

# **Adaptivní diferenciální evoluce v C++**

Bc. Radovan Fuchs

---

Diplomová práce  
2014



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
akademický rok: 2013/2014

## **ZADÁNÍ DIPLOMOVÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Radovan Fuchs**  
Osobní číslo: **A12387**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Informační technologie**  
Forma studia: **prezenční**

Téma práce: **Adaptivní Diferenciální evoluce v prostředí C/C++**

Zásady pro vypracování:

1. Vypracujte literární rešerši na dané téma.
2. Popište odlišnosti mezi kanonickou verzí a adaptivními verzemi algoritmu Diferenciální Evoluce.
3. Naprogramujte zvolenou verzi adaptivní Diferenciální Evoluce v prostředí C/C++.
4. Otestujte algoritmus na sadě vybraných testovacích funkcí.
5. Výsledky testování přehledně graficky a tabulkově zobrazte.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: tištěná/elektronická

Seznam odborné literatury:

1. ZELINKA, Ivan. Umělá inteligence v problémech globální optimalizace. BEN, 2002, 190 s. ISBN 80-7300-069-5.
2. ZELINKA, Ivan. Evoluční výpočetní techniky: principy a aplikace. 1. vyd. Praha: BEN – technická literatura, 2009, 534 s. ISBN 978-80-7300-218-3.
3. DE JONG, Kenneth A. Evolutionary computation: a unified approach. Cambridge: MIT Press, 2006, ix, 256 s. ISBN 02-620-4194-4.
4. MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: Umělá inteligence, Academia, 1993, ISBN 80-200-0496-3.
5. MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: Umělá inteligence 4., Academia, 2003, ISBN 80-200-1044-0.
6. ZELINKA, Ivan, Zuzana OPLATKOVÁ a Roman ŠENKERÍK. Aplikace umělé inteligence. Vyd. 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2010, 151 s. ISBN 978-80-7318-898-6.
7. PRICE, Kenneth V, Rainer M STORN a Jouni A LAMPINEN. Differential evolution: a practical approach to global optimization [online]. Berlin: Springer, 2005.
8. Handbook of Optimization: From Classical to Modern Approach. 2013. vyd. Editor Ivan Zelinka, Václav Snášel, Ajith Abraham. Berlin: Springer, 2013, xii, 1100 s. Intelligent systems reference library, 38. ISBN 978-3-642-30503-0.

Vedoucí diplomové práce:

doc. Ing. Roman Šenkeřík, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání diplomové práce:

21. února 2014

Termín odevzdání diplomové práce:

20. května 2014

Ve Zlíně dne 21. února 2014

prof. Ing. Vladimír Vašek, CSc.  
děkan



doc. Mgr. Roman Jašek, Ph.D.  
ředitel ústavu

## ABSTRAKT

Cílem této práce je vytvoření a testování základních adaptivních variant optimalizačního algoritmu diferenciální evoluce. Teoretická část práce se zabývá historií, úvodem do tématu evoluce, popisem základní diferenciální evoluce (DE) a popisem adaptivních variant DE. V závěru teoretické části jsou popsány metody testování a vyhodnocování a také jsou zde vypsány použité testovací funkce.

V praktické části práce je podrobně popsán program vytvořený v C++ a jsou provedeny testy na všech uvedených funkcích.

Klíčová slova: diferenciální evoluce, jDE, JADE, SADE, EPSDE

## ABSTRACT

The aim of this thesis is to develop and test the basic variant of adaptive optimization algorithm of differential evolution. The theoretical part deals with the history, introduction to the subject of evolution, describing the basic differential evolution (DE) a description of the adaptive DE variants. At the end of the theoretical section describes the methods for testing and evaluation and also are listed, used test functions.

In the practical part of the work is detailed in a program written in C++ and are tested for all of these functions.

Keywords: differential evolution, jDE, JADE, SADE, EPSDE

Velice děkuji doc. Ing. Romanu Šenkeříkovi, Ph.D. za odborné vedení diplomové práce, za ochotnou spolupráci při řešení problémů a za poskytnutí mnoha cenných připomínek a rad. Také bych rád poděkoval svým rodičům za podporu během celé doby mého studia.

**Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

**OBSAH**

<b>ÚVOD.....</b>	<b>9</b>
<b>I TEORETICKÁ ČÁST.....</b>	<b>10</b>
<b>1 HISTORIE .....</b>	<b>11</b>
<b>2 EVOLUČNÍ ALGORITMY .....</b>	<b>12</b>
2.1 ZÁKLADNÍ POJMY .....	12
2.1.1 Jedinec .....	12
2.1.2 Populace .....	12
2.1.3 Účelová funkce.....	13
2.1.4 Penalizace.....	13
2.2 CYKLUS EVOLUCE .....	14
2.3 APLIKACE EVOLUČNÍCH ALGORITMŮ .....	15
<b>3 DIFERENCIÁLNÍ EVOLUCE.....</b>	<b>17</b>
3.1 ZÁKLADNÍ DEFINICE .....	17
3.2 PARAMETRY .....	17
3.3 MUTACE.....	19
3.4 KŘÍŽENÍ.....	19
3.4.1 Binomické křížení .....	20
3.4.2 Exponenciální křížení.....	20
3.5 STAGNACE.....	21
3.6 PRINCIP ČINNOSTI.....	22
3.7 ADAPTIVNÍ MODIFIKACE .....	25
3.7.1 jDE .....	25
3.7.2 JADE .....	26
3.7.3 SaDE .....	28
3.7.4 EPSDE.....	31
3.8 UKONČOVACÍ PODMÍNKA .....	32
<b>4 TESTOVÁNÍ .....</b>	<b>33</b>
4.1 TESTOVACÍ FUNKCE .....	33
4.2 VYHODNOCENÍ.....	37
<b>II PRAKTICKÁ ČÁST .....</b>	<b>38</b>
<b>5 APLIKACE ADAPTIVNÍ DE .....</b>	<b>39</b>
5.1 POPIS IMPLEMENTACE .....	43
5.1.1 Třída DEMain .....	44
5.1.2 Třídy adaptivních modifikací .....	45
5.1.3 Možnosti rozšíření.....	45
5.2 VYHODNOCENÍ.....	46
<b>6 TESTOVÁNÍ .....</b>	<b>47</b>

6.1	FUNKCE SPHERE.....	48
6.2	ROTOVANÁ ELIPTICKÁ FUNKCE .....	49
6.3	ROTOVANÁ FUNKCE BENT CIGAR .....	51
6.4	ROTOVANÁ DISKOVÁ FUNKCE .....	52
6.5	FUNKCE ROZDÍLNÝCH MOCNIN.....	54
6.6	ROTOVANÁ ROSENBROCKOVA FUNKCE.....	55
6.7	ROTOVANÁ FUNKCE SCHAFFERS F7 .....	57
6.8	ROTOVANÁ ACKLEYHO FUNKCE .....	58
6.9	ROTOVANÁ GRIEWANKOVA FUNKCE.....	60
6.10	ROTOVANÁ RASTRIGINOVA FUNKCE .....	61
6.11	ROTOVANÁ SCHWEFELOVA FUNKCE .....	63
<b>7</b>	<b>VYHODNOCENÍ .....</b>	<b>65</b>
	<b>ZÁVĚR .....</b>	<b>66</b>
	<b>SEZNAM POUŽITÉ LITERATURY.....</b>	<b>67</b>
	<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>	<b>69</b>
	<b>SEZNAM OBRÁZKŮ .....</b>	<b>70</b>
	<b>SEZNAM TABULEK.....</b>	<b>71</b>
	<b>SEZNAM PŘÍLOH.....</b>	<b>72</b>



## ÚVOD

Již delší dobu je tématu evolučních algoritmů věnováno velké množství pozornosti. Jedná se o obor, jehož vývoj je velice pozitivně ovlivňován pokroky v oblasti výpočetní techniky. Tato postupná zlepšování zvyšují efektivnost a rozsah možností využití evolučních algoritmů. Obecně lze evoluční algoritmy aplikovat na každý problém, který můžeme popsat matematickým vztahem neboli účelovou funkcí. Úkolem evolučních algoritmů je optimalizace parametrů této funkce takovým způsobem, abychom dosáhli co nejlepšího výsledku.

Evolučních algoritmů existuje velké množství, kde každý má své specifické vlastnosti a je vhodný na odlišný typ problému. Není totiž možné vytvořit algoritmus, který by podával excelentní výkony na všech problémech. Jedním z těchto evolučních algoritmů je také diferenciální evoluce, využívající mutace a křížení k dosažení velice kvalitních výsledků.

Základním problémem diferenciální evoluce je způsob jakým pro daný problém nastavit řídicí parametry algoritmu, aby byly výsledky optimalizačního procesu pokud možno co nejlepší. Právě tímto problémem se zabývají v dnešní době vědci.

Při této snaze o vytvoření co nejlepšího algoritmu, schopného teoreticky fungovat nejlépe na daném problému bez nutnosti složitějšího nastavení parametrů vznikají tzv. adaptivní modifikace základního algoritmu diferenciální evoluce, schopné přizpůsobit se danému problému.

Cílem této práce je vytvořit plně funkční program schopný provádět výpočty variant diferenciální evoluce s různými základními adaptivními vylepšeními. Následně jsou tyto algoritmy otestovány na sadě vybraných testovacích funkcí.

## **I. TEORETICKÁ ČÁST**

## 1 HISTORIE

První zmínky o výpočetní technice, využívající principů známých z biologie, nacházíme koncem 50. let 20. století, viz práce pánů Bremermanna, Friedberga, Boxe a dalších. Avšak převážně díky nedostatečné výpočetní kapacitě tehdejší počítačové techniky se tato oblast vědy nedostává do povědomí širší veřejnosti po tři nadcházející desetiletí.[1]

Začátkem 60. let minulého století vzniká nový směr v rozvoji informatiky, tzv. genetické algoritmy. Tyto algoritmy vycházejí z Darwinovy teorie o vývoji druhu. Úspěšně se začaly používat k řešení optimalizačních problémů, kde dosahují kvalitních výsledků. Od dob, kdy jeden ze zakladatelů této problematiky John Holland popsal základní vlastnosti těchto algoritmů, prošla tato disciplína dlouholetým vývojem a v dnešní době se uplatňuje v mnoha optimalizačních algoritmech.[2]

Další vývoj, kterým optimalizační algoritmy prošly, nastal v 90. letech minulého století s vynálezem evolučního algoritmu (EA). Průkopníkem této problematiky se stal Kenneth Price, který se proslavil pokusem, vyřešit Chebychův polynom užitím evolučního algoritmu typu diferenční evoluce (DE) a Rainerem Stormem, který jej doplnil. Kritický zlom v evolučních algoritmech nastal v době, kdy Kenneth Price přišel s ideou použití rozdílového vektoru k zúžení populace vektoru. Tato původní idea vedla mezi Kennethem a Rainerem k rušným diskusím, nekonečným bádáním, počítání a simulacím, dokud nedosáhli lepších výsledků. Na základě jejich snahy je teď diferenční evoluce univerzálním a silným nástrojem pro řešení optimalizačních problémů. [3]

## 2 EVOLUČNÍ ALGORITMY

Základní princip evoluce v podobě známé z Darwinovy a Mendelovy teorie, spočívá v předávání rodičovského genomu (všech jeho genů a nekódující sekvence organismu) svým potomkům a následnému uvolnění jejich životního prostoru. Setkáváme se zde také s pojmem mutace, která ovlivňuje předávanou informaci a díky níž se generace vyvíjí. Jedinci, jejichž vlastnosti (parametry) nejsou pro aktuální životní prostředí vhodné, pak dle Darwinovy teorie vymírají. Jedinci schopní přizpůsobení pak přežívají. [2]

Algoritmy evoluční výpočetní techniky (EVT) z tohoto principu vycházejí. Oproti vývoji živých organismů však zde pracujeme s výpočetní problematikou. Jednotliví jedinci pak tedy nejsou živočichové, ale číselně vyjádřené parametry. Ty pak využitím různých základních metod evoluce (mutací a křížením) vyvíjíme a upravujeme do podoby, ve které v nové generaci dosáhnou optimálnějších výsledků. [2]

Algoritmů vycházejících z teorie evoluce je mnoho. Odlišují se především drobnými rozdíly v principu funkce, ale ať už se jedná o genetické algoritmy, evoluční strategie, rojení částic, SOMA (samo-organizující se migrační algoritmus) nebo diferenciální evoluci, základní myšlenka Darwinovy a Mendelovy teorie zůstává stále zachována.[3]

### 2.1 Základní pojmy

#### 2.1.1 Jedinec

Jedinec je numerický popis jednoho řešení daného problému. Každý jedinec je reprezentován hodnotami parametrů v definovaném počtu dimenzí a vhodností, představující výsledek ohodnocení zvolené účelové funkce vypočítaný z parametrů jedince. Samotného jedince si tedy můžeme představit jako číselný vektor parametrů s jeho ohodnocením. Každý jedinec je pak také omezen hranicemi prohledávaného prostoru. Při jeho vytváření bychom tedy měli používat korekci parametrů, které daný prostor překročí. Účelem evolučních algoritmů je najít jedince s ideálním ohodnocením účelové funkce.[3]

#### 2.1.2 Populace

Populace je pole jedinců dané generace včetně jejich parametrů. Může být znázorněna jako matice  $NP \times D$  (počet jedinců v populaci  $\times$  dimenze problému) s přidaným řádkem obsahujícím hodnotu účelové funkce jednotlivých jedinců (CV). Sloupce pak představují jednotlivé jedince.[4]

### 2.1.3 Účelová funkce

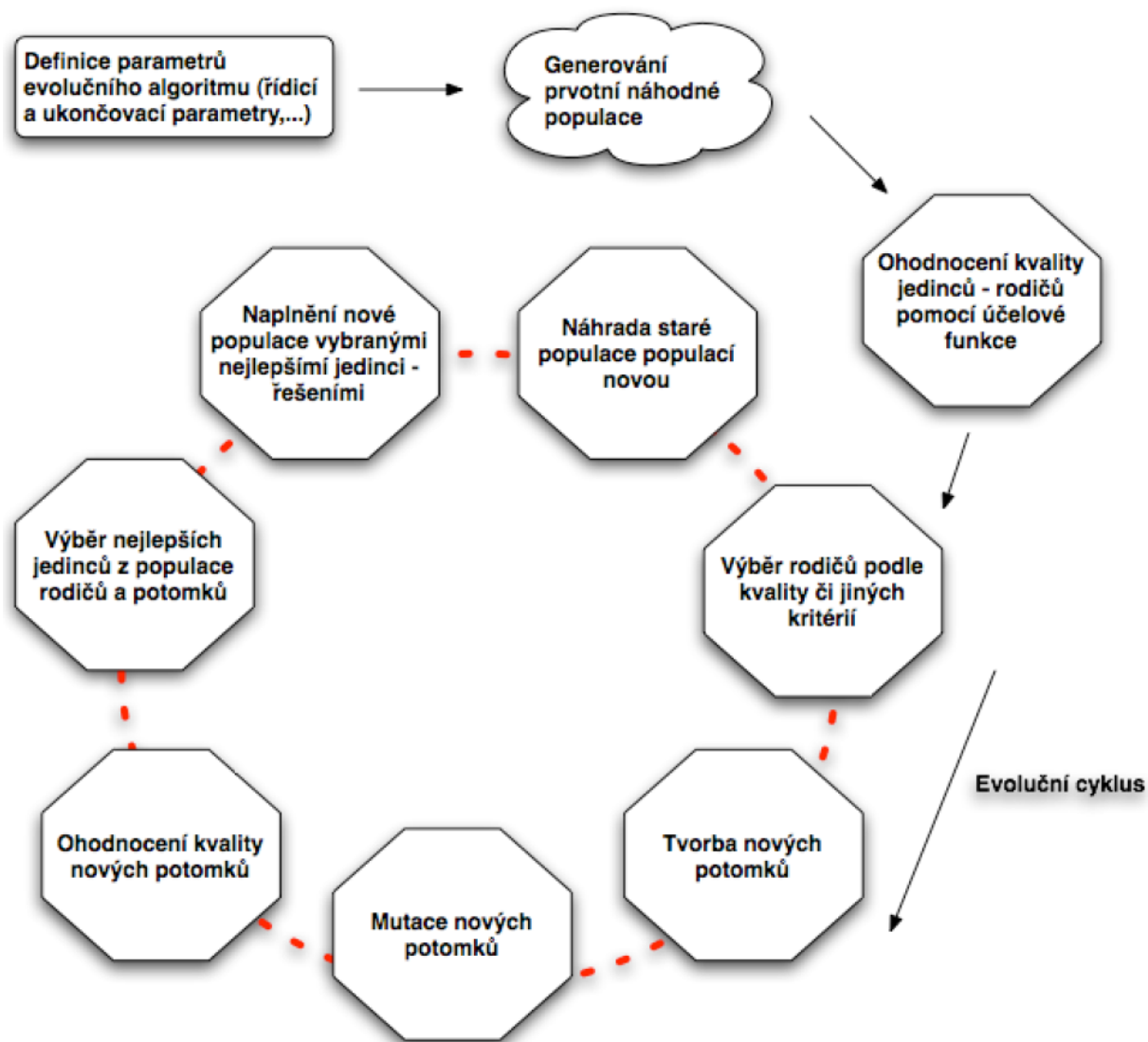
Účelová funkce hodnotí kvalitu jedince dle jeho parametrů. Označujeme ji jako  $f(x)$  nebo také  $f_{cost}(x)$  (podle anglického označení cost function). V rámci optimalizace nahlížíme na účelovou funkci jako na geometrický problém, ve kterém se snažíme nalézt minimum (maximum) dané funkce v  $N$  – dimenzionálním prostoru možných řešení. Počet dimenzí je dán počtem argumentů účelové funkce. [2]

Využití účelové funkce přesahuje rámec testovacích funkcí. V případě prediktivního řízení může účelová funkce například obsahovat veličiny akčního zásahu, výstupu a žádanou hodnotu, ke které se snažíme řídit výstupní veličinu. Vidíme tak, že možnost aplikace účelové funkce je velmi široká.

### 2.1.4 Penalizace

V průběhu evolučních výpočtů může dojít k situaci, že některý z jedinců se ocitne v zakázané oblasti. Neboli hodnota jednoho nebo více z jeho parametrů překročí vymezený interval. Penalizace může být řešena pomocí dvou různých řešení: hard constraints a soft constraints. V případě hard constraints jsou jedinci v penalizované oblasti zrušení a nahrazení novými. Při řešení soft constraints jedinci nejsou zrušeni, ale výpočet účelové funkce je v jejich případě znevýhodněn.[2]

## 2.2 Cyklus evoluce



Obr. 1: Obecný cyklus evoluce bez ukončovací podmínky [1]

### 1. Definice parametrů

V této části jsou definovány parametry pro řízení běhu vybraného algoritmu a také ukončovací podmínka

### 2. Generování prvotní náhodné populace

Podle počtu argumentů účelové funkce jsou vytvořeny vektory parametrů pro jednotlivé jedince v populaci

### 3. Ohodnocení kvality jedinců

Všichni jedinci jsou pomocí vybrané účelové funkce ohodnoceni a každému z nich je přiřazena hodnota účelové funkce

### 4. Výběr rodičů

Zde proběhne podle kvality jedinců, nebo jiných kritérií výběr rodičů

**5. Křížení**

Křížením rodičů jsou vytvořeni jejich potomci. Proces křížení se liší v závislosti na vybraném evolučním algoritmu

**6. Mutace**

Každý jedinec je zmutován, tedy je na něj aplikován vhodný náhodný proces.

