Tomas Bata University in Zlín
Faculty of Applied Informatics

Doctoral Thesis

Control Parameter Adaptation in Differential Evolution

Adaptace kontrolních parametrů v diferenciální evoluci

Author: **Ing. Adam Viktorin**
Branch of study: Engineering Informatics
Supervisor: doc. Ing. Roman Šenkeřík, Ph.D.
Advisor: doc. Ing. Zuzana Komínková Oplatková, Ph.D.

Zlín, 2021

At this place, I would like to thank all people that supported me throughout my studies and also throughout my life in general, whether it was family, friends, or colleagues. Without them, I would not be able to get this far, and I appreciate that.
A genuine thank you goes to my mentors, colleagues, and most importantly, friends - Alzbeta, Anezka, Zuzana, Michal, Roman, Tomas K. and Tomas T. who helped me countless times with everything, ranging from bureaucracy through research to personal matters.
Last but not least, I would like to thank the love of my life, my wife, for "adapting" to my character traits, bearing with me, pushing me forward, and for constantly reminding me that I should work on my dissertation when I was "running" away... to other projects.

"Correlation does not imply causation."

ABSTRAKT

Tato disertační práce popisuje autorovu výzkumnou aktivitu v oblasti adaptivních variant algoritmu diferenciální evoluce pro optimalizaci jednokriteriálních funkcí definovaných ve spojitém prostoru. První část práce popisuje oblast matematické optimalizace a její rozdělení do jednotlivých podkategorií podle charakteristik optimalizované funkce. Tyto charakterisitky jsou: počet optimializačních kritérií, typ vstupu, výpočetní složitost, typ prohledávaného prostoru řešení a počet optimalizovaných parametrů. Zároveň tato sekce zahrnuje popis typického zástupce metaheuristické optimalizace - evoluční výpočetní techniky.
Druhá část práce se věnuje variantám algoritmu diferenciální evoluce včetně variant s adaptivními kontrolními parametry. V jedné z podkapitol se autor věnuje i důvodům, proč si vybral algoritmus Success-History based Adaptive Differential Evolution jako základ své vědecké práce.
V experimentální části práce je navržen nástroj pro analýzu dynamiky populace evolučních algoritmů, který může být využit jak při tvorbě nových evolučních algoritmů, tak pro vyhodnocení vlastností algoritmů stávajících a aktuálně používaných. Mimo analýzu dynamiky populace obecně se autor zaměřil i na konkrétní algoritmy založené na diferenciální evoluci. Navrhl dvě úpravy vnitřní dynamiky - multi–chaotický framework pro výběr rodičů a adaptace kontrolních parametrů s využitím vzdálenosti jedinců. Obě techniky jsou zaměřeny na pomoc s hledáním správné rovnováhy mezi prohledáváním prostoru řešení do šířky a do hloubky. Na příkladu moderní verze diferenciální evoluce ve variantě jSO je ukázán přínos implementace adaptace kontrolních parametrů s využitím vzdálenosti jedinců. Takto upravený algoritmus byl nazván DISH a byl otestován na testovacích sadách spojených s celosvětovým kongresem evolučních technik - CEC (Congress on Evolutionary Computation). Výsledky ukazují, že využití nové adaptační strategie je vhodné především pro úlohy, které optimalizují větší množství vstupních parametrů.
Praktické využití algoritmu DISH je demonstrováno na příkladu hledání optimálního rozmístění spaloven odpadu v České republice. Upravený algoritmus DISH poskytuje pro menší instance problému srovnatelné řešení s deterministickými metodami. Pro větší instance problému již nejsou deterministické metody

schopny poskytnout řešení v akceptovatelném čase a proto je zde využití metaheuristického přístupu opodstatněno.
Výše zmíněné výsledky ukazují, že i v rámci jednoduchých změn vnitřní dynamiky algoritmu lze dosáhnout lepší výkonnosti. I proto si autor zvolil jako svůj budoucí výzkumný směr rozvíjení nástroje pro analýzu vnitřní populační dynamiky metaheuristických algoritmů.

SUMMARY

This doctoral thesis describes the author's research in the area of adaptive Differential Evolution variants for small–scale continuous single–objective optimization. The first part describes the topic of mathematical optimization and lists various problem domains according to the problem characteristics. Namely: number of objectives, input type, computational complexity, type of a search space, and problem scale. It also describes the area of metaheuristic optimization and Evolutionary Computation Techniques.
The Differential Evolution algorithm variants and control parameter adaptivity are described in the next part of this work and it also provides the justification of selecting Success–History based Adaptive Differential Evolution algorithm as a basis for author's research focus.
A novel population dynamic analysis tool is proposed in the experimental part. This tool can be used for the development process of new metaheuristic techniques as well as for the analysis of the state-of-the-art methods.
The experimental part also provides the proposal of multi–chaotic framework for parent selection for the Differential Evolution based algorithms and Distance based parameter adaptation, which can be implemented into adaptive variants of Differential Evolution algorithm to improve the balance between exploration and exploitation. The benefits of using Distance based parameter adaptation are shown on the improved jSO algorithm - DISH. The performance of both versions (jSO and DISH) is compared on the basis of Congress on Evolutionary Computation benchmark sets and shows that the DISH variant is more suitable

for optimization problems of a larger scale.
The practical use of the DISH algorithm is demonstrated on the operations research problem of finding optimal dislocation of waste–to–energy facilities in the Czech Republic. The improved DISH algorithm was able to provide comparable solutions for smaller instances of the problem and was also able to provide solutions for larger instances where traditional solvers failed.
Through the above–mentioned results, it can be seen that even simple changes in algorithms' inner dynamic can lead to significant improvements. Therefore, the research area of adaptive metaheuristics for optimization can benefit from knowledge gained through thorough algorithm analysis, which is the author's chosen research direction for the future.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AdapSS	Adaptive Strategy Selection in DE
ADE	Adaptive DE
AI	Artificial Intelligence
AIM-dDE	Adaptive Invasion–based Model for distributed DE
APTS	Adaptive Population Tuning Scheme
CDE-b6e6rl	Competitive DE
CEC	Congress on Evolutionary Computation
CoDE	Composite DE
CR	Crossover rate
D	Dimension
Db	Distance based parameter adaptation
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DE	Differential Evolution
DISH	Distance based parameter adaptation for Success–History based DE
DR_DISH	Distance Random DISH
ECT	Evolutionary Computation Technique
EPSDE	DE with Ensemble of Parameters and mutation Strategies
ESMDE	Evolving Surrogate Model–based DE
F	Scaling factor
FADE	Fuzzy Adaptive DE
FES	Function EvaluationS
G	Generation
HyDE–DF	Hybrid–adaptive DE with Decay Function
iL–SHADE	Improved L–SHADE
L-SHADE	SHADE with Linear decrease of the population size
LSHADE_EpSin	L–SHADE with Ensemble Sinusoidal parameter adaptation
LSHADE–RSP	LSHADE with Rank–based Selective Pressure strategy
MAXFES	MAXimum FES
MCO	Mean Cluster Occurrence
MC–SHADE	SHADE with Multi–Chaotic parent selection framework
MPD	Mean Population Diversity
MPEDE	Multi–Population based Ensemble of mutation strategies DE
NFL	No Free Lunch theorem
NP	Population size
PD	Population Diversity
PRNG	Pseudo–Random Number Generator
SADE	Self–Adaptive DE
SAMODE	Seld–Adaptive Multi–Operator DEE
SDE	Self–adaptive DE
SHADE	Success–History based Adaptive DE
SinDE	Sinusoidal DE
SPS–L–SHADE–EIG	L–SHADE with Successful–Parent–Selecting framework and EIGenvector–based crossover
SPSRDEMMS	Structured Population Size Reduction DE with Multiple Mutation Strategies

