

Knihovna IEEE CEC 2014 testovacích funkcí v prostředí Mathematica

Roman Málek

Bakalářská práce
2015



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2014/2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Roman Málek**
Osobní číslo: **A11610**
Studijní program: **B3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**
Forma studia: **prezenční**

Téma práce: **Knihovna IEEE CEC 2014 testovacích funkcí v prostředí Mathematica**

Téma anglicky: **The IEEE CEC 2014 Benchmark Functions Library in the Mathematica Environment**

Zásady pro vypracování:

1. Vypracujte literární rešerši na dané téma.
2. Seznamte se s distribucí IEEE CEC 2014 Benchmark sady testovacích funkcí a jejich implementacemi v jednotlivých prostředích.
3. Naprogramujte testovací benchmark v prostředí Mathematica.
4. Proveďte analýzu a testování, vizualizace dat a kontrolu dat oproti implementacím v Matlab/C prostředích.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. LIANG J. J., B-Y. Qu, P. N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Technical Report 2013-11, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, December 2013.
2. CHRAMCOV, Bronislav. Základy práce v prostředí Mathematica. Vyd. 2. Zlín: Univerzita Tomáše Bati, 2006, 122 s. ISBN 80-7318-510-5.
3. TROTT, Michael. The Mathematica guidebook for symbolics. New York: Springer, 2006, xxxviii, 1453 s. ISBN 0-387-95020-6.
4. TROTT, Michael. The Mathematica guidebook for numerics. New York: Springer, 2006, xxxv, 1208 s. ISBN 0-387-95011-7.
5. WOLFRAM, Stephen. The mathematica book. 5th ed. Champaign: Wolfram Media, 2003. ISBN 1579550223.
6. Wolfram [online]. 2015 [cit. 2015-01-30]. Dostupné z: <http://www.wolfram.com/>

Vedoucí bakalářské práce:

doc. Ing. Roman Šenkeřík, Ph.D.
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce:

6. března 2015

Termín odevzdání bakalářské práce:

22. května 2015

Ve Zlíně dne 6. března 2015



doc. Mgr. Milan Adámek, Ph.D.
děkan

L.S.

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

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 připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- 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/bakalářské 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 22.5.2015

Mašek

.....
podpis diplomanta

ABSTRAKT

Bakalářské práce se zabývá vytvořením rozsáhlé a komplexní knihovny IEEE CEC 2014 testovacích (benchmark) funkcí v prostředí Mathematica. Tyto testovací funkce budou vytvořeny tak, aby je bylo možné dále na FAI využít se stávajícími implementacemi výkonných algoritmů.

Klíčová slova: benchmark, testovací funkce

ABSTRACT

Bachelor's thesis deals with create an extensive and complex library IEEE CEC 2014 testing (benchmark) functions in Mathematica. These testing functions will be created so that they can continue to use the FAI with existing implementations powerful algorithms.

Keywords: benchmark, testing functions

Rád bych poděkoval vedoucímu doc. Ing. Romanu Šenkeříkovi, Ph.D. za cenné podněty, rady a připomínky při zpracovávání bakalářské práce.

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 OPTIMALIZACE	11
1.1 ÚČELOVÁ FUNKCE.....	12
1.2 SOFT-COMPUTING	12
1.3 HEURISTICKÉ ALGORITMY.....	12
1.3.1 Metoda lokálního hledání.....	13
1.3.2 Horolezecký algoritmus	13
1.3.3 Zakázaná prohledávání.....	13
1.3.4 Simulované žhání.....	14
1.3.5 ACO (Ant Colony Optimization).....	14
1.4 EVOLUČNÍ ALGORITMY	14
1.4.1 Genetické algoritmy	15
2 TESTOVACÍ FUNKCE	16
2.1 HISTORIE CEC BENCHMARKU	16
2.1.1 CEC 2005	16
2.1.2 CEC 2013	16
2.2 DEFINICE TESTOVACÍCH FUNKCÍ IEEE CEC 2014.....	17
2.3 DEFINICE ZÁKLADNÍCH FUNKCÍ.....	19
2.3.1 Rotated High Conditioned Elliptic Function	21
2.3.2 Rotated Bent Cigar Function.....	22
2.3.3 Rotated Discus Function	23
2.3.4 Shifted and Rotated Rosenbrock's Function.....	24
2.3.5 Shifted and Rotated Ackley's Function	25
2.3.6 Shifted and Rotated Weierstrass Function	26
2.3.7 Shifted and Rotated Griewank's Function	27
2.3.8 Shifted Rastrigin's Function	28
2.3.9 Shifted and Rotated Rastrigin's Function	29
2.3.10 Shifted Schwefel's Function	30
2.3.11 Shifted and Rotated Schwefel's Function.....	31
2.3.12 Shifted and Rotated Katsuura Function	32
2.3.13 Shifted and Rotated HappyCat Function.....	33
2.3.14 Shifted and Rotated HGBat Function	34
2.3.15 Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function.....	35
2.3.16 Shifted and Rotated Expanded Scaffer's F6 Function	36
II PRAKTICKÁ ČÁST	37
3 IMPLEMENTACE KNIHOVNY TESTOVACÍCH FUNKCÍ	38

3.1	FUNKCE SHIFTFUNC.....	38
3.2	FUNKCE ROTFUNC	38
3.3	FUNKCE CFCAL	39
3.4	FUNKCE SRFUNC.....	40
3.5	FUNKCE ELLIPSFUNC	40
3.6	FUNKCE HF01	41
3.7	FUNKCE CF01.....	42
4	VIZUALIZACE VÝSLEDKŮ.....	43
5	KONTROLA DAT OPROTI IMPLEMENTACÍM V MATLAB/C PROSTŘEDÍCH.....	45
5.1	SROVNÁNÍ VÝSLEDKŮ FUNKCÍ.....	45
	ZÁVĚR	47
	SEZNAM POUŽITÉ LITERATURY.....	48
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	49
	SEZNAM OBRÁZKŮ	50
	SEZNAM TABULEK.....	52
	SEZNAM PŘÍLOH.....	53

ÚVOD

Pro testování optimalizačních algoritmů se využívá dvou rozdílných postupů. V prvním se obvykle vychází z již existujících příkladů, které již byly řešeny jinými algoritmy. Výsledky právě testovaného algoritmu se pak porovnají s výsledky již existujícími. Druhý způsob spočívá v tom, že se použije množina testovacích funkcí, obsahující funkce s různými vlastnostmi, jako je nelinearita, různé patologie typu rovina okolo extrému apod. Vzhledem k tomu, že jsou známy analytické vztahy, je u většiny z nich velmi jednoduché vypočítat pozici a hodnotu extrému pro libovolnou dimenzi.

Tato práce se zabývá druhým uvedeným způsobem testování optimalizačních algoritmů. V této práci byla vytvořena knihovna IEEE CEC 2014 testovacích funkcí pro testování výkonných (optimalizačních) algoritmů. Testovací funkce jsou implementovány v prostředí softwaru Wolfram Mathematica.

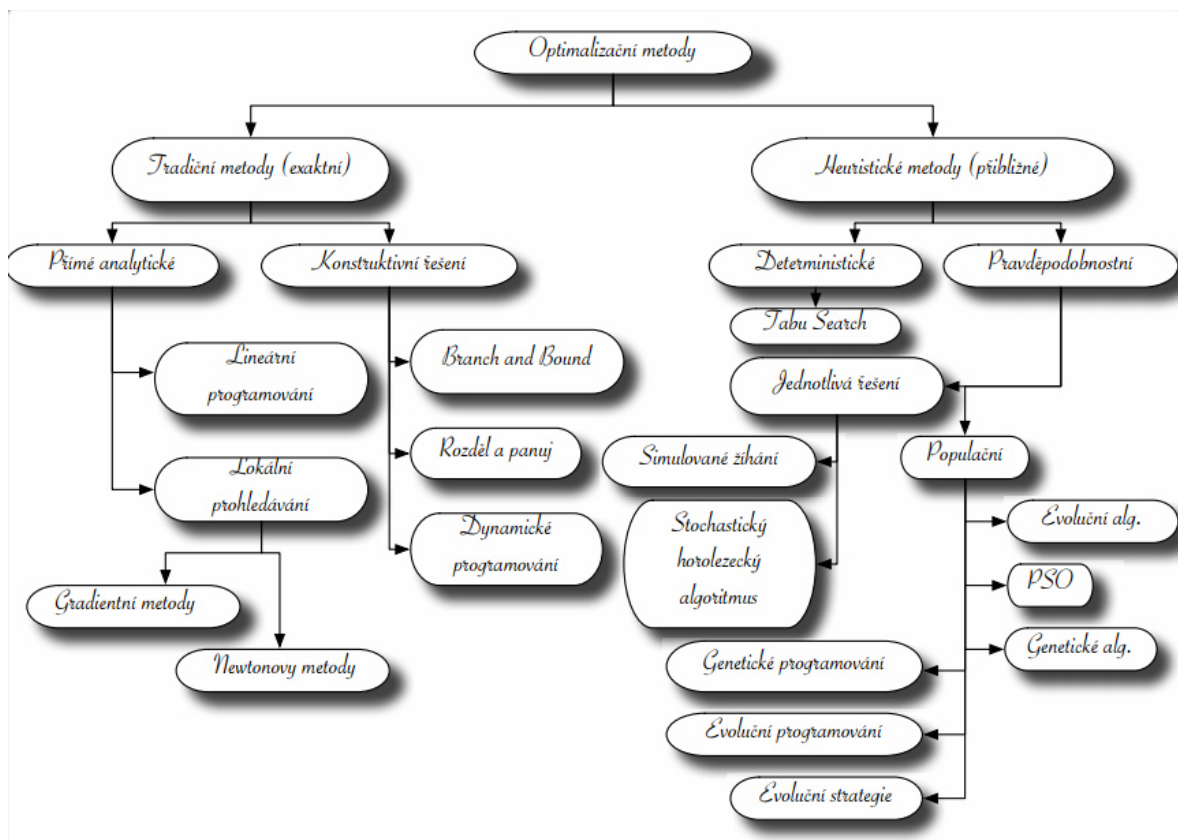
Teoretická část se zabývá problémem optimalizace, heuristickými metodami (pravděpodobnostními a evolučními algoritmy)

V praktické části jsou uvedeny příklady implementací několika testovacích funkcí a příklad použití navrhnuté aplikace pro zobrazení 1D a 2D řezu vybrané funkce. Na závěr byly srovnány výsledky funkcí implementovaných v prostředí Wolframu Mathematica s výsledky funkcí implementovaných v Matlabu/C.

I. TEORETICKÁ ČÁST

1 OPTIMALIZACE

Nemožnost nalézt deterministický algoritmus obecně řešící úlohu globální optimalizace v polynomiálním čase vedla k využití algoritmů stochastických, které sice nemohou garantovat nalezení řešení v konečném počtu kroků, ale často pomohou nalézt v přijatelném čase řešení prakticky použitelné. [6]



[8]

Obr. 1. Optimalizační metody

V praxi se rozlišuje mnoho typů optimalizačních metod, jako jsou například heuristické (pravděpodobnostní, deterministické, atd.) viz. Obr.1. Všechny typy optimalizačních úloh se však dají zapsat obecně takto:

$$\arg \min_{x \in \Omega} f$$

$$f : \Omega \rightarrow R$$

kde f je účelová funkce a Ω prostor přípustných řešení. Dělení úloh do zmíněných skupin je velmi podstatné pro klasické deterministické postupy řešení. Určuje tak, jaký deterministický postup může být použit, případně jakou je potřeba udělat transformaci, či aproxi-

maci. Heuristické algoritmy se zpravidla používají na ty typy optimalizace, kde klasické metody nejsou nebo selhávají. Obecně jsou však heuristické metody benevolentnější a dají se v podstatě nasadit na libovolnou optimalizační úlohu. [6]

1.1 Účelová funkce

Na každou účelovou funkci lze nahlížet jako na geometrický problém, v jehož rámci se hledá nejnížší (minimum) či nejvyšší (maximum) pozice na ploše ležící v $(N+1)$ - rozměrném prostoru, pro kterou se někdy používá výraz „hyperplocha“ či „prostor možných řešení“ daného problému. Počet dimenzí N je dán počtem optimalizovaných argumentů účelové funkce. Má-li optimalizovaná funkce např. šest argumentů (nezávisle proměnných), pak se hledá extrém na šestirozměrné ploše v sedmírozměrném prostoru, kde sedmá dimenze je návratová hodnota účelové funkce. [7]

1.2 Soft-Computing

Pro Soft-Computingu (SC) není důležitý model mechanismů, kterými se chování studovaného systému řídí (např. fyzikální model či ekonomický model ve formě rovnic popisujících daný mechanismus - tedy „vnitřní znalosti“). Důležitá je znalost o chování systému, nikoliv znalostí o příčinách chování.

Jedná se o tzv. praktickou optimalizaci, tedy stav, kdy není dosaženo teoretického optima, ale z praktického hlediska se jedná o již tak minimální rozdíl. Další vylepšování by ničemu příliš nepomohlo.

