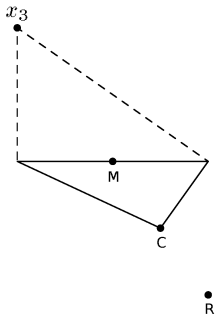
Pokud platí $f(\mathbf{R}) < f(\mathbf{x}_1)$ pokračuje se expanzí, dalším protažením reflexe v úspěšném směru může být dosažen ještě lepší výsledek

$$\mathbf{E} = \mathbf{M} + \chi\,(\mathbf{R} - \mathbf{M}). \tag{1.4}$$

Parametr χ ovlivňuje obdobně jako v předchozím případě vlastnosti metody. Standardně je voleno $\chi = 2$. V případě, že $f(\mathbf{E}) < f(\mathbf{R})$, je bod $\mathbf{x}_{n+1}$ nahrazen bodem $\mathbf{E}$, v opačném případě je nahrazen bodem $\mathbf{R}$.

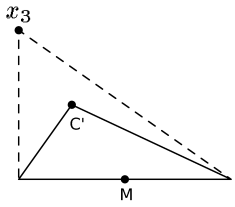
Pokud nebyla reflexe úspěšná $f(\mathbf{R}) \geq f(\mathbf{x}_n)$, pokračuje algoritmus vnitřní kontrakcí pokud $f(\mathbf{x}_n) \leq f(\mathbf{R}) < f(\mathbf{x}_{n+1})$, nebo vnější kontrakcí pokud $f(\mathbf{R}) \geq f(\mathbf{x}_{n+1})$. Vnější kontrakce je podobně jako reflexe zrcadlení nejhoršího bodu přes těžiště protější stěny, avšak do kratší vzdálenosti

$$\mathbf{C} = \mathbf{M} + \gamma\,(\mathbf{R} - \mathbf{M}). \tag{1.5}$$



Obrázek 1.3: Vnější zkrácení.

Standardně je parametr volen $\gamma = 1/2$. Bodem $\mathbf{C}$ je nahrazen bod $\mathbf{x}_{n+1}$ v případě, že $f(\mathbf{C}) \leq f(\mathbf{R})$, a iterace končí. Jinak pokračuje algoritmus zúžením simplexu.



Obrázek 1.4: Vnitřní zkrácení.

Vnitřní kontrakce je posunutí bodu $\mathbf{x}_{n+1}$ směrem k těžišti protější stěny $\mathbf{M}$

$$\mathbf{C}' = \mathbf{M} - \gamma\,(\mathbf{M} - \mathbf{x}_{n+1}). \tag{1.6}$$

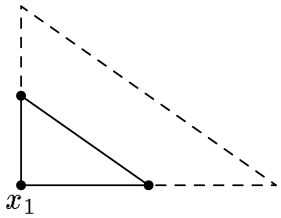
Parametr γ je stejný jako v předchozím případě. Bod $\mathbf{C}'$ nahradí bod $\mathbf{x}_{n+1}$, pokud $f(\mathbf{C}') < f(\mathbf{x}_{n+1})$, jinak pokračuje iterace zúžením simplexu.

Zúžení simplexu (shrink) je poslední transformací simplexu, která je provedena v případě, že žádná z předchozích transformací nebyla úspěšná. Tato transformace spočívá v tom, že je zachován nejlepší bod $\mathbf{x}_1$ a všechny ostatní body jsou posunuty směrem k bodu $\mathbf{x}_1$

$$\mathbf{x}'_i = \mathbf{x}_1 + \sigma\,(\mathbf{x}_i - \mathbf{x}_1), \quad i = 2, .., n+1. \tag{1.7}$$

Parametr σ je standardně volen $\sigma = 1/2$. Jelikož je po provedení této operace nutné vyhodnotit účelovou funkci v n nových bodech, je tato transformace nejdražší. Všechny předchozí transformace vyžadovaly pouze jedno vyhodnocení účelové funkce v novém bodě. Naštěstí je tento krok prováděn spíše výjimečně.

Aby měly transformace simplexu smysl naznačený na obr. 1.1 až obr. 1.5, musí splňovat následující nerovnosti [4]



Obrázek 1.5: Zúžení simplexu.

$$\rho > 0, \quad \chi > 1, \quad \chi > \rho, \quad 0 < \gamma < 1, \quad 0 < \sigma < 1. \tag{1.8}$$

Jak již bylo uvedeno výše, standardní volba parametrů metody je:

$$\rho = 1, \quad \chi = 2, \quad \gamma = \frac{1}{2}, \quad \sigma = \frac{1}{2}. \tag{1.9}$$

Vhodnou volbou parametrů lze ovlivnit rychlost konvergence simplexové metody, zejména ve vyšších dimenzích, jak bude naznačeno v následující části.

1.2 Konvergenční vlastnosti Nelder-Mead simplexové metody

Nelder Mead simplexová metoda je v podstatě heuristický algoritmus [5]. Z toho důvodu je velmi obtížné provést obecnou analýzu konvergence tohoto algoritmu do bodu minima funkce. Konvergence není obecně zajištěna. V praxi se také stává, že algoritmus zkonverguje do jiného, než optimálního bodu.

V roce 1996 byla McKinnonem publikována analýza [6], kde byla zkonstruována třída funkcí dvou proměnných, kdy při vhodné volbě počátečního simplexu (trojúhelníku) tento následně zkolaboval do úsečky. Funkce byly zkonstruovány tak, aby byla v každé iteraci provedena vnitřní kontrakce, přičemž nejlepší bod zůstává zachován. Po dostatečném počtu iterací se začal simplex blížit úsečce.

Další rozsáhlá studie konvergenčních vlastností byla publikována v roce 1998 J. C. Lagarisem se spolupracovníky [4]. Tato studie byla zaměřena na striktně konvexní funkce v 1D a 2D. V rámci studie bylo ukázáno, že v 1D konverguje algoritmus do minima striktně konvexní funkce. Ve 2D konvergují funkční hodnoty ve vrcholech simplexu ke stejné hodnotě a průměr simplexu konverguje k nule.

Z praktického hlediska velmi zajímavou práci publikovali v roce 2012 F. Gao a L. Han [7]. Publikace je zaměřena na vlastnosti Nelder-Mead metody při optimalizaci funkcí mnoha proměnných ($n > 5$). Analýza byla provedena pro stejnoměrně

konvexní funkce. Označíme-li $F(\Delta)$ součet funkčních hodnot ve vrcholech simplexu Δ:

$$F(\Delta) = f(\mathbf{x}_1) + f(\mathbf{x}_2) + ... + f(\mathbf{x}_{n+1}), \tag{1.10}$$

kde f je stejnoměrně konvexní funkce, a transformaci simplexu prostřednictvím extenze, vnitřní kontrakce nebo vnější kontrakce reprezentujeme pomocí operátoru $\hat{T}$, platí vztah [7]:

$$F(\Delta) - F(\hat{T}\,\Delta) \geq -\frac{n-1}{2n^2}\,\rho\left(\frac{1}{2}\,D(\Delta)\right), \tag{1.11}$$

kde n je dimenze parametrického prostoru, D je průměr simplexu a ρ je ostře rostoucí funkce s vlastností $\rho(0) = 0$. Ze vztahu (1.11) vyplývá, že pokles F je závislý na dimenzi n podle vztahu $(n-1)/2n^2$. Tento faktor klesá s rostoucím n a jde k nule pro $n \to \infty$. Z této vlastnosti vyplývá zhoršování efektivity Nelder-Mead simplexové metody pro rostoucí dimenzi parametrického prostoru, alespoň pro stejnoměrně konvexní funkce. Dále ze vztahu (1.11) vyplývá závislost horního odhadu poklesu F na průměru simplexu D. Proto je výhodné volit prvotní simplex spíše větší. K nejrychlejšímu poklesu účelové funkce pak dochází v počátečních iteracích, dokud se průměr simplexu výrazně nezmenší.

V rámci [7] byl dále za předpokladu stejnoměrně konvexní účelové funkce rozšířen závěr učiněný v [4], že průměr simplexu konverguje k nule pro dimenze $n > 2$. Na toto navazuje další analýza efektivity Nelder-Mead simplexové metody zohledňující počet kroků reflexe. Reflexe totiž nezmenší průměr simplexu, nepřispěje tudíž ke konvergenci průměru simplexu k nule. Čím více kroků reflexe bude učiněno, tím horší je efektivita metody. Na základě tohoto závěru navrhli autoři modifikaci Nelder-Mead simplexové metody spočívající v adaptivním nastavení parametrů metody ρ, χ, γ, σ v závislosti na dimenzi parametrického prostoru n:

$$\rho = 1, \quad \chi = 1 + \frac{2}{n}, \quad \gamma = 0.75 - \frac{1}{2n}, \quad \sigma = 1 - \frac{1}{n}. \tag{1.12}$$

Řadou numerických experimentů autoři ukázali, že počet kroků reflexe je touto volbou parametrů metody redukován. Pro $n = 2$ odpovídají parametry podle vztahu (1.12) parametrům standardním.

1.3 Ohraničení parametrického prostoru

Doposud popsaná metoda pracuje na neohraničené doméně. V praxi je však často nutné a vhodné omezit rozsah optimalizovaných parametrů. Omezení výpočetní domény lze řešit zavedením vazeb ve tvaru nerovností:

$$
\begin{aligned}
a_1 \leq x_1 \leq b_1,\\
a_2 \leq x_2 \leq b_2,\\
.....\\
a_n \leq x_n \leq b_n,
\end{aligned}
\tag{1.13}
$$

kde x_i je i-tá složka vektoru x a a_i, b_i, $i \in \{1..n\}$ jsou meze příslušné složky. Nerovnosti (1.13) lze přepsat do obecnějšího tvaru:

$$
\begin{aligned}
g_i(\mathbf{x}) = a_i - x_i \leq 0, \quad i = 1 \ ... \ n,\\
g_i(\mathbf{x}) = x_i - b_i \leq 0, \quad i = n+1 \ ... \ 2n.
\end{aligned}
\tag{1.14}
$$

Tyto nerovnosti omezí prohledávaný parametrický prostor na množinu
$\Omega = \{\mathbf{x} \mid g_i(\mathbf{x}) \leq 0 \ \forall i\}$. Nejjednoduššími metodami zahrnutí vazeb jsou metoda vnitřního bodu a metoda vnější penalty [1]. Obě metody konstruují modifikovanou účelovou funkci, ve které zohledňují míru nesplnění nerovností (1.14).

Metoda vnitřního bodu

Metoda vnitřního bodu konstruuje bariérovou funkci $b : \mathbb{R}^n \to \mathbb{R}$ s vlastnostmi $b|_\Omega \geq 0$ a $b(\mathbf{x}) \to \infty$ pro $\max_i g_i \to 0_-$.