1 INTRODUCTION

Artificial Intelligence (AI) has become a significant part of our everyday lives, even if we do not notice it sometimes. Our commute to work may be optimized according to the current traffic situation with intelligent path planning algorithms. Voice assistants in smart devices use AI to understand our speech and cater to our queries. Computer games we play use AI to offer worthy opponents. Virtual keyboards in mobile phones adapt to our style of writing, camera application is able to adjust its settings according to the current scene and focus on the faces of photographed people. Our emails are automatically filtered and categorized based on their content with clever algorithms. Antivirus is applying AI techniques for the detection of malicious software before installation. Advertisements we see during internet browsing are based on our search history and buyer preferences. Automatic text translation is becoming more precise with ever more powerful AI techniques. The newest cars are more often than not designed by computers. The production and manufacturing can be scheduled and managed by AI algorithms. Scanners at airports use AI to detect potentially dangerous items. Search engines use powerful AI to enhance their performance. Parking gates detect license plates and check whether they should open. Even electric toothbrushes use AI to learn user patterns. This work also demonstrates an AI approach to the waste–to–energy facility location problem.

The list could be endless, but all of these applications have one thing in common. They build on solid foundations in AI basic research and algorithms that were developed during the last few decades. This dissertation describes the author's contribution to a specific part of the research field of mathematical optimization and Evolutionary Computational Techniques (ECTs). This chapter provides a basic introduction into the research area and its subcategories. The second chapter specifies the main dissertation goal and selected methods for achieving it. Chapters 3, 4 and 5 provide more details about the Differential Evolution (DE) algorithm, control parameter adaptivity and modern adaptive DE variants. Chapter 6 describes methods developed during author's doctoral studies. Chapters 7 and 8 summarize author's contribution to the science and practice and fulfillment of the dissertation goal. And the last chapter contains concluding remarks with possible future expansion of this work.

1.1 Mathematical optimization

Mathematical optimization is a scientific research area that deals with searching for the problem parameter values combination that would yield the best result – objective function value (e.g., minimization of a cost or maximization of a profit). Of course, there are multiple subcategories of optimization tasks that require appropriate methods for their solving. These categories are divided according to [1] as follows:

- **The number of objectives:**
  - **Single–objective optimization** – the goal is to optimize one objective.
  - **Multi–objective optimization** – the goal is to simultaneously optimize two or three objectives.
  - **Many–objective optimization** – the goal is to simultaneously optimize more than three objectives.
- **The input parameter type:**
  - **Discrete/Combinatorial optimization** – optimized parameters have a finite number of possible values.
  - **Continuous/Real–valued/Numerical optimization** – optimized parameters are real–valued.
- **The computational complexity of the objective function:**
  - **Expensive optimization** – it is computationally expensive to evaluate the objective function of a single solution.
  - **Non–expensive optimization** – it is computationally inexpensive to evaluate the objective function of a single solution.

- **The search space type:**
  - **Unconstrained optimization** – the search space of parameter values is infinite.
  - **Bound–constrained optimization** – the search space is not constrained; individual parameters have only upper and lower bounds.
  - **Constrained optimization** – the search space is constrained by additional equalities or inequalities.
- **The scale of the problem (number of optimized parameters / dimensionality):**
  - **Small-scale optimization** – the dimensionality of the problem is between 1 and 100.
  - **Large-scale optimization** – the dimensionality of the problem is in hundreds or thousands.

Optimization algorithms are methods for solving optimization problems and can also be classified into subcategories. One of the main classifications might be by algorithms stochasticity into two groups - deterministic and stochastic [2]. Deterministic algorithms follow a rigorous mathematical approach and work with the mathematical model of the problem to provide the optimal solution. Unfortunately, the most complex tasks are unsolvable by deterministic optimization algorithms due to the time and computational constraints. Thus, stochastic optimization algorithms that use randomness in their core are employed. These algorithms can also be titled metaheuristics. Metaheuristics treat optimization problems as black boxes - trying to solve optimization tasks using only the information of an input/output combination and learning from that. Due to their stochastic nature, metaheuristics do not guarantee a finding of the global optimum.
ECTs form a particular metaheuristic class based on the principle of natural selection and are described in the next section.

1.2 Evolutionary computational techniques in optimization

ECTs are part of the soft computing field and are based on the Darwinian theory of evolution [3]. In this sense, ECTs often work with a population of individuals. Those individuals are combined via crossover operator (an analogy with breeding), and the resulting individuals are further mutated via mutation operator (analogous to gene mutation) to provide possibly fitter offspring for the next generation. This process is applied to the whole population to provide a new generation of solutions to the given optimization task. Thanks to this, ECTs can be used to optimize particularly hard optimization tasks that could not be solved, due to the computational complexity, by traditional deterministic methods.

ECTs are often employed for solving complex optimization tasks in various problem domains (e.g., load forecasting in smart grids [4], friction welding [5], underwater glider path-planning [6], species distribution modeling [7], large scale flexible scheduling [8], markerless human motion capture [9] or drug design [10]). The ECT's goal is to guide a search through a search space of feasible solutions and to find a satisfactory solution in a reasonable time. The solution mentioned here is a feature vector of values that correspond to the optimized parameters of the problem. The quality of a single solution (feature vector) is evaluated by the objective function, where the objective may be either minimization or maximization of the function value. The feature vector, along with its corresponding objective function value, makes up an individual of ECT. Therefore, the goal of each ECT run is to find an individual with a sufficient objective function value. Since ECTs are metaheuristic techniques, there is no guarantee that the found solution will be optimal. Each independent run of the evolutionary algorithm can also provide a different solution, and therefore, algorithms are often run multiple times.