Základními kritérii jsou jednoduchost, implementovatelnost a robustnost (např. při změně některých veličin je fyzikální model již nefunkční, zato pravidla nevycházejí z fyzikální podstaty a popisují vnější znalost, která zůstává víceméně stejná). [4]

1.3 Heuristické algoritmy

Heuristika je zkusmé řešení problému. Používá se nejčastěji tam, kde neexistuje přesný deterministický algoritmus, jak řešení nalézt. Nebo postup řešení znám je, ale jeho provedení není kvůli jeho náročnosti (většinou časové) přijatelné. Slovo heuristika pochází z

řeckého slova heuriskó, které má význam nalézt nebo objevit. Heuristické algoritmy jsou založeny na určitém způsobu prohledávání prostoru přípustných řešení. Tento způsob často vychází z náhody, intuice, analogie a zkušenosti. Heuristiky obecně nezaručují nalezení optimálního (nejlepšího) řešení. V praxi však často dávají dostatečně dobrá řešení v přijatelném čase. [5]

1.3.1 Metoda lokálního hledání

Je nejjednodušší, ale nejméně efektivní heuristická metoda. Tento algoritmus trpí základní nevýhodou gradientových metod, tzn. že nejspíše skončí v lokálním optimu a nedosáhne globálního optima.

Zvolí se počáteční přípustné, pro něj se vygenerují všechna sousední řešení a vypočítají se pro ně účelové funkce. Ze všech sousedních řešení vybereme takové, které má hodnotu účelové funkce co nejnižší a pokud je lepší jako aktuální řešení, zvolíme jej jako nové aktuální řešení. Dokud není splněna podmínka ukončení algoritmu, kterou je nalezení lepšího řešení mezi sousedními řešeními, pokračujeme v prohledání sousedních řešení. [5]

1.3.2 Horolezecký algoritmus

Je jednoduchou modifikací lokálního hledání. Obdobně jako u lokálního hledání v každém kroku vybíráme nejlepší řešení ze sousedních řešení, ale máme jinou podmínku ukončení. Podmínkou ukončení algoritmu je provedení určitého počtu iterací.

Základní nevýhodou horolezeckého algoritmu je, že se při určitém počtu iteračních kroků vrací k lokálně optimálnímu řešení, které se již vyskytlo v některém z předcházejících kroků (problém zacyklení). [5]

1.3.3 Zakázaná prohledávání

Je vylepšená verze horolezeckého algoritmu. Jejím tvůrcem je prof. Fred Glover z University of Colorado. Vylepšení spočívá v tom, že je do horolezeckého algoritmu zavedena tzv. krátkodobá paměť, jejímž úkolem je pamatovat si ty transformace, pomocí nichž byl vypočítán aktuální střed. To má v konečném důsledku ten efekt, že nedochází k zacyklení díky zakázanému použití těchto transformací. Odtud také plyne název „Tabu Search“. Tato metoda byla vylepšena ještě o tzv. dlouhodobou paměť, která obsahuje transformace, jež nejsou v paměti krátkodobé, ale byly často použity. Jejich použití je pak penalizováno, což

snižuje četnost jejich použití. Na rozdíl od algoritmu horolezeckého Tabu Search tak snadno neuvízne v lokálních extrémech. [7]

1.3.4 Simulované žíhání

Vylepšuje horolezecký algoritmus tím, že s jistou pravděpodobností připouští i „krok k horšímu“ (tím je umožněno opustit lokální extrém). Hlavní myšlenka algoritmu byla inspirovaná chováním krystalických látek během procesu žíhání. Při žíhání se látka výrazně zahřeje tak, že atomy s vysokou energií přestanou být vázané do původně defektních krystalických mřížek. Postupným chladnutím se atomy usadí v novém krystalickém uspořádání s minimální energií. [6]

1.3.5 ACO (Ant Colony Optimization)

Je to algoritmus, jehož činnost napodobuje chování mravenců v kolonii. Jeho princip je následující. Necht' existuje zdroj mravenců (mraveniště) a cíl jejich snažení (potrava). Když jsou vypuštěni, tak po nějaké době dojde k tomu, že všichni mravenci se pohybují po kratší (optimální) cestě mezi zdrojem a cílem. Tento efekt, kdy mravenci naleznou optimální cestu je dán faktem, že si svou cestu značkují feromonem. Pokud dorazí první mravenec k rozcestí dvou cest, které vedou ke stejnému cíli, pak je jeho rozhodnutí, po které cestě se vydá, náhodné. Ti, kteří najdou potravu začnou cestu značkovat a při návratu jsou díky těmto značkám při rozhodování ovlivněni ve prospěch této cesty. Při návratu ji označují podruhé, což opět zvyšuje pravděpodobnost rozhodnutí dalších mravenců v její prospěch. Tyto principy jsou použity v ACO algoritmu. Feromon je zde nahrazen vahou, která je přidělena dané cestě vedoucí k cíli. Tato váha je aditivní, což umožňuje přidávat další „feromony“ od dalších „mravenců“. V ACO algoritmu je zohledněn i fakt vypařování feromonů tak, že váhy u jednotlivých spojů s časem slábnou. To zvyšuje robustnost algoritmu z hlediska nalezení globálního extrému. [7]

1.4 Evoluční algoritmy

Evoluční algoritmy přísluší do třídy stochastických prohledávacích algoritmů a pracují s náhodnými změnami navrhovaných řešení. V případě že jsou nová řešení výhodnější, nahrazují řešení předchozí. Důležitým rysem evolučních algoritmů je jejich prohledávací strategie založena na populacích a využívají heuristiky, které upravují populaci tak, aby

vlastnosti populace zlepšovaly. Jejich uplatnění je především v praktických problémech, které nejsou jinými metodami řešitelné.

Každá populace obsahuje jedince, kde každý jedinec představuje jedno řešení daného problému. Jak populace probíhá evolucí, řešení se zlepšují (vznikají nové generace). Tento vývoj se uskutečňuje aplikací evolučních operátorů, nejdůležitějšími evolučními operátory jsou tyto:

- *selekce* – Silnější jedinci z populace mají větší pravděpodobnost svého přežití i předání svých vlastností potomkům.
- *křížení* (rekombinace) – Dva nebo více jedinců z populace si vymění informace a vzniknou tak noví jedinci kombinující vlastnosti rodičů.
- *mutace* – Genetická informace zakódovaná v jedinci může být náhodně modifikována. [2][3][6]

1.4.1 Genetické algoritmy

Základní myšlenkou genetických algoritmů je analogie s evolučními procesy probíhajícími v biologických systémech. Podle Darwinovy teorie přirozeného výběru přežívají jen nejlépe přizpůsobení jedinci populace. Mírou přizpůsobení je tzv. “fitness” jedince. V biologii je fitness chápána jako relativní schopnost přežití a reprodukce genotypu jedince. Biologická evoluce je změna obsahu genetické informace populace v průběhu mnoha generací směrem k vyšším hodnotám fitness. Jedinci s vyšší fitness mají větší pravděpodobnost přežití a větší pravděpodobnost reprodukce svých genů do generace potomků. Kromě reprodukce se v populačním vývoji uplatňuje mutace. V genetických algoritmech je fitness kladné číslo přiřazené genetické informaci jedince. Tato genetická informace jedince (chromozóm) se obvykle vyjadřuje bitovým řetězcem. Populaci jedinců pak modelujeme jako populaci chromozómů a každému chromozómu (bitovému řetězci) umíme přiřadit hodnotu účelové funkce a podle vhodného předpisu umíme přiřadit i jeho fitness. [2]

2 TESTOVACÍ FUNKCE

K testování stochastických algoritmů pro globální optimalizaci se užívá řada funkcí, u kterých je známé správné řešení (zná se globální extrém), který lze najít analyticky. Testovací funkce jsou často navrženy tak, aby je bylo možné použít pro problémy s různou dimenzí prohledávané domény.

Testovací funkce mohou mít různou obtížnost, nejjednodušší jsou konvexní funkce, ty mají jen jedno lokální minimum, které je tedy i minimem globálním. Takovým funkcím se říká unimodální např. (Bent Cigar Function). Funkce s více lokálními minimy se nazývají multimodální a nalezení globálního minima je obvykle obtížnější než u unimodálních funkcí např. (Rastrigin's Function).

Funkce u kterých je globální minimum ve středu prohledávané domény např.: (De Jongova, Ackleyho, Griewankova a Rastriginova funkce) jsou nevhodné pro testování stochastických algoritmů, neboť u nich lze najít globální minimum pouhým průměrováním náhodně vygenerovaných bodů v D . Tento problém jde jednoduše odstranit, místo vyhodnocování funkce v bodě x je funkce vyhodnocována v posunutém (shifted) bodě $x - o$. Globální minimum takovéto funkce je pak v bodě $x^* = o$. [2]

2.1 Historie CEC Benchmarku

Tato práce se zabývá benchmarkem CEC 2014 (Real Parameter Single Objective Optimization). Tímto optimalizačním problémem se v historii CEC benchmarku zabývali ještě benchmarky CEC 2005 a CEC 2013.

2.1.1 CEC 2005

Benchmark CEC 2005 Obsahuje 25 testovacích funkcí, z toho je:

Unimodálních funkcí (5) např.: Shifted Sphere Function

Multimodálních funkcí (20) např.: Shifted Rosenbrock's Function

2.1.2 CEC 2013

Benchmark CEC 2005 Obsahuje 28 testovacích funkcí.

Unimodálních funkcí (5) např.: Rotated Bent Cigar Function

Multimodálních funkcí (15) např.: Rotated Rosenbrock's Function

Složených funkcí(8) např.: Composition Function 1 (n=5,Rotated)

Seznam dalších CEC benchmarků:

CEC2006 Constrained Real Parameter single objective optimization

CEC2007 Performance Assessment of real-parameter MOEAs

CEC2008 large scale single objective global optimization with bound constraints

CEC2009 Dynamic Optimization (Primarily composition functions were used)

CEC2009 Performance Assessment of real-parameter MOEAs

CEC2010 large-scale single objective global optimization with bound constraints

CEC2010 Evolutionary Constrained Real Parameter single objective optimization

CEC2011 Algorithms on Real-world Numerical Optimization Problems

[1]

2.2 Definice testovacích funkcí IEEE CEC 2014

Všechny testovací funkce jsou minimalizační problémy definovány následujícím způsobem $Min f(x)$, $x = [x_1, x_2, \dots, x_D]^T$. Prohledávaná oblast se pohybuje v rozsahu $[-100,100]^D$.

Každá funkce je posunuta (shifted) o $o_{i1} = [o_{i1}, o_{i2}, \dots, o_{iD}]^T$, kde každé o_i představuje náhodné číslo v rozsahu $[-80,80]^D$.

Každá funkce má přidělené matice rotace pro několik dimenzí (2D, 10D, 20D, 30D, 50D, 100D). Vzhledem k tomu, že u reálných problémů jen zřídka existují vazby mezi všemi proměnnými, jsou v CEC'14 proměnné rozděleny do dílčích součástí náhodně. Matice rotace jsou pro každou dílčí součást generovány ze standardních normálně distribuovaných záznamů podle Gram-Schmidt ortho-normalizace se stavovým číslem, které je rovno 1 nebo 2. [1]

Tab. 1. Souhrn všech CEC'14 testovací funkcí

	No.	Functions	$F_i^*=F_i(x^*)$
Unimodal Functions	1	Rotated High Conditioned Elliptic Function	100
	2	Rotated Bent Cigar Function	200
	3	Rotated Discus Function	300
Simple Multimodal Functions	4	Shifted and Rotated Rosenbrock's Function	400
	5	Shifted and Rotated Ackley's Function	500
	6	Shifted and Rotated Weierstrass Function	600
	7	Shifted and Rotated Griewank's Function	700
	8	Shifted Rastrigin's Function	800
	9	Shifted and Rotated Rastrigin's Function	900
	10	Shifted Schwefel's Function	1000
	11	Shifted and Rotated Schwefel's Functio	1100
	12	Shifted and Rotated Katsuura Function	1200
	13	Shifted and Rotated HappyCat Function	1300
	14	Shifted and Rotated HGBat Function	1400
	15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
	16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
Hybrid Function 1	17	Hybrid Function 1 (N=3)	1700
	18	Hybrid Function 2 (N=3)	1800
	19	Hybrid Function 3 (N=4)	1900
	20	Hybrid Function 4 (N=4)	2000
	21	Hybrid Function 5 (N=5)	2100
	22	Hybrid Function 6 (N=5)	2200
Composition Functions	23	Composition Function 1 (N=5)	2300
	24	Composition Function 2 (N=3)	2400
	25	Composition Function 3 (N=3)	2500
	26	Composition Function 4 (N=5)	2600
	27	Composition Function 5 (N=5)	2700
	28	Composition Function 6 (N=5)	2800
	29	Composition Function 7 (N=3)	2900
	30	Composition Function 8 (N=3)	3000
Search Range: [-100,100] ^D			

2.3 Definice základních funkcí

V Tab.1. jsou uvedeny všechny funkce použité v benchmarku IEEE CEC 2014.

Použité účelové funkce jsou uvedeny rovnicemi (1-14).