**7. Ohodnocení nových potomků**

Noví jedinci jsou ohodnoceni stejně jako v kroku 3

**8. Výběr nejlepších jedinců z populace rodičů a potomků**

Podle hodnoty účelové funkce jsou vybráni nejlepší jedinci

**9. Naplnění nové populace**

Vybraní jedinci zaplní novou populaci

**10. Náhrada staré populace novou**

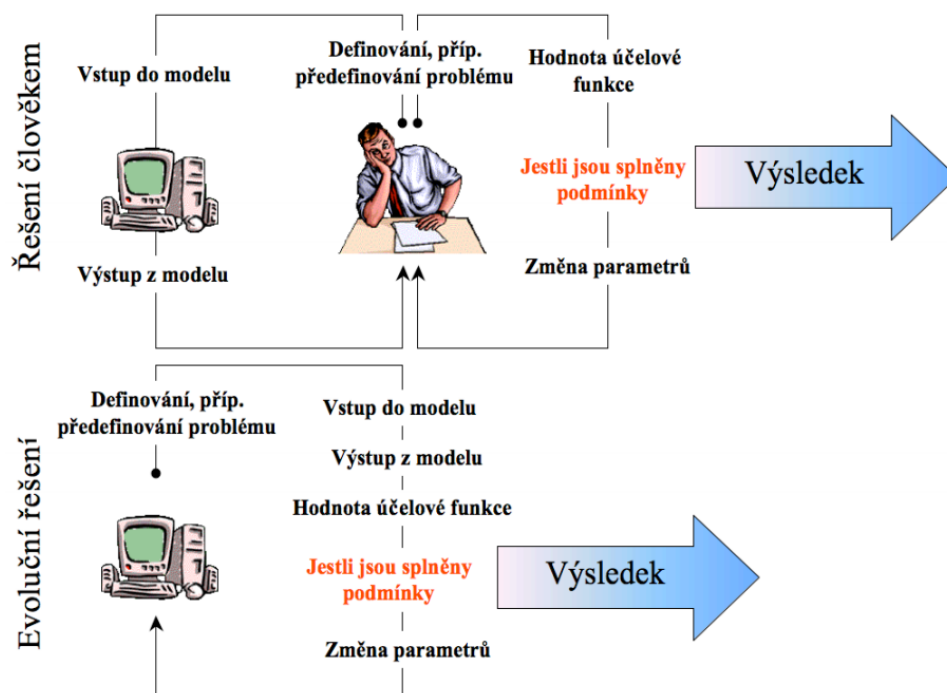
Stará populace je nahrazena novou populací

Kroky 4 – 10 se opakují do doby než je splněna ukončovací podmínka algoritmu (např. maximální počet generací). Uvedený princip evolučních algoritmů je obecný a v závislosti na konkrétním řešení se může více či méně odlišovat.[1]

**2.3 Aplikace evolučních algoritmů**

Tyto algoritmy nacházejí uplatnění v mnoha odvětvích lidské činnosti. Může se jednat o tvary různých výrobků, např. křidel letadla, nebo různé ekonomické problémy.

Evoluční algoritmy nejsou populární jen proto, že jsou moderní, ale také pro skutečnost že v případě vhodné aplikace jsou schopné nahradit člověka. Možnost tohoto využití je zobrazena na Obr. 2.[2]



Obr. 2: Porovnání řešení problémů pomocí evolučních algoritmů a člověkem [2]



### 3 DIFERENCIÁLNÍ EVOLUCE

#### 3.1 Základní definice

Diferenciální evoluci byla vytvořena v roce 1995 Kennethem V. Pricem a Dr. Rainerem Stormem. Algoritmus byl vyvinut pomocí úprav genetického žihání k řešení složitějších problémů. Původní binární reprezentace byla nahrazena dekadickou a logické operace vektorovými. Po přidání principu diferenciální mutace (viz 3.3) byly ovšem prvky genetického žihání uznány za nadbytečné a z diferenciální evoluce (dále jen DE) byly vypuštěny.

Z ostatních evolučních algoritmů se DE schematicky velmi podobá genetickým algoritmům a zároveň právě v porovnání s GA je velmi patrná rozdílnost obou postupů. Příkladem rozdílu obou postupů může být například odlišný počet rodičů, které algoritmy potřebují k vytvoření potomků. Zatímco GA využívá pouze dvou rodiče, DE vyžaduje čtyři.[5]

#### 3.2 Parametry

Parametry ovlivňující, jakým způsobem bude fungovat diferenciální evoluce, jsou následující:

- CR

Hodnotu CR využíváme v průběhu vytváření nového jedince, konkrétně při procesu křížení. Tento parametr určuje míru dědičnosti nově vzniklého jedince na aktuálním jedinci nebo na mutačním vektoru. V krajní hodnotě, pokud je CR rovno 0, bude potomek kopií aktuálního jedince. V opačném případě, když se hodnota CR rovná 1, nový potomek bude kopií mutačního vektoru a z aktuálního jedince nezíská žádnou informaci. V obou těchto případech dochází k problémům, protože se z evolučního algoritmu stává buď pouhé náhodné hledání, nebo jen kopírování starších generací bez možnosti jakéhokoliv vývoje. Doporučuje se proto, aby CR těchto krajních hodnot nikdy nenabývalo. V případě, že ošetřujeme separabilní funkci, je doporučeno nastavit tento parametr na hodnoty blížíící se k 0.[2]

- NP

Tento parametr udává počet jedinců v populaci. Jeho hodnota by nikdy neměla být nižší než 4, protože počet jedinců nutný pro výpočet nového potomka je 4.

- D

D udává počet argumentů účelové funkce.

- F

Parametr F je využíván v průběhu diferenciální mutace odlišně v závislosti na vybrané mutační strategii. Ovlivňuje tedy podobu vzniklého zkušebních vektorů.

- G

Udává počet evolučních cyklů šlechtění populace.

- Specimen

Udává typ čísel a interval, ze kterého jsou voleni jednotliví jedinci prvotní populace. V podstatě se jedná o omezení, v rámci kterého se jednotlivé parametry jedince mají hledat. Například:  $\{(Real, -10, 15), (Integer, -1, 1)\}$ . Díky tomuto omezení můžeme zajistit, aby byli jedinci generováni v rámci zadaného intervalu.

V Tab. 1 jsou uvedeny intervaly a doporučené hodnoty řídicích parametrů NP, F a CR.[2]

*Tab. 1: Intervaly a doporučené hodnoty řídicích parametrů DE [1]*

	Interval	Doporučeno
NP	[10D, 100D]	10D
F	[0, 2]	0,3 - 0,9
CR	[0, 1]	0,8 – 0,9

Parametry evolučních algoritmů přímo určují kvalitu a průběh evoluce. Při jejich chybném nastavení dochází ke stagnaci a k naprosto chybným výsledkům výpočtů. V základní verzi DE jsou parametry sice voleny jako pevné, v modifikovaných verzích DE je ale lze v závislosti na kvalitě probíhající evoluce a zadání aktuálního algoritmu příslušně měnit. V takovémto případě, pokud k vyhodnocení vhodných parametrů využijeme diferenciální evoluci samotnou, hovoříme o tzv. meta-diferenciální evoluci. Parametry této DE je trojice NP, CR, F. Samotná evoluce pak má sice efektivnější výsledky, ovšem za cenu prodlouženého evolučního procesu. Časově je meta-diferenciální evoluce mnohem náročnější než obyčejná DE.[1]

### 3.3 Mutace

Princip diferenciální mutace spočívá v náhodném výběru 3 nestejných jedinců z aktuální populace pro každého jedince z aktuální populace. Pomocí těchto 3 vybraných jedinců a mutační konstanty  $F$  je vytvořen tzv. šumový vektor. Seznam některých mutačních strategií je uveden v *Tab. 2.*[1]

*Tab. 2: mutační strategie[1]*

Best/1	$v = x_{best,j}^G + F(x_{r2,j}^G - x_{r3,j}^G)$	(1)
Rand/1	$v = x_{r1,j}^G + F(x_{r2,j}^G - x_{r3,j}^G)$	(2)
Rand-To-Best/1	$v = x_{i,j}^G + \lambda(x_{best,j}^G - x_{i,j}^G) + F(x_{r1,j}^G - x_{r2,j}^G)$	(3)
Best/2	$v = x_{best,j}^G + F(x_{r1,j}^G + x_{r2,j}^G - x_{r3,j}^G - x_{r4,j}^G)$	(4)
Rand/2	$v = x_{r5,j}^G + F(x_{r1,j}^G + x_{r2,j}^G - x_{r3,j}^G - x_{r4,j}^G)$	(5)

Doposud nejlepších výsledků dosahuje strategie Rand/1. Velice dobrých výsledků také dosahují strategie využívající nejlepšího jedince v aktuální populaci, tedy obsahující slovo Best.

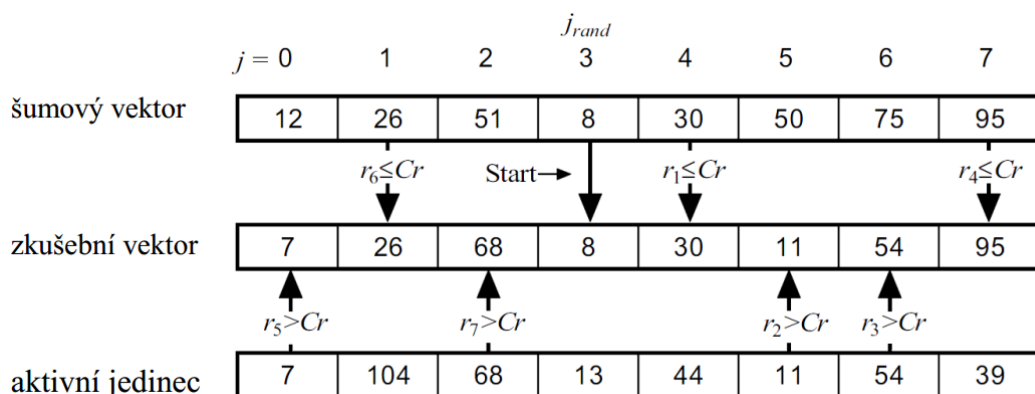
### 3.4 Křížení

Na rozdíl od GA, nastává křížení u DE až po mutaci. Křížením vytvoříme ze šumového vektoru a čtvrtého (dosud nepoužitého) rodiče nového jedince, který se nazývá zkušební (nebo také trial) vektor. Jeho vznik závisí na zvoleném typu křížení a velikosti CR. Zkušební vektor je zařazen do nové generace pouze v případě, že hodnota jeho účelové funkce je lepší než tato hodnota aktivního rodiče.[5]

### 3.4.1 Binomické křížení

Při použití binomického křížení využíváme náhodně generované číslo z intervalu  $<0,1>$  a hodnotu prahu křížení CR. Vždy pracujeme s odpovídajícími hodnotami šumového vektoru a aktivního rodiče. Z této dvojice vybereme do nově vytvářeného jedince jednu hodnotu podle hodnoty náhodně generovaného čísla. Pokud je toto číslo menší nebo rovno CR, bereme hodnotu z šumového vektoru. Pokud bude tato hodnota vyšší, bereme hodnotu z aktivního rodiče.

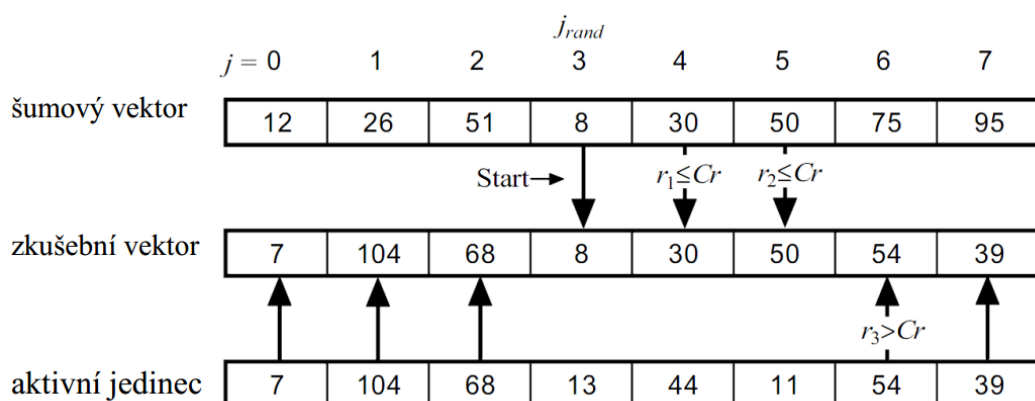
Tato metoda může být modifikovaná pomocí jednoho náhodně vybraného indexu, který bude sloužit jako počátek tohoto křížícího procesu. Tento prvek bude automaticky zařazen z šumového vektoru do zkušebního. Tím se zajistí, že bude z tohoto vektoru vždy alespoň jeden prvek v novém vektoru a ten tak nebude pouze kopií svého rodiče. Průběh tohoto křížení je názorně zobrazen na Obr. 3.[4]



Obr. 3: Binomické křížení[4]

### 3.4.2 Exponenciální křížení

Stejně jako u binomického křížení tento algoritmus začíná na náhodně vybraném prvku, který je automaticky zkopírován z šumového vektoru do zkušebního. V dalším průběhu kopírujeme hodnoty z šumového vektoru do zkušebního tak dlouho, dokud nevygenerujeme náhodné číslo větší než CR. Jakmile je toto číslo vygenerované, kopírujeme zbylé hodnoty z aktivního rodiče. Průběh tohoto křížení je názorně zobrazen na Obr. 4. [4]



Obr. 4: Exponenciální křížení[4]

Srovnáním exponenciálního a binárního křížení bylo dosaženo velice podobných výsledků, které se výrazně nelišily.

### 3.5 Stagnace

Evoluční algoritmy mohou nabývat nepřesných výsledků (nebylo dosaženo globálního extrému, ale pouze lokálního) při splnění jedné z následujících podmínek:[6]

- Celá populace se nachází v lokálním extrému dané účelové funkce.
- Populace ztratila diverzibilitu.
- Optimalizační proces probíhá pomalu, nebo vůbec neprobíhá

Tento případ se nazývá předčasná *konvergence k suboptimálnímu řešení* a je možné ho ovlivnit následujícími změnami:

- Změnou nastavení řídicích parametrů
- Zvýšením velikostí populace
- Zvýšením počtu generací
- Definicí omezení

Stagnací nazýváme jev, při kterém se vývoj hodnot účelové funkce zastavil, aniž by ještě byl nalezen globální extrém. Problémem je, že nebyla splněna žádná z výše uvedených podmínek, tedy populace je dále velice proměnlivá, nenachází se v lokálních extrémů funkce

ani nebylo zastaveno vytváření nových potomků. Neexistují tedy zřejmé důvody, proč došlo k zastavení optimalizačního procesu. [6]

Jednou z možných příčin stagnace je jev, při kterém všechny kombinace vzniklých jedinců mají horší ohodnocení, než jejich rodiče. V takovém případě do další populace postupují opět pouze původní rodiče, bez jakékoliv možnosti se vyvíjet. Riziko stagnace se tedy snižuje se zvyšujícím se počtem řešení (jedinců), které jsme schopni v rámci jedné generace vytvořit. [4]

Další možností boje proti stagnaci je změna řídicích parametrů evoluce, tj. kromě již zmíněného počtu jedinců v populaci (NP) i mutační faktor (F) a práh křížení (CR). Tyto parametry mají na vznik nových jedinců a jejich kvalitu velký vliv a jejich změna může opět obnovit optimalizační proces.[2]

Příkladem problematiky zvolených parametrů jsou například nastavení F či CR na hodnotu 1,0. V takových případech pozorujeme zvýšené riziko generování duplikátních řešení jedinců – bývá tedy praktičtější nastavit hodnotu daných parametrů raději na 0,99, nežli na číslo bez desetinného rozvoje.

### 3.6 Princip činnosti

Průběh diferenciální evoluce je obdobný jako u obecného evolučního algoritmu. Prvním krokem je stanovení parametrů:

- Mutační konstanta F z intervalu (0,2)
- Práh křížení CR z intervalu  $<0,1>$
- Velikost populace NP
- Počet dimenzí D
- Počet generací G
- Prototyp jedince Specimen

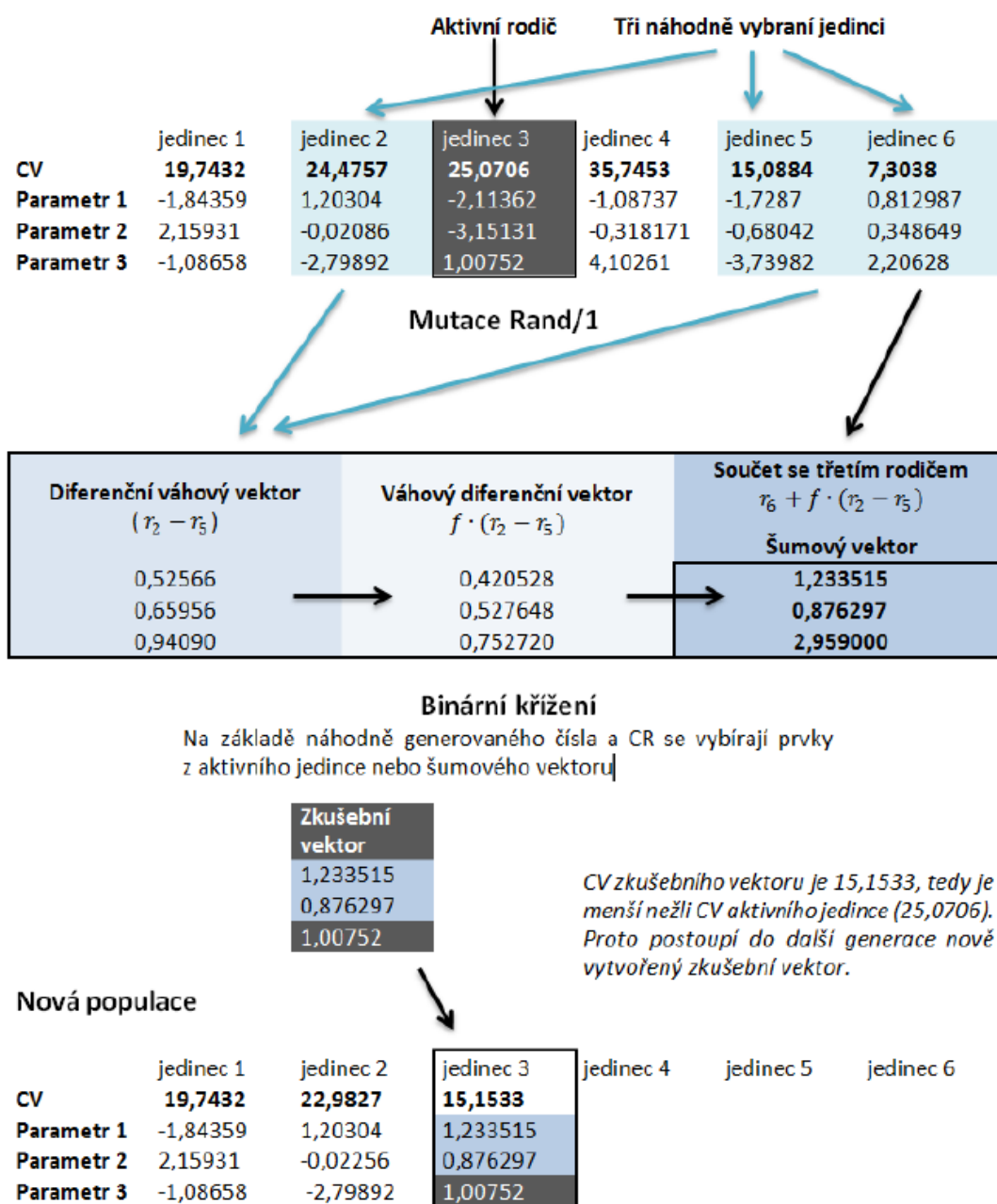
Následně je vygenerovaná první populace podle hodnot uvedených ve specimenu a její ohodnocení. Tímto končí počáteční nastavení a začíná evoluční cyklus.

Cyklus začíná výběrem aktivního rodiče. Ten je vybrán klasickým iteračním procesem, kdy postupně procházíme veškeré jedince v populaci. Aktivnímu rodiči pak ze stejné populace přiřadíme další tři nestejně náhodně vybrané jedince, na které aplikujeme mutaci. Mutační

(šumový) vektor zkřížíme s aktivním rodičem na základě náhodně vygenerované hodnoty a velikosti parametru CR. [4]

Vzniklého nového jedince opět ohodnotíme účelovou funkcí. Pokud tato hodnota účelové funkce bude horší, než v případě aktivního rodiče, postoupí do nové populace aktivní rodič. V opačném případě jsme úspěšně vytvořili nového jedince, který je lepší, než byl jeho rodič. Díky tomuto poslednímu kroku máme zajištěno, že populace v každé nové generaci bude vždy lepší nebo alespoň stejná než generace předchozí.

Následuje výběr dalšího aktivního rodiče a cyklus evoluce se opakuje až do vyčerpání populace. Ve své základní verzi je DE ukončena, pokud vyšlechtíme zadaný počet generací nebo dosáhneme jiné ukončovací podmínky. Programátor ale může ukončující podmínku také sám upravit. Příkladem takové úpravy může být ošetření případu, kdy nastává tzv. stagnace. Průběh tohoto algoritmu vedoucí ke vzniku nového potomka je zobrazen na *Obr. 5*. [4]



Obr. 5: Průběh algoritmu diferenciální evoluce [1]



### 3.7 Adaptivní modifikace

Existuje několik možných modifikací základního algoritmu diferenciální evoluce. Tyto modifikace spočívají převážně v rozdílné volbě řídicích parametrů  $F$  a  $CR$  a jejich úpravě za běhu algoritmu a mimo jiné také v možnosti využití více mutačních strategií při výpočtu nových jedinců. Základními typy modifikací jsou jDE, JADE, SADE a EPSDE. Tyto modifikace jsou dále v této práci popsány a otestovány.