One of the problems while using ECTs is a requirement for a control parameter setting. These parameters can significantly impact the algorithm's performance, and therefore their correct setting is essential. One of the latest trends in ECTs is to address this problem by adapting the algorithm's behavior (via adapting control parameter values) to the given optimization task. With the famous No

Free Lunch (NFL) theorem in mind [11], adaptive algorithms try to overcome the problem of correct parameter setting by incorporating knowledge of previously successful values of these parameters into the evolution process in an intelligent way. Thus, the user is no longer obliged to fine–tune these parameters manually. The Differential Evolution (DE) algorithm [12] is one of the main representatives of ECTs and has been thoroughly studied over the last 25 years. Moreover, its adaptive variants from the last decade show promising results in various problem domains, and that is why the DE was selected as the author's research focus. Particularly, this dissertation is focused on the DE algorithm and its adaptive variants for small–scale continuous single–objective optimization problems with a possible expansion to the area of large–scale optimization.

2 DISSERTATION GOAL

The prevailing trend in the metaheuristic optimization seems to be a constant development of new techniques without proper justification of their need. This was creditably described by Sörensen in [13]. A similar issue is a vast amount of new versions of existing successful algorithms. In author's opinion, the main problem is not the great volume of variants, but the lack of proper analysis of implemented changes and their influence on the algorithm's behavior. Therefore, the goal of this dissertation is to try and contribute to the scientific area of metaheuristic optimization by developing analysis tools which use datamining techniques to help with understanding the population dynamic of metaheuristic algorithms. More specifically, how the control parameter adaptation in Differential Evolution–based algorithms influences the population dynamic and whether this information can be used in the development and testing of new ideas.

Selected methods to achieve the above stated dissertation goal:

- **Analysis** – current state–of–the–art methods in adaptive DE field will be analyzed from the perspective of control parameter adaptation. What mechanism decides the direction of the adaptation and whether it is based on a greedy approach. The effect of the adaptation on exploration/exploitation abilities of the algorithm will be studied.
- **Programming** – selected state–of–the–art adaptive DE variants will be programmed in Java, Wolfram Mathematica, and Python in order to work with these algorithms and test the proposed modifications.
- **Testing** – the programmed code will be tested against possible errors and malfunctions.
- **Benchmarking** – proposed algorithm variants will be benchmarked on the basis of CEC benchmark sets of test functions.
- **Result evaluation** – evaluation of the results will be executed within the rules of used benchmark sets. This will create a basis for result comparison with the scientific community.

- **Result analysis** – the statistical analysis of obtained results will be performed. Population dynamic analysis will be used to asses the exploration/exploitation properties of the proposed framework.

3 CANONICAL DIFFERENTIAL EVOLUTION

The original algorithm of DE was proposed in a technical report in 1995 [12] and published in 1997 by Storn and Price [14]. Since its introduction, it is considered as one of the best performing algorithms for global optimization over continuous spaces. Its key features are simplicity and universality. The original paper [14] proposed DE with only three control parameters – scaling factor F, crossover rate CR, and population size NP. These three parameters have to be set by the user, and their correct setting is highly dependable on the optimization task [15, 16].

The DE algorithm is initialized with a random population of individuals $\boldsymbol{P}$, that represent solutions to the optimization problem. In continuous optimization, each individual is composed of a vector $\boldsymbol{x}$ of length D, which is a dimensionality (number of optimized parameters) of the problem and objective function value $f(\boldsymbol{x})$. Each vector component represents a value of the corresponding optimized parameter.

For each individual in a population, three mutually different individuals are selected for mutation, and the resulting mutated vector $\boldsymbol{v}$ is combined with the target vector $\boldsymbol{x}$ in the crossover step. The objective function value $f(\boldsymbol{u})$ of the resulting trial vector $\boldsymbol{u}$ is evaluated and compared to that of the target individual $\boldsymbol{x}$. When the quality (objective function value) of the trial individual $\boldsymbol{u}$ is better, it is placed into the next generation. Otherwise, the target individual $\boldsymbol{x}$ is placed there. This step is called selection. The process is repeated until the stopping criterion is met (e.g., the maximum number of objective function evaluations, the maximum number of generations, the low bound for diversity between objective function values in population or time restriction).

The following sections describe four steps of the DE: initialization, mutation, crossover, and selection.

3.1 Initialization

As aforementioned, the initial population $\boldsymbol{P}$, of size *NP*, is randomly generated. For this purpose, the individual vector $\boldsymbol{x}_i$ components are generated by Pseudo-Random Number Generator (PRNG) with uniform distribution from the range which is specified for the problem by lower $\boldsymbol{lo}$ and upper $\boldsymbol{up}$ bounds (3.1).

$$\boldsymbol{x}_{j,i} = U\left[\boldsymbol{lo}_j,\ \boldsymbol{up}_j\right] \text{ for } j = 1,\ \ldots,\ D \tag{3.1}$$

Where i is the index of a target individual, j is the index of current parameter and D is the dimensionality of the problem.
In the initialization phase, a scaling factor value F and crossover value CR has to be assigned as well. The typical range for F value is (0, 2] and for CR, it is [0, 1].

3.2 Mutation

In the mutation step, three mutually different individuals $\boldsymbol{x}_{r1},\ \boldsymbol{x}_{r2},\ \boldsymbol{x}_{r3}$ from a population are randomly selected and combined in accordance with the mutation strategy. The original mutation strategy of canonical DE is "rand/1" and is depicted in (3.2).

$$\boldsymbol{v}_i = \boldsymbol{x}_{r1} + F\left(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}\right) \tag{3.2}$$

Where $r1 \neq r2 \neq r3 \neq i$, F is the scaling factor value and $\boldsymbol{v}_i$ is the resulting mutated vector.

3.3 Crossover

In the crossover step, the mutated vector $\boldsymbol{v}_i$ is combined with the target vector $\boldsymbol{x}_i$ and produces a trial vector $\boldsymbol{u}_i$. The binomial crossover (3.3) is used in canonical DE.

$$\boldsymbol{u}_{j,i} = \begin{cases} \boldsymbol{v}_{j,i} & \text{if } U\left[0,1\right] \leq CR \text{ or } j = j_{rand} \\ \boldsymbol{x}_{j,i} & \text{otherwise} \end{cases} \tag{3.3}$$

Where CR is the used crossover rate value and j_{rand} is an index of a parameter that has to be taken from the mutated vector $\boldsymbol{u}_i$ (this ensures the generation of a vector with at least one component from the mutated vector in order to not present objective function with already evaluated solutions).

3.4 Selection

The selection step ensures that the optimization will progress towards better solutions because it allows only individuals of better or at least equal objective function value to proceed into the next generation *G+1* (3.4).