Pomocí bariérové funkce lze pak optimalizační úlohu s omezeními (1.13) převézt na úlohu bez omezení:

$$
\min_x \{f(\mathbf{x}) + \mu b(\mathbf{x})\}, \quad \mu \to 0_+,
\tag{1.15}
$$

kde $f(\mathbf{x})$ je původní účelová funkce. Díky neomezenosti bariérové funkce na hranici parametrické množiny nemůže optimalizační algoritmus tuto množinu opustit. Je však nutné zajistit, aby výchozí bod ležel uvnitř množiny Ω. Jako bariérové funkce lze volit například funkce [1]:

$$
b(\mathbf{x}) = -\sum_{i=1}^{2n} \frac{c_i}{g_i(\mathbf{x})}
\tag{1.16}
$$

$$
b(\mathbf{x}) = -\sum_{i=1}^{2n} c_i \ln(g_i(\mathbf{x})),
\tag{1.17}
$$

kde c_i jsou vhodné konstanty. Z podstaty bariérové funkce vyplývá, že bude vždy částečně ovlivňovat minimalizovanou funkci. Její vliv bude tím větší, čím blíže bude minimum původní cílové funkce hranici množiny Ω. Pokud by leželo minimum přímo

na hranici množiny, nebude mít algoritmus šanci k němu zkonvergovat. Vzdálenost od optima bude v tomto případě silně záviset na škálování a vlastnostech bariérové funkce. Je proto vhodné volit bariérové funkce takové, jejichž velikost je zanedbatelná uvnitř množiny, a prudce rostou až v těsné blízkosti hranice množiny. Další možností je zpracovávat úlohu iterativně a v každém kroku postupně zmenšovat parametr μ v závislosti na vlastnostech účelové funkce f. Bariérová funkce bude úlohu ovlivňovat tím méně, čím menší je parametr μ. Na druhou stranu metoda bariéry zajišťuje, že výsledek optimalizace bude vždy ležet uvnitř omezené množiny Ω.

Metoda vnější penalty

Obdobně jako metoda vnitřní bariéry, metoda vnější penalty přičítá k účelové funkci tzv. penaltovou funkci $p : \mathbb{R}^n \to \mathbb{R}$. Tato funkce zohledňuje porušení nerovností (1.14). Její vlastnosti jsou:

$$p\,|_{\Omega} = 0, \quad p\,|_{\mathbb{R}^n - \Omega} > 0. \tag{1.18}$$

Přičtením penaltové funkce k původní účelové funkci se změní omezená úloha na úlohu neomezenou:

$$\min_x \{f(\mathbf{x}) + \nu p(\mathbf{x})\}, \quad \nu > 0, \tag{1.19}$$

kde ν je parametr penalty. Na rozdíl od předchozího případu, v tomto případě bude řešení tím blíže řešení původní úlohy, čím větší bude parametr ν. Jako penaltová funkce se nejčastěji volí kvadratická funkce [1]:

$$p(\mathbf{x}) = \frac{1}{2}|\alpha(\mathbf{x})|^2, \quad \alpha(\mathbf{x}) = \max\{g_i(\mathbf{x}), 0\} \tag{1.20}$$

Jelikož je penaltová funkce uvnitř množiny Ω nulová, neovlivňuje nijak původní účelovou funkci. Oproti bariérové funkci však penaltová funkce nezajistí, že řešení bude ležet uvnitř množiny. Jako penaltovou funkci by bylo tedy vhodnější zvolit funkci, která má prudší nárůst vně množiny Ω.

Výhodou Nelder-Mead optimalizační techniky je, že je schopna zpracovat nespojité funkce. Toho lze s výhodou využít ke konstrukci penaltové funkce, která nemusí být spojitá. Jedinými operacemi, kterými se může simplexová metoda dostat mimo omezenou množinu Ω jsou reflexe, extenze a vnější kontrakce. Rozhodnutí o tom, zda bude nový bod přijat se provádí vesměs na základě porovnání s hodnotou účelové funkce ve druhém nejhorším bodě $f(\mathbf{x}_n)$. Cílem penaltové funkce tedy bude zajistit, že bod mimo množinu Ω bude mít vyšší hodnotu modifikované účelové funkce, než je hodnota nejhoršího bodu aktuálního simplexu. Na základě tohoto požadavku byla

navržena následující penaltová funkce:

$$\alpha(\mathbf{x}) = \max\{g_i(\mathbf{x}), 0\}$$
$$p(\mathbf{x}, f_{\max}) = \begin{cases} 0 & \alpha(\mathbf{x}) = 0 \\ |f(\mathbf{x})| + 2\,|f_{\max}| & \alpha(\mathbf{x}) > 0, \end{cases} \tag{1.21}$$

kde $f_{\max}$ je hodnota účelové funkce v nejhorším bodě simplexu $f_{\max} = f(\mathbf{x}_{n+1})$. Modifikovaná účelová funkce má pak tvar:

$$\tilde{f}(\mathbf{x}, f_{\max}) = f(\mathbf{x}) + p(\mathbf{x}, f_{\max}). \tag{1.22}$$

Je-li $f(\mathbf{x}) < 0$, zajistí přičtení $|f(\mathbf{x})|$ vyrovnání na 0 a přičtení $2\,|f_{\max}|$ pak zajistí, že nový bod bude horší, než všechny dosavadní body simplexu. Tato penaltová funkce vytváří pro simplexovou metodu na hranici množiny Ω nepropustnou bariéru. Zároveň je nulová uvnitř množiny Ω, takže nikterak neovlivňuje původní účelovou funkci. Řešení problému s modifikovanou účelovou funkci, bude tedy odpovídat řešení problému s omezeními s původní účelovou funkcí. Počáteční simplex musí pochopitelně ležet uvnitř množiny Ω, jinak se začne metoda chovat nepředvídatelně.

1.4 Úplný algoritmus Nelder-Mead simplexové metody

V předchozí části byla rozebrána jedna iterace Nelder-Mead simplexové metody a omezení výpočetní domény. Pro úspěšnou implementaci metody je však ještě nutné vyřešit generaci počátečního simplexu a ukončení optimalizace.

Počáteční simplex

Vytvoření prvotního simplexu je celkem jednoduché. Nejprve musí být stanoven bod $\mathbf{x}_0$, kde má optimalizace začínat. Ten je buď definován uživatelem, nebo například jako střed ohraničené výpočetní domény. Tento bod je vzat jako jeden z bodů simplexu. Dalších n bodů je vygenerováno jako posunutí tohoto bodu ve směru bázových vektorů n-rozměrného prostoru $\mathbf{e}_i$ o definovanou vzdálenost ξ:

$$\mathbf{x_i} = \mathbf{x}_0 + \xi\,\mathbf{e}_i. \tag{1.23}$$

Velikost posunutí ovlivní počáteční rychlost konvergence. Jak vyplývá z analýzy konvergenčních vlastností [7], je vhodnější spíše větší simplex než menší. V implementované metodě je velikost posunutí ξ stanovena jako 1/100 nejmenšího rozměru výpočetní domény. Ve všech takto vytvořených bodech je následně vyhodnocena účelová funkce a body setříděny podle její hodnoty.

Algoritmus dále iterativně pokračuje modifikacemi simplexu popsanými v části 1.1.

Ukončení optimalizace

Pro ukončení optimalizace je možné použít několik strategií. Nejjednodušší možností je provést pevně stanovený počet iterací. Tento přístup zajistí konečnost algoritmu, avšak žádným způsobem nezohledňuje vývoj simplexu.

Další možností je sledování velikosti simplexu. V tomto případě je sledována velikost nejdelší hrany simplexu a je porovnávána s předem stanovenou mezí. Za předpokladu, že simplex se při konvergenci do minima zmenšuje, mělo by být takto detekováno dosažení minima. Tento přístup však selže, pokud simplex začne degenerovat a kolabovat do nadroviny v daném prostoru, jak bylo diskutováno např. v [6]. Toto by pak vedlo k zacyklení.

Asi nejčastěji používaným přístupem je sledování rozdílu funkční hodnoty účelové funkce v nejhorším a nejlepším bodě simplexu [8]:

$$|f(\mathbf{x}_{n+1}) - f(\mathbf{x}_1)| < \varepsilon. \tag{1.24}$$

Obdobně jako v předchozím případě, ani tento postup nemusí zajistit ukončení algoritmu v případě degenerace simplexu. Zohledňuje však konvergenci metody do minima. Pro implementaci metody byla proto použita kombinace tohoto přístupu a omezení maximálního počtu iterací. Velmi důležitá je vhodná volba meze ε. Rozdíl největší a nejmenší hodnoty účelové funkce v rámci simplexu totiž nijak nesouvisí se vzdáleností simplexu od hledaného minima. Pokud je mez zvolena příliš velká, může se stát, že iterace skončí předčasně. Tento efekt se projevuje zejména pokud simplex v průběhu výpočtu narazí na hranici domény a minimum leží právě na této hranici. V takovém případě se totiž typicky začne simplex u hranice zmenšovat, než pokračuje prohledávání dál podél hranice. Pokud je mez příliš malá, metoda zase zbytečně dlouho zpřesňuje dosažené minimum.

Kapitola 2

Particle Swarm Optimization

Particle Swarm Optimization (PSO) je algoritmus patřící do kategorie evolučních algoritmů. Byl vyvinut na základě modelů sociálního chování včelích rojů či ptačích hejn při vyhledávání potravy [9].

Podstata základního PSO algoritmu je následující. Na prohledávaném prostoru je náhodně rozmístěno určité množství částic nazývaných agenti. Každý agent si pamatuje pozici, ve které při svém pohybu prostorem narazil na minimum účelové funkce. Tento bod je v terminologii PSO označován jako p_{best}. Dále má informaci o globálním nejlepším bodě nalezeném v průběhu optimalizace všemi agenty označeném g_{best}. Pohyb agentů prohledávaným prostorem je v podstatě popsán kinematikou hmotného bodu. Každý agent má v každé iteraci i definovanou rychlost svého pohybu $\boldsymbol{v}^i$. V každé iteraci pak změní každý agent svoji aktuální polohu podle vztahu $\boldsymbol{x}^i = \boldsymbol{x}^{i-1} + \boldsymbol{v}^i$. Rychlost agentů je na počátku generována náhodně. Rychlost nezůstává v průběhu optimalizace konstantní, ale mění s v každé iteraci podle vztahu:

$$v_n^i = w\, v_n^{i-1} + c_1\, \text{rnd}_{1,n}\, (p_{\text{best},n} - x_n^{i-1}) + c_2\, \text{rnd}_{2,n}\, (g_{\text{best},n} - x_n^{i-1}), \tag{2.1}$$

kde v_n^{i-1} je n-tá složka rychlosti v předchozí iteraci x_n^{i-1} je n-tá složka polohy v předchozí iteraci, $\text{rnd}_{1,n}$, $\text{rnd}_{2,n}$ jsou náhodná čísla z intervalu $< 0; 1 >$ s rovnoměrným rozdělením, pro každou složku jiná, c_1, c_2 jsou konstanty zohledňující vliv jednotlivých složek, faktor w definuje míru zpomalování agentů. Význam rovnice lze interpretovat následovně. Člen $(p_{\text{best},n} - x_n^{i-1})$ zohledňuje soukromou zkušenost každého agenta a urychluje jej ve směru jeho doposud nalezeného nejlepšího bodu. Oproti tomu člen $(g_{\text{best},n} - x_n^{i-1})$ zohledňuje kolektivní zkušenost celého hejna a urychluje každého agenta ve směru doposud nalezeného společného minima. Zavedení náhodných funkcí $\text{rnd}_{1,n}$ a $\text{rnd}_{2,n}$ dává pohybu částic nederministický charakter, jaký je například pro včelí roj přirozený. Díky tomuto znáhodnění pohybu je PSO schopné efektivně prohledat stavový prostor. Parametr w v prvním členu pravé strany rovnice 2.1 má významný vliv na konvergenci, určuje totiž rychlost zpomalování pohybu

agentů v průběhu optimalizace. Je volen z intervalu $< 0; 1 >$. Pokud je příliš velký, nezkonverguje roj ve vymezeném počtu iterací. Pokud je příliš malý, zkonverguje roj příliš rychle a nedojde k dostatečnému prohledání prostoru. Doporučuje se proto zvolit parametr w zpočátku spíše větší, a v průběhu optimalizace jej postupně snižovat [10].