#### 3.7.1 jDE

Jedná se o jednoduchou a velice efektivní adaptivní variantu diferenciální evoluce, kterou navrhl J. Brest et al. Princip tohoto algoritmu spočívá ve využití mutační strategie DE/rand/1/bin a samoadaptivních hodnot řídicích parametrů  $F$  a  $CR$ . Tato dvojice parametrů je přiřazena každému jedinci v populaci. [6]

Hodnoty parametrů  $F$  a  $CR$  jsou inicializovány náhodně na počátku algoritmu. V dalších generacích mohou jejich hodnoty být náhodně mutovány s pravděpodobnostmi  $\tau_1$  a  $\tau_2$ . Pokud tato podmínka nastane, jsou generovány nové hodnoty parametrů  $F$  a  $CR$  podle následujících pravidel:

$$F_i^{gen+1} = \begin{cases} F_l + rand_1 \cdot F_u & \text{pokud } rand_2 < \tau_1 \\ F_i^{gen} & \text{v opačném případě} \end{cases} \quad (6)$$

$$CR_i^{gen+1} = \begin{cases} rand_3 & \text{pokud } rand_4 < \tau_2 \\ CR_i^{gen} & \text{v opačném případě} \end{cases} \quad (7)$$

Čísla  $rand_1 - rand_4$  jsou náhodně generovaná z intervalu  $(0;1)$ . Konstantám  $\tau_1$ ,  $\tau_2$ ,  $F_l$  a  $F_u$  jsou přiřazeny hodnoty  $\tau_1 = \tau_2 = F_l = 0,1$  a  $F_u = 0,9$ . Nové hodnoty parametru  $F$  jsou tedy generovány z intervalu  $<0,1;1>$  a hodnoty  $CR$  z intervalu  $(0,1)$ . Parametr  $NP$  zůstává po celou dobu výpočtu nezměněn. Parametry  $F_i^{gen+1}$  a  $CR_i^{gen+1}$  jsou získány před provedením mutace, takže ovlivňují vznik jedince  $x_i^{gen+1}$ . [6]

### 3.7.2 JADE

Algoritmus JADE oproti základní verzi diferenciální evoluce obsahuje tři adaptivní vylepšení. Jedná se o možnost využití archivu, mutační strategie DE/current-to-pbest a proměnné hodnoty řídicích parametrů  $F$  a  $CR$ . [7]

Mutační strategie jako DE/current-to-best/k, nebo DE/best/k mohou trpět problémy spočívajícími v jejich příliš rychlé konvergenci. To může způsobit problémy s diverzibilitou populace. S ohledem na tyto velmi rychlé, ale méně spolehlivé mutační strategie, byla navržena nová strategie DE/current-to-pbest, která je využívána v této metodě diferenciální evoluce. Mutační vektor je tedy generován následujícím způsobem:

$$v_{i,g} = x_{i,g} + F_i(x_{best,g}^p - x_{i,g}) + F_i(x_{r1,g} - x_{r2,g}) \quad (8)$$

kde  $x_{best,g}^p$  je náhodně vybraný rodič jako jeden z 100p% nejlepších jedinců v aktuální populaci s hodnotou  $p$  v intervalu  $(0;1)$ .  $F_i$  zde představuje adaptivně proměnnou mutační konstantu, počítanou pomocí Cauchyho rozdělení jako:

$$F_i = randc_i(\mu_F; 0,1) \quad (9)$$

a následně oříznutou na hodnotu 1 pokud  $F_i > 1$  nebo znovu vygenerovanou pokud by hodnota byla menší než 0.

Další vylepšení této strategie přináší využití archivu, což způsobí změnu ve výpočtu mutačního vektoru následujícím způsobem:

$$v_{i,g} = x_{i,g} + F_i(x_{best,g}^p - x_{i,g}) + F_i(x_{r1,g} - \tilde{x}_{r2,g}) \quad (10)$$

Hodnoty  $x_{i,g}$ ,  $x_{best,g}^p$ ,  $x_{r1,g}$  jsou vybírány stále z aktuální populace  $P$ , ale hodnota  $\tilde{x}_{r2,g}$  je náhodně vybírána ze sjednocení  $P \cup A$  aktuální populace a archivu  $A$ . Archiv je zpracován velice jednoduše, aby se zabránilo větším výpočetním nárokům. Archiv je na počátku algoritmu prázdný. Pokud je rodič v procesu výběru lepšího jedince nahrazen zkušebním vektorem v následující generaci, pak je tento rodič přidán do archivu. V případě, kdy velikost archivu na konci generace je větší než velikost populace  $NP$ , pak jsou z něj náhodně odebírány prvky, dokud tato velikost neklesne pod danou hranici. [7]

Kromě archivu jsou v průběhu výpočtu nové generace uchovávány také hodnoty řídicích konstant  $F$  a  $CR$  do polí  $S_F$  a  $S_{CR}$ . Z nich jsou na konci každé generace počítány střední hodnoty  $\mu_F$  a  $\mu_{CR}$  pomocí rovnic (11) a (12).

$$\mu_F = (1 - c)\mu_F + c \cdot mean_L(S_F) \quad (11)$$

$$\mu_{CR} = (1 - c)\mu_{CR} + c \cdot \text{mean}_A(S_{CR}) \quad (12)$$

kde  $c$  je kladná konstanta z intervalu  $\langle 0,05;0,2 \rangle$ ,  $\text{mean}_A$  je aritmetický průměr a  $\text{mean}_L$  je Lehmerův průměr počítaný pomocí:

$$\text{mean}_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \quad (13)$$

Hodnota řídicí konstanty  $CR$  je v každé generaci počítaná pomocí:

$$CR_i = \text{randn}_i(\mu_{CR}; 0,1) \quad (14)$$

kde  $\text{randn}$  je normální rozdělení se střední hodnotou  $\mu_{CR}$  a rozptylem 0,1. Celý průběh výpočtu JADE je zobrazen ve formě pseudokódu na Obr. 6.

**Begin**

Nastav  $\mu_{CR} = 0.5$ ;  $\mu_F = 0.5$ ;  $A = \emptyset$

Vytvoř náhodnou prvotní populaci  $\{\mathbf{x}_{i,0} \mid i = 1, 2, \dots, NP\}$

**For**  $g = 1$  to  $G$

$S_F = \emptyset$ ;  $S_{CR} = \emptyset$ ;

**For**  $i = 1$  to  $NP$

Generuj  $CR_i = \text{randn}_i(\mu_{CR}, 0.1)$ ,  $F_i = \text{randc}_i(\mu_F, 0.1)$

Náhodně vyber  $\mathbf{x}^{p_{\text{best},g}}$  jako jeden z 100p% nejlepších vektorů

Náhodně vyber  $\mathbf{x}_{r1,g} \neq \mathbf{x}_{i,g}$  z aktuální populace  $\mathbf{P}$

Náhodně vyber  $\mathbf{x}_{r2,g} \neq \mathbf{x}_{r1,g} \neq \mathbf{x}_{i,g}$  z  $\mathbf{P} \cup \mathbf{A}$

$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i \cdot (\mathbf{x}^{p_{\text{best},g}} - \mathbf{x}_{i,g}) + F_i \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g})$

Generuj  $j_{\text{rand}} = \text{randint}(1, D)$

**For**  $j = 1$  to  $D$

**If**  $j = j_{\text{rand}}$  or  $\text{rand}(0, 1) < CR_i$

$u_{j,i,g} = v_{j,i,g}$

**Else**

$u_{j,i,g} = x_{j,i,g}$

**End If**

**End For**

Vypočítej hodnotu účelové funkce zkušebního vektoru  $\mathbf{u}_{i,g}$

**If**  $f(\mathbf{x}_{i,g}) \leq f(\mathbf{u}_{i,g})$

$\mathbf{x}_{i,g+1} = \mathbf{x}_{i,g}$

**Else**

$\mathbf{x}_{i,g+1} = \mathbf{u}_{i,g}$ ;  $\mathbf{x}_{i,g} \rightarrow \mathbf{A}$ ;  $CR_i \rightarrow S_{CR}$ ,  $F_i \rightarrow S_F$

**End If**

**End for**

Náhodně odebírej hodnoty z  $\mathbf{A}$  dokud  $|\mathbf{A}| \leq NP$

$\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot \text{mean}_A(S_{CR})$

$\mu_F = (1 - c) \cdot \mu_F + c \cdot \text{mean}_L(S_F)$

**End for**

**End**

Obr. 6: Pseudokód JADE [7]

### 3.7.3 SaDE

Vzhledem k tomu, že v průběhu evolučního procesu může docházet k situaci, kdy jsou určité strategie generování zkušebního vektoru vhodnější než jiné, byla vyvinuta adaptivní modifikace nazvaná SaDE. Princip této strategie spočívá ve využití několika strategií pro výpočet zkušebního vektoru. Tyto strategie jsou přidělovány členům populace v každé generaci podle jejich vypočítaných pravděpodobností. Strategie použité v tomto algoritmu jsou DE/rand/1/bin(15), DE/rand-to-best/2/bin(16), DE/rand/2/bin(17) a DE/current-to-rand/bin(18).[8]

$$u_{i,j} = \begin{cases} x_{r1,j} + F \cdot (x_{r2,j} + x_{r3,j}) & \text{pokud } \text{rand}[0,1] < CR \text{ nebo } j = j_{rand} \\ x_{i,j} & \text{v opačném případě} \end{cases} \quad (15)$$

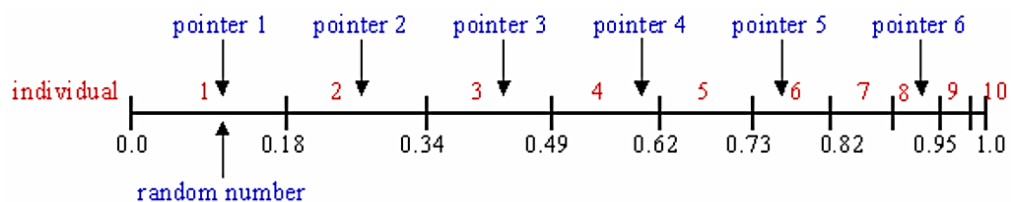
$$u_{i,j} = \begin{cases} x_{i,j} + F \cdot (x_{best,j} - x_{i,j}) + F \cdot (x_{r1,j} - x_{r2,j}) + F \cdot (x_{r3,j} + x_{r4,j}) & \text{pokud } \text{rand}[0,1] < CR \text{ nebo } j = j_{rand} \\ x_{i,j} & \text{v opačném případě} \end{cases} \quad (16)$$

$$u_{i,j} = \begin{cases} x_{r1,j} + F \cdot (x_{r2,j} - x_{r3,j}) + F \cdot (x_{r4,j} - x_{r5,j}) & \text{pokud } \text{nebo } j = j_{rand} \\ x_{i,j} & \text{v opačném případě} \end{cases} \quad (17)$$

$$U_{i,G} = X_{i,G} + K \cdot (X_{r1,j} - X_{i,j}) + F \cdot (X_{r2,j} - X_{r3,j}) \quad (18)$$

Na počátku algoritmu je stanovena délka učící periody a vytvořen okruh použitelných strategií, kde každé je přiřazena stejná pravděpodobnost, že bude využita pro výpočet nového jedince. Na konci každé generace je počet zkušebních vektorů, které mohou úspěšně projít do další generace, uložen do paměti úspěchů jako  $ns_{k,G}$  a počet neúspěšných zkušebních vektorů je uložen do paměti neúspěchů jako  $nf_{k,G}$ .

K výběru strategií se využívá metody stochastického univerzálního vzorkování, založené na požadavku rovnoměrně rozdělených náhodných čísel v daném intervalu. První hodnota v této metodě je zvolena náhodně a další jsou od ní vzdáleny o pevný počet daných kroků. Délka kroku je dána jako  $1/\text{počet jedinců}$ . Princip této metody je také zobrazen na Obr. 7.



Obr. 7: Stochastické univerzální vzorkování

Po uplynutí učicí periody jsou naplněny paměti úspěchů a neúspěchů podle *Tab. 3* a *Tab. 4*. S využitím těchto hodnot jsou aktualizovány pravděpodobnosti výběru jednotlivých strategií podle rovnice (19) a jsou vymazány nejstarší hodnoty (první řádek) z obou pamětí.[8]

$$p_{k,G} = \frac{S_{k,G}}{\sum_{k=1}^K S_{k,G}} \quad (19)$$

kde

$$S_{k,G} = \frac{\sum_{g=G-LP}^{G-1} nS_{k,G}}{\sum_{g=G-LP}^{G-1} nS_{k,G} + \sum_{g=G-LP}^{G-1} nf_{k,G}} + \varepsilon \quad (20)$$

*Tab. 3: Tabulka úspěchů[8]*

Index	Strategie 1	Strategie 2	...	Strategie K
1	$nS_{1,G-LP}$	$nS_{2,G-LP}$	...	$nS_{1,G-LP}$
2	$nS_{1,G-LP+1}$	$nS_{2,G-LP}$	...	$nS_{k,G-LP+1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
LP	$nS_{1,G-1}$	$nS_{2,G-1}$	...	$nS_{k,G-1}$

*Tab. 4: Tabulka neúspěchů[8]*

Index	Strategie 1	Strategie 2	...	Strategie K
1	$nf_{1,G-LP}$	$nf_{2,G-LP}$	...	$nf_{1,G-LP}$
2	$nf_{1,G-LP+1}$	$nf_{2,G-LP}$	...	$nf_{k,G-LP+1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
LP	$nf_{1,G-1}$	$nf_{2,G-1}$	...	$nf_{k,G-1}$

Hodnota řídicího parametru F je počítaná pro každého jedince v populaci pomocí normálního rozdělení se střední hodnotou 0,5 a rozptylem 0,3. Tedy s pravděpodobností 99,7% spadá hodnota tohoto parametru do intervalu  $\langle -0,4; 1,4 \rangle$ .

Výpočet parametru CR je také prováděn pomocí normálního rozdělení. Střední hodnota je inicializovaná hodnotou 0,5 a rozptyl je 0,1. Pokud vygenerovaná hodnota klesne pod 0 nebo

přesáhne 1, je regenerovaná. Po uplynutí učicí periody je střední hodnota určována jako medián z pole CRMemory obsahujícího hodnoty CR pro každou zadanou strategii, ukládané v případě, že byl zkušební vektor vybrán do nové generace. Celý průběh výpočtu SaDE je zobrazen ve formě pseudokódu na Obr. 8.[8]

```

Begin
Inicializuj hodnotu medianu CR(CRMk), pravděpodobnosti strategií pk,g, k = 1,...,K (K je počet dostupných strategií) a
délku periody učení LP a Vytvoř náhodnou prvotní populaci (xi,0, i = 1, 2,..., NP);
For g = 1 to G
//Výpočet pravděpodobnosti strategií pk,g a aktualizace paměti úspěšných a neúspěšných hodnot
If g > LP
For k = 1 to K
Aktualizuj pravděpodobnost pk,g podle rovnice;
Odeber k-tou hodnotu z prvních řádků paměti úspěchů a neúspěchů;
End For
End If
// Přiřazení strategie generování zkušebního vektoru a parametrů každému jedinci
If g >= LP
For k = 1 to K
Vypočítej CRMk = median( CRMemoryk );
End For
End If

Pomocí stochastického univerzálního vzorkování vyber strategii k pro každého jedince

For i = 1 to NP
Vypočítej odpovídající hodnotu Fi = Normrnd(0,5;0,3);
For k = 1 to K
Vypočítej CRk,i = Normrnd(CRMk;0,1);
While CRk,i < 0 nebo CRk,i > 1
Vypočítej CRk,i = Normrnd(CRMk;0,1);
End While
End For

Vypočítej mutační vektor pomocí zvolené strategie a parametrů v předchozím kroku;

Generuj jrand = randint(1, D);
For j = 1 to D
If j = jrand or rand(0, 1) < CRk,i
ukj,i,g = vj,i,g;
Else
ukj,i,g = xj,i,g;
End If
End For

Vypočítej hodnotu účelové funkce zkušebního vektoru uki,g;
If f(xi,g) > f(uki,g)
xi,g+1 = xi,g; nfk,g = nfk,g + 1;
Else
xi,g+1 = uki,g; CRk,i → CRMemoryk; nsk,g = nsk,g + 1;
End If
End for
Ulož nsk,g a nfk,g do paměti úspěchů a neúspěchů;
End for
End

```

Obr. 8: Pseudokód SaDE[8]

### 3.7.4 EPSDE

Poslední z adaptivních modifikací diferenciální evoluce zpracovávaných v rámci této práce je EPSDE. Jak již bylo řečeno, tak velkou míru na kvalitě evolučního procesu má volba mutační strategie a řídicích parametrů. Metoda EPSDE k tomuto problému přistupuje pomocí několika okruhů. Jeden z nich je určen pro mutační strategie, další pro hodnoty parametrů  $F$  a poslední pro hodnoty parametrů  $CR$ . [10]

V prvotní populaci je každému z jedinců náhodně přiřazena mutační strategie a hodnoty řídicích parametrů vybrané z příslušných okruhů. Pokud vygenerovaný zkušební vektor příslušného rodiče projde do další generace, zůstává mu přiřazena vybraná mutační strategie a příslušné parametry. V opačném případě vybrány nové hodnoty. Celý průběh výpočtu EPSDE je zobrazen ve formě pseudokódu na Obr. 9.[10]

```

Begin
Vytvoř náhodnou prvotní populaci  $\{x_{i,0} \text{ , } i = 1, 2, \dots, NP\}$  ;
Vytvoř okruh použitelných mutačních strategií a řídicích parametrů  $F$  a  $CR$  pro dané strategie;
Přiřaď každému členu populace náhodně mutační strategii a řídicí parametry;
For  $g = 1$  to  $G$ 
  For  $i = 1$  to  $NP$ 
    Vypočítej mutační vektor v závislosti na přiřazené strategii a konstantě  $F$ ;
    Generuj  $j_{rand} = \text{randint}(1, D)$ ;
    For  $j = 1$  to  $D$ 
      If  $j = j_{rand}$  or  $\text{rand}(0, 1) < CR_i$ 
         $u_{j,i,g} = v_{j,i,g}$ ;
      Else
         $u_{j,i,g} = x_{j,i,g}$ ;
      End If
    End For
    Vypočítej hodnotu účelové funkce zkušebního vektoru  $u_{i,g}$ 
    If  $f(x_{i,g}) < f(u_{i,g})$ 
       $x_{i,g+1} = x_{i,g}$ ;
      Náhodně přiřaď nezměněnému jedinci v nové populaci novou mutační strategii a řídicí parametry
    Else
       $x_{i,g+1} = u_{i,g}$  ;
      If  $f(x_{i,g}) > f(u_{i,g})$ 
         $x_{best,g} = u_{i,g}$ ;
      End If
    End If
  End for
End for
End

```

Obr. 9: Pseudokód EPSDE [10]

### 3.8 Ukončovací podmínka

Algoritmus diferenciální evoluce je možné ukončit podle několika možných kritérií. Za předpokladu, že známe umístění a hodnotu globálního extrému již při výpočtu funkce, můžeme výpočet ukončit v okamžiku, kdy se nalezená hodnota tomuto extrému přiblíží v přijatelné toleranci.[1]

Není-li hodnota extrému předem známá, obvykle se pro ukončení výpočtu používá stanovený počet generací. Tento počet by měl být dostatečný, aby byl algoritmus schopen nalézt extrém.