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G} & \text{if } f\left(\boldsymbol{u}_{i,G}\right) \leq f\left(\boldsymbol{x}_{i,G}\right) \\ \boldsymbol{x}_{i,G} & \text{otherwise} \end{cases} \tag{3.4}$$

Where G is the index of the current generation.
For easier understanding, the basic concept of the DE algorithm is depicted in the pseudo–code in Appendix A.

4 DIFFERENTIAL EVOLUTION AND ADAPTIVITY

Troublesome fine–tuning of control parameters soon became a problem for researchers and practitioners who were trying to accommodate DE for solving complex optimization problems. Therefore, researchers started working on this problem by studying DE's behavior on different types of objective function landscapes and tried to come up with a simple guide for the setting of control parameter values.

In the original technical report by Storn and Price from 1995 [12], authors recommend population size NP between 5*D and 10*D, the initial choice of a scaling factor $F = 0.5$ and an effective range for F from 0.4 to 1. The initial choice for crossover rate CR was suggested to 0.1, but for fast convergence 0.9 or 1. In 2002, Gamperle et al. [15] tested not only the setting of control parameters but also the choice of mutation and crossover operators. They suggested population size NP between 3*D and 8*D, scaling factor $F = 0.6$, and crossover rate CR between 0.3 and 0.9. Moreover, the authors proposed using "DE/best/2/bin" and "DE/rand/1/bin" variants for the mutation and crossover strategies. In [17], Ronkkonen et al. used a testbed of 25 scalable objective functions to evaluate the performance of DE, and they recommended scaling factor $F = 0.9$ as a good first choice and range from 0.4 to 0.95, crossover rate CR from 0 to 0.2 for separable functions and 0.9 to 1 for non–separable functions.

As can be seen, suggestions from different authors vary and are highly dependent on the choice of objective function testbed used in the study. This fact only supports the NFL theorem [11], which roughly states that there is no universal algorithm or algorithm parameter setting, that would solve all the different types of optimization problems optimally.

The solution to these problems may lie in the adaptive behavior of the DE algorithm. Since the setting of control parameters and mutation and crossover operators is dependent on the optimized objective function, these variables might be set during the optimization run according to the success of the currently implemented settings. Adaptivity in DE is a current trend in the field and has shown auspicious results right from its beginning around the year 2005. The following paragraphs present the evolution timeline of selected adaptive DE–based algorithms for a continuous single-objective optimization along with their authors and a short description of each adaptive scheme.

- 2004
  - **Fuzzy Adaptive DE (FADE)** [18] by Liu and Lampinen – uses fuzzy logic controllers to adapt F and CR values.
- 2005
  - **Self–adaptive DE (SDE)** [19] by Omran et al. – self–adaptation of scaling factor F, which is sampled from a normal distribution $N(0.5, 0.15)$ at the beginning, but each individual remembers its scaling factor for future generations. Before mutation, scaling factor F is given by a recombination of 3 randomly selected values from population.
  - **Self–Adaptive DE (SaDE)** [20] by Qin and Suganthan – this algorithm adapts mutation and crossover operators as well as CR values during the optimization run. Mutation and crossover operator combinations are selected from a pool of strategies according to their previous success in generating better offspring (in this version, the pool contained only two strategies). Scaling factor F is sampled from a normal distribution $N(0.5, 0.3)$, and the crossover rate values CR are also sampled from a normal distribution, but with the mean value dependent on the successful values from previous generations.
- 2006
  - **jDE** [21] by Brest et al. – in this work, authors proposed an algorithm with adaptive F and CR values according to two probabilities of parameter adjustment – $\tau 1$ for scaling factor and $\tau 2$ for crossover rate. These parameters are usually set to 0.1 and thus correspond to the probability of a parameter change of 10%. Furthermore, each individual has its combination of F and CR values, that are stored alongside the feature vector.

- 2009
  - **Improved SaDE** [22] by Qin et al. – an improved version of SaDE algorithm from 2005 implemented four strategies into the strategy pool and the same scheme for F and adaptation for CR.
  - **JADE** [23] by Zhang and Sanderson – authors proposed a novel mutation strategy "current-to-$p$best/1" with an optional archive of inferior solutions. Both scaling factor F and crossover rate CR are adapted according to previously successful values and are sampled from Cauchy distribution and normal distribution, respectively. Both distributions are based on the mean of successful values from the previous generation.
- 2011
  - **DE with Ensemble of Parameters and mutation Strategies (EPSDE)** [24] by Mallipeddi et al. – this algorithm uses an ensemble of parameter values and mutation strategies. Each individual is assigned a mutation strategy and parameter combination randomly at the beginning. When the offspring produced by an individual succeeds in the selection, mutation strategy and parameter values are stored within it for the next evaluation. When the offspring is worse than its parent, mutation strategy and parameter values are reinitialized.
  - **Composite DE (CoDE)** [25] by Wang et al. – this algorithm is not essentially using an adaptation of any of its parameters. It uses three distinct strategies and three popular parameter settings. Each individual provides three trial vectors based on each of the strategies with randomly selected parameter values from the parameter value pool. The best outcome in terms of objective function value is tested against its parent in the selection step. The adaptation there is, therefore, based on the usability of a strategy for a given objective function.
  - **Adaptive Strategy Selection in DE (AdapSS)** [26] by Gong et

al. – in this work, the authors proposed an adaptive strategy selection framework for DE–based algorithms. They tested two approaches – probability matching and adaptive pursuit and showed that both are capable of adjusting the strategy selection.

- **Self–Adaptive Multi–Operator DE (SAMODE)** [27] by Elsayed et al. – this algorithm uses self–adaptive scaling factor F and CR values but also divides the population into four subpopulations. Each subpopulation uses a different mutation strategy, and the subpopulation sizes are determined based on their individual quality. Thus, better performing mutation strategies will get a reward in the form of a larger population.

- 2013

  - **Competitive DE (CDE-b6e6rl)** [28] by Tvrdik and Polakova – a DE variant with twelve competing strategies. It uses two mutation strategies "DE/randrl/1/bin" and "DE/ranrdl/1/exp", each with six different settings of scaling factor F and crossover rate CR. Each strategy has a probability of selection adapted according to their success in generating better offspring.
  - **Structured Population Size Reduction DE with Multiple Mutation Strategies (SPSRDEMMS)** [29] by Zamuda and Brest – algorithm based on the jDE [21], but with population size reduction and multiple mutation strategies using a structured population. The population size reduction is performed once after a few generations, and the population is halved; where the worst individuals in the population are the ones discarded. Two mutation strategies are used in this algorithm – "rand/1" and "best/1".
  - **Adaptive Population Tuning Scheme (APTS) for DE** [30] by Zhu et al. – authors proposed a new population management scheme, which is based on solution–searching status. It dynamically adjusts the population size, removes redundant individuals, and perturbs the population by generating "fine" individuals.