Průběhy vybraných funkcí jsou zobrazeny na Obr. 2-33. U každé funkce je uvedena rovnice (15-30), která určí zda se budou vstupní hodnoty posouvat nebo rotovat.

1) High Conditioned Elliptic Function

$$f_1(x) = \sum_{i=1}^D (10^6)^{\frac{i-1}{D-1}} x_i^2 \quad (1)$$

2) Bent Cigar Function

$$f_2(x) = x_1^2 + 10^6 \sum_{i=2}^D x_i^2 \quad (2)$$

3) Discus Function

$$f_3(x) = 10^6 x_1^2 + \sum_{i=2}^D x_i^2 \quad (3)$$

4) Rosenbrock's Function

$$f_4(x) = \sum_{i=2}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \quad (4)$$

5) Ackley's Function

$$f_5(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e \quad (5)$$

6) Weierstrass Function

$$f_6(x) = \sum_{i=1}^D \left(\sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k (x_i + 0.5))] \right) - D \sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k \times 0.5)] \quad (6)$$

$$a = 0.5, b = 3, k_{\max} = 20$$

7) Griewank's Function

$$f_7(x) = \sum_{i=1}^D \frac{x_i^2}{4000} - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (7)$$

8) Rastrigin's Function

$$f_8(x) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10) \quad (8)$$

9) Modified Schwefel's Function

$$f_9(x) = 418,9829 \times D - \sum_{i=1}^D g(z_i) \quad z_i = x_i + 4.209687462275036 \times 10^2 \quad (9)$$

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

10) Katsuura Function

$$f_{10}(x) = \frac{10}{D^2} \prod_{i=1}^D \left(1 + i \sum_{j=1}^{32} \frac{|2^j x_i - \text{round}(2^j x_i)|}{2^j} \right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} \quad (10)$$

11) HappyCat Function

$$f_{11}(x) = \left| \sum_{i=1}^D x_i^2 - D \right|^{1/4} + \left(0.5 \sum_{i=1}^D x_i^2 + \sum_{i=1}^D x_i \right) / D + 0.5 \quad (11)$$

12) HGBat Function

$$f_{12}(x) = \left| \left(\sum_{i=1}^D x_i^2 \right)^2 - \left(\sum_{i=1}^D x_i \right)^2 \right|^{1/2} + \left(0.5 \sum_{i=1}^D x_i^2 + \sum_{i=1}^D x_i \right) / D + 0.5 \quad (12)$$

13) Expanded Griewank's plus Rosenbrock's Function

$$f_{13}(x) = f_7(f_4(x_1, x_2)) + f_7(f_4(x_2, x_3)) + \dots + f_7(f_4(x_{D-1}, x_D)) + f_7(f_4(x_D, x_1)) \quad (13)$$

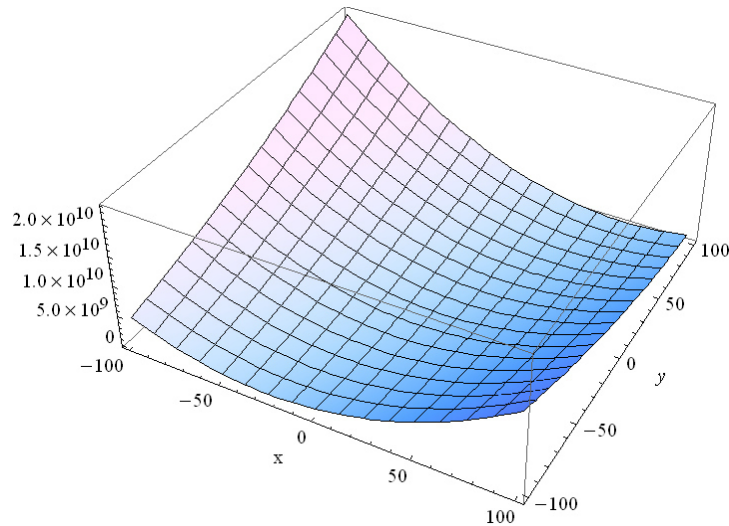
14) Expanded Scaffer's F6 Function

$$\text{Scaffer's F6 Function: } g(x, y) = 0.5 + \frac{(\sin^2(\sqrt{x^2 + y^2}) - 0.5)}{(1 + 0.001(x^2 + y^2))^2}$$

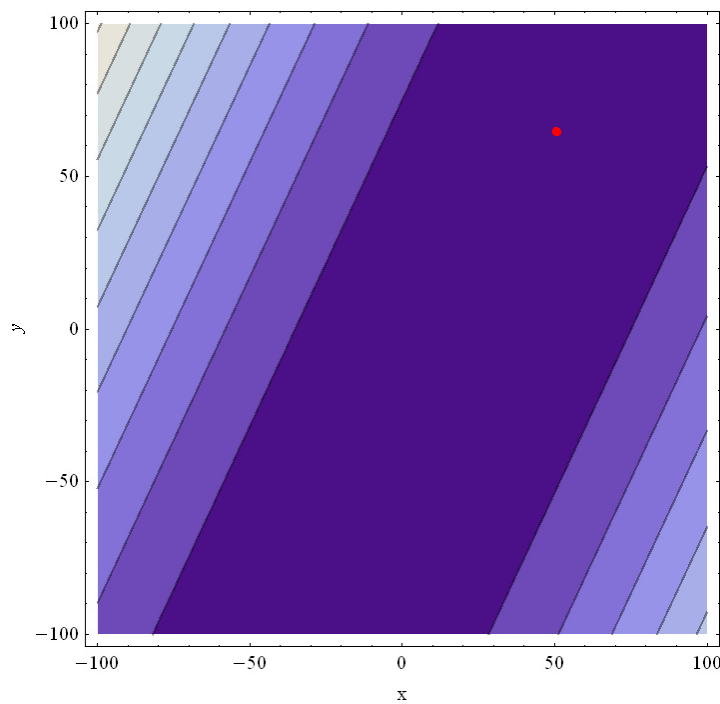
$$f_{14}(x) = g(x_1, x_2) + g(x_2, x_3) + \dots + g(x_{D-1}, x_D) + g(x_D, x_1) \quad (14)$$

2.3.1 Rotated Hight Conditioned Elliptic Function

$$F_1(x) = f_1(M(x - o_1)) + F_1^* \quad (15)$$



Obr. 2. Elliptic function (3D map)



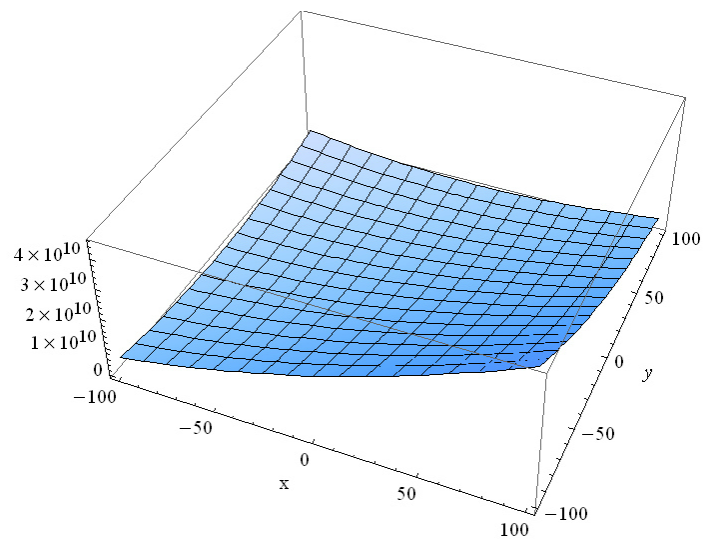
Obr. 3. Elliptic function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

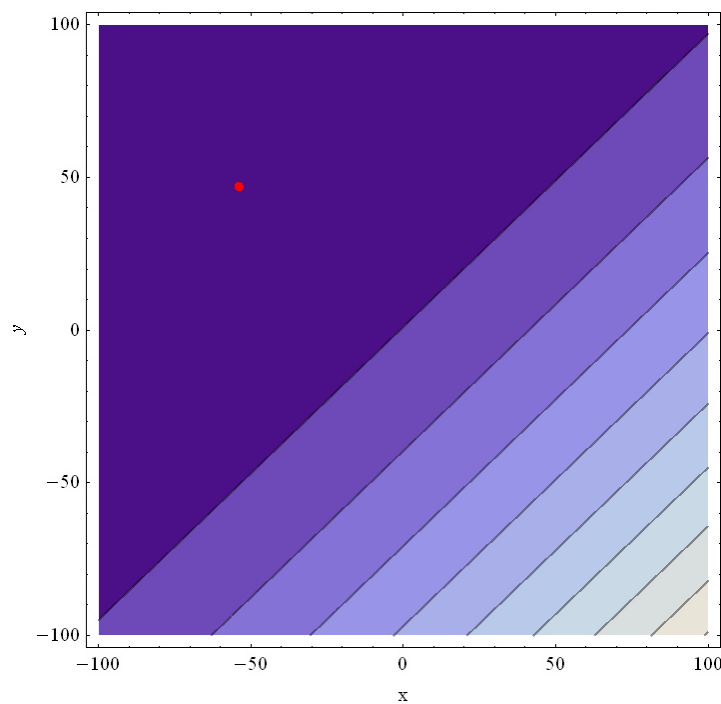
Globální minimum v E_2 : 100

2.3.2 Rotated Bent Cigar Function

$$F_2(x) = f_2(M(x - o_2)) + F_2^* \quad (16)$$



Obr. 4. Bent Cigar function (3D map)



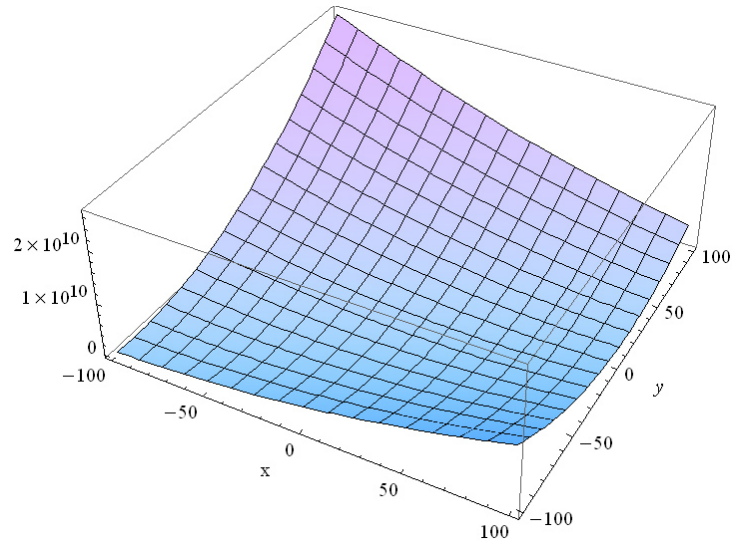
Obr. 5. Bent Cigar function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

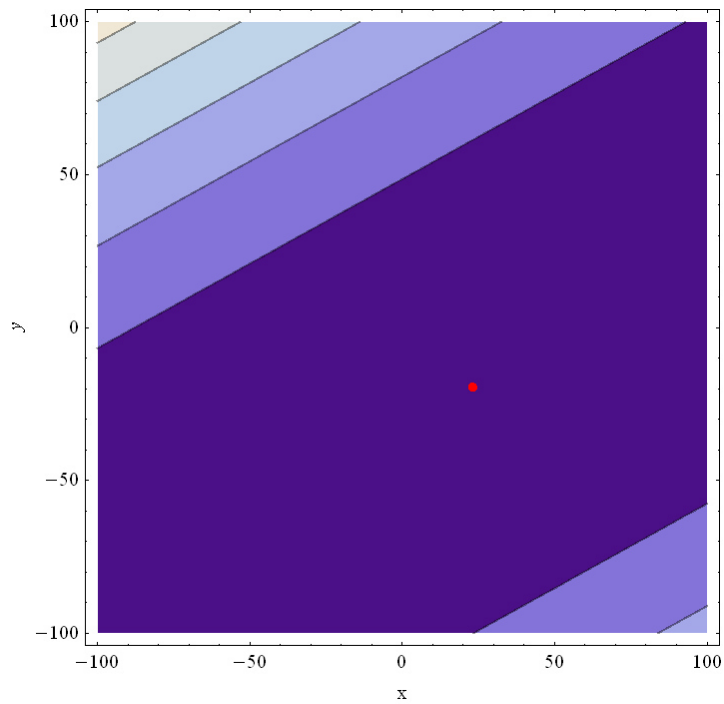
Globální minimum v E_2 : 200

2.3.3 Rotated Disc Function

$$F_3(x) = f_3(M(x - o_3)) + F_3^* \quad (17)$$



Obr. 6. Discus function (3D map)