Ukončovací podmínkou mohou být také statistické údaje, například rozdíl jedinců v populaci, nebo rozdíl nejlepších jedinců v několika generacích. Tato metoda by ale měla být používána jen s velice přesným nastavením, protože algoritmus by mohl být zastaven při nalezení lokálního extrému.[1]



## 4 TESTOVÁNÍ

Testování je důležité z mnoha rozličných důvodů. Můžeme pomocí něj zjistit, jakým způsobem reaguje výkonnost algoritmu na změny parametrů účelové funkce. Testováním se také zjistí, jaké kombinace řídicích parametrů jsou v daných situacích nejlepší, což představuje velmi užitečnou informaci. Na základě simulačních pokusů lze porovnat různé typy algoritmů a ve výsledku nám testování může přinést nové poznatky, které můžeme následně využít pro zdokonalení optimalizačního procesu.[3]

Samotné testování algoritmu se provádí tak, že mnohonásobně opakujeme hledání extrému na stejné testovací funkci a statistické výsledky těchto pokusů pak zobrazuje v grafické podobě nebo v tabulce.[11]

### 4.1 Testovací funkce

Využitím testovacích funkcí můžeme zjistit, jak je daný algoritmus úspěšný na daném problému. Pro otestování algoritmů používáme soubor testovacích funkcí, které záměrně trpí různými patologiemi nebo jsou často nelineární. U mnoha testovacích funkcí známe hodnoty globálních extrémů. Porovnáním těchto předem známých hodnot s globálními extrémy nalezenými konkrétním algoritmem získáváme informaci o úspěšnosti algoritmu. Testovací funkce mohou být jak unimodální (obsahující jediný extrém) tak také multimodální (obsahují více extrémů). Samostatnou skupinou pak představují fraktální testovací funkce, u kterých předem neznáme hodnoty globálního extrému, ale lze je rovněž použít pro porovnání úspěšnosti jednotlivých algoritmů. V této práci je k testování adaptivních modifikací diferenciální evoluce využito několik testovacích funkcí vytvořených pro účel testování evolučních algoritmů. Tyto funkce jsou uvedeny v *Tab. 5* a *Tab. 6*. Veškeré funkce jsou penalizovány hodnotou  $f^*$  uvedenou před předpisem každé funkce.[2]

Tab. 5: Testovací funkce s uvedenou hodnotou penalizace [12]

Sphere function	$f^* = -1400$	$f_1(x) = \sum_{i=1}^D z_i^2 + f_1^*, z = x - o$	(21)
--------------------	---------------	--	------

Rotated Elliptic function	$f^* = -1300$	$f_2(x) = \sum_{i=1}^D (10^6)^{\frac{i-1}{D-1}} \cdot z_i^2 + f_2^*, z = T_{osz}(M_1(x - o))$	(22)
---------------------------------	---------------	---	------

Rotated Bent Cigar function	$f^* = -1200$	$f_3(x) = z_1^2 + 10^6 \sum_{i=2}^D z_i^2 + f_3^*, z = M_2 T_{asy}^{0.5}(M_1(x - o))$	(23)
-----------------------------------	---------------	---	------

Rotated Discus function	$f^* = -1100$	$f_4(x) = 10^6 \cdot z_1^2 + \sum_{i=2}^D z_i^2 + f_4^*, z = T_{osz}(M_1(x - o))$	(24)
-------------------------------	---------------	---	------

Different Powers function	$f^* = -1000$	$f_5(x) = \sqrt{\sum_{i=1}^D [z_i]^{2+4\frac{i-1}{D-1}}} + f_5^*, z = x - o$	(25)
---------------------------------	---------------	--	------

Rotated Rosenbrock function	$f^* = -900$	$f_6(x) = \sum_{i=1}^D (100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + f_6^*,$ $z = M_1\left(\frac{2048(x - o)}{100}\right) + 1$	(26)
-----------------------------------	--------------	--	------

Rotated Schaffers F7 function	$f^* = -800$	$f_7(x) = \left(\frac{1}{D-1} \sum_{i=1}^{D-1} (\sqrt{z_i} + \sqrt{z_i} \sin^2(50z_i^2))\right)^2 + f_7^*,$ $z_i = \sqrt{y_i^2 + y_{i+1}^2}, y = \Lambda^{10} M_2 T_{asy}^{0.5}(M_1(x - o))$	(27)
-------------------------------------	--------------	---	------

Rotated Ackley function	$f^* = -700$	$f_8(x) = -20 \exp\left(-0,2 \sqrt{\frac{1}{D} \sum_{i=1}^D z_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi z_i)\right) +$ $+20 + e + f_8^*$ $z = \Lambda^{10} M_2 T_{asy}^{0.5}(M_1(x - o))$	(28)
-------------------------------	--------------	--	------

Rotated Griewank function	$f^* = -500$	$f_9(x) = \sum_{i=1}^D \frac{z_i^2}{4000} - \prod_{i=1}^D \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f_{10}^*$ $z = \Lambda^{100} M_1 \frac{600(x - o)}{100}$	(29)
---------------------------------	--------------	---	------

Tab. 6: Testovací funkce s uvedenou hodnotou penalizace [12]

Rotated  
Rastrigin  
function

$$f^* = -300 \quad f_{10}(x) = \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i) + 10) f_{10}^* \quad (30)$$

$$z = M_1 \wedge^{10} M_2 T_{asy}^{0,2}(T_{osz} \left( M_1 \frac{5,12(x - o)}{100} \right))$$

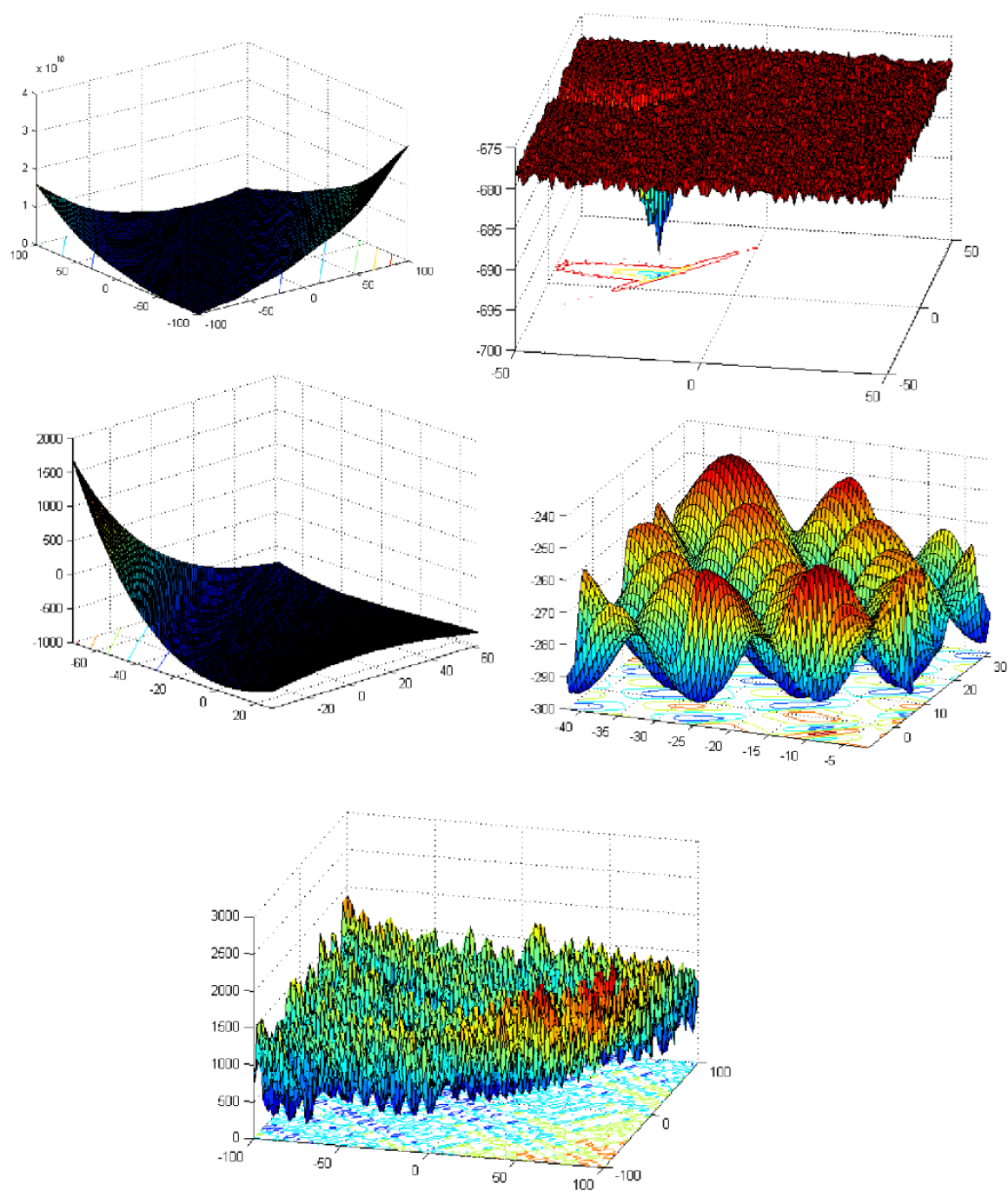
Rotated  
Schwefel  
function

$$f^* = 100 \quad f_{11}(x) = 418,9829xD - \sum_{i=1}^D g(z_i) + f_{11}^* \quad (31)$$

$$z = \wedge^{10} M_1 \left( \frac{1000(x - o)}{100} \right) + 4,209687462275036e + 002$$

$$g(z_i) = \begin{cases} z_i \sin(|z_i|^{1/2}) & \text{pokud } |z_i| \leq 500 \\ (500 - \text{mod}(z_i, 500)) \sin(\sqrt{|500 - \text{mod}(z_i, 500)|}) + \frac{(z_i - 500)^2}{10000D} & \text{pokud } z_i > 500 \\ (\text{mod}(|z_i|, 500) - 500) \sin(\sqrt{|\text{mod}(|z_i|, 500) - 500|}) + \frac{(z_i + 500)^2}{10000D} & \text{pokud } z_i < -500 \end{cases}$$

Na Obr. 10 jsou grafické průběhy eliptické, Ackleyho, Rosenbrockovy, Rastriginovy a Schwefelovy funkce vykresleny ve 3D.



Obr. 10: Grafy vybraných testovacích  
funkcí z Tab. 5 a Tab. 6[12]

## 4.2 Vyhodnocení

Kvalita průběhu evolučního procesu se zobrazuje pomocí historie vývoje hodnoty účelové funkce zanesené do grafu. Graf může být vytvořen několika odlišnými způsoby. Jednou z možností je zobrazení nejhorších a nejlepších jedinců ze všech populací, což nám dává informaci o celkové konvergenci populace. Jako lepší řešení se však ukázalo zobrazení závislosti hodnoty účelové funkce na počtu jejího ohodnocení. Je to dáno tím, že každý algoritmus provádí během jednoho cyklu různý počet ohodnocení účelové funkce. Nedochází tak ke zkreslení informace o skutečné kvalitě evolučního procesu a můžeme si dovolit porovnávat algoritmy lišící se vnitřní strukturou. [11]

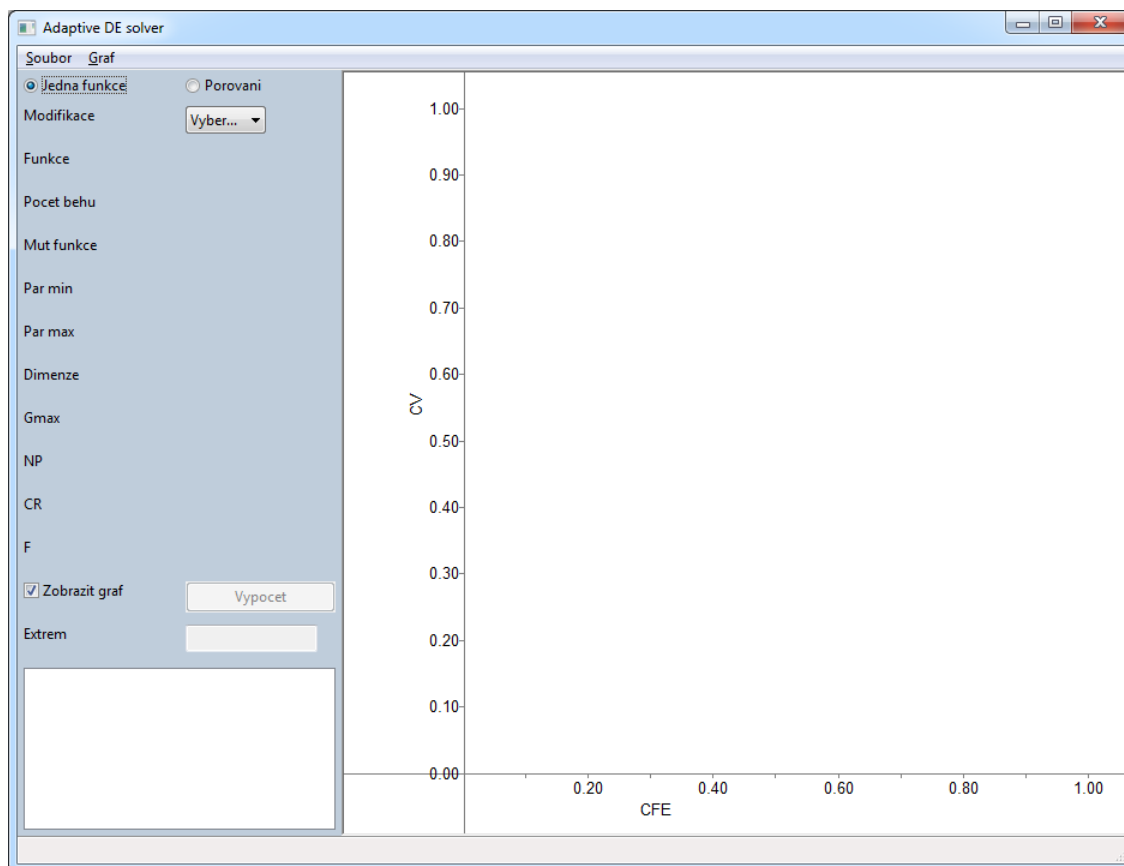
Další možností vyhodnocení je zobrazení pouze nejlepších jedinců ze všech realizovaných simulací formou histogramu. Při použití této formy prezentace výsledků je dobře viditelné, jaký nejlepší výsledek našly jednotlivé simulace. [2]

Pro hodnocení kvality algoritmů je také velmi důležitý čas výpočtu, který se odvíjí od počtu provedených elementárních operací. Tento čas bývá ovlivňován složitostí účelové funkce a konkrétním nastavením jednotlivých parametrů algoritmu. S výhodou se zde používá paralelizace algoritmu, která snižuje celkový čas výpočtu.[11]

## **II. PRAKTICKÁ ČÁST**

## 5 APLIKACE ADAPTIVNI DE

Program je ovládán přes GUI, ve kterém jsou zadány veškeré potřebné parametry a pokud byly všechny zadány bezchybně, má uživatel možnost provést výpočet zvolené varianty diferenciální evoluce se zadanými parametry.



Obr. 11: Prostředí programu po spuštění

První možností, kterou má uživatel možnost zvolit je, zda chce výpočet pouze jedné varianty DE nebo porovnání. Při výběru první možnosti je zobrazen průběh nejlepších jedinců v každém běhu. Příklad tohoto zobrazení je uveden na Obr. 13. V případě, že je zvolena možnost porovnání, jsou zobrazeny průběhy průměrných běhů ze zadaného počtu běhů pro každou z variant DE. Příklad této varianty je pro srovnání uveden na Obr. 14.

Po zadání jedné ze základních možností zobrazení musí uživatel vybrat požadovanou metodu DE, resp. funkce v případě porovnání. Zadáním této volby je zpřístupněno zadání veškerých zbývajících parametrů.

Možnosti vložených hodnot jsou následující:

- Modifikace – V případě výběru výpočtu na jediné funkci je tento parametr první, který musí být zadán. Z rozevíracího seznamu musí být vybraná varianta DE k výpočtu
- Funkce – V případě výběru porovnání první nutný výběr. Stejně jako v případě modifikace jsou zde vybírány možnosti z rozevíracího seznamu. Zde je vybírána funkce, na které bude algoritmus počítat
- Počet běhů – Udává počet opakování běhu výpočtu. V případě zvoleného výpočtu jediné funkce jsou všechny běhy zobrazeny. V případě zvolení porovnání je zobrazen pouze průměrný běh
- Mut funkce – Tato možnost je dostupná pouze v případě zvolení výpočtu na jediné funkci a varianty výpočtu DE. Z rozevíracího seznamu je nutné vybrat mutační funkci
- Par. Min – Tento parametr udává dolní hranici pro náhodně generovaná data na počátku výpočtu DE. Vstupní data jsou očekávána ve formě desetinného nebo celého čísla
- Par. Max – Hodnota tohoto parametru udává horní hranici pro generovaná data na počátku výpočtu DE. Vstupní data jsou očekávána ve formě desetinného nebo celého čísla.
- Dimenze – Nastavení dimenze účelové funkce zadávané výše. Vstup je očekáván pouze ve formě celého čísla. Pro zadané testovací funkce je nutné zadat hodnotu z pole (2, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100). Jiné hodnoty nebudou akceptovány
- Gmax – Nastavení počtu evolučních cyklů výpočtu. Vstup je přípustný pouze ve formě celého čísla.
- NP – Nastavení velikosti populace. Vstup je přípustný pouze ve formě celého čísla.
- CR – Nastavení řídicího parametru evoluce CR. Vstupní data jsou očekávána ve formě desetinného nebo celého čísla.
- F – Nastavení řídicího parametru evoluce F. Vstupní data jsou očekávána ve formě desetinného nebo celého čísla.
- Zobrazit graf – Při zatržení této položky jsou zobrazena vypočítaná data ve formě grafu. V opačném případě je graf skryt.



Možnost výpočtu spouštěná stiskem tlačítka Vypocet je povolena až po bezchybném zadání poslední požadované hodnoty. Pod možnými poli pro zadávání se nachází pole extrému, kde je vypsána neoptimálnější nalezená hodnota v průběhu výpočtu.

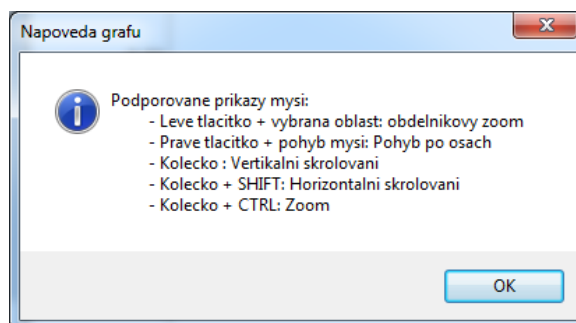
Po dokončení výpočtu jsou v seznamu uvedeny veškeré parametry, které byly použity pro aktuální výpočet. Pokud jsou zadány vyšší hodnoty Gmax, populace a dimenze nebo komplexnější testovací funkce, zabere výpočet delší dobu, vzhledem ke složitosti výpočtů.

Provedením prvního výpočtu a vykreslením odpovídajícího grafu jsou také zpřístupněny položky menu pro ukládání dat nebo obrázku. V menu jsou následující možnosti výběru:

- Soubor
  - Save data – Uložení dat ve formátu \*.txt. Ukázka těchto dat je na *Obr 15*
  - Exit – Ukončení celé aplikace
- Graf
  - Save image – Uložení obrázku ve formátu \*.bmp
  - Fit – Změní měřítko grafu, tak aby byl roztažený a kompletně zobrazený v grafu

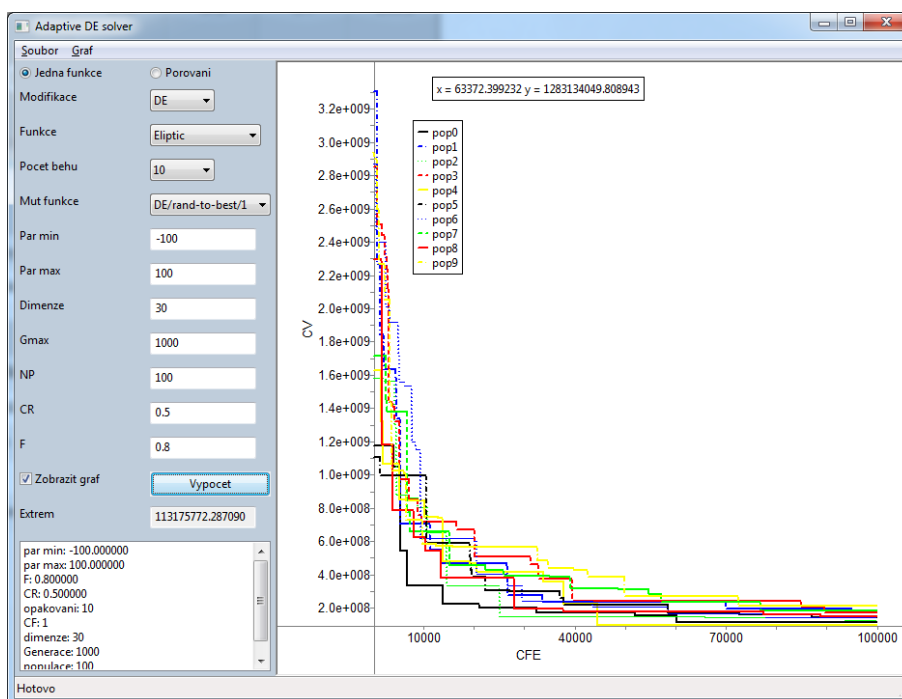
Při stisknutí pravého tlačítka myši v oblasti grafu je zobrazeno kontextové menu nabízející následující možnosti:

- Center – Vycentrování grafu podle aktuální pozice myši
- Fit – Změní měřítko grafu, tak aby byl roztažený a kompletně zobrazený v grafu
- Zoom in – Přiblížení grafu, stejně funguje i otočení kolečkem myši
- Zoom out – Oddálení grafu, stejně funguje i otočení kolečkem myši
- Lock aspect – Aktivováním této možnosti probíhá posun po ose souměrně
- Show mouse commands – Zobrazí tabulku s možnými příkazy myši podle *Obr. 12*

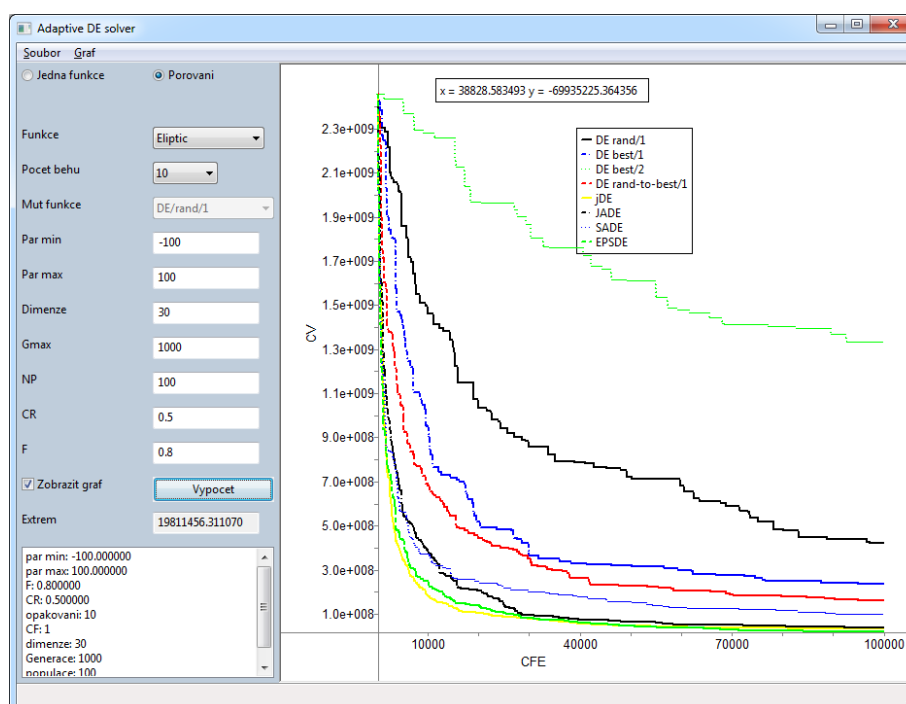


*Obr. 12: Nápořěda grafu*

Aktuální poloha myši v oblasti grafu je zobrazena ve formě souřadnic  $x$  a  $y$  v tabulce v oblasti grafu.