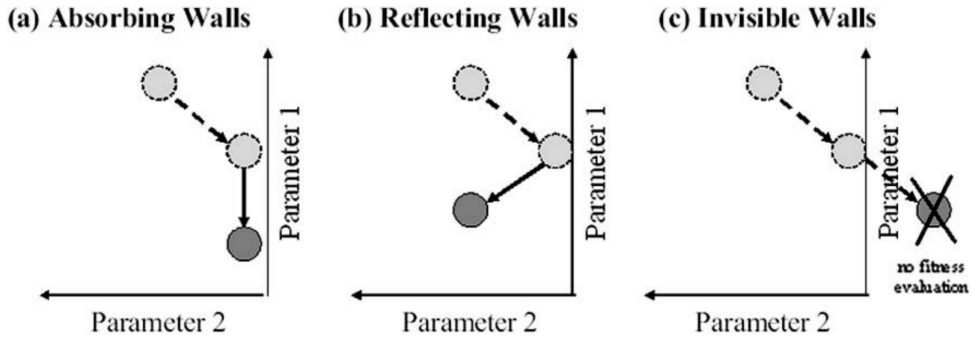
Velmi zajímavý pohled nabízí interpretace rovnice (2.1) z hlediska dynamiky hmotného bodu o jednotkové hmotnosti [11]:

$$F = m\frac{\Delta v_n}{\Delta t} = (w-1)\, v_n^{i-1} + c_1\, \mathrm{rnd}_{1,n}\, (p_{\mathrm{best},n} - x_n^{i-1}) + \\ + c_2\, \mathrm{rnd}_{2,n}\, (g_{\mathrm{best},n} - x_n^{i-1}). \tag{2.2}$$

První člen pravé strany představuje sílu závislou na rychlosti. Ta má pro $w < 1$ disipativní charakter a způsobuje tlumení systému. Druhý a třetí člen pravé strany jsou síly závislé na poloze. To v podstatě odpovídá charakteru lineární pružiny. Rovnice tedy představuje rovnici tlumeného lineárního harmonického oscilátoru. Toto chování se projevuje zejména ke konci optimalizace, kdy mají agenti tendenci oscilovat okolo rovnovážné polohy dané g_{best}.

Omezení výpočetní domény PSO

Pro PSO algoritmus byly navrženy tři varianty omezení výpočetní domény [12], tzv. absorbční stěna, odrazná stěna a neviditelná stěna. Charakter všech tří variant je naznačenen na obrázku 2.1. Podmínky pro jednotlivé varianty jsou aplikovány



Obrázek 2.1: Typy omezení výpočetní domény v PSO [12].

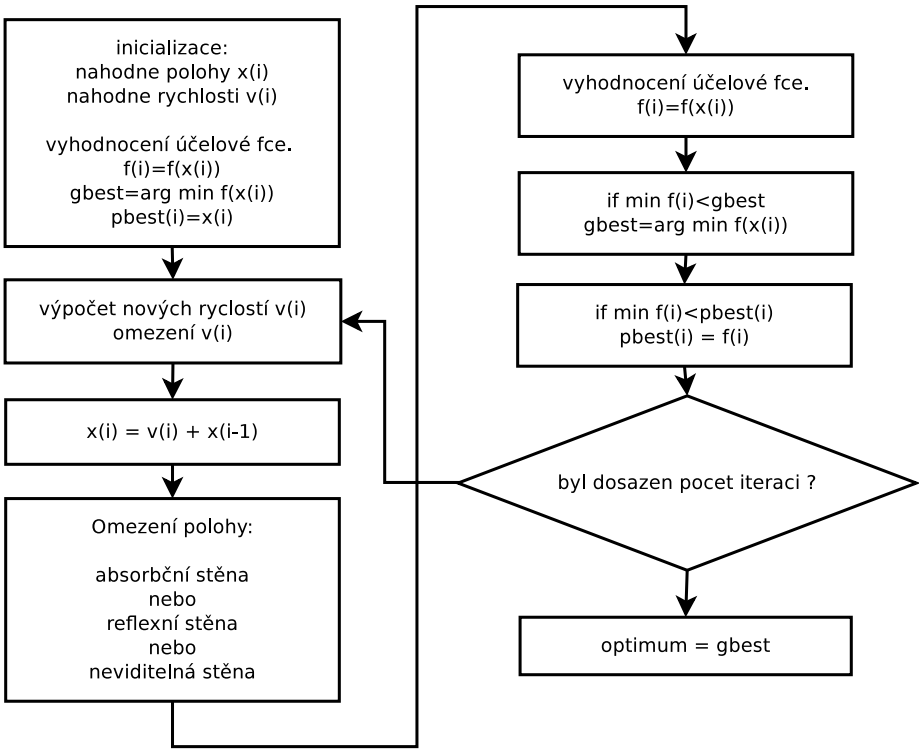
po výpočtu posunutí agentů. Dojde li v některé dimenzi k překročení stanovené hranice, je v případě absorbční stěny posunuta poloha v této dimenzi na zadané maximum a rychlost je v této dimenzi vynulována. V případě reflexní stěny je přesah polohy v dané dimenzi ozrcadlen zpět do výpočetní oblasti a je změněno znaménko

příslušné složky rychlosti. Neviditelná stěna znamená, že pro agenty, kteří opustili hranice výpočetní domény není vypočítávána hodnota účelové funkce. Ta je znovu vypočítívána až po jejich návratu do výpočetní domény.

Vedle omezení polohy do hranic výpočetní domény je vhodné omezit také rychlost. Bez omezení rychlosti se může stát, že agent během jednoho posunutí několikrát překročí rozměr výpočetní domény v dané dimenzi. Rychlost v daném rozměru je proto vhodné omezit tak, aby nepřekročila rozsah daného rozměru. Agent má pak možnost v rámci jedné iterace přeletět doménu v daném rozměru právě jednou.

2.1 Algoritmus PSO

Celkový vývojový diagram PSO algoritmu je zobrazen na obrázku 2.2.



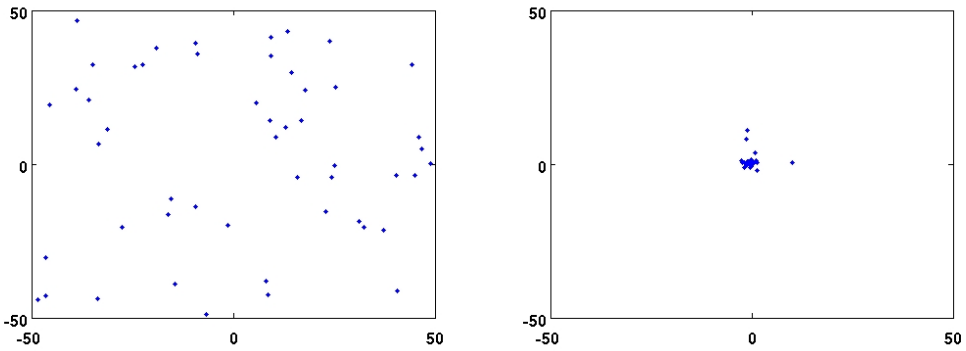
Obrázek 2.2: Vývojový diagram PSO.

Na počátku v rámci inicializace jsou náhodně vygenerovány polohy agentů v rámci výpočetní domény a jejich počáteční rychlosti. Dále jsou vypočítány hodnoty účelové funkce v aktuálních polohách a nalezen bod g_{best}. Body p_{best} odpovídají v této inicializační části aktuálním polohám. Následuje iterační část algoritmu. Nejprve jsou vypočítány nové složky rychlostí podle vztahu (2.1) a jsou aplikovány omezujících podmínky na jednotlivé složky rychlostí. Po výpočtu rychlostí jsou agenti posunuti do nových poloh a jsou aplikována pravidla pro omezení výpočetní domény.

Následně jsou vypočítány hodnoty účelové funkce v aktuálních polohách. Poté jsou aktuální hodnoty porovnány s hodnotami v bodech g_{best} a p_{best}, které jsou v případě lepší hodnoty aktualizovány. Zde se iterační cyklus uzavírá. V případě, že nebylo dosaženo předepsaného počtu iterací, pokračuje algoritmus výpočtem nových rychlostí. V opačném případě je nalezené minimum dáno aktuálním g_{best}.

2.2 Průběh optimalizace a vlastnosti PSO

Jak vyplývá z popisu aloritmu, agenti jsou nejprve náhodně rozmístěni po celé výpočetní doméně a následně se kvazinedeterministickým způsobem pohybují uvnitř výpočetní domény. Typicky z počátku optimalizace procházejí agenti doménu celou. Postupně dochází k útlumu pohybu agentů. Zmenšuje se průměr oblasti prohledávané agenty, ti se shlukují kolem polohy g_{best} a jemně prohledávají okolí.



(a) počátek optimalizace (b) konec optimalizace

Obrázek 2.3: Ilustrace rozmístění agentů v průběhu optimalizace PSO. Zpočátku agenti prochází celou doménu(a), na konci se shluknou v okolí nalezeného minima (b).

Náhodný způsob prohledávání výpočetní domény dodává PSO metodě odolnost proti uvíznutí v lokálním minimu. To z ní dělá globální optimalizační nástroj. Konvergence metody do globálního minima je však ve srovnání s Nelder-Mead metodou pomalá a potřebný počet vyčíslení účelové funkce je vyšší.

Vlastnosti metody jsou závislé na nastavení parametrů c_1, c_2, w. Podle obecného doporučení [12] je vhodná volba $c_1 = 2, c_2 = 2$. Další doporučení navrhuje volit konstanty tak, aby $c_1 + c_2 \leq 4$ [10]. Praktické zkušenosti toto potvrzují. Vyšší hodnota sociálního parametru c_2 urychluje konvergenci, klesá tím však efektivnost prohledání výpočetní domény. Na rychlost konvergence má dále významný vliv charakter optimalizované účelové funkce. Pokud je účelová funkce relativně hladká s dobře definovaným lokálním minimem, jako například Rosenbrockova funkce popsaná v

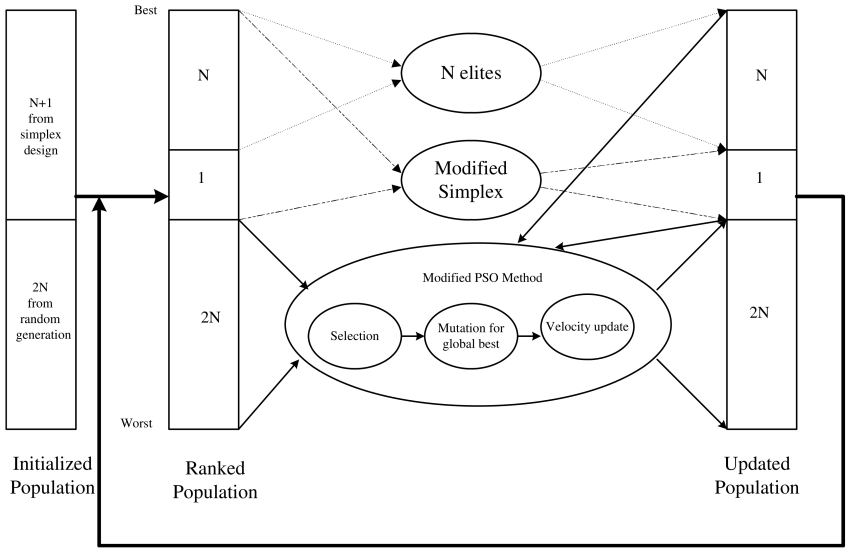
kapitole 5, konverguje PSO rychle k malému shluku agentů, kteří dále lokálně zpřesňují výsledek. Pokud má účelová funkce velké množství blízkých lokálních minim, jako např. funkce Griewank, která je taktéž popsaná v kapitole 5, má prohledávání spíše globální charakter a teprve v závěrečné fázi zkonvergují agenti k nalezenému minimu.