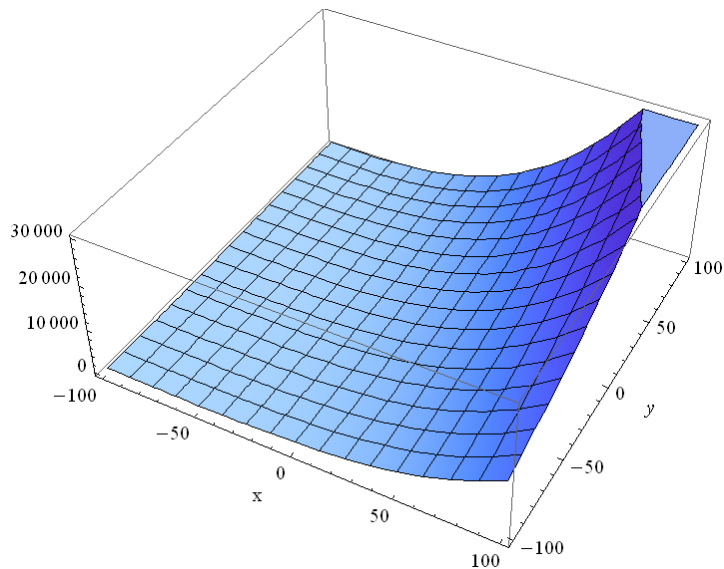
Obr. 7. Discus function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

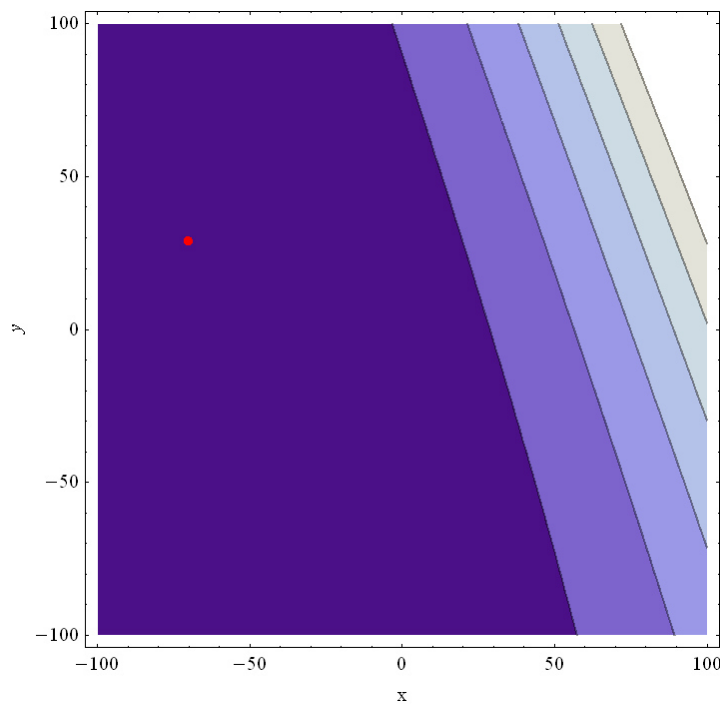
Globální minimum v E_2 : 300

2.3.4 Shifted and Rotated Rosenbrock's Function

$$F_4(x) = f_4 \left(M \left(\frac{2.048(x - o_4)}{100} \right) + 1 \right) + F_4^* \quad (18)$$



Obr. 8. Rosenbrock's function (3D map)



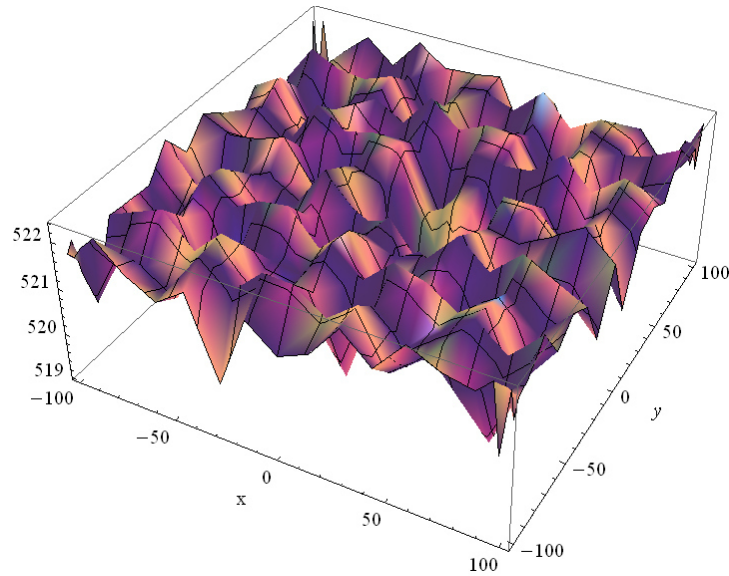
Obr. 9. Rosenbrock's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

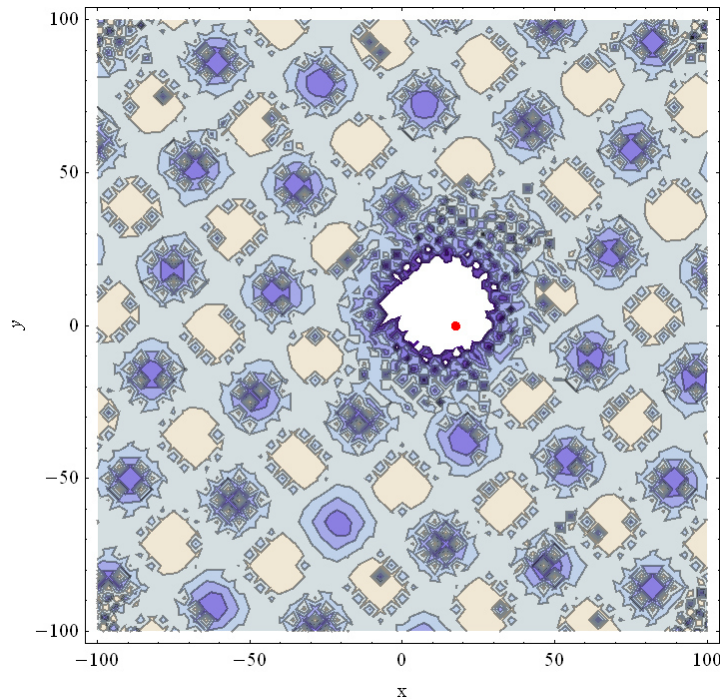
Globální minimum v E_2 : 400

2.3.5 Shifted and Rotated Ackley's Function

$$F_5(x) = f_5(M(x - o_5)) + F_5^* \quad (19)$$



Obr. 10. Ackley's function (3D map)



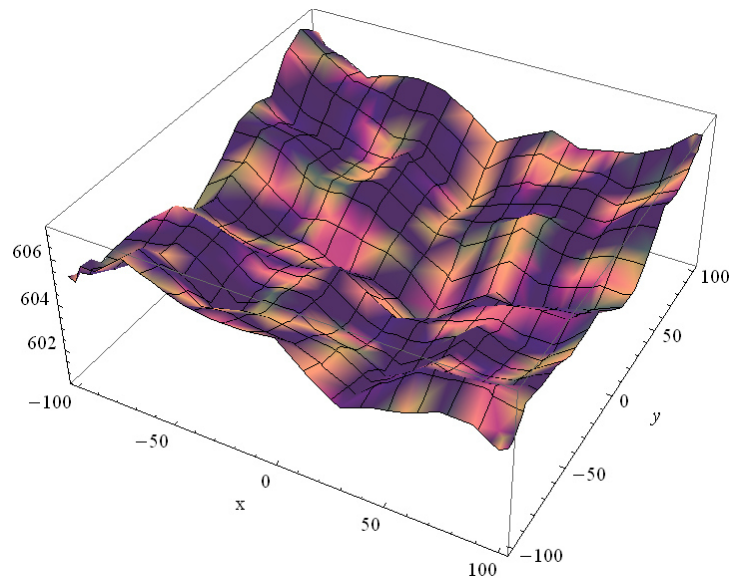
Obr. 11. Ackley's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

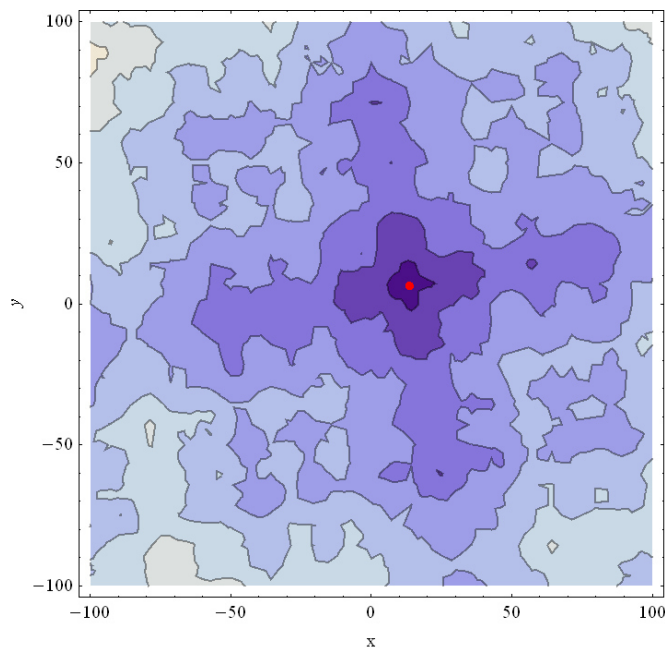
Globální minimum v E_2 : 511.437

2.3.6 Shifted and Rotated Weierstrass Function

$$F_6(x) = f_6\left(M\left(\frac{0.5(x - o_6)}{100}\right)\right) + F_6^* \quad (20)$$



Obr. 12. Weierstrass function (3D map)



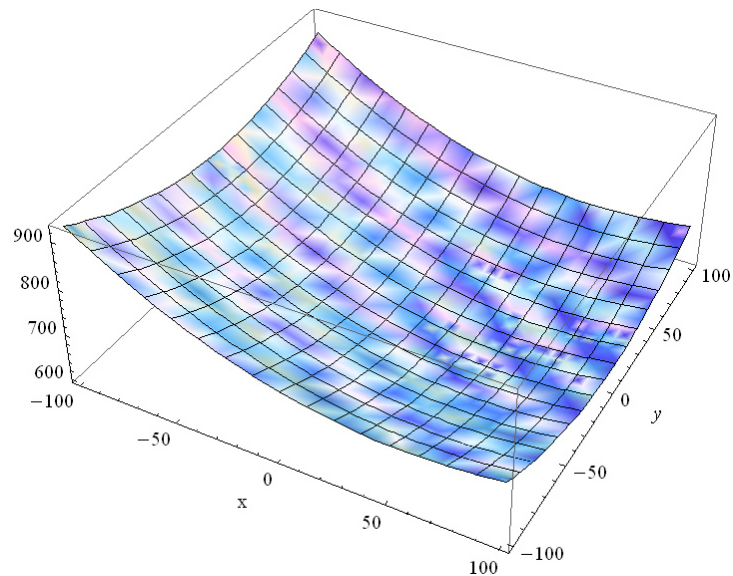
Obr. 13. Weierstrass function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

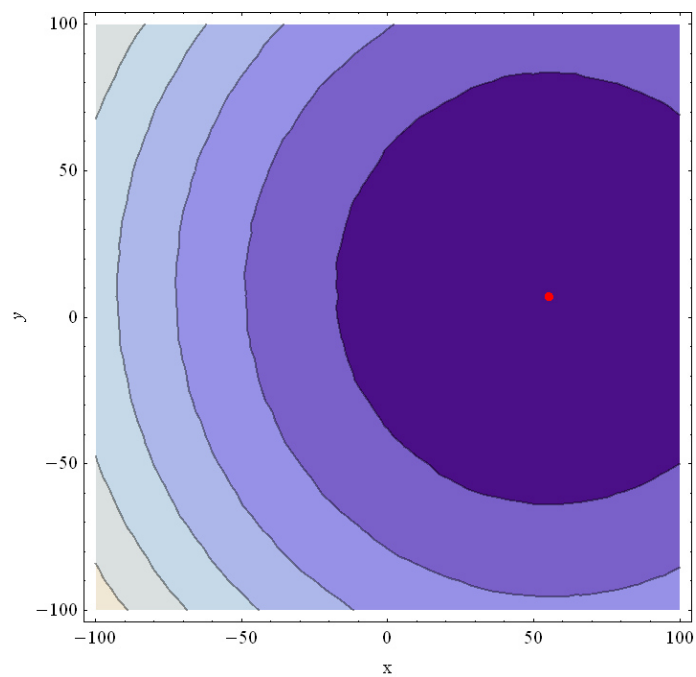
Globální minimum v E_2 : 600.058

2.3.7 Shifted and Rotated Griewank's Function

$$F_7(x) = f_7\left(M\left(\frac{600(x - o_7)}{100}\right)\right) + F_7^* \quad (21)$$



Obr. 14. Griewank's function (3D map)