Obr. 13: Prostředí programu při volbě výpočtu jediné varianty DE



Obr. 14: Prostředí programu při volbě výpočtu porovnání

## 5.1 Popis implementace

Celý program je napsán s využitím frameworku wxwidgets sloužícím pro vytvoření GUI programu a zajišťujícím funkčnost programu na více platformách, ale vzhledem k využití souboru s testovacími funkcemi vyžadujícímu knihovnu windows.h je tento projekt spustitelný pouze v prostředí Windows. Pro vytvoření grafu je pak využito knihovny wxMathPlot.

Program je řízen z hlavní třídy aplikace MainFrame. V konstruktoru této třídy jsou inicializovány seznamy pro výběr variant DE, testovací funkce a mutační funkce. Tato inicializace je provedena v rámci metody SetModsCFs, kde jsou veškeré nastavitelné možnosti uložené v polích podle příslušných seznamů, ve kterých mají být zobrazeny a pomocí funkce Append, definované nad prvkem rozevíracího seznamu ve frameworku, jsou do těchto seznamů přidány. Také jsou zde nastaveny základní parametry a je vykresleno prostředí grafu. Dále jsou ve třídě MainFrame uvedeny metody pro ovládání všech prvků v GUI. Po výběru příslušné modifikace DE (funkce v případě porovnání) z rozbalovacího výběru je v příslušné metodě vytvořena instance této modifikace (u porovnání vždy instance DE).

V metodě spouštěné po stisku tlačítka pro výpočet DE je nejprve zvolena podle výběru uživatele možnost výpočtu. Při nastavené jediné funkce je spuštěn výpočet metody evoluce třídy příslušející vybrané modifikaci a vypočítaná data jsou zkopírována do proměnných grafu a vykreslena.

Pokud je však prováděn výpočet porovnání, jsou zde postupně vytvářeny instance jednotlivých modifikací, kde jsou každé přiřazena totožná vstupní data. Poté jsou pro každou modifikaci načteny parametry z GUI a je proveden výpočet evoluce.

Pokud je spuštěn výpočet opakovaně a nejsou změněny všechny parametry, jsou nezadané hodnoty načteny z polí v GUI pomocí metody LoadPrevious. Součástí této metody je i výpis hodnot parametrů do seznamu pod poli pro zadávání.

Metoda pro export dat obdobně jako předchozí popsané začíná rozdělením podle počítaného způsobu. Ten má však vliv pouze na délku cyklu pro výpis hodnot. Vypočítaná data jsou pak pomocí 2 vnořených cyklů vypsána do souboru \*.txt. Příklad exportovaných hodnot je uveden na *Obr 15*. Pro snížení paměťových nároků v průběhu všech výpočtů vždy uchovávány pouze hodnoty ohodnocení účelové funkce (CV). Počet sloupců je dán zadaným

počtem běhů (pro jedinou funkci) nebo počtem modifikace (pro případ porovnání) a počet řádků je dán počtem generačních cyklů.

V případě exportu obrázku je uložena aktuálně zobrazená část grafu do souboru \*.bmp o pevné velikosti 800 x 500. Příkladem vyexportovaného grafu jsou *Obr. 17 - Obr. 27*. V algoritmu je nejprve vytvořen dialog jako proměnná typu `wxFileDialog`, která vytváří dialog ve svém konstruktoru. Pokud proběhne operace s dialogem v pořádku, je přes obrázkový buffer uložena bitmapa do souboru na zadané cestě.

```

1  |-1336.460227 -1366.764346 -1339.397467 -1352.067196 -1358.762803
2  -1343.787956 -1366.764346 -1339.397467 -1352.067196 -1358.762803
3  -1343.787956 -1366.764346 -1339.397467 -1372.348177 -1358.762803
4  -1343.787956 -1366.764346 -1339.397467 -1372.348177 -1358.762803
5  -1345.156269 -1366.764346 -1339.397467 -1372.348177 -1358.762803
6  -1345.156269 -1366.764346 -1342.283430 -1372.348177 -1358.762803
7  -1345.156269 -1366.764346 -1362.793614 -1372.348177 -1368.840366
8  -1345.156269 -1366.764346 -1374.036391 -1372.348177 -1370.262622
9  -1345.156269 -1371.397712 -1374.036391 -1372.348177 -1370.262622
10 -1352.673263 -1377.241065 -1374.036391 -1372.348177 -1370.262622
11 -1352.673263 -1377.241065 -1374.036391 -1379.910777 -1373.336554
12 -1369.134112 -1380.472501 -1374.036391 -1379.910777 -1373.336554
13 -1369.134112 -1380.472501 -1374.036391 -1383.744497 -1373.336554
14 -1369.134112 -1380.472501 -1374.036391 -1383.744497 -1379.827315
15 -1369.134112 -1381.835317 -1374.036391 -1383.744497 -1379.827315

```

*Obr 15: Ukázka exportovaných dat*

### 5.1.1 Třída `DEMain`

Tato abstraktní třída obsahuje veškeré proměnné a metody, které jsou společné pro všechny modifikace DE. Jedná se především o vektory pro ukládání dat v průběhu výpočtů a vykreslování. Je zde také uvedena metoda `writedata` sloužící k zápisu vypočítaných dat do vektoru k vykreslení.

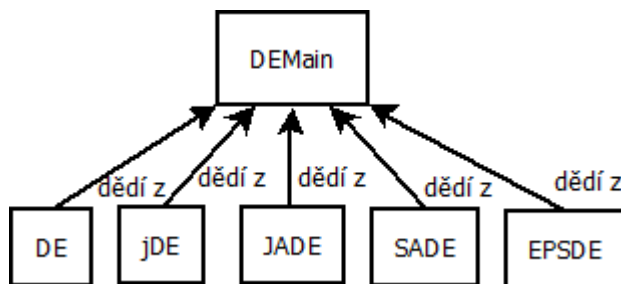
Další uvedenou metodou je `pickn`, která slouží k výběru  $n$  nestejných jedinců při výpočtu mutační funkce lišící se podle zadání. Tato metoda pracuje pouze s indexy jedinců a vybírá náhodně k aktuálnímu rodiči daný počet jedinců, případně snížený o výběr nejlepšího.

Jsou zde také deklarovány virtuální metody:

- `evoluce` - obsahuje celý proces výpočtu evoluce

- setstart - nastavení a inicializace veškerých proměnných využívaných v dané modifikaci DE
- zerogen - vygenerování prvotní populace

Těla těchto metod se s každou modifikací více či méně odlišují a proto jsou uvedeny v každé odvozené třídě zvlášť. Na Obr. 16 je zobrazena hierarchie použitých tříd s výjimkou třídy MainFrame.



Obr. 16: Schéma dědění tříd

### 5.1.2 Třídy adaptivních modifikací

Pro každou z modifikací DE (jDE, JADE, SADE a EPSDE) je vytvořena vlastní třída odvozená od DEMain. Podrobný popis funkce adaptivních modifikací je uveden v 3.7.1, 3.7.2, 3.7.3 a 3.7.4, V případě posledních 3 i včetně pseudokódu.

V každé třídě je metoda `mutace_funkce`, využívaná k výpočtu mutačního vektoru, nebo v případě, kdy je možnost využití více mutačních funkcí, i výběr této funkce a následné křížení. Dále každá modifikace obsahuje proměnné a metody specifické k výpočtu daného algoritmu. V případě SADE se jedná o metodu `vybermut`, zajišťující rozdělení mutačních metod pomocí stochastického univerzálního vzorkování.

U EPSDE se pak jedná o metodu `setvalues`, která přiřazuje jednotlivým jedincům v populaci náhodně zvolené strategie a řídicí konstanty. Tato metoda pracuje s dvourozměrným polem, kde je první rozměr 3 a druhý je dán velikostí populace. Pro každého jedince jsou vybírány hodnoty z okruhů pomocí vygenerování náhodného čísla modulo délkou pole.

Aplikaci je možné rozšířit několika způsoby:

### 5.1.3 Možnosti rozšíření

- Přidání dalších variant DE
- Rozšíření počtu testovacích funkcí

Přidání další varianty DE je možní provést vytvořením nové třídy požadované modifikace a jejím odvozením od základní třídy DEMain. Vzhledem k tomu, že tato třída obsahuje virtuální metody, je nutné, aby nově vytvořená třída obsahovala implementace těchto základních metod. Pro zpřístupnění třídy v menu aplikace je nutné požadovaný název přidat do příslušného pole v metodě SetModsCFs ve třídě MainFrame a přidat další možnost volby do switchu v metodě OnChoiceSet také ve třídě MainFrame.

V případě přidání nové testovací funkce je možné buď přidat další funkci do souboru test\_func.cpp do stejně nazvané funkce. Je také potřeba přidat další možnost do switchu v této funkci. Pro zobrazení této nově vytvořené třídy je stejně jako v případě nové varianty DE nutné aktualizovat pole v metodě SetModsCFs a přidat další možnost do switchu v metodě test\_fcn třídy DEMain.

## 5.2 Vyhodnocení

Pro vyhodnocení získaných dat v průběhu všech výpočtů jsem vytvořil v programu Wolfram Mathematica aplikaci. Tato aplikace načte požadovaná data ze zadané cesty a vypočítá pro tyto data statistické údaje, které jsou následně uloženy do přehledné tabulky.

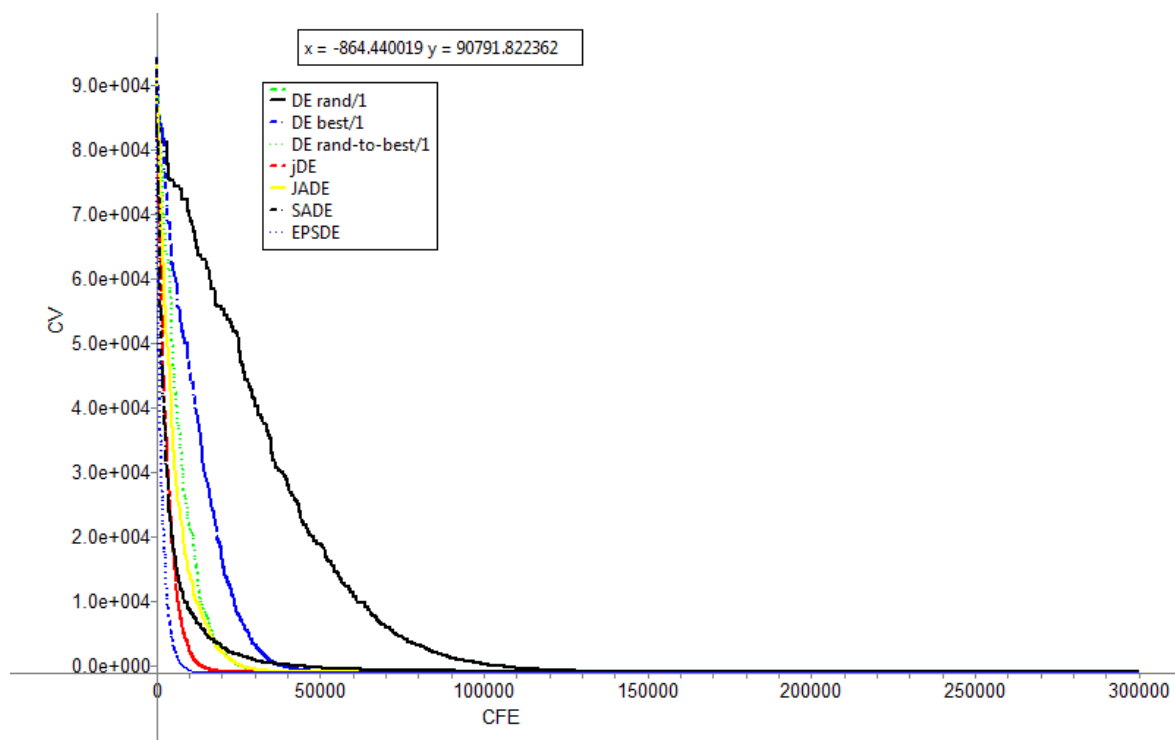
## 6 TESTOVÁNÍ

Testování modifikací DE jsem prováděl postupně na 12 testovacích funkcích. Nejprve jsem srovnal všechny metody graficky srovnáním průběhu průměrného nejlepšího jedince z 10 běhů pro každou modifikaci. Následně jsem vypsál do tabulky statisticky zpracované hodnoty pro 10 běhů z každé modifikace. Z průběhů na každé modifikaci jsem určoval průměr, medián, standardní odchylku, max. hodnotu CF a min hodnotu CF. Parametry pro testy jsem volil podle *Tab. 7*.

*Tab. 7: Parametry pro testování*

Specimen	{{Real, {-100,100}}}
Počet generací	1500, 3000
Počet opakování	10
Velikost populace	100
Počet dimenzí	30
CR	0,5
F	0,8

## 6.1 Funkce Sphere



Obr. 17: Srovnání variant DE na funkci sphere

Z grafu na Obr. 17 můžeme vidět, že nejlepších výsledků na této funkci (21)(14) dosahují modifikace EPSDE a jDE, kterým stačilo na ustálení 20 – 25 tisíc ohodnocení účelové funkce. Oproti tomu velice pomalou optimalizací má na této funkci základní DE s mutační strategií rand/1/bin, která potřebuje více než 100 tisíc ohodnocení účelové funkce k ustálení.

Tab. 8: Srovnání statistických dat po 1500 generacích

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-8,7618E+02	7,1587E+04	-8,6312E+02	-8,6627E+02	1,0286E+01
DE best/1	-9,1667E+02	6,2369E+04	-9,1667E+02	-9,1667E+02	4,7140E-07
DE rand-to-best/1	-9,1667E+02	7,5436E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
jDE	-9,1667E+02	6,5298E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
JADE	-9,1667E+02	6,9712E+04	-9,1667E+02	-9,1667E+02	4,2164E-07
SADE	-8,4860E+02	6,9527E+04	-8,1800E+02	-8,3091E+02	3,9816E+01
EPSDE	-9,1667E+02	6,1897E+04	-9,1667E+02	-9,1667E+02	1,1984E-13

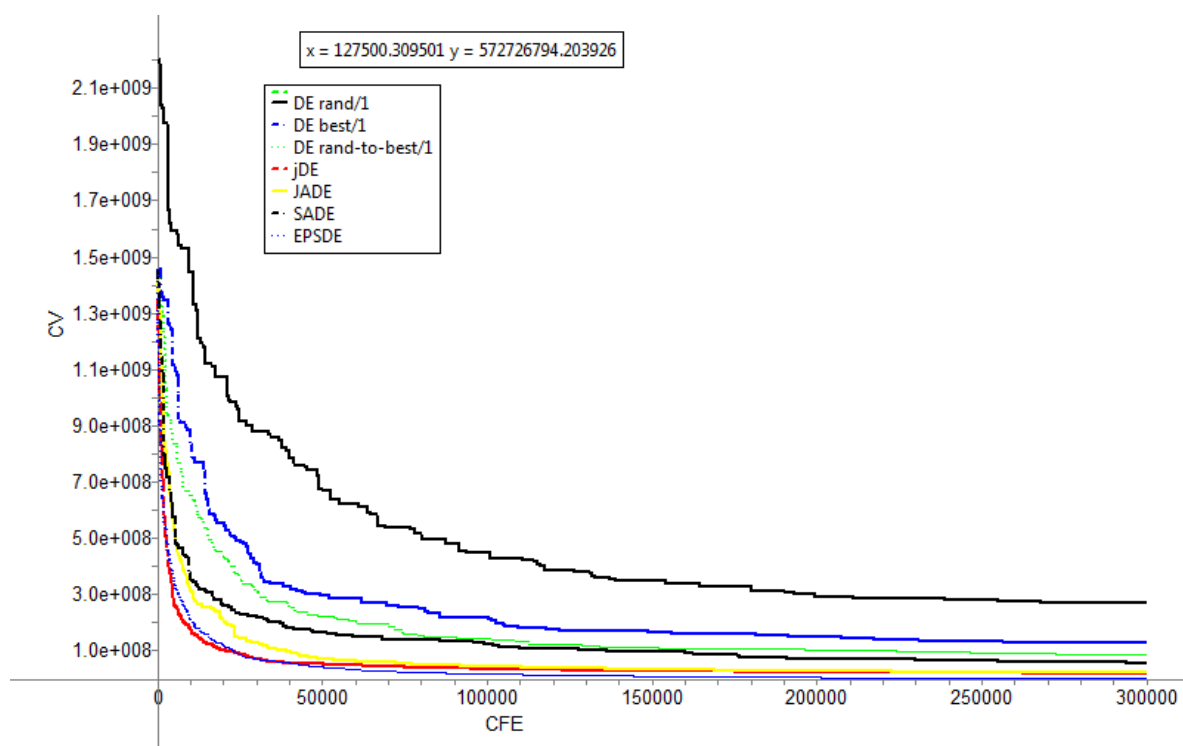


Tab. 9: Srovnání statistických dat po 1500 generacích

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-9,1666E+02	7,1587E+04	-9,1666E+02	-9,1666E+02	7,4780E-04
DE best/1	-9,1667E+02	6,2369E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
DE rand-to-best/1	-9,1667E+02	7,5436E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
jDE	-9,1667E+02	6,5298E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
JADE	-9,1667E+02	6,9712E+04	-9,1667E+02	-9,1667E+02	1,1984E-13
SADE	-9,0109E+02	6,9527E+04	-8,8406E+02	-8,9158E+02	2,5356E+01
EPSDE	-9,1667E+02	6,1897E+04	-9,1667E+02	-9,1667E+02	1,1984E-13

Z Tab. 8 a Tab. 9 je zřejmé, že na této velice jednoduché unimodální testovací funkci je dostatečný počet generací k nalezení minimální hodnoty menší než 1500, protože dle všech statistických hodnot průběhy všech variant DE již u 1500 generací téměř splývají. Pouze v případě SADE je i zde potřeba více generací k ustálení.

## 6.2 Rotovaná eliptická funkce



Obr. 18: Srovnání variant DE na funkci sphere

Na eliptické testovací funkci (22) lze vidět, že rozptyl dosažených minimálních hodnot je poměrně velký, tedy že daný počet ohodnocení účelové funkce není dostatečný k nalezení

globálního extrému Nejvíce vzdálená od hodnoty minima je zde klasická varianta DE s mutační strategií rand/1/bin. Naopak nejlepších hodnot zde dosáhly varianty EPSDE, JADE s jDE, jejichž výsledky jsou velice podobné.