Parametr w, jak již bylo diskutováno, ovlivňuje tlumení systému. S rostoucím w se tlumení zeslabuje a naopak. Pro tento parametr jsou doporučeny hodnoty z intervalu $w \in < 0.1; 0.9 >$. Velikost parametru w je nutné volit také s ohledem na použitý typ stěny. Je li použita absorbční stěna, která přidává další tlumení, mělo by být w vyšší, je li použita reflexní stěna, mělo by být w spíše menší. Vhodná obecnější strategie je, aby bylo w na počátku nastaveno na vyšší hodnotu, např $w = 0.9$ a v průběhu algoritmu klesalo k dostatečně malé hodnotě, např $w = 0.3$. Tento postup zajistí konvergenci metody v závěrečné fázi. Pokud je w nastaveno fixně a je příliš velké, nemusí PSO v zadaném počtu iterací vůbec zkonnvergovat. PSO metoda obvykle končí dosažením zadaného počtu iterací.

Kapitola 3

Hybridní Nelder-Mead PSO algoritmus

Motivací pro tvorbu hybridního Nelder-Mead PSO algoritmu je využít výhody obou algoritmů v rámci jedné metody. Nelder-Mead metoda je poměrně efektivní lokální algoritmus. Zcela však selhává na funkcích s více lokálními minimy. Oproti tomu PSO je schopné prohledávat parametrický prostor globálně, tzn. nalézt globální minimum i u funkcí, které obsahují vedlejší lokální minima. Jeho nevýhodou je ve srovnání s Nelder-Mead metodou pomalá konvergence a vysoký počet vyčíslení účelové funkce.



Obrázek 3.1: Schéma hybridního Nelder-Mead PSO algoritmu [13].

Hybridní Nelder-Mead PSO algoritmus byl navržen v [13, 14]. Podstata metody je znázorněna na obrázku 3.1. Na počátku je vygenerováno $3N + 1$ agentů. Populace

je setříděna podle velikosti účelové funkce. Je rozdělena na prvních $N+1$ agentů a zbylých $2N$ agentů. Agent g_{best} s nejlepší hodnotou účelové funkce je vybrán z celé populace. Z posledních $2N$ agentů jsou vybráni tzv. „neighbour best“ agenti n_{best}, kteří budou popsáni později. Prvních $N+1$ agentů vytvoří simplex a vstupuje do simplexové metody. Zbylých $2N$ agentů je zpracováno PSO algoritmem. Následně je populace znovu setříděna a celý postup se opakuje.

Autoři v rámci této hybridní metody navrhli modifikace jak pro simplexovou metodu, tak pro PSO algoritmus [13].

Modifikace Nelder-Mead metody

Modifikace spočívá v provedení druhé expanze. Jednou z operaci prováděných se simplexem v rámci simplexové metody je expanze viz. obr. 1.2. Pokud je expanze úspěšná a platí, že $f(E) < f(1)$, je provedena ještě druhá expanze do dvojnásobné vzdálenosti, která je v případě lepšího výsledku přijata. Tento postup má za cíl dále urychlit konvergenci simplexové metody.

Modifikace PSO algoritmu

Pro PSO algoritmu byly použity dvě modifikace. První z nich je zavedení tzv. „neighbour best“ agentů. Populace $2N$ agentů vstupujících do PSO metody je rozdělena do dvojic. Z každé dvojice je vybrán agent s lepší hodnotou účelové funkce v bodě p_{best}, ten je pak označován jako „neighbour best“ n_{best}. V rovnici pro aktualizaci rychlosti agentů 2.1 je pak pro obě částice nahrazen p_{best} příslušným bodem n_{best}.

Druhá modifikace PSO metody spočívá v zavedení mutace pro bod g_{best}. Z polohy g_{best} je odvozeno 5 nových bodů $g^{\text{new}}_{\text{best},i,k} = g_{\text{best},i} + \varepsilon_{i,k}$, kde $\varepsilon_{i,k}$ jsou náhodné vektory. Jako nový g_{best} je následně přijat nejlepší z bodů $g^{\text{new}}_{\text{best},i}$, pokud má nižší hodnou účelové funkce než starý g_{best}.

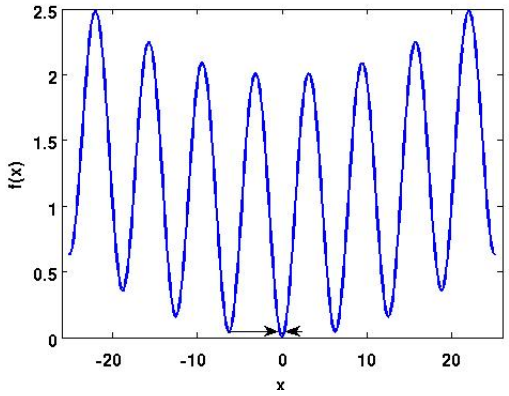
3.1 Vlastnosti hybridního algoritmu

Výše popsaný algoritmus byl implementován a byly testovány jeho základní vlastnosti. V implementaci však nebyla zahrnuta modifikace Nelder-Mead simplexové metody, ani mutační heuristika pro body g_{best}. Nebyly použity ani neighbour best částice. Místo toho byla použita permutace pro body p_{best} mezi jednotlivými agenty.

Jak již bylo naznačeno, konverguje Nelader-Mead metoda k minimu mnohem rychleji, než PSO. Z toho vyplývá obvyklý průběh optimalizace. Na začátku je populace rozdělena na $N+1$ nejlepších částic a $2N$ ostatních. Díky rychlé konvergenci simplexové metody, nabyde poměrně rychle, během několika iterací, prvních $N+1$ agentů tvořících simplex výrazně menší hodnoty účelové funkce, než zbytek

populace. Tím dojde k rozdělení populace na dvě množiny, kdy nad první probíhá Nelder-Mead algoritmus a nad druhou PSO algoritmus. K promíchání množin dochází pouze zpočátku, nebo v případě, že simplexová metoda uvízne v lokálním minimu a PSO nalezne jinde lepší hodnotu.

Druhým problémem přináší aktualizace polohy g_{best} z Nelder-Mead algoritmu. Situace je naznačena na obrázku obr. 3.2. V případě funkce s mnoha lokálními mi-



Obrázek 3.2: Ilustrace účelové funkce s mnoha blízkými lokálními minimy a vymezení oblasti s nižší hodnotou účelové funkce, než je hodnota prvního lokálního minima.

nimy „uvízne“ typicky simplexová metoda v některém z lokálních minim. Pokud je hodnota lokálního minima blízká hodnotě globálního minima, jak je tomu například u testovací funkce Griewank, která bude popsána v kapitole 5, musí některý z agentů PSO projít velmi blízko globálního minima, aby nalezl lepší hodnotu účelové funkce a vyprostil tak simplex z lokálního minima, což je poměrně nepravděpodobné. Pravděpodobnost nalezení globálního minima dále snižuje to, že agenti jsou přitahováni prostřednictvím g_{best} k lokálnímu minimu. Tímto způsobem narušuje Nelder-Mead metoda prohledávací funkci PSO. Navíc, pokud je simplex zachycen v lokálním minimu, nepodílí se již dále na prohledávání výpočetní domény.

3.2 Upravený hybridní Nelder-Mead PSO algoritmus

Cílem úprav hybridního algoritmu bylo překonat výše zmíněné obtíže. Na základě předchozí analýzy byly stanoveny následující cíle:

- Snížit vliv Nelder-Mead metody na PSO, aby negativně neovlivňovala prohledávací schopnosti PSO.
- Zvýšit podíl Nelder-Mead Metody na prohledávání oblasti.

Na základě uvedených pozorování a stanovených cílů vznikla následující myšlenka. Simplexová metoda má schopnost efektivně vyhledávat lokální minima. Pokud tedy PSO nalezne nějaký nový bod g_{best}, mělo by být jeho okolí prohledáno pomocí simplexové metody. Každý nový bod g_{best}, který je nalezen PSO je zaznamenán do zásobníku. Body g_{best} jsou následně setříděny podle velikosti účelové funkce. Je vybrán bod s nejmenší hodnotou, jehož okolí ještě nebylo prohledáno simplexovou metodou a v jeho okolí je sestaven simplex. Za tím účelem je celá populace setříděna podle velikosti. Lze předpokládat, že prvních $N + 1$ agentů tvořilo simplex v předchozí iteraci simplexové metody a budou mít výrazně nižší hodnotu účelové funkce, než zbytek populace. Vytvářený simplex je tedy doplněn body s pořadím $N + 2$ až $2N + 1$. Tím je vytvořen simplex a simplexová metoda i PSO probíhají nadále nezávisle. Simplexová metoda probíhá až do předem dané přesnosti nalezeného minima, to znamená, že rozdíl mezi nejvyšší a nejnižší hodnotou účelové funkce daného simplexu je menší, než stanovená mez. Výsledné minimum nalezené simplexovou metodou je zaznamenáno zpět do zásobníku bodů g_{best} namísto původní hodnoty a daný bod je označen jako již zpracovaný simplexovou metodou. Vedle bodu g_{best} byl definován ještě bod g_{NMbest}, který je bodem s nejlepší hodnotou účelové funkce nalezený Nelder-Mead metodou. Nalezené minimum je tedy ještě porovnáno s hodnotou bodu g_{NMbest} a pokud je jeho hodnota menší, je bod g_{NMbest} aktualizován. Body g_{best} ani p_{best} nejsou aktualizovány ze simplexové metody ale pouze z PSO metody. Tím je odstraněn silný vliv simplexové metody na PSO algoritmus. Aby byla zajištěna vazba ze simplexové do PSO algoritmu, je rozšířena rovnice pro rychlost agentů o člen obsahující bod g_{NMbest}

$$\begin{aligned} v_n^i = {} & (w + c_0\,\text{rnd}_{0,n})\,v_n^{i-1} + c_1\,\text{rnd}_{1,n}\,(P_{nk}\cdot p_{\text{best},k} - x_n^{i-1}) \\ & + c_2\,\text{rnd}_{2,n}\,(g_{\text{best},n} - x_n^{i-1}) + c_3\,\text{rnd}_{3,n}\,(g_{\text{NMbest},n} - x_n^{i-1}). \end{aligned} \tag{3.1}$$

Hodnotou parametru c_3 lze nastavit míru ovlivňování PSO simplexovou metodou. Pokud je nastaven na hodnotu 0, není PSO ovlivněno vůbec. V průběhu testování se jako rozumná hodnota ukázalo $c_3 = 0.2$. V rovnici byl k parametru w přičten ještě náhodný člen $c_0 \cdot \text{rnd}_{0,n}$ mající za cíl dále narušit deterministický pohyb agentů [14]. V průběhu testování bylo voleno $c_0 = 0.2$. Ostatní parametry odpovídají parametrům diskutovaným v rámci PSO algoritmu. Jako poslední úprava inspirovaná „neighbour best“ agenty popsanými výše byla použita permutace p_{best} každých dvou sousedních agentů z hlediska pořadí. V rovnici (3.1) je úprava zohledněna členem $P_{nk} \cdot p_{\text{best},k}$ s využitím Einsteinovy sumační konvence a permutační matice P_{nk}.