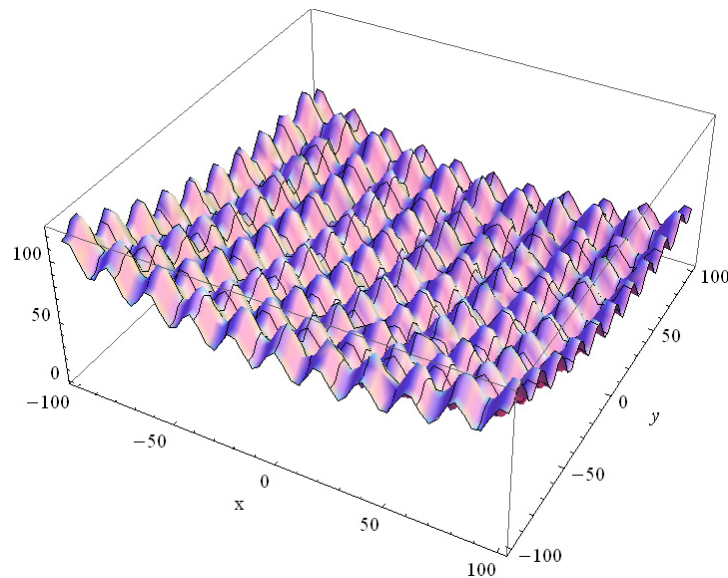
Obr. 15. Griewank's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

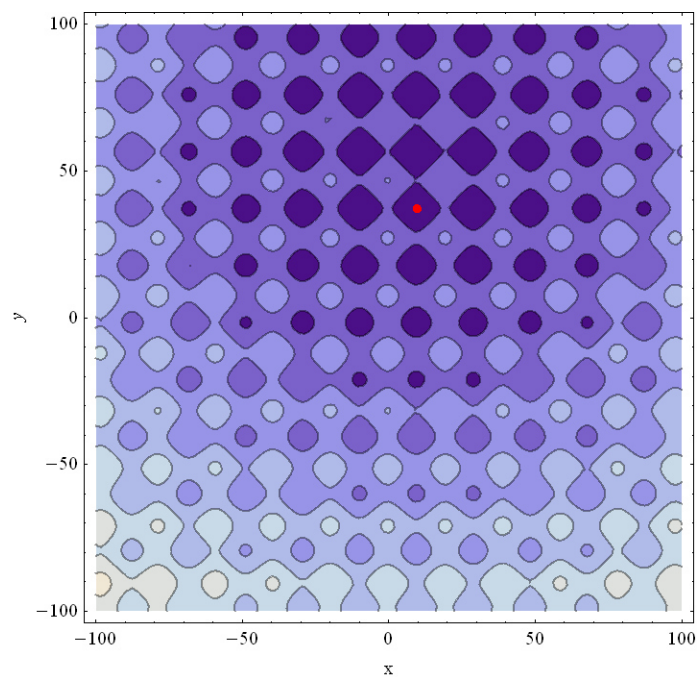
Globální minimum v E_2 : 700.067

2.3.8 Shifted Rastrigin's Function

$$F_8(x) = f_8\left(M\left(\frac{5.12(x - o_8)}{100}\right)\right) + F_8^* \quad (22)$$



Obr. 16. Rastrigin's function (3D map)



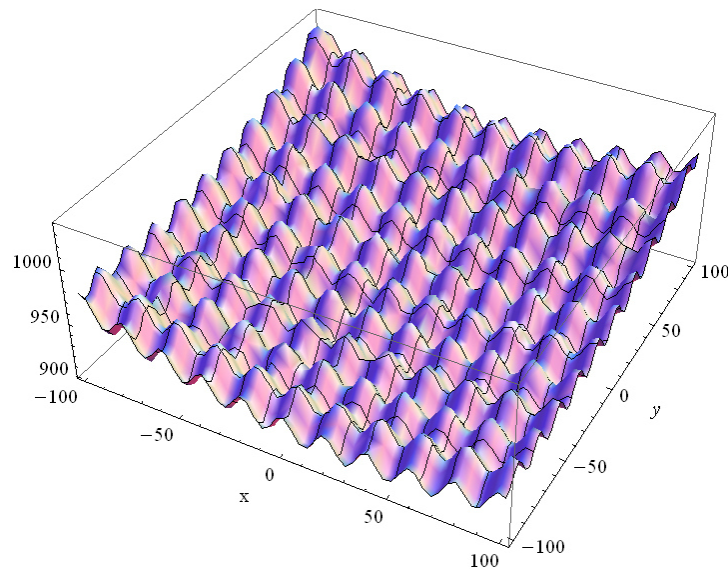
Obr. 17. Rastrigin's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

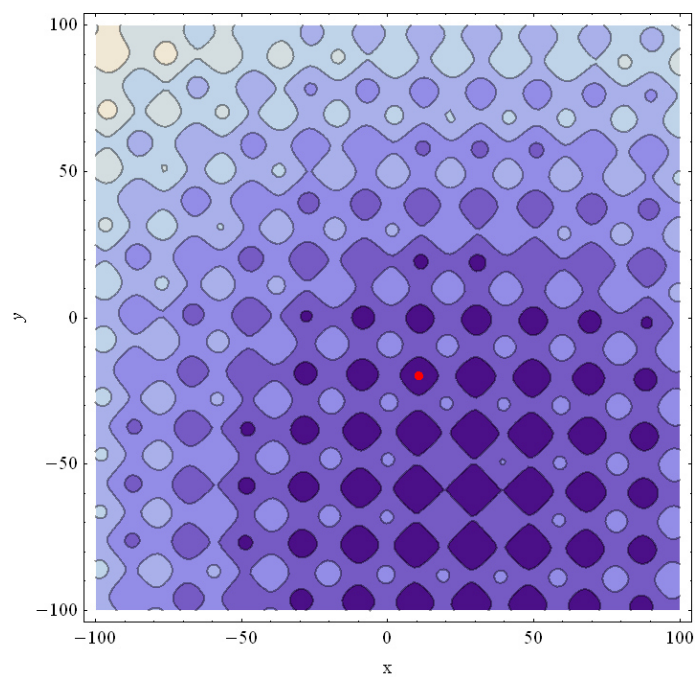
Globální minimum v E_2 : 800.995

2.3.9 Shifted and Rotated Rastrigin's Function

$$F_9(x) = f_8\left(M\left(\frac{5.12(x - o_9)}{100}\right)\right) + F_9^* \quad (23)$$



Obr. 18. Rotated Rastrigin's function (3D map)



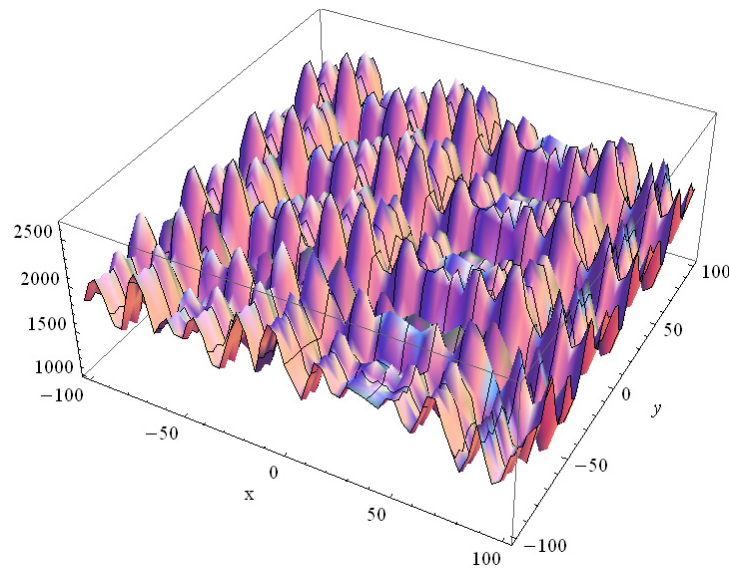
Obr. 19. Rotated Rastrigin's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

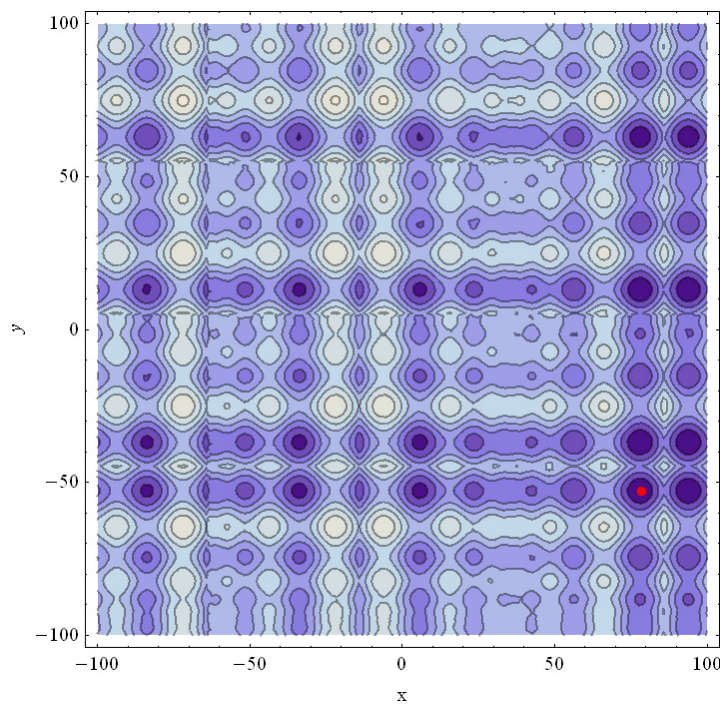
Globální minimum v E_2 : 904.975

2.3.10 Shifted Schwefel's Function

$$F_{10}(x) = f_9 \left(M \left(\frac{1000(x - o_{10})}{100} \right) \right) + F_{10}^* \quad (24)$$



Obr. 20. Schwefel's function (3D map)



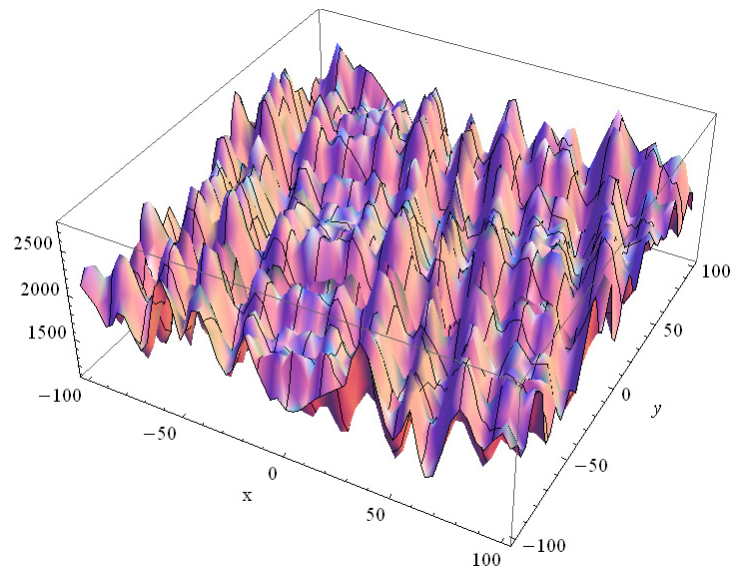
Obr. 21. Schwefel's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

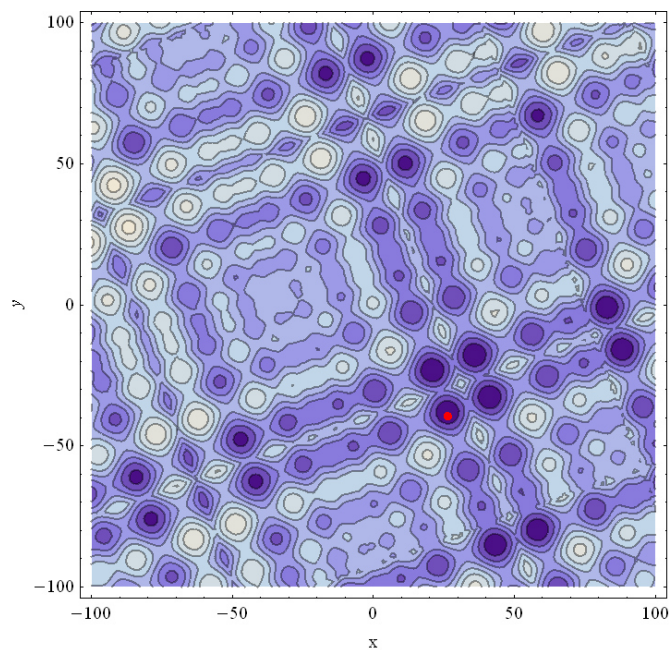
Globální minimum v E_2 : 1185.33

2.3.11 Shifted and Rotated Schwefel's Function

$$F_{11}(x) = f_9 \left(M \left(\frac{1000(x - o_{11})}{100} \right) \right) + F_{11}^* \quad (25)$$



Obr. 22. Rotated Schwefel's function (3D map)