  - **Success–History based Adaptive DE (SHADE)** [31] by Tanabe and Fukunaga – this algorithm is based on the JADE [23], but implements historical memories for storing previously successful scaling factor F and crossover rate CR values from previous generations. These memories are later used for the generation of F and CR values for each individual.

- 2014
  - **Adaptive DE (ADE)** [32] by Yu et al. – in this work, authors propose adaptive DE with two–level parameter adaptation. This algorithm uses a new strategy, "DE/Lbest/1", which is a variation to the original "DE/best/1" with multiple sub–populations, each having its locally best individual. The two–level parameter adaptation is based on the estimation of optimization state, whether it is exploiting or exploring the search space. The scaling factor F and crossover rate CR are generated on the population level and serve as a base for individual–level parameter values.
  - **Adaptive Invasion–based Model for distributed DE (AIM–dDE)** [33] by Falco et al. – the adaptive model uses three updating schemes for setting the scaling factor F and crossover rate CR values. As the name suggests, this adaptive model is suitable for distributed DE, where the network is created from the nodes represented by different DE strategies. After a predefined number of iterations, important knowledge is transferred between the connected nodes. In this case, important knowledge consists of better individuals and useful control parameter values.
  - **SHADE with Linear decrease of the population size (L–SHADE)** [34] - by Tanabe and Fukunaga – authors further improved their SHADE [31] algorithm by implementing linear decrease of the population size. The population is linearly decreased during the optimization, and thus, the starting large population provides explorative capabilities, whereas a smaller population in the later stages of optimization promotes the exploitation of the search region.

- 2015
  - **Multi–Population based Ensemble of mutation strategies DE (MPEDE)** [35] by Wu et al. – this algorithm dynamically partitions the population into four sub–populations. Three sub–populations are given different mutation strategies, and the fourth sub–population is a reward sub–population, which uses the currently best performing strategy and serves as a computational resource.
  - **Evolving Surrogate Model–based DE (ESMDE)** [36] by Mallipeddi and Lee – this DE variant uses a surrogate model created from the current population by simple Kriging model [37] in order to select appropriate mutation and crossover strategies.
  - **Sinusoidal DE (SinDE)** [38] by Draa et al. – authors propose sinusoidal formulas for automatic adjustment of the scaling factor F and crossover rate CR values in order to balance the algorithm's exploration and exploitation capabilities.
  - **L-SHADE with Successful–Parent–Selecting framework and EIGenvector–based crossover (SPS–L–SHADE–EIG)** [39] by Guo et al. – authors combined popular L–SHADE [34] algorithm with eigenvector–based crossover, which is useful while solving optimization problems with highly correlated variables and successful–parent–selection framework helps to overcome the problem of stagnation of the population. Thus, this algorithm not only adapts control parameter values but also its crossover strategy and parent selection in mutation strategy.
- 2016
  - **L–SHADE with Ensemble Sinusoidal parameter adaptation (LSHADE_EpSin)** [40] by Awad et al. – authors combined the L-SHADE [34] algorithm with adaptive sinusoidal control parameter adjustment and local search. The sinusoidal adjustment is used for the first half of generations, while the original L–SHADE adaptation is used for the remaining half. The local search is run when the

population size NP is equal to or less than 20 for the first time.

- **SHADE with Multi–Chaotic parent selection framework (MC–SHADE)** [41] by Viktorin et al. – this variant utilizes five chaotic maps as PRNGs for the selection of parent individuals for the mutation strategy in SHADE algorithm [31]. The proposed MC framework is suitable for all SHADE–based algorithms.
- **LSHADE44** [42] by Polakova et al. – a similar approach as in [28] was proposed for L–SHADE [34] in this work. It is an L–SHADE algorithm with two different types of mutation and crossover operators combined. The selection of a suitable strategy is based on the previous success in generating trial offspring.
- **Improved L–SHADE (iL–SHADE)** [43] by Brest et al. – authors propose few tweaks to the original L–SHADE [34] algorithm. The historical memories of scaling factor F and crossover rate CR are initialized to higher values, and one cell of the memory remains unchanged during the optimization. Large values of F and small CR values are not allowed in the early stage of optimization to avoid premature convergence to local optima. There is also a small change to the mutation strategy "current-to-$p$best/1", where the p value is computed differently and is based on the optimization stage.

• 2017

- **Distance based parameter adaptation (Db)** [44] by Viktorin et al. – authors proposed an update to the weighting scheme for SHADE–based [31] algorithms. Instead of the greedy approach, which calculates the weights of successful scaling factor F and crossover rate CR values from the improvement in objective function value, the distance based parameter adaptation promotes exploration of the search space by incorporating the distance between the original and trial individuals as a basis for the weights.
- **jSO** [45] by Brest et al. – another iteration of iL–SHADE [43], which uses an updated weighted mutation strategy "current-to-$p$best-w/1"

and slightly changes the predefined fixed values of scaling factor F and crossover rate CR in historical memories.

- 2018
  - **LSHADE with Rank–based Selective Pressure strategy (LSHADE–RSP)** [46] by Stanovov et al. – This algorithm uses an updated mutation strategy from jSO with rank–based selection of random individuals from the population. The strategy is titled "current-to-$p$best/r" and is inspired by ranked selection form genetic algorithms. It also incorporates the same adaptive scheme for control parameters as jSO.
  - **Distance based parameter adaptation for Success–History based DE (DISH)** [47] by Viktorin et al. – DISH algorithm was created by implementing Db adaptation into jSO and led to the improvement of the algorithm mainly for optimization problems of larger scale.
- 2019
  - **jDE100** [48] by Brest et al – as the name suggests, it is based on the previously mentioned jDE [21] algorithm with few updates. jDE100 uses two sub–populations with one–way migration of the best individual. It also introduces new limits for scaling factor F and crossover rate CR values and utilizes restart strategy if the variance of best population members in terms of objective function value decreases under a specified threshold.
  - **Hybrid–adaptive DE with Decay Function (HyDE–DF)** by Lezama et al. – the algorithm incorporates the same adaptive mechanism for control parameters as it was in jDE, but uses a novel mutation strategy titled "target-to-perturbed_best/1". This strategy perturbs the vector of the best individual by a perturbation factor taken from a normal distribution, calculates its difference to a target vector and gradually decreases the influence of this difference in the mutation strategy with algorithm generations. It also implements a

restart strategy if a specified number of successive generations show no improvement in objective function value.

– **DISHchain 1e+12** [49] by Zamuda – DISH algorithm with larger population size and a big computational budget designed specifically for CEC2019 competition [50].

A quick summary of the algorithms with their characteristics concerning parameter adaptivity is given in the following Tab. 4.1.