Díky tomu, že Nelder-Mead simplexová metoda má v rámci hybridního algoritmu omezenou přesnost, nezůstane neúměrně dlouho v jednom lokálním minimu, ale prohledá v průběhu optimalizace několik minim. Tím se zvyšuje pravděpodobnost na nalezení globálního minima. Jako optimum nalezené algoritmem slouží lepší

z bodů g_{NMbest}, g_{best}. Stejně jako v případě PSO končí hybridní algoritmus dosažením stanoveného počtu iterací. Pro zpřesnění nalezeného minima proběhne po konci hybridní metody ještě samostatná Nelder-Mead simplexová metoda, která má nastavenou nižší mez přesnosti, než měla uvnitř hybridního algoritmu.

Kapitola 4

Program NM-PSOptimizer

Jedním z hlavních cílů práce bylo vytvořit program, který implementuje hybridní Nelder-Mead PSO metodu, jejíž algoritmus byl diskutován v předchozí kapitole. Program je koncipován jako univerzální optimalizační nástroj a jeho rozhraní vychází z koncepce programu PSOptimizer [15, 16]. Program je implementován v prostředí Matlab. Možné syntaxe spuštění optimalizace jsou následující:

```
res=NM_PSO(NMdata,'fitnessFunction');
res=NM_PSO(NMdata,'fitnessFunction','PropertyName',PropertyValue,...);
```

Struktura NMdata obsahuje parametry optimalizace a bude popsána níže. Výstupní struktura res obsahuje výsledky optimalizace a bude také popsána dále. Parametr 'fitnessFunction' obsahuje jméno m-souboru s účelovou funkcí. Této funkci je předávána řetězcová proměnná a struktura ve formátu NMdata a vrací hodnotu účelové funkce v daném bodě.

Funkci NM_PSO je možné volat i v rozšířené variantě s parametry upravujícími další vlastnosti optimalizace. Prvním parametrem je vždy název vlastnosti, druhým parametrem je pak hodnota dané vlastnosti. Možné parametry a rozsahy hodnot jsou uvedeny v tabulce 4.1.

Struktura NMdata

Struktura NMdata vychází ze struktury PsoData [16] a je s ní plně kompatibilní. Struktura obsahuje následující vstupní parametry optimalizace: hranice výpočetní oblasti, dimenzi optimalizace a vazby mezi jednotlivými proměnnými. Jednotlivé složky jsou popsány v tabulce 4.2.

Matice data1-data3 obsahují optimalizované proměnné. Na jejich základě je vyhodnocována účelová funkce. Struktura bound obsahuje hranice optimalizační domény. Jejich počet musí odpovídat dimenzi optimalizace.

Vlastnost	Meze	Popis
'MaxIter'	$1..\infty$	Maximální počet iterací
'Accuracy'	$> 1e-15$	Mez přesnosti závěrečné zpřesňující Nelder-Mead metody
'IntNMacc'	$> 1e-15$	Mez přesnosti vnitřní Nelder-Mead metody
'IntNMsteps'	$1..\infty$	Počet kroků vnitřní Nelder-Mead metody v rámci jedné iterace hybridního algoritmu
'Wall'	$0; 1$	0- absorpční stěna 1- reflexní stěna
'PSOParams'	$[c_0, c_1, c_2, c_3, w]$	Vektor parametrů PSO metody
'SaveLoc'	$0; 1$	0- nalezená lokální minima neukládána 1- nalezená lokální minima ukládána

Tabulka 4.1: Volitelné parametry funkce NM_PSO.

NMdata.data1	matice typu (m, n)	1. slot dat pro úlohu
NMdata.data2	matice typu (u, v)	2. slot dat pro úlohu
NMdata.data3	matice typu (x, y)	3. slot dat pro úlohu
NMdata.bound	cell typu $(1, a)$	obsahuje matice s hranicemi optim. oblasti [min max]
NMdata.cond	cell typu $(1, a)$	obsahuje matice, které ukazují na optimalizované parametry
NMdata.rank	integer	dimenze optimalizovaného problému
NMdata.type	'psopt'	pomocný řetězec

Tabulka 4.2: Složky struktury NMdata.

NMdata.bound$\{k\}$=[min$\{x_k\}$, max$\{x_k\}$]

Proměnná rank je nepovinná. Je li použita, musí být v souladu s počtem hranic oblasti. Specifické postavení má struktura cond. Tato struktura umožňuje spřáhnout vybrané členy matic data1-data3 tak, že vystupují jako jedna proměnná. Tím je následně snížena dimenze optimalizace. Tento postup v podstatě implementuje vazby ve tvaru rovností $x_i = x_j$. Každý člen struktury cond sestává ze tří sloupcových

vektorů o výšce i.

$$\mathsf{NMdata.cond}\{k\}=[A_k\ B_k\ C_k]$$

Výška vektorů i znamená počet svázaných proměnných v dané dimenzi optimalizace. Pokud tedy dané dimenzi odpovídá pouze jedna proměnná, je i v dané dimenzi 1. Trojice A_k, B_k, C_k, musí být definována pro každou dimenzi optimalizovaného problému. Složky sloupce $A_k(i)$ obsahují čísla z množiny $\{1, 2, 3\}$, která jsou ukazateli na jednu ze složek Data1, Data2, Data3. Složky sloupce $B_k(i)$ obsahují čísla řádků matic v datových složkách příslušných k $A_k(i)$. Složky sloupce $C_k(i)$ obsahují čísla sloupců (tedy pozici na daném řádku) matic v datových složkách příslušných k $A_k(i)$. Trojicí $[A_k(i)\ B_k(i)\ C_k(i)]$ je tedy pro dané i adresována jedna proměnná v příslušné datové složce.

Struktura res

res.data1	matice typu (m, n)	1. slot zoptimalizovaných dat
res.data2	matice typu (u, v)	2. slot zoptimalizovaných dat
res.data3	matice typu (x, y)	3. slot zoptimalizovaných dat
res.score	double	hodnota minima účelové funkce
res.evals	vector	akumulovaný počet vyhodnocení účelové funkce v iteracích
res.valhist	vector	vývoj minima v průběhu optimalizace
res.done	logical	bezchybné dokončení optimalizace
totTime	double	celkový čas optimalizace[s]
res.type	'optim'	pomocný řetězec
res.locmindata1	cell	složka data1 lokálních minim
res.locmindata2	cell	složka data2 lokálních minim
res.locmindata3	cell	složka data3 lokálních minim
res.scorelocmin	vektor	hodnoty účelové funkce nalezených lokálních minim

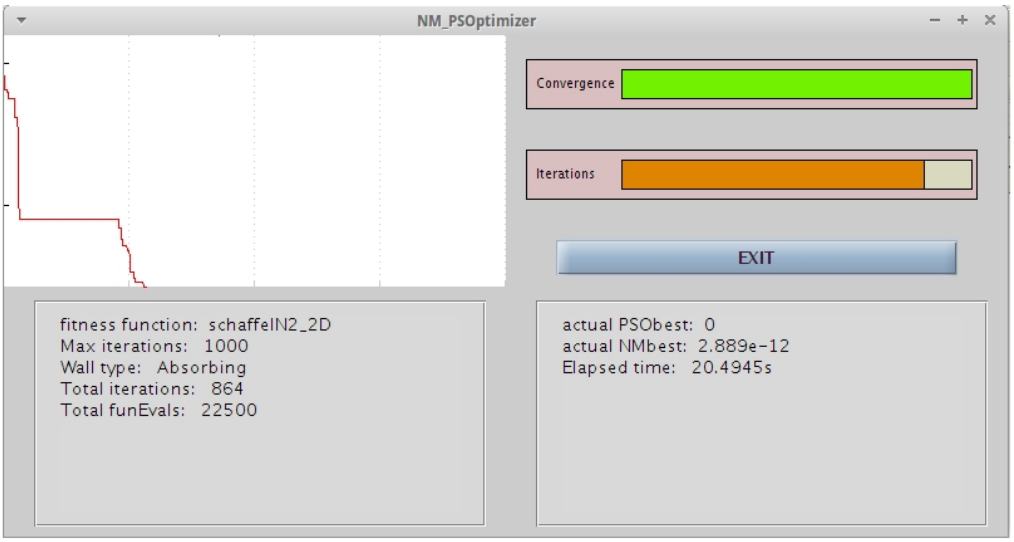
Tabulka 4.3: Složky struktury res.

Do struktury res jsou uloženy výsledky optimalizace dané úlohy. I zde byla snaha zachovat co největší kompatibilitu s výstupem programu PSOptimizer. Struktura

obsahuje následující výstupy: datové složky se zoptimalizovanými proměnnými, hodnotu nalezeného minima účelové funkce, akumulovaný počet vyhodnocení účelové funkce v průběhu jednotlivých iterací, vývoj nalezeného minima v průběhu iterací ,příznak, zda byla optimalizace úspěšně dokončena, celkový čas optimalizace. Pokud je ve volbách programu NM-PSOptimizer nastaveno ukládání lokálních minim, obsahuje výstupní struktura ještě datové složky obsahující polohy lokálních minim a složku s příslušnými hodnotami účelové funkce v jednotlivých minimech. Jednotlivé složky a jejich význam jsou popsány v tabulce 4.3.

Grafické uživatelské rozhraní

K programu NM-PSOptimizer bylo vytvořeno grafické uživatelské rozhraní (GUI). Toto rozhraní zobrazuje informace o průběhu optimalizace a objeví se po spuštění funkce NM_PSO. Grafické rozhraní je znázorněno na obrázku 4.1.



Obrázek 4.1: Grafické uživatelské rozhraní programu NM_PSOptimizer.

Grafické rozhraní opět vychází z rozhraní programu PSOptimizer. Jak bylo vysvětleno v popisu algoritmu, probíhá nejprve zadaný počet iterací hybridního algoritmu a následně maximálně stejný počet iterací zpřesňující Nelder-Mead simplexové metody. Panel convergence zobrazuje v průběhu hybridního algoritmu poloměr hejna. Ten je určen jako podíl maxima vzdáleností agentů od g_{best} a maximálního rozměru výpočetní domény. V průběhu závěrečné Nelder-Mead metody pak zobrazuje v logaritmickém měřítku rozdíl nejlepšího a nejhoršího bodu simplexu: $\log|f(\mathbf{x_{n+1}}) - f(\mathbf{x_1})|$.

Panel Iterations odměřuje iterace do maximálního počtu iterací. Nejprve pro hybridní metodu, která končí dosažením stanoveného počtu iterací a následně pak

pro Nelder-Mead metodu, která končí, pokud je dosaženo požadované meze konvergence, nebo pokud je dosaženo maximálního počtu iterací. V grafu v levé horní části okna je zobrazován průběh účelové funkce. Stupnice grafu je v logaritmickém měřítku.