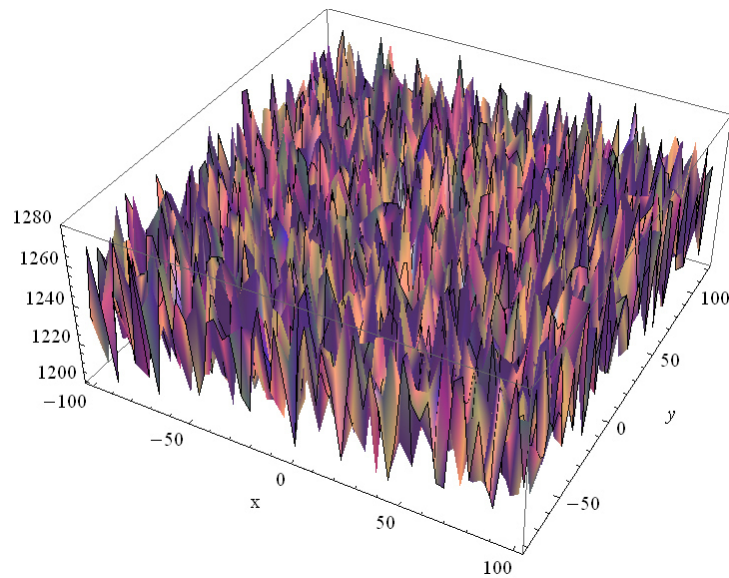
Obr. 23. Rotated Schwefel's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

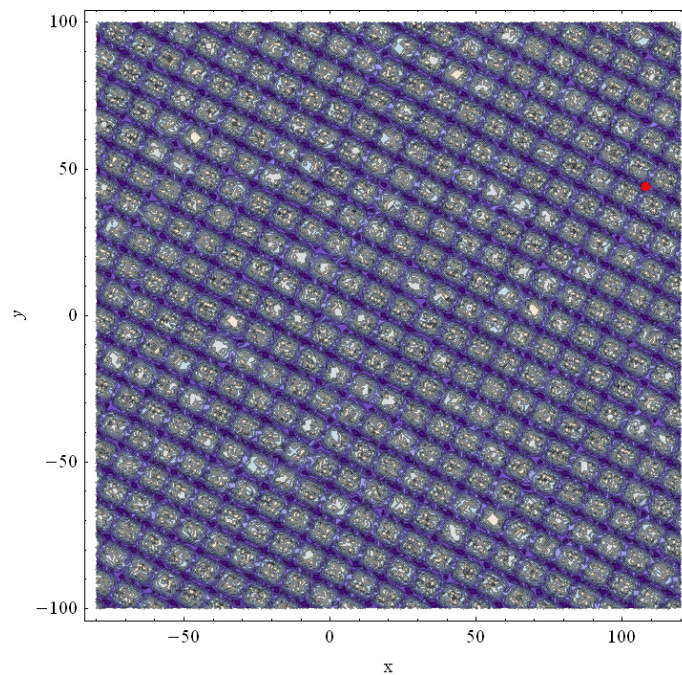
Globální minimum v E_2 : 1224.82

2.3.12 Shifted and Rotated Katsuura Function

$$F_{12}(x) = f_{10} \left(M \left(\frac{5(x - o_{12})}{100} \right) \right) + F_{12}^* \quad (26)$$



Obr. 24. Katsuura function (3D map)



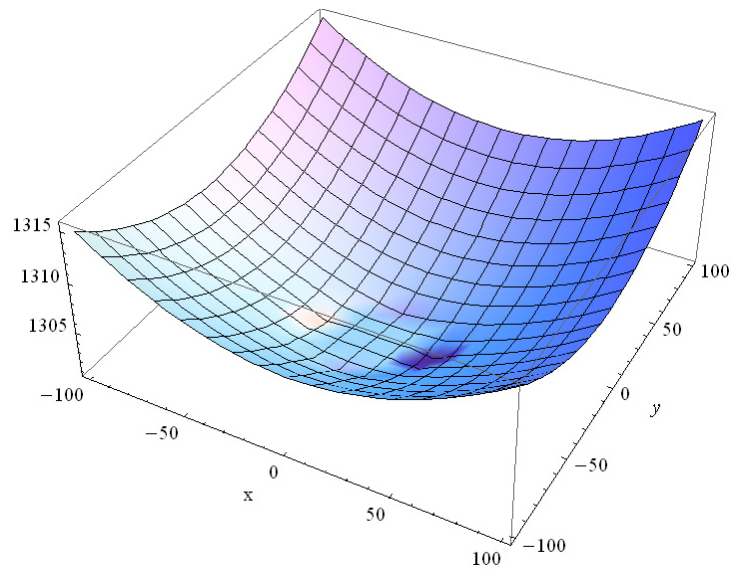
Obr. 25. Katsuura function (2D map)

Prohledávaný interval: $-80 \leq x_i \leq 120$

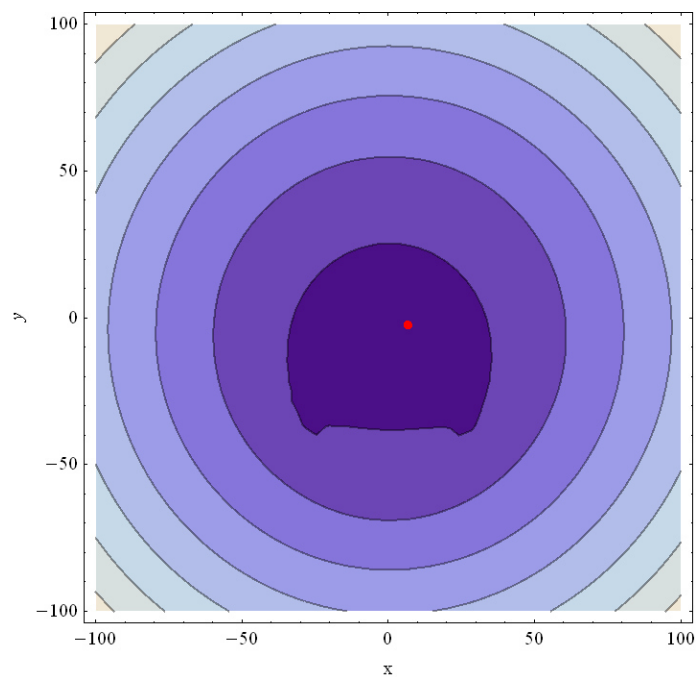
Globální minimum v E_2 : 1202.00

2.3.13 Shifted and Rotated HappyCat Function

$$F_{13}(x) = f_{11}\left(M\left(\frac{5(x - o_{13})}{100}\right)\right) + F_{13}^* \quad (27)$$



Obr. 26. HappyCat function (3D map)



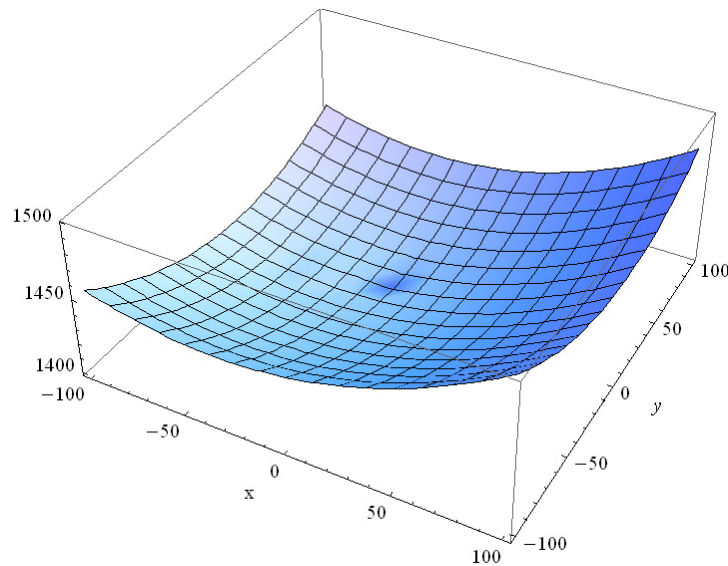
Obr. 27. HappyCat function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

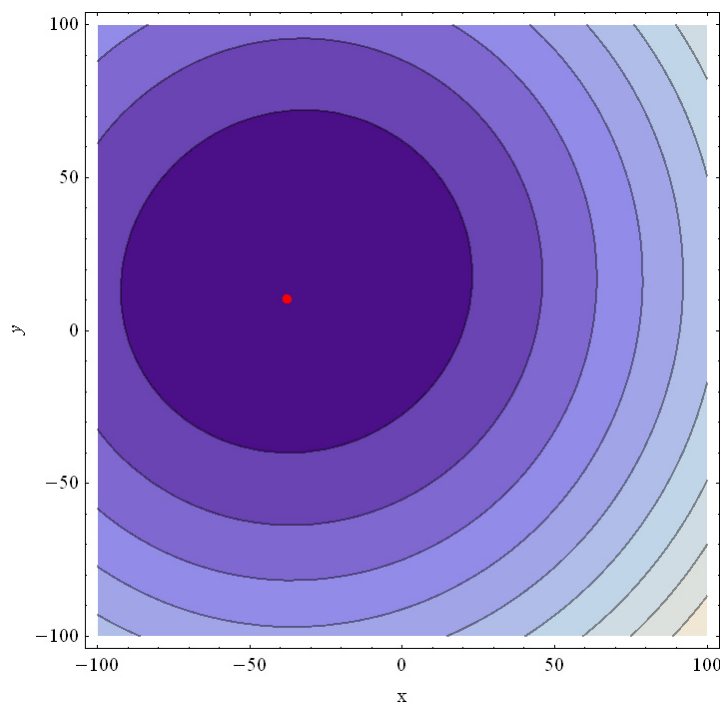
Globální minimum v E_2 : 1324.82

2.3.14 Shifted and Rotated HGBat Function

$$F_{14}(x) = f_{12}\left(M\left(\frac{5(x - o_{14})}{100}\right)\right) + F_{14}^* \quad (28)$$



Obr. 28. HGBat function (3D map)



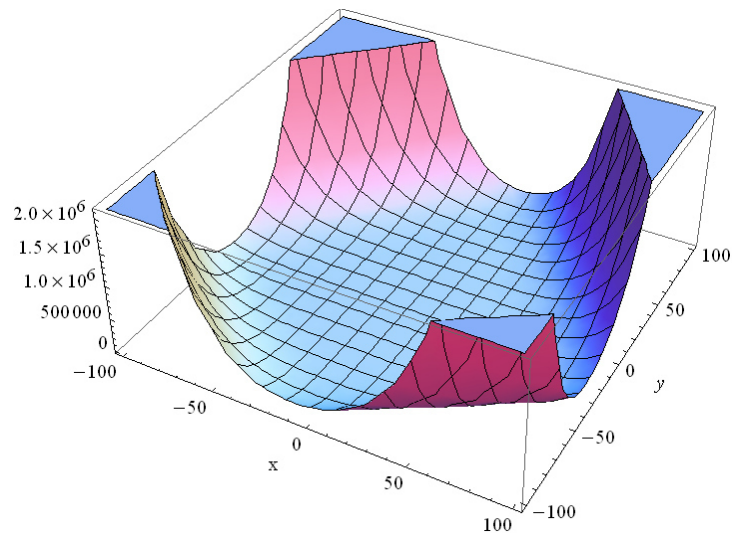
Obr. 29. HGBat function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

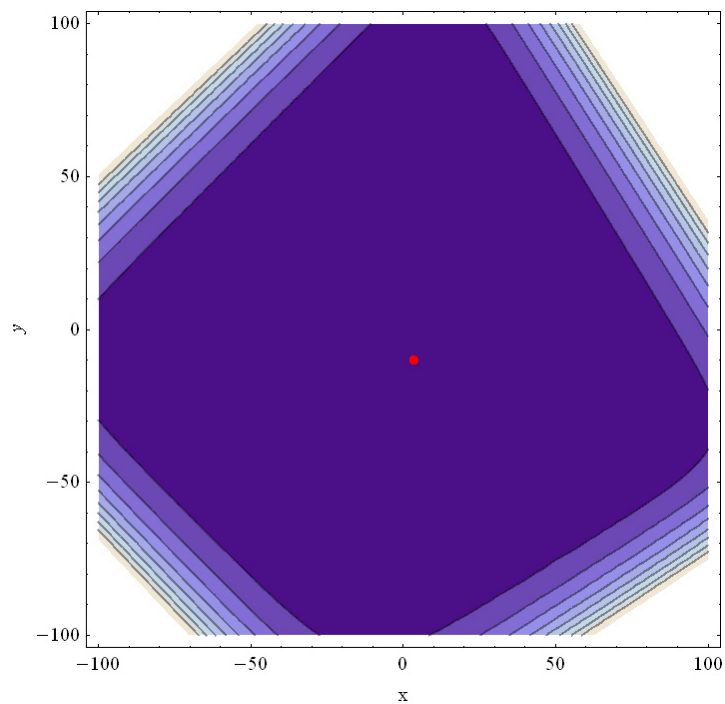
Globální minimum v E_2 : 1404.19

2.3.15 Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function

$$F_{15}(x) = f_{13} \left(M \left(\frac{5(x - o_{15})}{100} \right) + 1 \right) + F_{15}^* \quad (29)$$



Obr. 30. Griewank's plus Rosenbrock's function (3D map)