*Tab. 10: Srovnání statistických dat po 1500 generacích  
na eliptické funkci*

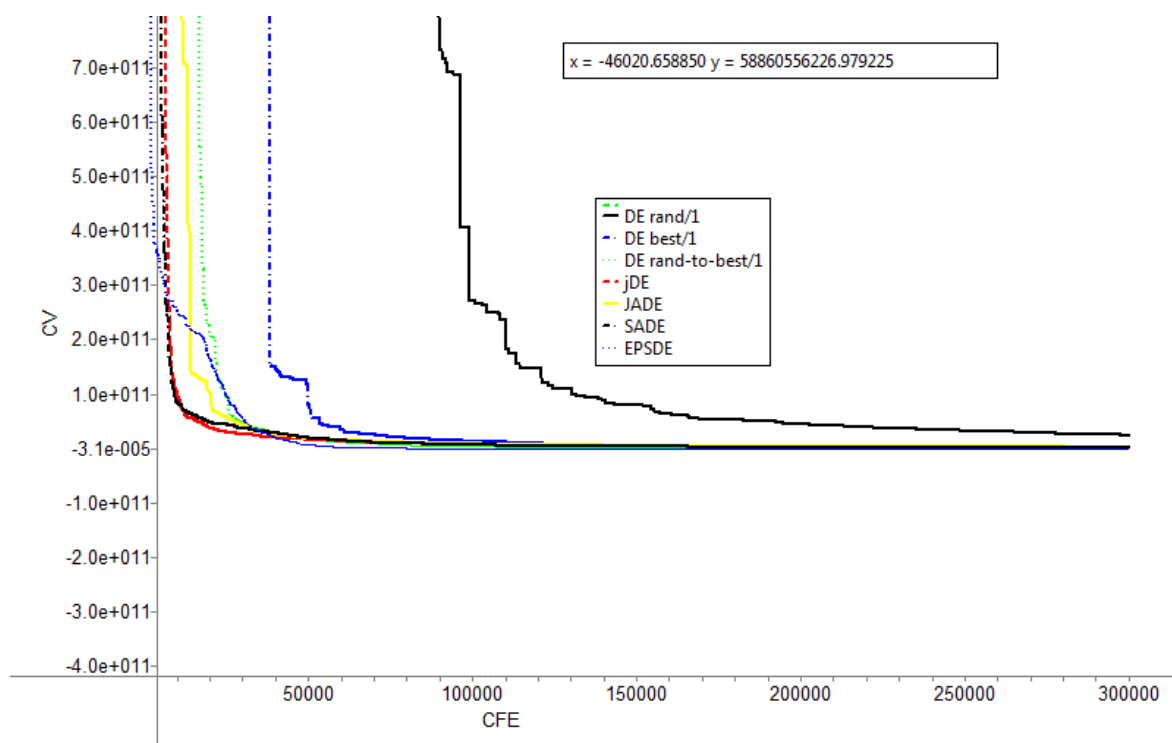
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	2,4182E+08	1,9245E+09	3,4751E+08	3,5445E+08	6,1356E+07
DE best/1	8,3269E+07	1,2549E+09	1,6978E+08	1,8079E+08	5,2491E+07
DE rand-to-best/1	8,0133E+07	1,3771E+09	1,2807E+08	1,2623E+08	3,0072E+07
jDE	1,6626E+06	8,9483E+08	5,2393E+06	4,9116E+06	2,1104E+06
JADE	2,2102E+07	1,6589E+09	3,0352E+07	2,8601E+07	6,7706E+06
SADE	7,2553E+07	2,3835E+08	8,9938E+07	8,8483E+07	1,4938E+07
EPSDE	5,5052E+06	5,6647E+08	1,1524E+07	1,2188E+07	4,0005E+06

*Tab. 11: Srovnání statistických dat po 3000 generacích  
na eliptické funkci*

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	1,7041E+08	1,9245E+09	2,4140E+08	2,3877E+08	5,6488E+07
DE best/1	7,3472E+07	1,2549E+09	1,1750E+08	1,1433E+08	3,6018E+07
DE rand-to-best/1	5,5687E+07	1,3771E+09	8,2460E+07	8,7298E+07	2,0224E+07
jDE	1,9385E+05	8,9483E+08	1,5141E+06	1,3590E+06	8,5233E+05
JADE	1,3415E+07	1,6589E+09	2,0970E+07	1,9516E+07	6,0799E+06
SADE	4,3916E+07	2,3835E+08	6,4974E+07	6,9815E+07	1,2856E+07
EPSDE	1,3164E+06	5,6647E+08	4,1049E+06	4,0953E+06	1,9445E+06

Rozptyl hodnot můžeme pozorovat i v *Tab. 10* a *Tab. 11*. Stejně jako v grafickém znázornění průběhů všech variant DE můžeme i zde vidět, že nejlepších výsledků na této testovací funkci dosahují varianty jDE, JADE a EPSDE, kde nejnižší hodnoty dosáhla po 3000 generacích jDE.

### 6.3 Rotovaná funkce Bent Cigar



Obr. 19: Srovnání variant DE na funkci Bent Cigar

Na testovací funkci Bent Cigar (23) vidíme, že rozptyl dosažených minimálních hodnot je ještě vyšší než v případě předchozí funkce, tedy že daný počet ohodnocení účelové funkce není dostatečný k nalezení globálního extrému. Nejvíce vzdálená od hodnoty minima je zde klasická varianta DE s mutační strategií rand/1/bin. Naopak nejlepších hodnot zde dosáhly varianty EPSDE, JADE s jDE, jejichž výsledky jsou velice podobné.

Tab. 12: Srovnání statistických dat po 1500 generacích  
na funkci Bent Cigar

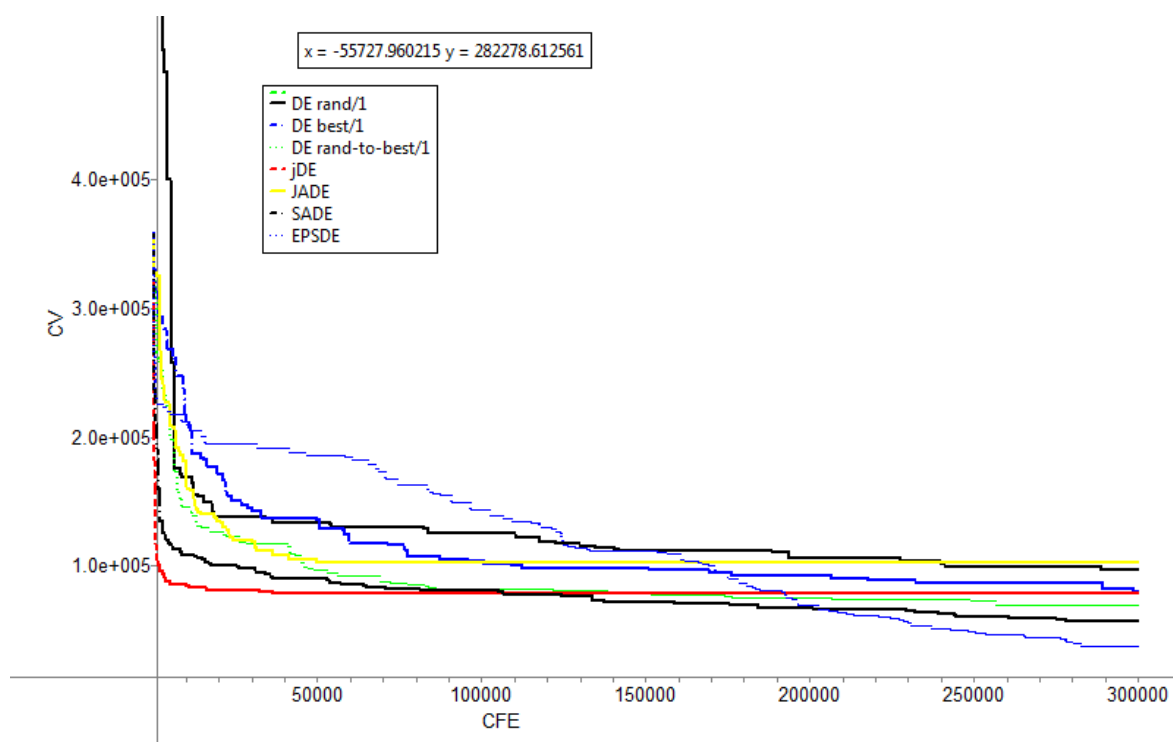
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	5,0746E+10	1,2243E+15	6,6655E+10	5,7645E+10	1,5274E+10
DE best/1	4,1443E+09	5,9374E+16	6,0777E+09	5,8596E+09	1,8142E+09
DE rand-to-best/1	5,9424E+08	8,3358E+15	1,0674E+09	1,0910E+09	3,0537E+08
jDE	1,2168E+07	1,1887E+18	1,6908E+07	1,3450E+07	9,7813E+06
JADE	4,0928E+09	1,1507E+18	6,1603E+09	5,8257E+09	1,4017E+09
SADE	1,6101E+09	2,4773E+17	3,6163E+09	3,4693E+09	1,3597E+09
EPSDE	1,0290E+07	8,4219E+17	1,9602E+07	1,7336E+07	8,3948E+06

Tab. 13: Srovnání statistických dat po 3000 generacích  
na funkci Bent Cigar

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	1,2582E+10	1,2243E+15	1,7963E+10	1,6725E+10	3,4118E+09
DE best/1	4,0026E+08	5,9374E+16	1,1367E+09	1,0333E+09	4,9940E+08
DE rand-to-best/1	1,1587E+08	8,3358E+15	2,0137E+08	2,0038E+08	6,4157E+07
jDE	1,0537E+07	1,1887E+18	1,4172E+07	1,0800E+07	1,0392E+07
JADE	1,3538E+09	1,1507E+18	2,5124E+09	2,4219E+09	6,5186E+08
SADE	4,0027E+08	2,4773E+17	1,5844E+09	1,6834E+09	7,0958E+08
EPSDE	7,3243E+06	8,4219E+17	1,3624E+07	1,0692E+07	5,2715E+06

Rozptyl hodnot můžeme pozorovat i v Tab. 12 a Tab. 13. Stejně jako v grafickém znázornění průběhů všech variant DE lze zde pozorovat, že nejlepších výsledků na této testovací funkci dosahují varianty jDE, JADE a EPSDE, kde nejnižší hodnoty dosáhla po 3000 generacích jDE.

#### 6.4 Rotovaná disková funkce



Obr. 20: Srovnání variant DE na diskové funkci

Obdobně jako u dvou předchozích funkcí pozorujeme i na diskové testovací funkci (24) rozptýl dosažených minimálních hodnot, který je však daleko nižší než u předchozí funkce. Stejně jako v minulých případech platí, že daný počet ohodnocení účelové funkce není dostatečný k nalezení globálního extrému. Nejvíce vzdálená od hodnoty minima je zde klasická varianta DE s mutační strategií rand/1/bin. Naopak nejlepších hodnot zde dosahují varianty EPSDE a SADE.

Tab. 14: Srovnání statistických dat po 1500 generacích  
na diskové funkci

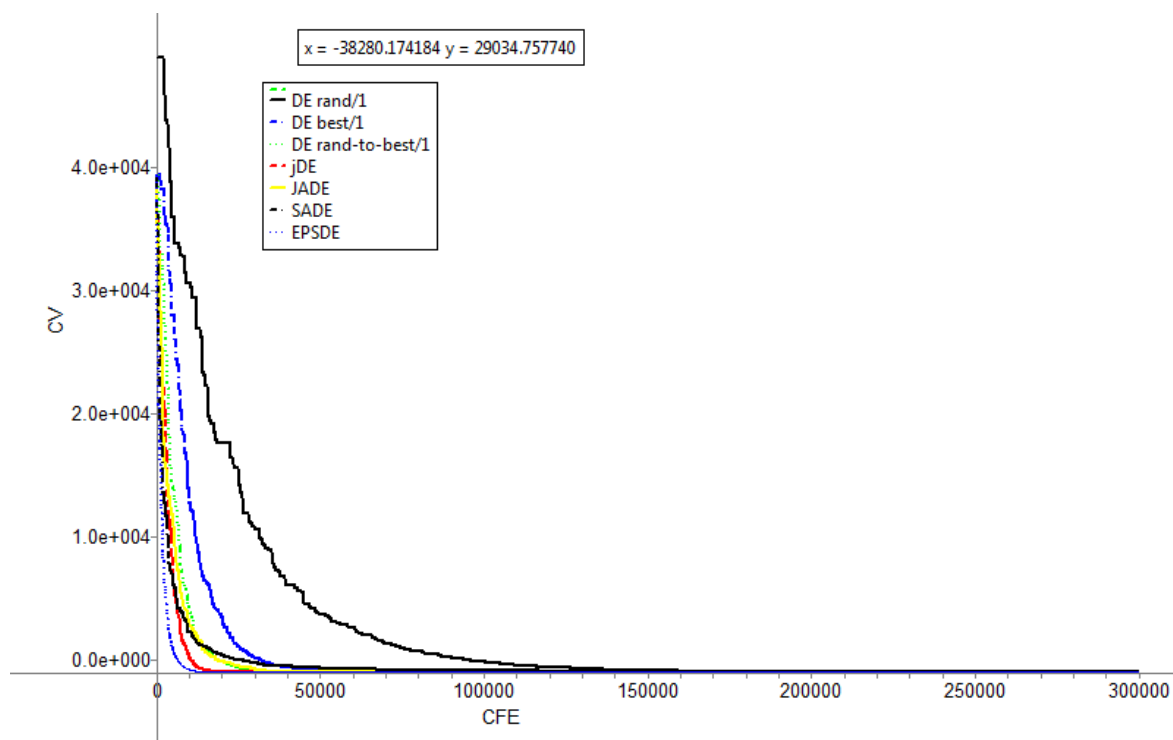
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	7,4519E+04	1,1996E+05	1,0781E+05	1,0189E+05	2,1408E+04
DE best/1	7,9637E+04	1,7121E+05	9,3294E+04	9,5005E+04	9,2859E+03
DE rand-to-best/1	4,8128E+04	2,0309E+05	8,0652E+04	8,4583E+04	1,4114E+04
jDE	9,9823E+03	1,2969E+05	2,0731E+04	1,8656E+04	1,1114E+04
JADE	5,2128E+04	1,3271E+05	7,3574E+04	6,8361E+04	1,5723E+04
SADE	5,0598E+04	1,0839E+05	6,6901E+04	6,5551E+04	1,1209E+04
EPSDE	6,7440E+04	1,1506E+05	1,0768E+05	1,0679E+05	2,4599E+04

Tab. 15: Srovnání statistických dat po 3000 generacích  
na diskové funkci

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	7,1063E+04	1,1996E+05	9,5492E+04	9,5804E+04	1,1833E+04
DE best/1	5,4644E+04	1,7121E+05	8,0623E+04	8,1165E+04	1,1917E+04
DE rand-to-best/1	4,8128E+04	2,0309E+05	6,8652E+04	6,9554E+04	9,2817E+03
jDE	2,0004E+02	1,2969E+05	2,7838E+03	1,3087E+03	3,9805E+03
JADE	5,0752E+04	1,3271E+05	7,2875E+04	6,8173E+04	1,5471E+04
SADE	3,6338E+04	1,0839E+05	5,6820E+04	5,8684E+04	1,2419E+04
EPSDE	2,6756E+04	1,1506E+05	3,7534E+04	3,2859E+04	1,0364E+04

Skutečnosti pozorovatelné v grafickém znázornění porovnání variant DE na této funkci jsou přesněji vyjádřené v Tab. 14 a Tab. 15. Můžeme zde vidět, že zatímco u funkce jako například DE s mutační strategií již mezi 1500 a 3000 generacemi nedochází ke zlepšení extrémní hodnoty a celkový postup je velice pomalý, pro variantu jDE nebo EPSDE dochází k velice razantnímu zlepšení.

## 6.5 Funkce rozdílných mocnin



Obr. 21: Srovnání variant DE na funkci rozdílných mocnin

Z grafu na Obr. 21 můžeme vidět, že nejlepších výsledků na funkci rozdílných mocnin (25) dosahují opět modifikace EPSDE a jDE, kterým stačilo na ustálení 20 – 25 tisíc ohodnocení účelové funkce. Oproti tomu velice pomalou optimalizaci má na této funkci základní DE s mutační strategií rand/1, která potřebuje více než 100 tisíc ohodnocení účelové funkce k ustálení.

Tab. 16: Srovnání statistických dat po 1500 generacích  
na funkci rozdílných mocnin

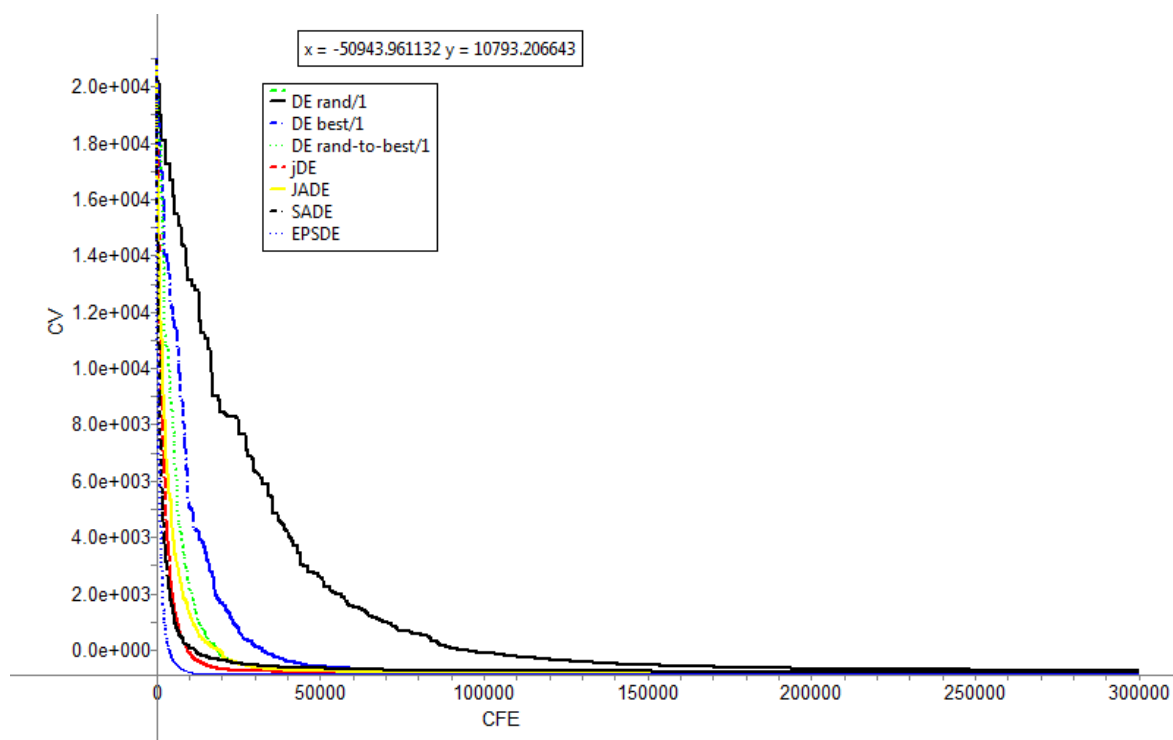
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-8,2416E+02	2,8612E+04	-7,8682E+02	-7,8809E+02	2,7229E+01
DE best/1	-9,7802E+02	2,7566E+04	-9,7802E+02	-9,7802E+02	4,2164E-07
DE rand-to-best/1	-9,7802E+02	2,4648E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
jDE	-9,7802E+02	2,3966E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
JADE	-9,7802E+02	2,6656E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
SADE	-9,2639E+02	2,2190E+04	-8,7620E+02	-9,1470E+02	7,6648E+01
EPSDE	-9,7802E+02	2,4386E+04	-9,7802E+02	-9,7802E+02	1,1984E-13

Tab. 17: Srovnání statistických dat po 3000 generacích  
na funkci rozdílných mocnin

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-9,7799E+02	2,8612E+04	-9,7798E+02	-9,7798E+02	1,5556E-02
DE best/1	-9,7802E+02	2,7566E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
DE rand-to-best/1	-9,7802E+02	2,4648E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
jDE	-9,7802E+02	2,3966E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
JADE	-9,7802E+02	2,6656E+04	-9,7802E+02	-9,7802E+02	1,1984E-13
SADE	-9,5624E+02	2,2190E+04	-9,3510E+02	-9,4359E+02	2,1891E+01
EPSDE	-9,7802E+02	2,4386E+04	-9,7802E+02	-9,7802E+02	1,1984E-13

Tato funkce je podobně jako první testovaná sférová funkce velice jednoduchá, takže k nalezení a ustálení zde dochází velice rychle a s výjimkou základní varianty DE se strategií rand/1 a SADE jsou již při 1500 generacích všechny metody na optimální hodnotě s minimálním rozptylem přes všech 10 běhů. Po 3000 generacích je již i základní varianta DE velice blízko optimální hodnotě, ale u SADE probíhá optimalizace velice pomalu.