V panelech ve spodní části okna jsou vypisovány informace o průběhu optimalizace: název účelové funkce, maximální počet iterací, typ stěny PSO, počet provedených iterací, počet provedených vyhodnocení účelové funkce, aktuální hodnota účelové funkce v bodě g_{best} (označená PSObest), aktuální hodnota v bodě g_{NMbest} (označená NMbest) a čas optimalizace. Tyto hodnoty jsou aktualizovány v průběhu optimalizace.

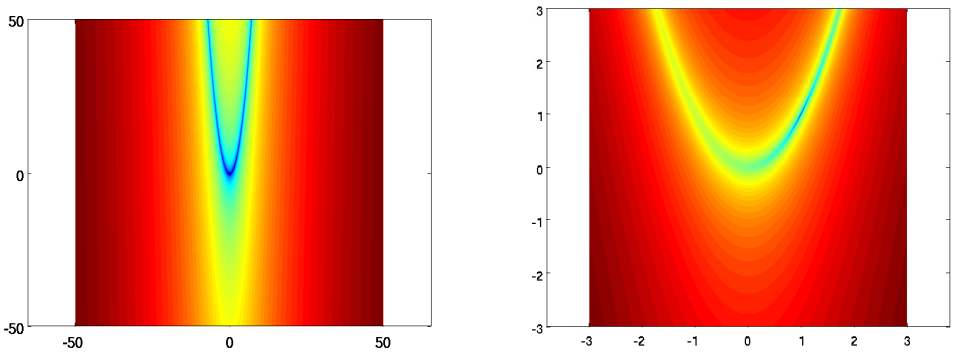
Program je možné kdykoli ukončit stisknutím tlačítka EXIT s tím, že aktuální výsledky budou zapsány do struktury res.

Kapitola 5

Testovací funkce

Program NM-PSOptimizer byl testován a porovnáván s ostatními optimalizačními algoritmy na několika funkcích, které jsou běžně používány pro testování optimalizačních metod [17]. Ve všech případech se jedná o spojité funkce.

Rosenbrockova funkce



(a) Rosenbrockova funkce (b) Detail globálního minima

Obrázek 5.1: Dvojrozměrná Rosenbrockova funkce.

První testovací funkcí byla Rosenbrockova funkce. Tato funkce je velmi obvyklou testovací funkcí pro optimalizační algoritmy. Dvojrozměrná varianta funkce je zobrazena na obrázku 5.1. Barevná škála je pro lepší názornost v logaritmickém měřítku. Funkční předpis je dán vztahem (5.1)

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} [(100x_{i+1} - x_i^2)^2 + (1 - x_i)^2]. \tag{5.1}$$

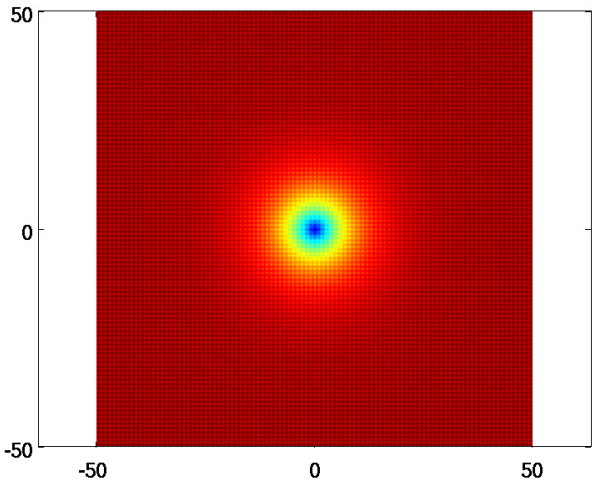
Rosenbrockova funkce nabývá globálního minima $f(\mathbf{x}) = 0$ v bodě $x_i = 1$, $i = 1, ..., n$. Pro dimenzi $n \geq 4$ obsahuje další lokální minima, jejichž počet a polohy

závisí na dimenzi prostoru [18]. Pro $n \geq 4$ již tedy Rosenbrockova funkce není unimodální funkcí.

Jak je vidět z obrázku 5.1, má rosenbrockova funkce charakter dlouhého úzkého údolí. Spád funkce v údolí je malý ve srovnání se spádem ve směrech kolmých k údolí. V tom spočívá náročnost hledání minima Rosenbrockovy funkce. Optimalizátor poměrně rychle najde údolí, pak ale kvůli malému spádu dlouho prochází údolím, než nalezne globální minimum.

Ackleyho funkce

Druhou použitou testovací funkcí je Ackleyho funkce. Její dvojrozměrná varianta je znázorněna na obrázku 5.2. Funkce je definována předpisem:



Obrázek 5.2: Dvourozměrná Ackleyho funkce.

$$f(\mathbf{x}) = -20\, e^{-\frac{1}{5}\cdot\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)} + 20 + e, \tag{5.2}$$

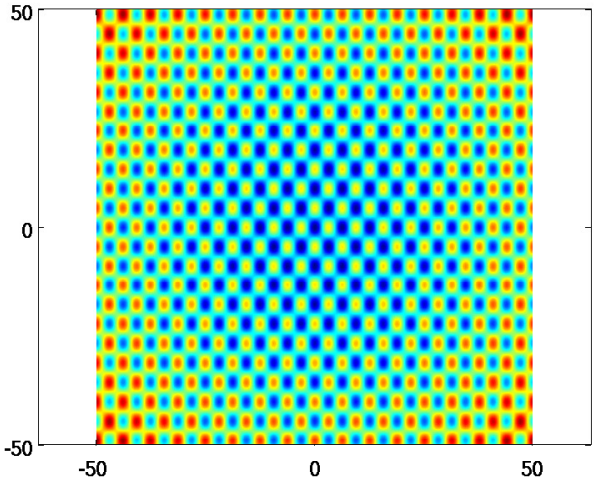
kde n je dimenze prostoru.

Ačkoli to není z obrázku 5.2 dobře patrné, obsahuje Ackleyho funkce velké množství lokálních minim. Má však výrazné globální minimum $f(\mathbf{x}) = 0$ v bodě $\mathbf{x} = \mathbf{0}$.

Griewank funkce

Další testovací funkcí je funkce Griewank. Je dána funkčním předpisem:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \tag{5.3}$$

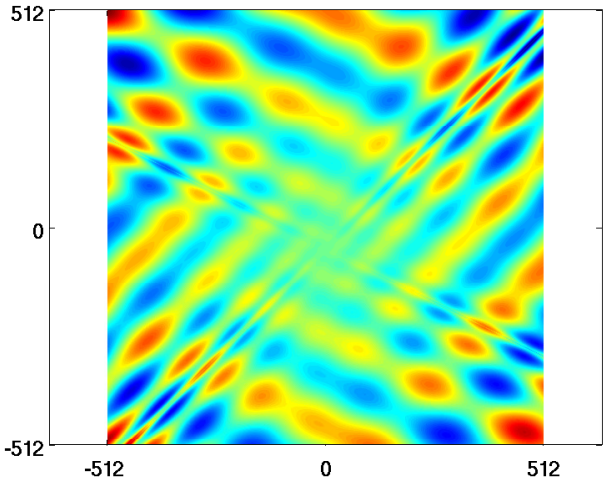


Obrázek 5.3: Funkce Griewank.

Funkce nabývá globálního minima $f(\mathbf{x}) = 0$ v bodě $\mathbf{x} = \mathbf{0}$.

Obdobně jako předchozí funkce obsahuje i funkce Griewank velké množství lokálních minim. Rozdíl je v tom, že v tomto případě mají lokální minima velmi blízkou hodnotu k minimu globálnímu. Pro optimalizátor je pak obtížné globální minimum najít. Obtížnost hledání globálního minima této funkce výrazně roste se zvyšující se dimenzí.

Funkce Eggholder



Obrázek 5.4: Funkce Eggholder.

Poslední použitou testovací funkcí je funkce Eggholder. Je znázorněna na obrázku 5.4. Funkční předpis je:

$$f(x,y) = -(y+47)\,\sin\left(\sqrt{\left|y+\frac{x}{2}+47\right|}\right) - x\,\sin\left(\sqrt{|x-y-47|}\right). \tag{5.4}$$

Funkce se obvykle definuje na množině $< -512; 512 > \times < -512; 512 >$. Globální minimum $f(\mathbf{x}) = -959.6407$ se nachází v bodě $\mathbf{x} = (512, 404.2319)$. Globální minimum se tedy nachází na hranici výpočetní domény vedle lokálního minima s velmi blízkou hodnotou účelové funkce.

Kapitola 6

Porovnání Nelder-Mead-PSO, Nelder-Mead a PSO metod

V této části jsou prezentovány výsledky porovnání hybridního algoritmu se samostatnými Nelder-Mead metodou a PSO algoritmem.

6.1 Metodika testování

Optimalizační metody byly testovány na uvedených testovacích funkcích. Funkce Griewank byla použita v dvojrozměrné i čtyřrozměrné variantě. S výjimkou funkce Eggholder probíhalo testování na kartézském součinu intervalů $< -50; 50 >$ potřebné dimenze. Funkce Eggholder byla použita na množině, která je pro ni standardní $< -512; 512 > \times < -512; 512 >$. Použité testovací funkce spolu s dalšími parametry optimalizace jsou v tabulce 6.1.

Číslo fce.	Popis funkce	Podmínka nalezení globálního minima
1	Griewank 2D, odrazná stěna	$f(x) < 0.0001$
2	Griewank 4D, odrazná stěna	$f(x) < 0.0001$
3	Ackley 4D, odrazná stěna	$f(x) < 0.001$
4	Rosenbrock 10D, odrazná stěna	$f(x) < 0.001$
5	Eggholder 2D, odrazná stěna	$f(x) < -959.65$

Tabulka 6.1: Označení použitých testovacích funkcí s příslušnými parametry.

Každý optimalizační algoritmus byl vždy pro danou testovaci funkci s danými parametry spuštěn 100×. Maximální počet iterací byl nastaven na 1500 pro NM-PSO a PSO a na 7500 pro Nelder-Mead metodu. V průběhu jedné iterace hybridního algoritmu totiž probíhaly 4 iterace Nelder-Mead a na závěrečné zpřesnění dalších maximálně 1500 iterací. Jako úspěšný pokus byl přijat takový, který splňuje podmínku nalezení globálního minima uvedenou pro každou funkci v tabulce 6.1. Podmínka je pro každou funkci jiná, aby byly zohledněny její vlastnosti.

Sledované ukazatele byly průměrovány podle svého charakteru buď přes všechny pokusy, nebo jen přes úspěšné pokusy. Nejvýznamnějším parametrem je úspěšnost optimalizačního algoritmu pro daný problém. Druhýn sledovaným parametrem je čas běhu optimalizace, který byl průměrován přes všechny pokusy t_{tot}. Dalším významným parametrem je počet vyčíslení účelové funkce v průběhu optimalizace f_{tot}. Jeho hodnota byla průměrována přes všechny pokusy. Posledním parametrem je počet vyčíslení účelové funkce potřebný pro konvergenci do globálního minima f_{suc}, ten byl průměrován pouze přes úspěšné pokusy.