Tab. 4.1 Summary - selected adaptive DE variants and their characteristics.

Year	**Algorithm**	**Adaptive CR**	**Adaptive F**	**Adaptive operators (mutation & crossover)**	**Adaptive NP**
2004	FADE	Fuzzy logic controllers			
2005	SDE		recombination		
	SaDE	sampling		pool of 2 strategies	
2006	jDE	stochastic change, otherwise retain			
2009	Improved SaDE	sampling		pool of 4 strategies	
	JADE	sampling			
2011	EPSDE	pool		pool of 3 strategies	
	CoDE	best from 3 strategies			
	AdapSS	based on underlying algorithm		pool of 4 strategies	
	SAMODE	recombination		4 competing sub–populations	
2013	CDE-b6e6rl	12 competing strategies			
	SPSRDEMMS	jDE scheme		2 sub–populations	deterministic halving
	APTS				dynamic adjustment
	SHADE	sampling with historical memories			
2014	ADE	two–level adaptation		sub–populations	
	AIM–dDE	3 updating schemes		distributed populations	
	L–SHADE	SHADE scheme			deterministic linear decrease
2015	MPEDE			3 competeing sub–populations, 1 reward	
	ESMDE			surrogate selection	
	SinDE	deterministic sinusoidal change			
	SPS–L–SHADE–EIG	SHADE scheme		eigenvector crossover and successful parent selection	deterministic linear decrease
2016	LSHADE_EpSin	SHADE + SinDE scheme			deterministic linear decrease
	MC–SHADE	SHADE scheme		chaos–enhanced parent selection	
	LSHADE44	SHADE scheme		4 competing strategies	deterministic linear decrease
	iL–SHADE	updated SHADE scheme			deterministic linear decrease
2017	Db	distance based SHADE adaptation			
	jSO	updated iL–SHADE scheme			deterministic linear decrease
2018	LSHADE–RSP	jSO scheme		current-to-$p$best/r mutation	deterministic linear decrease
	DISH	jSO scheme with Db implementation			deterministic linear decrease
2019	jDE100	jDE adaptation with new limits		two sub–populations	
	HyDE–DF	jDE adaptation		target-to-perturbed_best/1 mutation	
	DISHchain 1e+12	DISH scheme			deterministic linear decrease

As perceivable, there are a plethora of different variants of adaptive DEs, and the question is, how to select a suitable algorithm for the problem at hand. Luckily, since 2005, there is an annual competition in numerical optimization held within the Congress on Evolutionary Computation (CEC), which provides a benchmark incorporating multiple test functions from various domains. The benchmarks are named after the year they were used in the competition, e.g., CEC2015 benchmark set. These benchmarks also provide a good testbed for researchers who can easily compare their algorithms with the community. Practitioners can also use the competition results as useful guidance when searching for a suitable algorithm for their problem.
An interesting pattern is visible when observing the results of the CEC competition since 2013. In 2013, the SHADE [31] was proposed and sent for the competition, where it placed 3rd [51]. The next year, Tanabe and Fukunaga proposed L–SHADE [34] and won the CEC2014 competition [52]. In 2015, the competition had three different test scenarios - learning-based optimization, expensive optimization, and multi–niche optimization. Closest to the original bound–constrained numerical optimization was the learning–based test case [53], which allowed the competitors to fine–tune the algorithm's parameters for each problem. This competition was won by SPS–L–SHADE–EIG [39], but there were two other algorithms based on the L–SHADE [34] – DesPA [54], which ranked 2nd and LSHADE–ND [55], which shared the 3rd rank with MVMO [56]. In 2016, the competition had four test scenarios, and the single–objective numerical optimization returned to the set with the same testbed as in 2014 [52]. This time, five out of nine algorithms were based on either SHADE [31], or L–SHADE [34] and the LSHADE_EpSin [40] became the joint winner with UMOEAII [57]. In 2017, algorithms based on L–SHADE [34] ended on 2nd (jSO [45]), 3rd (LSHADE_cnEpSin [58]) and 4th place (LSHADE_SPACMA [59]). And in 2018, 2nd and 3rd places also belonged to L–SHADE–based [34] algorithms - LSHADE–RSP [46] and ELSHADE–SPACMA (not published) respectively. In 2019, the CEC competition changed its format and prepared a whole new benchmark - 100–digit challenge [50], which presented a new approach of evaluating algorithms. The goal was to optimize 10 different objective functions in various dimensions up to the accuracy of 10 decimal digits. If an algorithm was able to

do this at least 25 times out of 50 tries on one function, it got the full score of 10 points. Thus, the maximum score was 100 points (10 functions, 10 points each). The total of 18 algorithms were sent for this competition and whole 13 of them were based on adaptive DE variants [60]. Joined 1st place with 100 points was obtained by jDE100 [48] (jDE–based) and DISHchain 1e+12 (L–SHADE–based) [49], joined 2nd place with 93 points was obtained by HyDE-DF [61] (jDE–based) and SOMA T3A [62].

It is apparent that SHADE [31] and L–SHADE [34] algorithms created an excellent basis to start on when designing an efficient optimization algorithm for single–objective bound–constrained numerical optimization, since they are still core parts of the best performing algorithms from the latest CEC competitions. Therefore, their selection as a starting point for the author's research in 2015 was continually justified.

5 SHADE/L–SHADE

As aforementioned, the SHADE algorithm was proposed with a self–adaptive mechanism of some of its control parameters to avoid their fine–tuning. The control parameters in question are scaling factor F and crossover rate CR. It is fair to mention that the SHADE algorithm is based on Zhang and Sanderson's JADE [23] and shares a lot of its mechanisms. The main difference is in the historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ for successful scaling factor and crossover rate values with their update mechanism.
The following subsections describe individual steps of the SHADE algorithm: initialization, mutation, crossover, selection, and historical memory update.

5.1 Initialization

The initialization of the population $\boldsymbol{P}$ is the same as in the case of canonical DE (3.1). However this step also initializes the historical memories M_{CR} and M_F (5.1).

$$\boldsymbol{M}_{CR,i} = \boldsymbol{M}_{F,i} = 0.5 \text{ for } i = 1, \ldots, H \tag{5.1}$$

Where H is a user–defined size of historical memories.
Also, the external archive of inferior solutions $\boldsymbol{A}$ has to be initialized. Because of no previous inferior solutions, it is initialized empty, $\boldsymbol{A} = \emptyset$. And index k for historical memory updates is initialized to 1.