## 6.6 Rotovaná Rosenbrockova funkce



Obr. 22: Srovnání variant DE na Rosenbrockově funkci

V grafu na Obr. 22 můžeme pozorovat, že nejlepších výsledků na Rosenbrockově funkci (26) dosahuje modifikace EPSDE, které stačilo na ustálení cca 20 tisíc ohodnocení účelové funkce. Oproti tomu velice pomalou optimalizaci má na této funkci základní DE s mutační strategií rand/1, která potřebuje více než 100 tisíc ohodnocení účelové funkce k ustálení.

*Tab. 18: Srovnání statistických dat po 1500 generacích  
na Rosenbrockově funkci*

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-5,8012E+02	9,9830E+03	-4,9490E+02	-4,8930E+02	5,8712E+01
DE best/1	-8,4155E+02	1,3257E+04	-8,2925E+02	-8,3764E+02	1,6384E+01
DE rand-to-best/1	-8,4245E+02	1,2789E+04	-8,3331E+02	-8,3976E+02	1,7080E+01
jDE	-8,4250E+02	1,3329E+04	-8,4245E+02	-8,4250E+02	1,5059E-01
JADE	-8,2942E+02	1,0541E+04	-8,1627E+02	-8,1424E+02	6,1454E+00
SADE	-7,7046E+02	1,3952E+04	-7,5193E+02	-7,5446E+02	1,3268E+01
EPSDE	-8,6115E+02	1,3785E+04	-8,6115E+02	-8,6115E+02	0

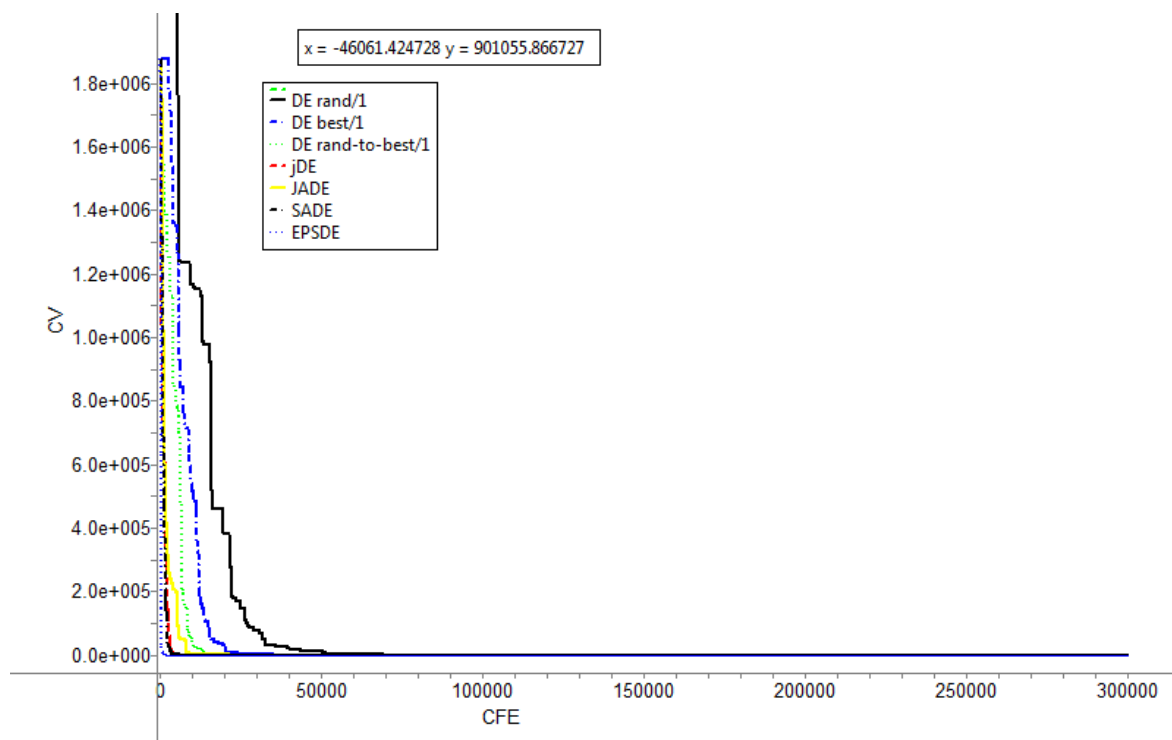
*Tab. 19: Srovnání statistických dat po 3000 generacích  
na Rosenbrockově funkci*

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-7,7145E+02	9,9830E+03	-7,2964E+02	-7,2923E+02	2,4511E+01
DE best/1	-8,4250E+02	1,3257E+04	-8,3489E+02	-8,4250E+02	1,3891E+01
DE rand-to-best/1	-8,4250E+02	1,2789E+04	-8,4010E+02	-8,4250E+02	7,4714E+00
jDE	-8,4250E+02	1,3329E+04	-8,4250E+02	-8,4250E+02	1,1984E-13
JADE	-8,3832E+02	1,0541E+04	-8,3243E+02	-8,3256E+02	4,7366E+00
SADE	-7,9145E+02	1,3952E+04	-7,7777E+02	-7,8106E+02	1,0863E+01
EPSDE	-8,6115E+02	1,3785E+04	-8,6115E+02	-8,6115E+02	0

Jak již bylo řečeno, tak na Rosenbrockově funkci dosahuje nejrychlejší optimalizace EPSDE, což je velice dobře viditelné na nulové hodnotě směrodatné odchylky. Tato hodnota znamená, že již v 1500 generacích je ve všech 10 bězích hodnota účelové funkce ustálena na totožné hodnotě.



## 6.7 Rotovaná funkce Schaffers F7



Obr. 23: Srovnání variant DE na funkci Schaffers F7

Stejně jako v mnoha předešlých případech je i na funkci Schaffers F7 (27) optimalizace velice rychlá. Nejlepších výsledků tu také dosahuje varianta EPSDE, následovaná dvěma velice podobnými funkcemi, co se týká rychlosti nalezení minima, jDE a SADE. Nejhorších výsledků i zde dosahuje základní varianta DE s mutační strategií rand/1/bin.

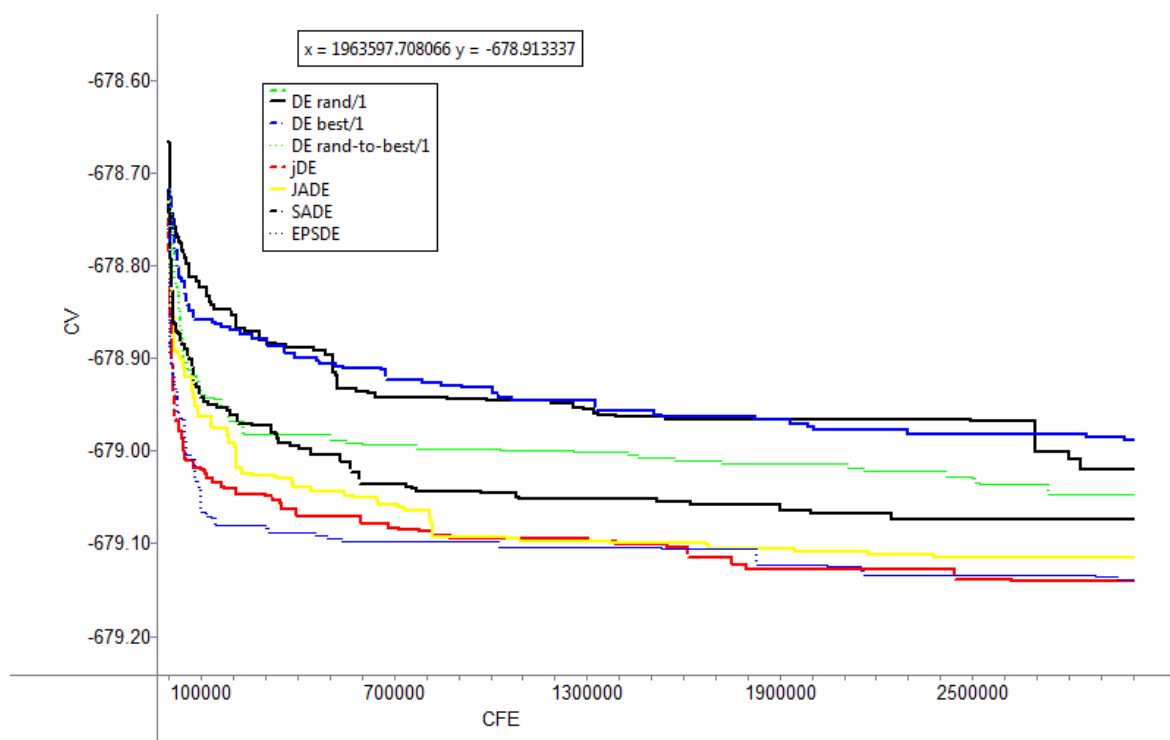
Tab. 20: Srovnání statistických dat po 1500 generacích  
na funkci Schaffers F7

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-5,8310E+02	6,4421E+05	-5,4977E+02	-5,4660E+02	2,1393E+01
DE best/1	-7,3217E+02	7,3799E+05	-7,0676E+02	-7,0372E+02	1,6547E+01
DE rand-to-best/1	-7,6339E+02	5,9926E+05	-7,3864E+02	-7,3400E+02	1,3500E+01
jDE	-7,9556E+02	7,0167E+04	-7,9139E+02	-7,9142E+02	2,6380E+00
JADE	-7,0090E+02	3,8855E+05	-6,8785E+02	-6,9021E+02	1,1954E+01
SADE	-7,3369E+02	9,5340E+05	-7,2137E+02	-7,1705E+02	8,7260E+00
EPSDE	-7,7511E+02	4,7811E+05	-7,4476E+02	-7,4668E+02	2,4615E+01

Tab. 21: Srovnání statistických dat po 3000 generacích  
na funkci Schaffers F7

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-6,9337E+02	6,4421E+05	-6,5521E+02	-6,5739E+02	2,0037E+01
DE best/1	-7,5434E+02	7,3799E+05	-7,4438E+02	-7,4643E+02	7,9714E+00
DE rand-to-best/1	-7,7435E+02	5,9926E+05	-7,6442E+02	-7,6556E+02	8,7231E+00
jDE	-7,9813E+02	7,0167E+04	-7,9630E+02	-7,9642E+02	1,5170E+00
JADE	-7,1521E+02	3,8855E+05	-6,9860E+02	-7,0041E+02	1,1520E+01
SADE	-7,5077E+02	9,5340E+05	-7,3958E+02	-7,4281E+02	1,1261E+01
EPSDE	-7,9453E+02	4,7811E+05	-7,8946E+02	-7,9060E+02	5,6264E+00

## 6.8 Rotovaná Ackleyho funkce



Obr. 24: Srovnání variant DE na Ackleyho funkci

Na Ackleyho funkci (28) i přes opticky velký rozptyl hodnot vidíme poměrně přesnou optimalizaci. Zde se již také jedná o multimodální funkci, kde bývá lokalizace extrému přes jejich větší počet velmi obtížná. Velmi podobných výsledků tu dosahují varianty jDE a EPSDE končící i po 30000 generacích na nejlepší hodnotě těsně následovány variantou JADE.

*Tab. 22: Srovnání statistických dat po 1500 generacích  
na Ackleyho funkci*

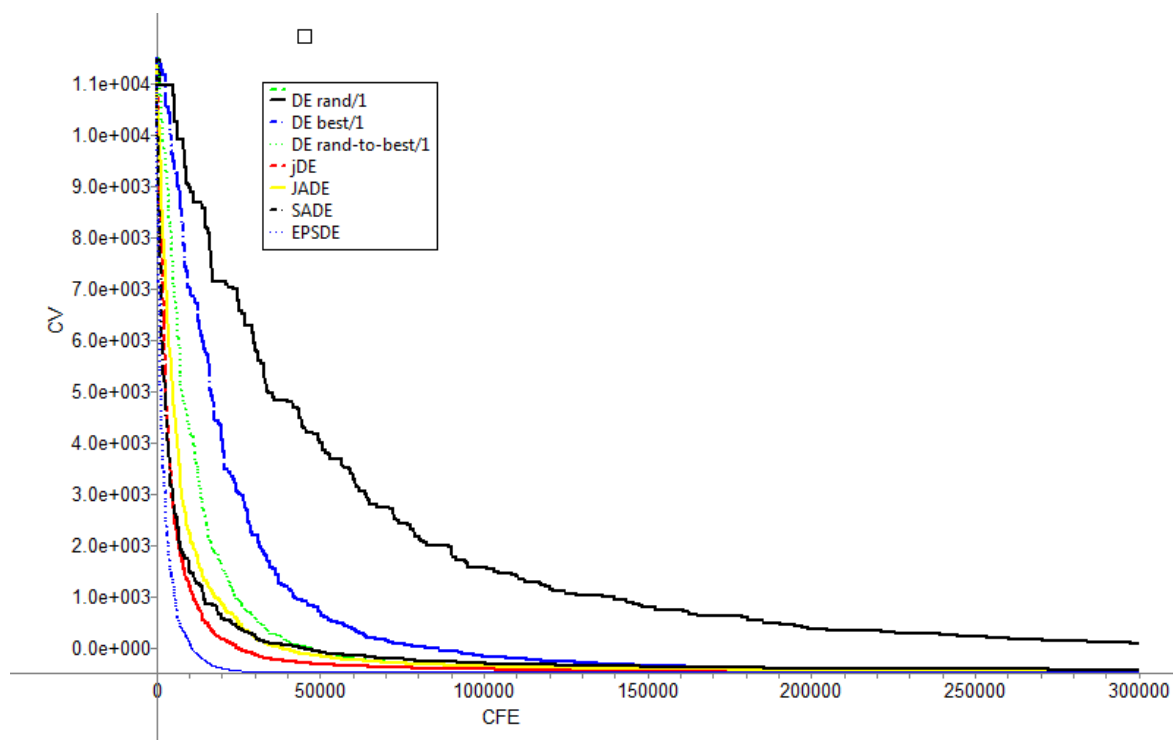
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-6,7896E+02	-6,7882E+02	-6,7886E+02	-6,7886E+02	8,0478E-02
DE best/1	-6,7901E+02	-6,7874E+02	-6,7890E+02	-6,7888E+02	5,9057E-02
DE rand-to-best/1	-6,7920E+02	-6,7872E+02	-6,7891E+02	-6,7890E+02	1,2105E-01
jDE	-6,7903E+02	-6,7869E+02	-6,7895E+02	-6,7895E+02	4,0108E-02
JADE	-6,7906E+02	-6,7883E+02	-6,7899E+02	-6,7898E+02	4,7957E-02
SADE	-6,7904E+02	-6,7881E+02	-6,7894E+02	-6,7893E+02	5,4357E-02
EPSDE	-6,7908E+02	-6,7881E+02	-6,7901E+02	-6,7900E+02	3,9958E-02

*Tab. 23: Srovnání statistických dat po 3000 generacích  
na Ackleyho funkci*

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-6,7896E+02	-6,7882E+02	-6,7888E+02	-6,7887E+02	5,5976E-02
DE best/1	-6,7901E+02	-6,7874E+02	-6,7890E+02	-6,7889E+02	5,6267E-02
DE rand-to-best/1	-6,7920E+02	-6,7872E+02	-6,7898E+02	-6,7898E+02	9,1301E-02
jDE	-6,7908E+02	-6,7869E+02	-6,7900E+02	-6,7899E+02	3,8609E-02
JADE	-6,7908E+02	-6,7883E+02	-6,7902E+02	-6,7902E+02	4,8294E-02
SADE	-6,7906E+02	-6,7881E+02	-6,7897E+02	-6,7899E+02	4,7632E-02
EPSDE	-6,7914E+02	-6,7881E+02	-6,7903E+02	-6,7903E+02	4,6635E-02

Z tabulek pro 1500 a 3000 generací pro tuto testovací funkci vidíme stejně jako z grafu velmi přesné hodnoty nalezených minim s drobnými rozdíly. Velice zajímavou hodnotou na této funkci je směrodatná odchylka, která pro všechny varianty DE vychází velmi nízká. To znamená, že rozptyl hodnot účelových funkcí napříč všemi provedenými běhy je minimální.

## 6.9 Rotovaná Griewankova funkce



Obr. 25: Srovnání variant DE na Griewankově funkci

Z grafu na Obr. 25 můžeme vidět, že nejlepší výsledky na Griewankově funkci (29) dosahuje opět modifikace EPSDE, které stačilo na ustálení cca 30 tisíc ohodnocení účelové funkce. V porovnání s ní většina modifikací potřebovala většina funkcí 100 až 150 tisíc ohodnocení účelové funkce. Výjimku tvoří velice pomalou optimalizující základní verze DE s mutační strategií rand/1/bin, která má na této funkci, která nestačilo 300 tisíc ohodnocení účelové funkce k ustálení.

Tab. 24: Srovnání statistických dat po 1500 generacích na Griewankově funkci

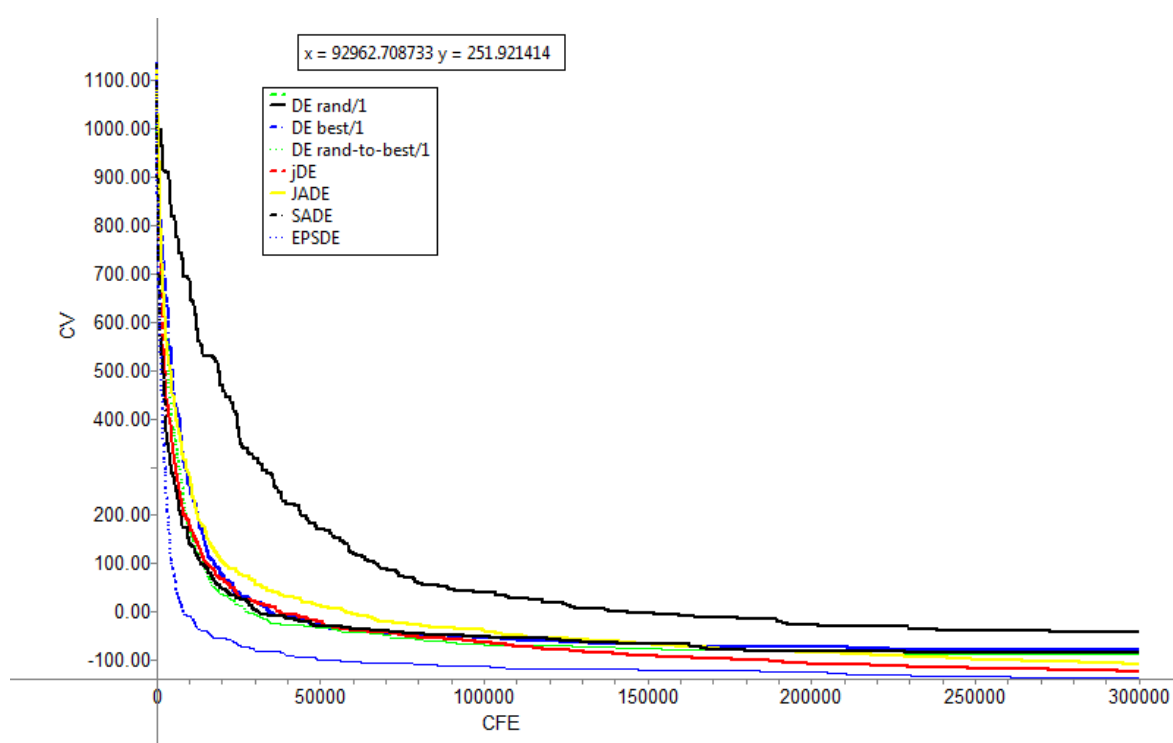
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	6,7029E+02	9,5676E+03	9,6330E+02	9,1401E+02	2,4360E+02
DE best/1	-4,0753E+02	7,9188E+03	-3,5810E+02	-3,5608E+02	3,1800E+01
DE rand-to-best/1	-4,4662E+02	1,1134E+04	-4,2823E+02	-4,3246E+02	2,0339E+01
jDE	-4,6869E+02	9,6191E+03	-4,6869E+02	-4,6869E+02	1,2449E-03
JADE	-4,3911E+02	6,5633E+03	-4,0922E+02	-4,0957E+02	1,6550E+01
SADE	-4,0805E+02	7,7007E+03	-3,7932E+02	-3,7674E+02	1,1822E+01
EPSDE	-4,6869E+02	7,5054E+03	-4,6869E+02	-4,6869E+02	5,9918E-14

Tab. 25: Srovnání statistických dat po 3000 generacích  
na Griewankově funkci

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-7,298E+01	9,568E+03	7,158E+01	8,007E+01	1,070E+02
DE best/1	-4,652E+02	7,919E+03	-4,568E+02	-4,583E+02	8,711E+00
DE rand-to-best/1	-4,682E+02	1,113E+04	-4,668E+02	-4,669E+02	1,151E+00
jDE	-4,687E+02	9,619E+03	-4,687E+02	-4,687E+02	5,992E-14
JADE	-4,601E+02	6,563E+03	-4,490E+02	-4,500E+02	6,341E+00
SADE	-4,484E+02	7,701E+03	-4,272E+02	-4,293E+02	1,317E+01
EPSDE	-4,687E+02	7,505E+03	-4,687E+02	-4,687E+02	5,992E-14

Z tabulek pro tuto funkci velice minimální postup většiny variant s výjimkou základní DE se strategií rand/1. Nejlepších výsledků zde dosahuje EPSDE i co do rozptylu napříč všemi běhy, kdy již v 1500 generacích jsou všechny tyto hodnoty téměř shodné. Po 3000 generacích již dosáhly téměř optimální hodnoty všechny varianty kromě již zmíněné DE využívající strategii rand/1.