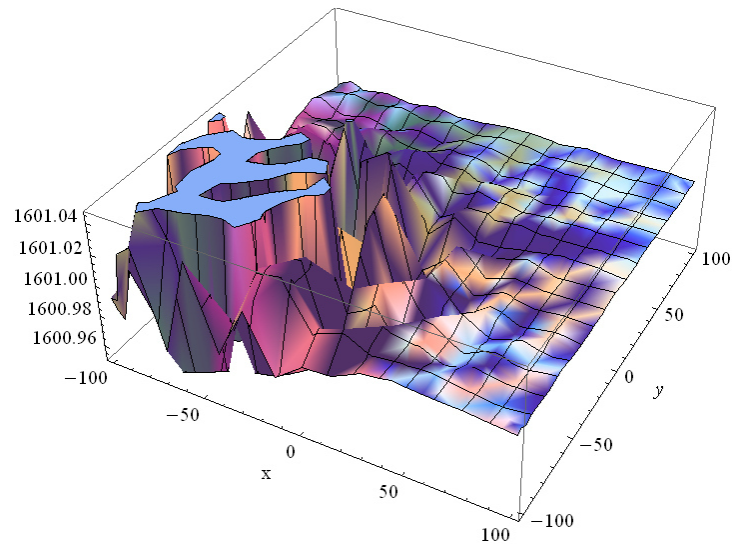
Obr. 31. Griewank's plus Rosenbrock's function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

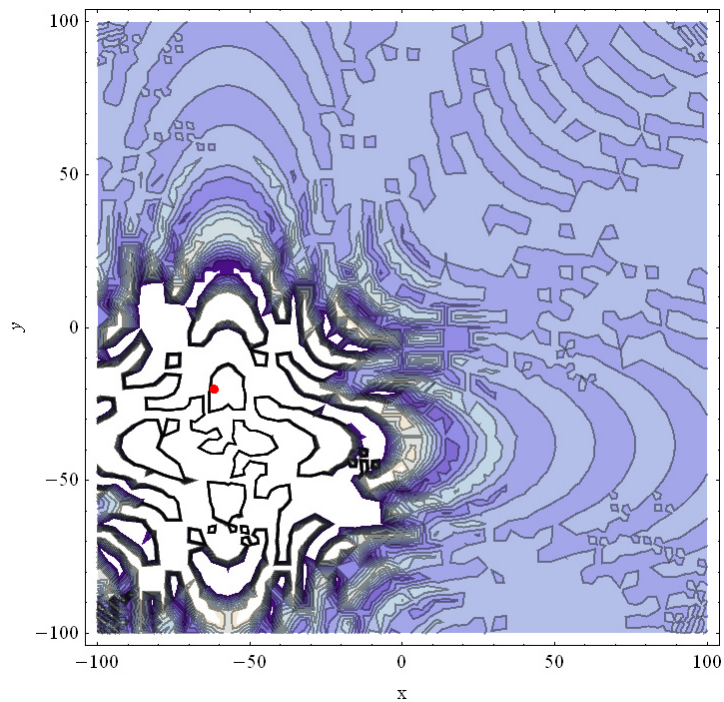
Globální minimum v E_2 : 1500.91

2.3.16 Shifted and Rotated Expanded Scaffer's F6 Function

$$F_{16}(x) = f_{14}(M(x - o_{16}) + 1) + F_{16}^* \quad (30)$$



Obr. 32. Scaffer's F6 function (3D map)



Obr. 33. Scaffer's F6 function (2D map)

Prohledávaný interval: $-100 \leq x_i \leq 100$

Globální minimum v E_2 : 1600.45

II. PRAKTICKÁ ČÁST

3 IMPLEMENTACE KNIHOVNY TESTOVACÍCH FUNKCÍ

V soubor CEC2014_Benchmark.nb jsou definovány všechny testovací a pomocné funkce.

K souboru je přiložen adresář se soubory pro posunutí a rotaci (shifted, rotated).

3.1 Funkce ShiftFunc

Funkce ShiftFunc od zvolených hodnot odečítá (posouvá) hodnoty načtené ze souboru shift_data_i.txt, kde i je číslo funkce.

```
ShiftFunc[nx_, Os_, X_] := Module[{shift, n, i, x, y},
  shift = Os;
  n = nx;
  x = X;
  y = Array[0 &, n];
  For[i = 1, i <= n, ++i, y[[i]] = x[[i]] - shift[[i]]];
  Return[y];
];
```

3.2 Funkce RotFunc

Funkce RotFunc rotuje dané hodnoty pomocí matice načtené ze souboru M_i_Dn.txt, kde i je číslo funkce a n je dimenze ve které se výpočet odehrává.

```
RotFunc[nx_, Mr_, shift_] := Module[{m, n, i, j, s, rot},
  m = Mr;
  n = nx;
  s = shift;
  rot = Array[0 &, n];
  For[i = 1, i <= n, i++,
    rot[[i]] = 0;
    For[j = 1, j <= n, ++j,
      rot[[i]] = rot[[i]] + s[[j]] * m[[i, j]];
    ];
  ];
  Return[rot];
];
```

3.3 Funkce Cfcf

Funkce Cfcf volá se ve všech funkcích Cf (Composition Functions). Bylo by zbytečné v každé funkci Cf psát stejnou část kódu.

```

Cfcf[nx_, shiftdata_, Delta_, Bias_, Pol_, NN_, X_] :=
Module[{i, j, w, wmax, wsum, v, n, x, shift, delta, bias, pol, N},
  n = nx;
  shift = shiftdata;
  delta = Delta;
  bias = Bias;
  pol = Pol;
  N = NN;
  x = X;
  v = 0;
  wmax = 0;
  wsum = 0;
  w = Array[0 &, N];
  For[i = 1, i <= N, i++,
    pol[[i]] = pol[[i]] + bias[[i]];
    w[[i]] = 0;
    For[j = 1, j <= n, j++,
      w[[i]] = w[[i]] + Power[x[[j]] - shift[[i, j]], 2];];
    If[w[[i]] != 0,
      w[[i]] =
        Power[1 / w[[i]], 0.5] * Exp[-w[[i]]/(2 * n * Power[delta[[i]], 2])],
      w[[i]] = 1 * 10^99;
    ];
    If[w[[i]] > wmax, wmax = w[[i]];
  ];
  For[i = 1, i <= N, i++,
    wsum = wsum + w[[i]];
  ];
  If[wmax == 0,
    For[i = 1, i <= N, i++,
      w[[i]] = 1;
    ];
    wsum = N;
  ];
  For[i = 1, i <= N, i++,
    v = v + w[[i]] / wsum * pol[[i]];
  ];
  Return[v];
];

```

3.4 Funkce SrFunc

Funkce SrFunc slouží ke zvolení, zda se budou v dané funkci vstupní hodnoty posouvat (shifted), nebo rotovat (rotated).

```

SrFunc[nx_, Os_, Mr_, shrate_, sflag_, rflag_, X_] := Module[{n, shift, s, m, sh, r, i, x, y},
  n = nx;
  shift = Os;
  m = Mr;
  sh = shrate;
  s = sflag;
  r = rflag;
  x = X;
  y = Array[0 &, n];
  If[s == 1,
    If[r == 1,
      y = ShiftFunc[n, shift, x];
      For[i = 1, i <= n, i++, y[[i]] = y[[i]] * sh];
      Return[RotFunc[n, m, y]],
      y = ShiftFunc[n, shift, x];
      For[i = 1, i <= n, i++, y[[i]] = y[[i]] * sh];
      Return[y];],
    If[r == 1,
      For[i = 1, i <= n, i++, y[[i]] = y[[i]] * sh];
      Return[RotFunc[n, m, y]],
      y = m;
      For[i = 1, i <= n, i++, y[[i]] = y[[i]] * sh];
      Return[y];
  ];];
];

```

3.5 Funkce EllipsFunc

```

EllipsFunc[M_, shiftdata_, nx_, sflag_, rflag_, X_] := Module[{i, v, m, shift, n, s, r, x, z},
  m = M;
  shift = shiftdata;
  n = nx;
  s = sflag;
  r = rflag;
  x = X;
  z = Array[0 &, n];
  v = 0;
  z = SrFunc[n, shift[[1, 1 ;; n]], m, 1, s, r, x];
  For[i = 1, i <= n, i++,
    v = v + (Power[1000000, ((i - 1) / (n - 1))]) * (z[[i]] * z[[i]]);
    v = v + 100;
  Return[v];
];

```

3.6 Funkce Hf01

```

Hf01[M_, shiftdata_, S_, nx_, sflag_, flag, X_] :=
Module[{i, v, m, n, r, s, x, z, shuffle, shift, tmp, N, G, Gp, Gnx},
  m = M;
  shift = shiftdata;
  n = nx;
  s = sflag;
  r = rflag;
  shuffle = S;
  x = X;
  N = 3;
  tmp = 0;
  v = 0;
  z = Array[0&, n];
  Gp = {0.3,0.3,0.4};
  G = Array[0&, N];
  Gnx = Array[0&, N];
  For[i = 1, i < N, i++,
    Gnx[[i]] = Gp[[i]] * n;
    tmp = tmp + Gnx[[i]];
  ];
  Gnx[[N]] = n - tmp;
  G[[1]] = 0;
  For[i = 2, i < N + 1, i++,
    G[[i]] = G[[i - 1]] + Gnx[[i - 1]];
  ];
  z = SrFunc[n, shift[[1, 1;;n]], m, 1, s, r, x];
  For[i = 1, i <= n, i++,
    x[[i]] = z[[shuffle[[1, i]]]];
  ];
  i = 1;
  v = v + SchwefelFunc[m, shift, Rationalize[Gnx[[i]], 0, 0, x];
  i = 2;
  v = v + RastriginFunc[m, shift, Rationalize[Gnx[[i]], 0, 0, Drop[x, Rationalize[G[[i]]]]];
  i = 3;
  v = v + EllipsFunc[m, shift, Rationalize[Gnx[[i]], 0, 0, Drop[x, Rationalize[G[[i]]]]];
  v = v + 1700;
  Return[v];
];

```

3.7 Funkce Cf01

```

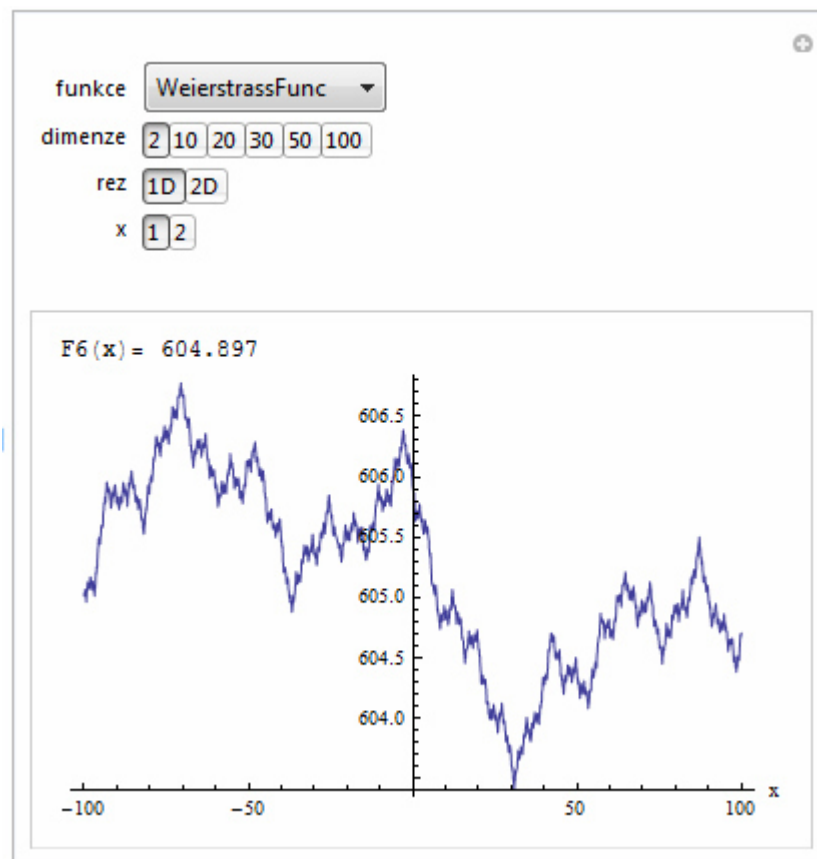
Cf01[M_, shiftdata_, nx_, rflag_, X_] := Module[{i, v, m, n, r, x, shift, N, delta, bias, pol},
  m = M;
  shift = shiftdata;
  n = nx;
  r = rflag;
  x = X;
  N = 5;
  pol = Array[0&, N];
  delta = {10, 20, 30, 40, 50};
  bias = {0, 100, 200, 300, 400};
  i = 1;
  pol[[i]] = RosenbrockFunc[m[[i ;; n]], {shift[[i]]}, n, 1, r, x];
  i = 2;
  pol[[i]] = EllipsFunc[m[[((i - 1) * n) + 1 ;; i * n]], {shift[[i]]}, n, 1, r, x]/(1*10^6);
  i = 3;
  pol[[i]] = BencigarFunc[m[[((i - 1) * n) + 1 ;; i * n]], {shift[[i]]}, n, 1, r, x]/(1*10^26);
  i = 4;
  pol[[i]] = DiscusFunc[m[[((i - 1) * n) + 1 ;; i * n]], {shift[[i]]}, n, 1, r, x]/(1*10^6);
  i = 5;
  pol[[i]] = EllipsFunc[m[[((i - 1) * n) + 1 ;; i * n]], {shift[[i]]}, n, 1, 0, x]/(1*10^6);
  v = Cfcad[n, shift, delta, bias, pol, N, x];
  v = v + 2300;
  Return[v];
];