6.2 Porovnání výsledků

Výsledky testování jednotlivých optimalizačních metod jsou uvedeny v tabulkách 6.2, 6.3. Z hlediska úspěšnosti nalezení globálního minima lze říci, že výsledky hybridního algoritmu jsou lepší, nebo minimálně srovnatelné, než u obou nezávislých algoritmů. Například pro Rosenbrockovu funkci nebyl PSO algoritmus schopen ve stanoveném počtu iterací nalézt dostatečně přesně globální minimum, jeho úspěšnost je tudíž na této funkci mizivá. Oproti tomu Nelder-Mead metoda konvergovala do minima poměrně rychle, v některých případech však byla zachycena v lokálním minimu. V hybridním algoritmu, kde se obě metody vzájemně doplňují, byla úspěšnost výrazně vyšší. Nelder-Mead metoda rychle konvergovala do minima a PSO zajišťovalo překonání lokálních minim. Na ostatních funkcích které mají velké množství lokálních minim byla úspěšnost Nelder-mead metody prakticky nulová. Globální prohledání a hrubá lokalizace globálního minima je v těchto případech úkolem PSO algoritmu. Přínos Nelder-Mead metody v hybridním algoritmu spočívá u těchto funkcí v rychlejším zpřesnění globálního minima. To je dobře patrné z porovnání parametru f_{suc}, zejména pro Ackleyho funkci a funkci Eggholder, zrychlení oproti PSO je patrné i v případě Rosenbrockovy funkce. Provnání průběhu optimalizace Rosenbrockovy funkce jednotlivými algoritmy je znázorněno na obrázku 6.1. Obrázek ukazuje poměrně rychlé nalezení minima hybridním algoritmem. Následně už se pouze čeká, až proběhne předepsaný počet iterací. Zde je nutné poukázat na to, že se nepodařilo vymyslet žádný obecný způsob detekce dosažení globálního mi-

nima a využít jej k ukončení hybridního algoritmu. Uživatel má alespoň možnost sledovat hodnotu nalezeného minima a v případě dostatečného poklesu ukončit optimalizaci ručně. Z obrázku 6.1 je dále patrná pomalá konvergence PSO v případě Rosenbrockovy funkce, kdy algoritmus prohledává úzké sedlo.

	PSO				Nelder-Mead			
fce.	usp. [%]	t_{tot} [s]	f_{tot} [-]	f_{suc} [-]	usp. [%]	t_{tot} [s]	f_{tot} [-]	f_{suc} [-]
1	66	35	36027	18871	1	0.3	77	38
2	7	38	42033	30861	0	0.8	243	-
3	100	36	42033	25811	0	1	283	-
4	1	37	60501	58971	74	61.3	3818	3239
5	68	31.4	36027	13201	0	0.3	104	-

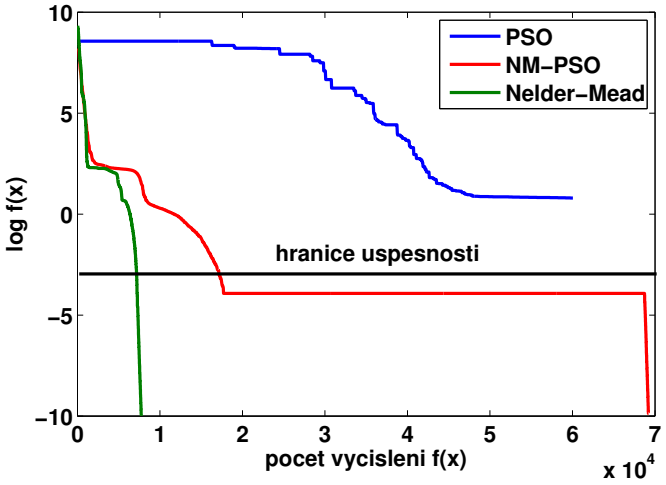
Tabulka 6.2: Porovnání optimalizačních algoritmů.

	NM-PSO			
fce.	usp. [%]	t_{tot} [s]	f_{tot} [-]	f_{suc} [-]
1	69	37	36518	15236
2	4	39	44870	30908
3	100	41	47391	6325
4	96	70	70170	34588
5	90	33	37326	3645

Tabulka 6.3: Porovnání optimalizačních algoritmů.

Z hlediska celkového počtu vyčíslení účelové funkce není hybridní algoritmus o mnoho náročnější, než samotné PSO. V případě Rosenbrockovy funkce byl nárůst přibližně 12 procent. Nárůst je však vykoupen větší úspěšností algoritmu. U ostatních testovaných funkcí byl nárůst menší.

V rámci dalšího vývoje by bylo vhodné nalézt ukončovací kritérium pro hybridní algoritmus, které rozpozná konvergenci do globálního minima. Jako první námět se nabízí použít rozdíl účelové funkce mezi nejhorším a nejlepším bodem populace.



Obrázek 6.1: Porovnání průběhu optimalizace Rosenbrockovy funkce jednotlivými metodami.

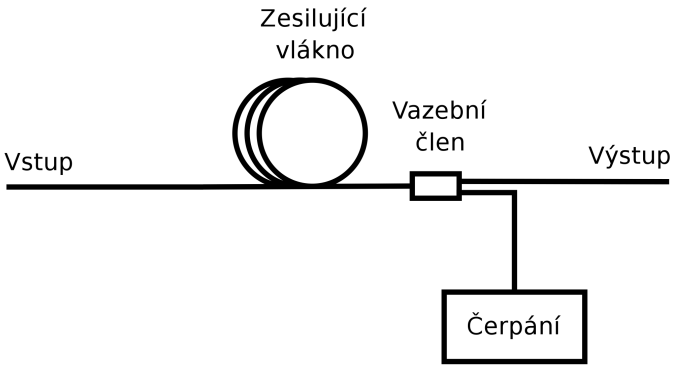
Tento rozdíl však zůstává v průběhu optimalizace na rozdíl od Nelder-Mead poměrně velký. Dále by bylo užitečné implementovat kvaziabsorbční stěnu, která nebude pohlcovat celou rychlost agentů, ale například desetinu rychlosti odrazí zpět. Při použití absorbční stěny se totiž objevoval problém, že částice ztratily v jedné dimenzi veškerou rychlost a zachytily se v té dimenzi na hranici výpočetní domény. Reflexní stěna oproti tomu ztěžuje konvergenci do minima, které se nachází na hranici, jak lze pozorovat na příkladu funkce eggholder.

Kapitola 7

Optimalizace zisku ramanovského vláknového zesilovače

Jako praktický příklad použití vytvořeného optimalizačního nástroje byla zvolena úloha optimalizace zisku ramanovského vláknového zesilovače.

Ramanovské vláknové zesilovače jsou optické zesilovače používané v optických komunikačních systémech [19] . Jejich fyzikálním principem je stimulovaný Ramanův rozptyl. Ramanův rozptyl je nelineární optický jev a není pevně vázán na atomární či molekulární energetické hladiny. Díky tomu umožňují ramanovské zesilovače při vhodném čerpání pokrýt i spektrální oblasti, které jsou nedosažitelné s použitím např. erbiem dopovaných vláknovách zesilovačů. K Ramanovu rozptylu dochází v každém optickém vlákně. Obzvláště účinný je díky užšímu jádru a vyšší dotaci germánia ve vláknech používaných pro kompenzaci chromatické disperze, tzv. DCF. Lze tak zkonstruovat víceúčelové zařízení, které bude zároveň kompenzovat chromatickou disperzi trasy a zesilovat procházející signál. Schematické uspořádání uvažovaného ramanovského zesilovače je na obrázku 7.1. Zesilovač se skládá ze zesilovacího



Obrázek 7.1: Uspořádání ramanovského zesilovače.

vlákna typu DCF, vazebního členu pro navázání čerpání a čerpacích laserových diod. Případně může být na vstupu a na výstupu doplněn optickými izolátory. Zesilovač uvedený na obrázku 7.1 je tzv. zpětně čerpaný zesilovač, kdy čerpací výkon je zaveden v protisměru k šířícím se signálům. Vhodně zvoleným čerpáním, jak z hlediska vlnových délek čerpacích diod, tak z hlediska jejich výkonů, lze docílit vyrovnaného zisku zesilovače v poměrně širokém spektrálním pásmu [20]. Návrh čerpání je podstatou optimalizační úlohy.

7.1 Formulace optimalizační úlohy

Předpokládejme, že do zesilovače vstupuje 10 signálů o výkonu 0.1 mW, jejichž vlnové délky odpovídají rastru DWDM multiplexu. Požadujeme, aby na výstupu zesilovače měly všechny signály výstupní výkon 3 mW. Úkolem je navrhnout vlnové délky a výkony čerpacích diod tak, aby byl splněn požadavek na výstupní výkon. Parametry vstupních signálů jsou přehledně uvedeny v tabulce 7.1.

Vlnová délka λ [nm]	Vstupní výkon P_{in} [mW]	Požadovaný výstupní výkon $P_{\text{out}}^{\text{opt}}$ [mW]
1548.5	0.1	3
1551.8	0.1	3
1554.8	0.1	3
1558.2	0.1	3
1561.4	0.1	3
1564.7	0.1	3
1567.9	0.1	3
1571.2	0.1	3
1574.5	0.1	3
1577.0	0.1	3

Tabulka 7.1: Parametry zesilovaných signálů.

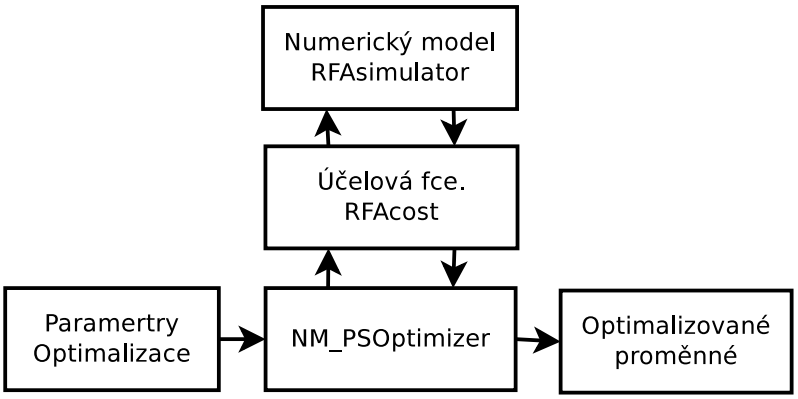
Učelová funkce byla zvolena nejjednodušším možným způsobem, jako součet kvadrátů rozdílů výstupního výkonu $P_{i,\text{out}}$ a požadovaného výstupního výkonu $P_{i,\text{out}}^{\text{opt}}$ pro jednotlivé vlnové délky:

$$f(P_p, \lambda) = 10^6 \cdot \sum_{i=1}^{n} (P_{i,\mathrm{out}}(P_p, \lambda) - P_{i,\mathrm{out}}^{\mathrm{opt}})^2. \tag{7.1}$$

Z definice účelové funkce je patrné, že v globálním minimu nabývá hodnoty 0. Prostřednictvím výstupního výkonu signálů závisí účelová funkce na parametrech čerpání. Meze optimalizace byly nastaveny tak, aby odpovídaly vlnovým délkám a výkonům dostupných čerpacích diod. Čerpací vlnové délky byly omezeny na interval $\lambda_i \in <1420\,\mathrm{nm}; 1500\,\mathrm{nm}>$ a čerpací výkony na interval $P_{i,p} \in <0\,\mathrm{mW}; 150\,\mathrm{mW}>$.