## 6.10 Rotovaná Rastriginova funkce



Obr. 26: Srovnání variant DE na Rastriginově funkci

Podobně jako u eliptické a diskové funkce vidíme také na Rastriginově funkci (30), že daný počet ohodnocení účelové funkce není dostatečný k ustálení všech variant DE na optimální

hodnotě. Nejlepších výsledků obdobně jako v předchozích případech dosahují i zde varianty DE EPSDE, jDE a JADE.

Tab. 26: Srovnání statistických dat po 1500 generacích  
na Rastriginově funkci

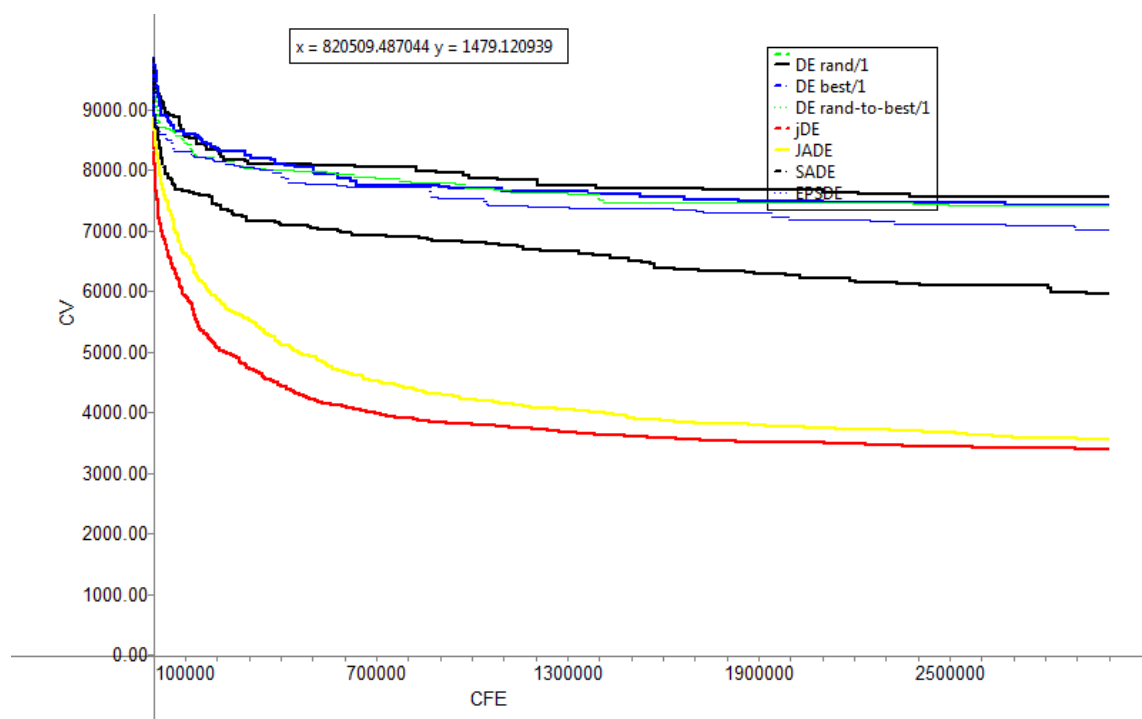
Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-3,9243E+01	7,7248E+02	-1,0655E+01	-7,2751E+00	2,2418E+01
DE best/1	-1,0845E+02	6,9078E+02	-6,5027E+01	-5,8895E+01	1,6721E+01
DE rand-to-best/1	-9,7371E+01	6,2633E+02	-7,9790E+01	-8,0666E+01	1,2444E+01
jDE	-1,8598E+02	9,7836E+02	-1,4453E+02	-1,4156E+02	2,5301E+01
JADE	-1,3946E+02	8,1752E+02	-1,0564E+02	-1,0819E+02	1,8718E+01
SADE	-1,0946E+02	7,9038E+02	-9,2274E+01	-9,0439E+01	1,0731E+01
EPSDE	-1,4718E+02	8,2910E+02	-1,2188E+02	-1,2070E+02	1,6498E+01

Tab. 27: Srovnání statistických dat po 3000 generacích  
na Rastriginově funkci

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	-8,4490E+01	7,7248E+02	-4,6695E+01	-4,4869E+01	1,5010E+01
DE best/1	-1,0845E+02	6,9078E+02	-8,7435E+01	-8,6341E+01	1,2433E+01
DE rand-to-best/1	-1,1750E+02	6,2633E+02	-9,3296E+01	-9,3406E+01	1,2423E+01
jDE	-2,6400E+02	9,7836E+02	-2,2426E+02	-2,2925E+02	3,4756E+01
JADE	-1,7296E+02	8,1752E+02	-1,4114E+02	-1,3389E+02	1,9023E+01
SADE	-1,4367E+02	7,9038E+02	-1,0327E+02	-1,0086E+02	1,7674E+01
EPSDE	-1,5941E+02	8,2910E+02	-1,3901E+02	-1,3991E+02	1,5712E+01

V tabulkách 26 a 27 pro tuto funkci vidíme velmi podobné hodnoty rozptylů napříč všemi variantami DE a to jak pro 1500 tak i 3000 generací. Mezi oběma body, ve kterých odečítáme statistická data, dosahuje téměř nulových změn základní varianta DE se strategií best/1. Naproti tomu největší změny v tomto intervalu dosáhla jDE.

## 6.11 Rotovaná Schwefelova funkce



Obr. 27: Srovnání variant DE na Schwefelově funkci

Na Schwefelově funkci (31) stejně jako na Ackleyho funkci pozorujeme mnohem větší počet ohodnocení účelové funkce nutný k nalezení optimální hodnoty než u ostatních variant. Na této testovací funkci můžeme jako na jediné vidět, že varianta EPSDE nedosahuje optimálních výsledků, ale spíše horších. Nejlepší výsledky mají na této funkci varianty jDE a JADE, jejichž hodnota je na konci zkoumaného intervalu nejlepší.

Tab. 28: Srovnání statistických dat po 1500 generacích  
na Schwefelově funkci

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	8,3711E+03	9,2449E+03	8,7158E+03	8,7240E+03	2,2563E+02
DE best/1	7,8673E+03	9,7665E+03	8,4435E+03	8,4337E+03	4,2906E+02
DE rand-to-best/1	7,3668E+03	8,1286E+03	8,0237E+03	8,0970E+03	3,4664E+02
jDE	6,0329E+03	8,4682E+03	7,2092E+03	7,3596E+03	4,7105E+02
JADE	5,4523E+03	9,0912E+03	5,9981E+03	6,0337E+03	2,9561E+02
SADE	7,0714E+03	9,3768E+03	7,7084E+03	7,7849E+03	2,8532E+02
EPSDE	7,4313E+03	8,7931E+03	8,1212E+03	8,2246E+03	3,1812E+02

*Tab. 29: Srovnání statistických dat po 1500 generacích  
na Schwefelově funkci*

Modifikace	Min	Max	Prumer	Median	Std odchylka
DE rand/1	7,6541E+03	9,2449E+03	8,3637E+03	8,5010E+03	3,9464E+02
DE best/1	7,8587E+03	9,7665E+03	8,1619E+03	8,0733E+03	2,3225E+02
DE rand-to-best/1	7,3668E+03	8,1286E+03	7,8884E+03	7,9924E+03	3,2977E+02
jDE	5,7495E+03	8,4682E+03	6,8660E+03	6,9582E+03	4,4900E+02
JADE	4,5747E+03	9,0912E+03	5,1254E+03	5,0967E+03	3,1941E+02
SADE	7,0714E+03	9,3768E+03	7,6564E+03	7,7416E+03	2,5778E+02
EPSDE	7,4313E+03	8,7931E+03	7,8691E+03	7,9008E+03	2,5288E+02

V tabulkách pro tuto funkci jasně vidíme, že na rozdíl od variant DE jDE a JADE, dosahujících v intervalu mezi 1500 a 3000 generacemi poměrně velké změny optimalizace, většina ostatních funkcí je již zde s velice minimálními změnami. I přes uvedené velké změny některých variant DE a relativní stálost jiných jsou pro všechny křivky velice podobné hodnoty odchylek napříč všemi běhy.



## 7 VYHODNOCENÍ

Jak bylo vidět na všech 12 testovacích funkcích, různé varianty DE dosahovaly různě přesných výsledků. Při srovnání na celé škále testovacích funkcí jsou až na drobné výjimky velmi schopné varianty DE EPSDE a jDE. Velmi použitelných výsledků dosahují většinou také varianty JADE, SADE a DE s mutační strategií rand-to-best/1. Naproti tomu, nejhorších výsledků téměř vždy dosáhla základní varianta DE se strategií rand/1.

Celkově se však dá říct, že pro dané typy funkcí je možné tyto základní varianty, až na výjimky na několika funkcích, využít. Je však třeba také dodat, že v případě eliptické nebo Schwefelovy funkce, kde by k nalezení minima pomocí všech variant DE bylo potřeba i několik desítek milionů ohodnocení účelové funkce, by tyto velmi složitě definované funkce zabraly velice dlouhou dobu výpočtu.

Oproti tomu, na jednodušších funkcích jsou tyto varianty velmi výkonné a schopné nalézt optimální hodnoty před dosažením 100 tisíc ohodnocení účelové funkce téměř v každém případě.

## ZÁVĚR

Cílem diplomové práce bylo vytvořit a porovnat klasickou verzi DE s jejími základními adaptivními modifikacemi. Na základě těchto porovnání se ukázalo, že adaptivní modifikace mohou být v algoritmu DE velice prospěšné, protože samotná DE trpí nutností vždy správně určit řídicí parametry. Ale i zde může volba špatné modifikace na danou funkci vést k problémům a nižší kvalitě optimalizace. Volba správného algoritmu je tedy stejně důležitá, jako je u základní verze volba parametrů, jelikož mezi nimi mohou vznikat na určitých problémech propastné rozdíly.

Adaptivní algoritmy SADE a JADE v porovnání s diferenciální evolucí byly většinou poměrně srovnatelné nebo lepší než základní varianty DE využívající mutačních strategií best/1 a rand-to-best/1. Naprosto nejhorších výsledků dosahoval v průběhu testů základní algoritmus DE se strategií rand/1. Nejlepších a většinou srovnatelných výsledků dosahovaly algoritmy jDE a EPSDE.

Adaptivní algoritmus diferenciální evoluce EPSDE mohu doporučit jako náhradu klasické varianty diferenciální evoluce. I přesto, že EPSDE nedosáhlo nejlepších výsledků v každém testu jej lze doporučit, protože v celkovém porovnání je tento algoritmus nejlepší.

Do práce nebylo možné zahrnout všechny adaptivní varianty diferenciální evoluce, protože těchto algoritmů postupně vzniká velké množství. Cílem této práce bylo pouze poskytnout náhled do této problematiky a tvorba základních algoritmů, ze kterých dnešní algoritmy vycházejí.

**SEZNAM POUŽITÉ LITERATURY**

- [1] ZELINKA, Ivan, et al *Handbook of Optimization: From Classical to Modern Approach*. 2012. vyd. Berlín: Springer-Verlag, 2012. ISBN 978-3-642-30503-0.
- [2] ZELINKA, Ivan, et al. *Evoluční výpočetní techniky : Principy a aplikace*. Praha : BEN, 2008. 536 s. ISBN 978-80-7300-218-3.
- [3] PRICE, Kenneth V, Rainer M STORN a Jouni A LAMPINEN. *Differential evolution: a practical approach to global optimization*. 1. vyd. Berlin: Springer, 2005, xix, 538 s. ISBN 35-402-0950-6.
- [4] ZELINKA, Ivan. *Umělá inteligence v problémech globální optimalizace*. 1. vyd. Praha: BEN - technická literatura, 2002, 189 s. ISBN 80-730-0069-5.
- [5] ZELINKA, Ivan, Zuzana OPLATKOVÁ a Roman ŠENKEŘÍK. *Aplikace umělé inteligence*. Vyd. 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2010, 151 s. ISBN 978-80-7318-898-6.
- [6] Kazíková, Anežka. *Algoritmus diferenciální evoluce s prvky deterministického chaosu (CHaosDE) v prostředí C/C++*. Zlín, 2013. *Bakalářská práce*. UTB ve Zlíně, Fakulta aplikované informatiky. Vedoucí práce Ing. Roman Šenkeřík, Ph.D.
- [7] BREST, J, S GREINER, B BOSKOVIC, M MERNIK a ZUMER. *Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems*. In: *IEEE Transactions on Evolutionary Computation*. New York: Institute of Electrical and Electronics Engineers, Inc., 646 - 657. ISSN 1089-778x.
- [8] ZHANG, J a A.C. SANDERSON. *JADE: Adaptive differential evolution with optional external archive*. In: *IEEE Transactions on Evolutionary Computation*. New York: Institute of Electrical and Electronics Engineers, Inc., 945 - 958. ISSN 1089-778x
- [9] QIN, A, V HUANG, P.N. SUGANTHAN. *Differential evolution algorithm with strategy adaptation for global numerical optimization*. In: *IEEE Transactions on Evolutionary Computation*. New York: Institute of Electrical and Electronics Engineers, Inc., s. 398-417. ISSN 1089-778x.
- [10] R. Mallipeddi , P. N. Suganthan , Q. K. Pan and M. F. Tasgetiren "Differential evolution algorithm with ensemble of parameters and mutation strategies", *Appl. Soft Comput.*, vol. 11, no. 2, pp.1679 -1696 2011
- [11] KARGER, Michal. *Algoritmus Diferenciální Evoluce s prvky deterministického chaosu (ChaosDE) v prostředí Mathematica [online]*. 2011. vyd. Zlín, 2011. 75 s.

*Dostupné z: <http://dspace.k.utb.cz/handle/10563/16562>. Diplomová práce. UTB ve Zlíně, Fakulta aplikované informatiky. Vedoucí práce Ing. Roman Šenkeřík, Ph.D.*

- [12] J. J. Liang, B-Y. Qu, P. N. Suganthan, Alfredo G. Hernández-Díaz, *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session and Competition on Real-Parameter Optimization*, Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, January 2013.
- [13] DE JONG, Kenneth A. *Evolutionary computation: a unified approach*. Vyd. 1. Cambridge: MIT Press, 2006, ix, 256 s. ISBN 02-620-4194-4

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

DE	Diferenciální evoluce.
F	Mutační konstanta.
CR	Práh křížení
jDE	Adaptivní varianta diferenciální evoluce
JADE	Adaptivní varianta diferenciální evoluce
SADE	Adaptivní varianta diferenciální evoluce
EPSDE	Adaptivní varianta diferenciální evoluce

**SEZNAM OBRÁZKŮ**

<i>Obr. 1: Obecný cyklus evoluce bez ukončovací podmínky .....</i>	<i>14</i>
<i>Obr. 2: Porovnání řešení problémů pomocí .....</i>	<i>16</i>
<i>Obr. 3: Binomické křížení .....</i>	<i>20</i>
<i>Obr. 4: Exponenciální křížení .....</i>	<i>21</i>
<i>Obr. 5: Průběh algoritmu diferenciální evoluce .....</i>	<i>24</i>
<i>Obr. 6: Pseudokód JADE .....</i>	<i>27</i>
<i>Obr. 7: Stochastické univerzální vzorkování .....</i>	<i>28</i>
<i>Obr. 8: Pseudokód SaDE .....</i>	<i>30</i>
<i>Obr. 9: Pseudokód EPSDE .....</i>	<i>31</i>
<i>Obr. 10: Prostředí programu po spuštění .....</i>	<i>39</i>
<i>Obr. 11: Prostředí programu při volbě výpočtu jediné varianty DE .....</i>	<i>42</i>
<i>Obr. 12: Prostředí programu při volbě výpočtu porovnání .....</i>	<i>42</i>
<i>Obr. 13: Ukázka exportovaných dat .....</i>	<i>44</i>
<i>Obr. 14: Srovnání variant DE na funkci sphere .....</i>	<i>48</i>
<i>Obr. 15: Srovnání variant DE na funkci sphere .....</i>	<i>49</i>
<i>Obr. 16: Srovnání variant DE na funkci Bent Cigar .....</i>	<i>51</i>
<i>Obr. 17: Srovnání variant DE na diskové funkci .....</i>	<i>52</i>
<i>Obr. 18: Srovnání variant DE na funkci rozdílných mocnin .....</i>	<i>54</i>
<i>Obr. 19: Srovnání variant DE na Rosenbrockově funkci .....</i>	<i>55</i>
<i>Obr. 20: Srovnání variant DE na funkci Schaffers F7 .....</i>	<i>57</i>
<i>Obr. 21: Srovnání variant DE na Ackleyho funkci .....</i>	<i>58</i>
<i>Obr. 22: Srovnání variant DE na Griewankově funkci .....</i>	<i>60</i>
<i>Obr. 23: Srovnání variant DE na Rastriginově funkci .....</i>	<i>61</i>
<i>Obr. 24: Srovnání variant DE na Schwefelově funkci .....</i>	<i>63</i>

**SEZNAM TABULEK**

<i>Tab. 1: Intervaly a doporučené hodnoty řídicích parametrů DE .....</i>	<i>18</i>
<i>Tab. 2: mutační strategie .....</i>	<i>19</i>
<i>Tab. 3: Tabulka úspěchů.....</i>	<i>29</i>
<i>Tab. 4: Tabulka neúspěchů.....</i>	<i>29</i>
<i>Tab. 5: Testovací funkce s uvedenou hodnotou penalizace .....</i>	<i>34</i>
<i>Tab. 6: Testovací funkce .....</i>	<i>35</i>
<i>Tab. 7: Parametry pro testování.....</i>	<i>47</i>
<i>Tab. 8: Srovnání statistických dat po 1500 generacích.....</i>	<i>48</i>
<i>Tab. 9: Srovnání statistických dat po 1500 generacích.....</i>	<i>49</i>
<i>Tab. 10: Srovnání statistických dat po 1500 generacích.....</i>	<i>50</i>
<i>Tab. 11: Srovnání statistických dat po 3000 generacích.....</i>	<i>50</i>
<i>Tab. 12: Srovnání statistických dat po 1500 generacích.....</i>	<i>51</i>
<i>Tab. 13: Srovnání statistických dat po 3000 generacích.....</i>	<i>52</i>
<i>Tab. 14: Srovnání statistických dat po 1500 generacích.....</i>	<i>53</i>
<i>Tab. 15: Srovnání statistických dat po 3000 generacích.....</i>	<i>53</i>
<i>Tab. 16: Srovnání statistických dat po 1500 generacích.....</i>	<i>54</i>
<i>Tab. 17: Srovnání statistických dat po 3000 generacích.....</i>	<i>55</i>
<i>Tab. 18: Srovnání statistických dat po 1500 generacích.....</i>	<i>56</i>
<i>Tab. 19: Srovnání statistických dat po 3000 generacích.....</i>	<i>56</i>
<i>Tab. 20: Srovnání statistických dat po 1500 generacích.....</i>	<i>57</i>
<i>Tab. 21: Srovnání statistických dat po 3000 generacích.....</i>	<i>58</i>
<i>Tab. 22: Srovnání statistických dat po 1500 generacích.....</i>	<i>59</i>
<i>Tab. 23: Srovnání statistických dat po 3000 generacích.....</i>	<i>59</i>
<i>Tab. 24: Srovnání statistických dat po 1500 generacích.....</i>	<i>60</i>
<i>Tab. 25: Srovnání statistických dat po 3000 generacích.....</i>	<i>61</i>
<i>Tab. 26: Srovnání statistických dat po 1500 generacích.....</i>	<i>62</i>
<i>Tab. 27: Srovnání statistických dat po 3000 generacích.....</i>	<i>62</i>
<i>Tab. 28: Srovnání statistických dat po 1500 generacích.....</i>	<i>63</i>
<i>Tab. 29: Srovnání statistických dat po 1500 generacích.....</i>	<i>64</i>

## SEZNAM PŘÍLOH

*PI CD s textem práce a zdrojovými kódy*