5.2 Mutation

Mutation strategy "current–to–$p$best/1" was introduced in [23] and it combines four mutually different vectors in the creation of the mutated vector $\boldsymbol{v}$. Therefore, $\boldsymbol{x}_{pbest} \neq \boldsymbol{x}_{r1} \neq \boldsymbol{x}_{r2} \neq \boldsymbol{x}_i$ (5.2).

$$\boldsymbol{v}_i = \boldsymbol{x}_i + F_i\left(\boldsymbol{x}_{pbest} - \boldsymbol{x}_i\right) + F_i\left(\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}\right) \tag{5.2}$$

Where $\boldsymbol{x}_{pbest}$ is randomly selected individual from the best $NP \times p$ individuals in the current population. The p value is randomly generated for each mutation by PRNG with uniform distribution from the range $[p_{min}, 0.2]$ and $p_{min} = 2/NP$. Vector $\boldsymbol{x}_{r1}$ is randomly selected from the current population $\boldsymbol{P}$. Vector $\boldsymbol{x}_{r2}$ is randomly selected from the union of the current population $\boldsymbol{P}$ and external archive $\boldsymbol{A}$. The scaling factor value F_i is given by (5.3).

$$F_i = C\left[\boldsymbol{M}_{F,r}, 0.1\right] \tag{5.3}$$

Where $\boldsymbol{M}_{F,r}$ is a randomly selected value (index r is generated by PRNG from the range 1 to H) from $\boldsymbol{M}_F$ memory, and C stands for Cauchy distribution. Therefore the F_i value is generated from the Cauchy distribution with location parameter value $\boldsymbol{M}_{F,r}$ and scale parameter value of 0.1. If the generated value F_i is higher than 1, it is truncated to 1, and if it is less or equal to 0, it is generated again by (5.3).

5.3 Crossover

In the crossover step, the trial vector $\boldsymbol{u}$ is created from the mutated $\boldsymbol{v}$ and the target $\boldsymbol{x}$ vector. For each vector component, a PRNG with uniform distribution $U[0,1]$ is used to generate a random value. If this random value is less or equal to a given crossover rate value CR_i, the current vector component will be taken from a trial vector. Otherwise, it will be taken from the target vector (5.4). There is also a safety measure, which ensures that at least one vector component will be taken from the trial vector. This is given by a randomly generated component index j_{rand}.

$$\boldsymbol{u}_{j,i} = \begin{cases} \boldsymbol{v}_{j,i} & \text{if } U\left[0,1\right] \leq CR_i \text{ or } j = j_{rand} \\ \boldsymbol{x}_{j,i} & \text{otherwise} \end{cases} \tag{5.4}$$

Overall, the crossover step is very similar to that of the canonical DE. The only difference is, that the crossover rate value CR_i is generated from a normal distribution with a mean parameter value $\boldsymbol{M}_{CR,r}$ selected from the crossover rate historical memory $\boldsymbol{M}_{CR}$ by the same index r as in the scaling factor case and standard deviation value of 0.1 (5.5).

$$CR_i = N\left[\boldsymbol{M}_{CR,r}, 0.1\right] \tag{5.5}$$

When the generated CR_i value is less than 0, it is replaced by 0, and when it is greater than 1, it is replaced by 1.

5.4 Selection

There is no change between the selection step of canonical DE and SHADE, and therefore, the equation (3.4) also applies.

5.5 Update of historical memories

Historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ are initialized according to (5.1), but their components change during the evolution. These memories serve to hold successful values of F and CR used in mutation and crossover steps. Successful regarding producing trial individual better than the target individual. During every single generation, these successful values are stored in their corresponding arrays $\boldsymbol{S}_F$ and $\boldsymbol{S}_{CR}$. After each generation, one cell of $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ memories is updated. This cell is given by the index k, which starts at 1 and increases by 1 after each generation. When it overflows the memory size H, it is reset to 1. The new value of k-th cell for $\boldsymbol{M}_F$ is calculated by (5.6) and for $\boldsymbol{M}_{CR}$ by (5.7).

$$\boldsymbol{M}_{F,k} = \begin{cases} \text{mean}_{WL}\left(\boldsymbol{S}_F\right) & \text{if } \boldsymbol{S}_F \neq \emptyset \\ \boldsymbol{M}_{F,k} & \text{otherwise} \end{cases} \tag{5.6}$$

$$\boldsymbol{M}_{CR,k} = \begin{cases} \text{mean}_{WL}\left(\boldsymbol{S}_{CR}\right) & \text{if } \boldsymbol{S}_{CR} \neq \emptyset \\ \boldsymbol{M}_{CR,k} & \text{otherwise} \end{cases} \tag{5.7}$$

Where $\text{mean}_{WL}()$ stands for weighted Lehmer mean (5.8).

$$\mathrm{mean}_{WL}\left(\boldsymbol{S}\right) = \frac{\sum_{n=1}^{|\boldsymbol{S}|} \boldsymbol{w}_n \cdot \boldsymbol{S}_n^2}{\sum_{n=1}^{|\boldsymbol{S}|} \boldsymbol{w}_n \cdot \boldsymbol{S}_n} \tag{5.8}$$

Where the weight vector $\boldsymbol{w}$ is given by (5.9) and is based on the improvement in objective function value between trial and target individuals in the current generation G.

$$\boldsymbol{w}_n = \frac{\mathrm{abs}\left(f\left(\boldsymbol{u}_{n,G}\right) - f\left(\boldsymbol{x}_{n,G}\right)\right)}{\sum_{m=1}^{|\boldsymbol{S}_{CR}|} \mathrm{abs}\left(f\left(\boldsymbol{u}_{m,G}\right) - f\left(\boldsymbol{x}_{m,G}\right)\right)} \tag{5.9}$$

Since both arrays $\boldsymbol{S}_F$ and $\boldsymbol{S}_{CR}$ have the same size, it is arbitrary which size will be used for the upper boundary for m in (5.9).
Once again, for easier understanding, the pseudo–code of the SHADE algorithm is depicted in Appendix B.

5.6 Linear decrease of the population size

A linear decrease of the population size was introduced to SHADE in [34] to improve its performance. The basic idea is to reduce the population size to promote exploitation in later phases of the evolution. Therefore, a new formula to estimate the population size was formed (5.10) and is calculated after each generation. Whenever the new population size NP_{new} is smaller than current population size NP, the population is sorted according to the objective function value and the worst $NP - NP_{new}$ individuals are discarded. Also, the size of an external archive is reduced to the NP.

$$NP_{new} = \mathrm{round}\left(NP_{init} - \frac{FES}{MAXFES} \cdot \left(NP_{init} - NP_f\right)\right) \tag{5.10}$$

The NP_{init} value is the initial population size, and NP_f is the final population size. FES and $MAXFES$ are current objective function number evaluations and the maximum number of objective function evaluations respectively. The pseudo–code of the L–SHADE algorithm is depicted in Appendix C.

6 PROPOSED METHODS

Previous sections were dedicated to the specification of the author's selected scientific area. This section provides a detailed description of proposed analysis tools and adaptive frameworks for adaptive DE–based algorithms.