Zesilující prostředí bylo tvořeno 13 km DCF vlákna, které tvoří modul pro kompenzaci disperze EWBDK-C společnosti OFS. Jako numerický model ramanovského zesilovače byl použit program RFAsimulator, jehož implementace byla předmětem předchozí práce autora [21] . V rámci této práce byla také experimentálně ověřena přesnost simulačního programu pro uvažované zesilující vlákno.

Výměna informací mezi jednotlivými programy účastnícími se na optimalizačním procesu je schematicky naznačena na obrázku 7.2. Optimalizačnímu programu



Obrázek 7.2: Výměna informací mezi jednotlivými programy v rámci optimalizace.

jsou nejprve předány parametry jako je omezení výpočetní oblasti a použitá účelová funkce. Optimalizační program pak předává účelové funkci testované parametry čerpání a ta při svém vyhodnocení spouští numerický model ramanovského zesilovače RFAsimulator. Výsledkem numerického modelu jsou výstupní výkony sledovaných signálů, na jejichž základě je vyčíslena účelová funkce (7.1). Získaná hodnota účelové funkce je předána zpět optimalizačnímu programu. Výstupem optimalizačního programu jsou prametry čerpání, pro keré nabývala účelová funkce minimální nalezené hodnoty.

7.2 Výsledky optimalizace

Doposud neřešeným parametrem zůstal počet čerpacích vlnových délek. Je pochopitelné, že čím více čerpacích diod bude použito, tím vyrovnanější zisk bude možné dosáhnout. Volným parametrem jsou jak vlnová délka, tak čerpací výkon v příslušných mezích. Nejprve byla optimalizace prováděna orientačně s 500 iteracemi. Pro 5 čerpacích signálů nebyly výsledky uspokojivé. Při 6 čerpacích signálech bylo dosaženo poměrně slušného výsledku. Následně byla spuštěna optimalizace jemně ve 3000 iteracích pro 6 čerpacích vlnových délek. Optimalizace tedy probíhala ve dvanácti dimenzích. Výsledek dopadl poměrně překvapivě.

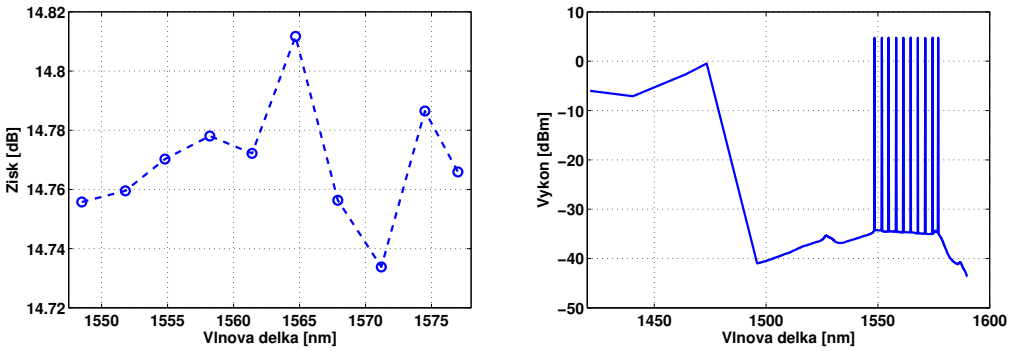
Optimalizované parametry čerpání jsou uvedeny v tabulce 7.2. Výsledná hodnota účelové funkce (7.1) byla $f(P_p, \lambda) = 0.0017$.

Vlnová délka čerpání λ [nm]	Čerpací výkon P_p [mW]
1421.0	100.1
1421.3	150.0
1440.4	112.3
1464.0	132.0
1473.4	118.4
1495.0	0

Tabulka 7.2: Optimalizované parametry čerpání ramanovského zesilovače.

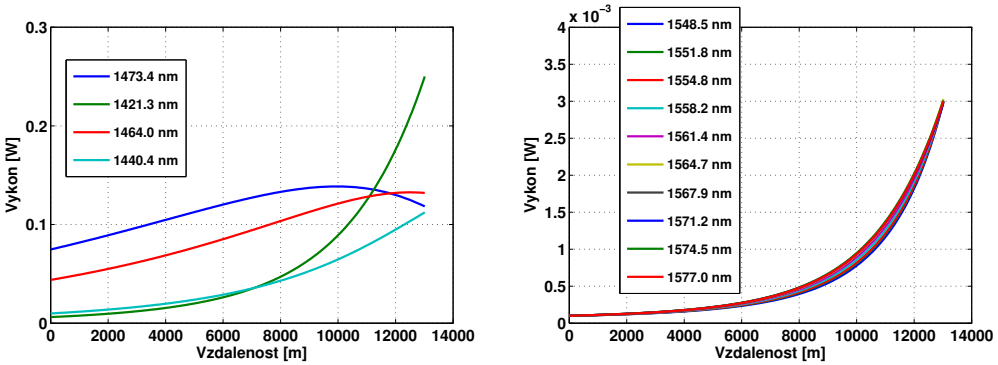
Z výsledků vyplývá, že jeden z čerpacích signálů má nulový výkon. První dvě vlnové délky čerpání jsou navíc velmi blízko sebe. Zde je jasně patrné, že optimalizátor narazil na hranici výpočetní domény, což kompenzoval tím, že posunul jiný signál do bezprostřední blízkosti. Pokud budou tyto dva čerpací signály sloučeny do jednoho se součtovým výkonem $P_{p_{1421.3}} = 250.1\,\mathrm{mW}$, nedojde k výraznému zhoršení hodnoty účelové funkce. Hodnota účelové funkce pro sloučené čerpací signály je: $f(P_p, \lambda) = 0.0019$. Zhoršení oproti původní hodnotě je zanedbatelné. Budou-li k dispozici čerpací diody s dostatečným výkonem, budou pro čerpání zesilovače stačit pouze 4 vlnové délky, což je výhodné jak z hlediska ceny, tak i spolehlivosti zařízení.

Celkový zisk je společně s výstupním spektrem optimalizovaného zesilovače zobrazen na obrázku 7.3. Zisk zesilovače se pohybuje v rozmezí
$\Delta G \approx 0.1\,\mathrm{dB}$, což odpovídá rozdílu $\Delta P \approx 0.05\,\mathrm{mW}$ ve výstupním výkonu. Na obrázku 7.4 je zobrazeno podélné rozložení čerpání a signálů v daném zesilovači.



(a) Zisk zesilovače (b) Výstupní spektrum

Obrázek 7.3: Zisk a výstupní spektrum optimalizovaného ramanovského zesilovače.



(a) Rozložení čerpání (b) Rozložení signálů

Obrázek 7.4: Podélné rozložení čerpání a signálů v optimalizovaném ramanovském zesilovači.

Z obrázků je patrné, že optimalizací bylo velmi úspěšně dosaženo požadovaného cíle.

Závěr

V rámci diplomové práce byla prostudována problematika Nelder-Mead simplexové metody a jejích základních vlastností. Nelder-Mead metoda byla také implementována. Dále byl prostudován PSO algoritmus a hybridní Nelder-Mead-PSO algoritmus. Byly navrženy úpravy hybridního algoritmu zlepšující kooperaci obou metod. Upravený hybridní algoritmus byl implementován v podobě programu NM-PSOptimizer. Efektivita hybridního algoritmu byla porovnána s efektivitou samostatných Nelder-Mead a PSO algoritmů pomocí testovacích funkcí standardně používaných v problematice optimalizace. Ukázalo se, že za cenu 12% nárůstu počtu vyčíslení účelové funkce oproti PSO, dosahuje hybridní algoritmus větší úspěšnosti nalezení globálního minima, než PSO. Hlavním nedostatkem je, že se nepodařilo nalézt vhodné kritérium, které by spolehlivě detekovalo konvergenci do globálního minima a umožnilo tak hybridní algoritmus automaticky ukončit. Nalezení takového kritéria by výrazně zvýšilo efektivitu hybridního algoritmu.

Vytvořený program byl následně úspěšně použit pro návrh a optimalizaci čerpání ramanovského vláknového zesilovače, coby příkladu z technické praxe. Všechny body zadání diplomové práce byly splněny.

Literatura

[1] Z. Dostál, P. Bermelijski: *Metody optimalizace*, VŠB-TUO, (2012)

[2] J. A. Nelder, R. Mead: *A simplex method for function minimization*, The Computer Journal 7, 308 (1966)

[3] J. Pytlíček: *Lineární algebra a geometrie*, ČVUT, (2007)

[4] J. C. Lagais, J. A. Reeds, M. H. Wright, P. E. Wright : *Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions*, SIAM Journal on Optimization 9, 112 (1998)

[5] L. Nazareth, P. Tseng: *Gildyng the Lily: A variant of the Nelder-Mead Algorithm Based on Golden-Section Search*, Computational Optimization and Applications 22, 133 (2002)

[6] K. I. M. Mckinnon : *Convergence of the Nelder-Mead simplex method to a nonstationary point*, SIAM Journal on Optimization 9, 148 (1998)

[7] F. Gao, L. Han: *Implementing the Nelder-Mead simplex algorithm with adaptive parameters*, Computational Optimization and Applications 51, 259 (2012)

[8] M. Luersen, R. Riche: *Globalized Nelder–Mead method for engineering optimization*, Computers & Structures 82, 2251 (2004)

[9] J. Kennedy, R. Eberhart: *Particle Swarm Optimization*, IEEE Transaction Int. Neural Networks 4, 1942 (1995)

[10] K. E. Parsopoulos, N. M. Vrahatis: *Recent approaches to global optimization problems through Particle Swarm Optimization*, IEEE Transaction Int. Neural Networks 4, 1942 (1995)

[11] N. Jin, Y. Rahmat-Samii: *Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations*, IEEE Transactions on Antenas and Propagation 55, 556 (2007)

[12] J. Robinson, Y. Rahmat-Samii: *Particle Swarm Optimization in Electromagnetics*, IEEE Transactions on Antenas and Propagation 52, 397 (2004)

[13] S.S. Fan, E. Zahara: *A hybrid simplex search and particle swarm optimization for unconstrained optimization*, European Journal of Operational Research 181, 527 (2007)

[14] A. Liu, M. Yang: *A new Hybrid Nelder-Mead Particle Swarm Optimization for Coordination Optimization of Directional Overcurrent Relays*, Math. Problems in Engineering 2012, ID 46047 (2012)

[15] M. Čapek, P. Hazdra: *PSO optimalizace v Matlabu*, Technical Computing Prague, (2008)

[16] M. Čapek, P. Hazdra: *PSOptimizer*, elmag.org, ČVUT, (2013),
Dostupné z:
http://www.old.elmag.org/doku.php/wiki:user:capek:psoptimizer

[17] Testovací funkce In: I. Zelinka, Z. Oplatková, M. Šeda, P. Ošmera, F. Včelař: *Evoluční výpočetní techniky*, BEN 281 (2009) ISBN: 978-80-7300-218-3

[18] Y. W. Shang, Y. H Qiu: *A Note on the Extended Rosenbrock Function*, Evolutionary Computation 14, 119 (2006)

[19] M. N. Islam: *Raman Amplifiers for Telecommunications 1*, Springer-Verlag, (2004) ISBN: 0-387-00751-2

[20] M. Karásek, J. Kaňka, P. Honzátko, P. Peterka: *Time-domain simulation of power transients in Raman fibre amplifiers*, International Journal of Numerical Modelling: Electronic Networks, Devices and Fields 17, 165 (2004)

[21] P. Koška: *Přechodové jevy v ramanovských vláknových zesilovačích*, ČVUT, (2008)