```

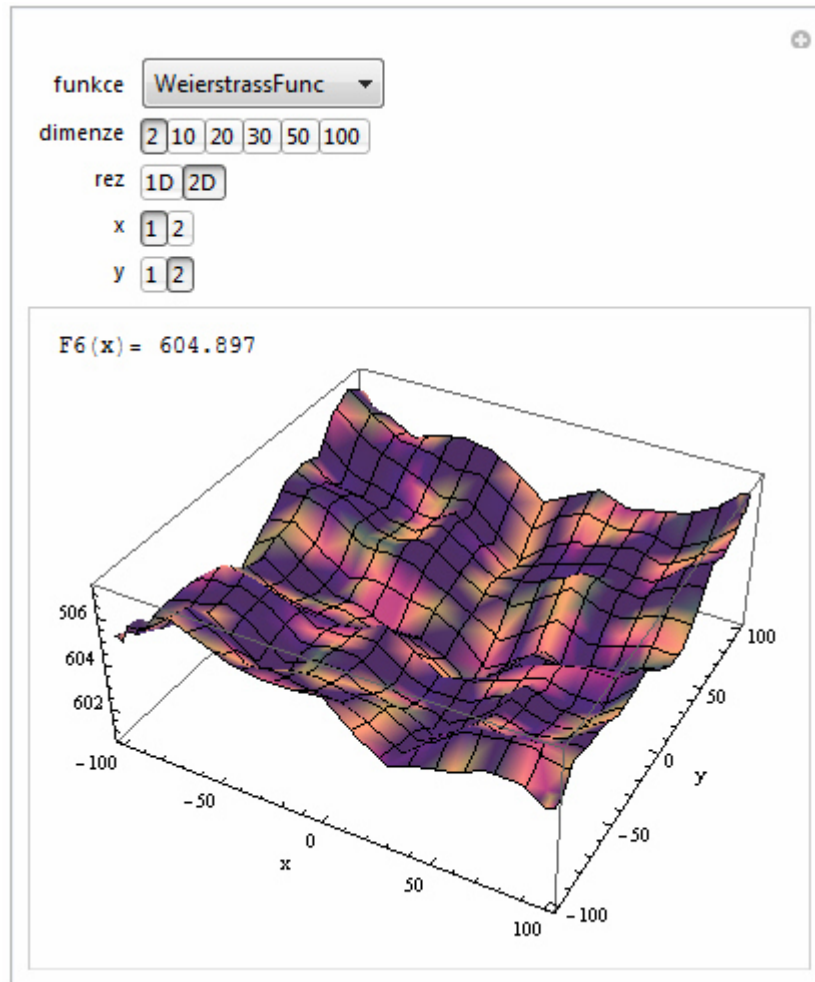
4 VIZUALIZACE VÝSLEDKŮ

Pro vizualizaci výsledků je pomocí funkce Manipulate vytvořena jednoduchá aplikace viz. obr. 37. Ve výsledné aplikaci je možné si vybrat funkci, v jaké dimenzi se bude počítat a dále zobrazení daná funkce v 1D nebo 2D. Po zvolení řezu je možné si vybrat, která dimenze se bude vykreslovat v případě 1D zobrazení, pro 2D zobrazení se vybírají 2 dimenze viz. obr. 38. Při zvolení dimenze větší než 20 je možnost výběru dimenze pro vykreslení grafu omezená jen na prvních 20 dimenzích.

Nad grafem je výsledek funkce pro danou dimenzi při jednom průchodu dané funkce. Vstupní vektor pro danou dimenzi je generován náhodně v rozsahu od -100 do 100.



Obr. 34. 1D Řez funkce



Obr. 35. 2D Řez funkce

5 KONTROLA DAT OPROTI IMPLEMENTACÍM V MATLAB/C PROSTŘEDÍCH

Pro ověření správné funkčnosti funkcí vytvořených v Matematice jsou funkce otestovány ve dvou dimenzích (2D, 10D). viz.(Tab.1., Tab.2.) Jako vstupní parametr byly pro každou dimenzi zvoleny hodnoty umístěné v poli x, které pak byly použity pro všechny funkce jak v Mathematce tak v Matlab/C.

5.1 Srovnání výsledků funkcí

Srovnání výsledků z Matematiky a výsledků z Matlab/C pro 2D.

Jako vstup bylo zvoleno: $x = \{-34, 72\}$

Tab. 2. Srovnání pro 2D

Funkce	Výsledek Matematice	Výsledek Matlab/C
EllipsFunc	6305813963.28	6305813963.28
BentcigarFunc	16700467.38	16700467.38
DiscusFunc	11584840061.50	11584840061.50
RosenbrockFunc	1230.51	1230.51
AckleyFunc	521.26	521.26
WeierstrassFunc	604.47	604.47
GriewankFunc	809.80	809.80
RastriginFunc (r=0)	821.74	821.74
RastriginFunc	988.68	988.68
SchwefelFunc (r=0)	1805.42	1805.42
SchwefelFunc	1655.98	1655.98
KatsuuraFunc	1211.12	1211.12
HappycatFunc	1306.42	1306.42
HgbatFunc	1409.93	1409.93
GrierosenFunc	175422.00	175422.00
Escaffer6Func	1601.00	1601.00

Srovnání výsledků z Matematiky a výsledků z Matlab/C pro 10D.

Jako vstup bylo zvoleno: $x = \{12, 3, 10, -42, 22, 40, 22, -68, -8, -47\}$

Tab. 3. Srovnání pro 10D

Funkce	Výsledek Mathematice	Výsledek Matlab/C
EllipsFunc	10993340088.74	10993340088.74
BentcigarFunc	41192968435.12	41192968435.12
DiscusFunc	46745693.45	46745693.45
RosenbrockFunc	44880.12	44880.12
AckleyFunc	521.80	521.80
WeierstrassFunc	617.07	617.07
GriewankFunc	1236.13	1236.13
RastriginFunc (r=0)	969.55	969.55
RastriginFunc	1128.32	1128.32
SchwefelFunc (r=0)	5157.70	5157.70
SchwefelFunc	4862.75	4862.75
KatsuuraFunc	1209.34	1209.34
HappycatFunc	1309.76	1309.76
HgbatFunc	1553.54	1553.54
GrierosenFunc	1763715.95	1763715.95
Escaffer6Func	1604.87	1604.87
Hf01	336632896.10	336632896.10
Hf02	197908693.64	197908693.64
Hf03	8817.00	8817.00
Hf04	733875858.94	733875858.94
Hf05	9517224115.32	9517224115.32
Hf06	5504.36	5504.36
Cf01	5235.01	5235.01
Cf02	2733.55	2733.55
Cf03	2785.33	2785.33
Cf04	2920.07	2920.07
Cf05	6961.87	6961.87
Cf06	9111.57	9111.57
Cf07	1965068837.34	1965068837.34
Cf08	305951137.18	305951137.18

ZÁVĚR

Cílem práce bylo naprogramovat testovací benchmark v prostředí Wolfram Mathematica a následně porovnat výsledky benchmarku z Mathematici s výsledky benchmarku implementovaném v Matlab/C.

V praktické části byly uvedeny příklady implementace některých funkcí v Mathematice. Dále byly uvedeny implementace pomocných funkcí a popis jejich funkčnosti. Tyto pomocné funkce slouží ke zkrácení a zpřehlednění kódu.

V práci měla být původně zahrnuta prezentace pomocí nadstavby Wolfram WebMathematica. Během vypracování této práce přestala být tato nadstavba podporována. Z tohoto důvodu byla vytvořena jednoduchá aplikace přímo v prostředí Mathematici pomocí funkce Manipulate. V této aplikaci je možné zobrazit průběhy všech implementovaných funkcí v 1D nebo 2D řezu a to pro různé dimenze.

Ve dvou tabulkách byly porovnány výsledky funkcí implementovaných v Mathematice s funkcemi z Matlab/C. V obou případech se výsledky shodovaly. Z toho vyplývá, že funkce napsané v Mathematice jsou implementovány správně.

Vytvořená knihovna obsahuje řadu různých testovacích funkcí. Tyto testovací funkce jsou vytvořeny tak, aby je bylo možné dále na FAI využít se stávajícími implementacemi výkonných algoritmů.

SEZNAM POUŽITÉ LITERATURY

- [1] *Special Session & Competition on Real-Parameter Single Objective (Expensive) Optimization at CEC-2014, Beijing, PR-China 6-11 July 2014*. [online]. [cit. 2015-05-17]. Dostupné z: <http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/CEC-2014/Definitions%20of%20%20CEC2014%20benchmark%20suite%20Part%20A.pdf>
- [2] TRVDÍK Josef. *Stochastické algoritmy pro globální optimalizaci*. [online]. 2010 [cit. 2015-05-14]. Dostupné z: http://www1.osu.cz/~tvrdik/wp-content/uploads/STAGO_10.pdf
- [3] TRVDÍK Josef. *Evoluční algoritmy* [online]. 2004 [cit. 2015-05-14]. Dostupné z: http://prf.osu.cz/doktorske_studium/dokumenty/Evolutionary_Algorithms.pdf
- [4] VOLNÁ Eva. *Základy Softcomputingu* [online]. 2012 [cit. 2015-05-19]. Dostupné z: http://www1.osu.cz/~volna/Zaklady_softcomputingu_skripta.pdf
- [5] VOLNÁ Eva. *Evoluční algoritmy a neuronové sítě* [online]. 2012 [cit. 2015-05-19]. Dostupné z: http://www1.osu.cz/~volna/Evolucni_algoritmy_a_neuronove_site.pdf
- [6] KOMÍNEK Jan. *Heuristické algoritmy pro optimalizaci*. Brno 2012. Diplomová práce. Fakulta strojního inženýrství, Vysoké učení technické v Brně.
- [7] ZELINKA, Ivan, Zuzana OPLATKOVÁ, Miloš ŠEDA, Pavel OŠMERA a František VČELARĚ. *Evoluční výpočetní techniky: Principy a aplikace*. Praha: BEN, 2009. ISBN 978-80-7300-218-3.
- [8] ZELINKA, Ivan. *Optimalizační a heuristické algoritmy* [online]. 2012 [cit. 2015-05-20]. Dostupný z: <http://arg.vsb.cz/data/Vyuka/02%20BIV%20Evoluce%20-%20Uvod.pdf>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

Hf	Hybrid function
Cf	Composition functions
D	Dimenze
SC	Soft computing

SEZNAM OBRÁZKŮ

Obr. 1. Optimalizační metody.....	11
Obr. 2. Elliptic function (3D map).....	21
Obr. 3. Elliptic function (2D map).....	21
Obr. 4. Bent Cigar function (3D map).....	22
Obr. 5. Bent Cigar function (2D map).....	22
Obr. 6. Discus function (3D map).....	23
Obr. 7. Discus function (2D map).....	23
Obr. 8. Rosenbrock's function (3D map).....	24
Obr. 9. Rosenbrock's function (2D map).....	24
Obr. 10. Ackley's function (3D map).....	25
Obr. 11. Ackley's function (2D map).....	25
Obr. 12. Weierstrass function (3D map).....	26
Obr. 13. Weierstrass function (2D map).....	26
Obr. 14. Griewank's function (3D map).....	27
Obr. 15. Griewank's function (2D map).....	27
Obr. 16. Rastrigin's function (3D map).....	28
Obr. 17. Rastrigin's function (2D map).....	28
Obr. 18. Rotated Rastrigin's function (3D map).....	29
Obr. 19. Rotated Rastrigin's function (2D map).....	29
Obr. 20. Schwefel's function (3D map).....	30
Obr. 21. Schwefel's function (2D map).....	30
Obr. 22. Rotated Schwefel's function (3D map).....	31
Obr. 23. Rotated Schwefel's function (2D map).....	31
Obr. 24. Katsuura function (3D map).....	32
Obr. 25. Katsuura function (2D map).....	32
Obr. 26. HappyCat function (3D map).....	33
Obr. 27. HappyCat function (2D map).....	33
Obr. 28. HGBat function (3D map).....	34
Obr. 29. HGBat function (2D map).....	34
Obr. 30. Griewank's plus Rosenbrock's function (3D map).....	35
Obr. 31. Griewank's plus Rosenbrock's function (2D map).....	35
Obr. 32. Scaffer's F6 function (3D map).....	36

Obr. 33. Scaffer's F6 function (2D map).....	36
Obr. 38. 1D Řez funkce	43
Obr. 39. 2D Řez funkce	44

SEZNAM TABULEK

Tab. 1. Souhrn všech CEC'14 testovacích funkcí	18
Tab. 2. Srovnání pro 2D.....	45
Tab. 3. Srovnání pro 10D.....	46

SEZNAM PŘÍLOH

Příloha P I: CD se spustitelnými zdrojovými kódy pro SW Mathematica.