6.1 Population dynamic analysis

The main disadvantage of modern adaptive DE algorithms lies in their susceptibility to fast convergence towards local optima. In such a case, the algorithm loses its ability to explore the search space and aims only at the exploitation of the currently most promising area. On the other hand, algorithms that mainly explore are deemed to fail on complex and rugged objective function landscapes. Therefore, researcher's frequent goal is to find the optimal balance between their algorithm's exploration and exploitation abilities. The author believes that the exploration/exploitation abilities can be analyzed through studying the population dynamic over generations and thus, a new tool for that purpose was developed.
In order to study the speed of population convergence towards the same point in the search space (part of exploitation), the clustering of the population members was proposed. This technique is based on the Density Based Spatial Clustering of Applications with Noise algorithm (DBSCAN) and is further described in section 6.1.1 [63]. For the purpose of studying the population exploration abilities, the population diversity metric can be used and is described in section 6.1.2 [64].

6.1.1 Cluster analysis

Datamining technique – the DBSCAN algorithm [63] was selected for the population cluster analysis because it conveniently works with inner cluster density rather than a cluster center. This allows DBSCAN to discover clusters of arbitrary shapes, which is beneficial for discovering clusters of population members

on complex objective function landscapes.
The complete description of the DBSCAN algorithm is available in the original paper [63], this section provides only a brief summary.

Glossary:

1. ***Eps*–neighborhood of a point** – This denotes an area of the parameter space, which surrounds point p to the maximum specified distance by *Eps* parameter.

2. **Minimal number of points to form a cluster *MinPts*** – Parameter, which defines how many points are at least needed in the *Eps*–neighbourhood to form a cluster.

3. **Core points** – Points inside of the cluster. These points have at least *MinPts* in their *Eps*–neighborhood.

4. **Border points** – Points on the border of a cluster. These points do not have *MinPts* points in their *Eps*–neighborhood, but are in an *Eps*–neighborhood of at least one core point.

5. **Directly density–reachable points** – Point p is directly density–reachable if it is in the *Eps*–neighbourhood of point q and q is a core point.

6. **Density–reachable points** – Point p is density–reachable from a core point q if there is a chain of core points connecting them.

7. **Density–connected points** – Point p is density–connected to point q if there is a chain of core points connecting them.

8. **Cluster** – A set of points forms a cluster if all possible tuples are density–connected.

9. **Noise** – Noise points are points that do not belong to any cluster.

Algorithm:
The DBSCAN algorithm starts from an arbitrary point p from the set S. In the

metaheuristic optimizer case, set S is composed of individuals in the population, and their point representations are their parameter values. DBSCAN retrieves all density–reachable points from p, and if the size of this set is bigger than or equal to $MinPts$, then those points are labeled as a cluster. This is done for all unlabeled points in the set S. Points that do not belong to any cluster are labeled as noise.

For population clustering analysis, the setting of control parameters of the DBSCAN algorithm is recommended as follows:

1. Set of points S – Individuals in one population form a set of points for clustering analysis. Each point p is given by parameter values of an individual. Since the goal is to discover spatial clusters in the search space, an individual's objective function value is not considered.

2. Eps = 1% of the parameter space – e.g., for the CEC2015 benchmark set with bounds $\{-100, 100\}^D$, $Eps = 2$,

3. $MinPts = 4$ (minimal number of individuals for mutation for most common DE schemes),

4. Chebyshev distance [65] – if the distance between any corresponding parameters of two individuals is higher than 1% of the parameter space, they are not considered as being in the Eps–neighborhood, therefore, cannot be part of the same cluster.

Clustering analysis is used to evaluate algorithms transition from exploration (ideally no clusters) to exploitation phase (clusters occur). In order to evaluate that, the DBSCAN algorithm is run on each generation of the population during the optimization run and the number of clusters and the index of generation of their first occurrence is recorded. This gives a metric, which was titled Mean Cluster Occurrence (MCO) [66]. This metric represents the average index of generation in which clusters occurred over all algorithm runs on given optimized function.

An example of cluster occurrence is shown in Fig. 6.1, where it can be seen that the SHADE algorithm with distance based parameter adaptation (Db_SHADE) maintains the exploration phase longer and that clusters occur later than in the original SHADE algorithm. The figure shows mean cluster occurrence by the bold line with confidence interval pictured by the same color with lighter shade.

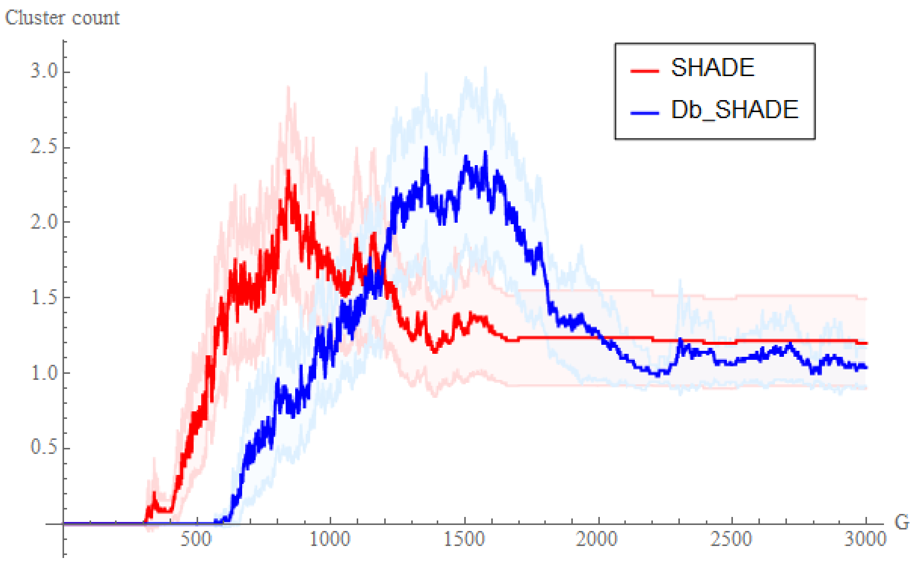

Fig. 6.1 Example of cluster occurrence comparison between SHADE and Db_SHADE algorithms on CEC 2015 benchmark, function 8, 30D.

6.1.2 Population diversity

In order to evaluate the population's exploration ability, a population diversity metric can be used. The combination of population diversity with cluster occurrence can give a clear picture of the state in which the population resides in each generation. Also, these two metrics are connected - when there is no cluster, the population diversity has to be higher than when there is one cluster of all population members. When comparing DE–based algorithms, higher population diversity in the time of first cluster occurrence suggests that the algorithm can still escape the local optima and explore the search space further. This is also

true for many other metaheuristics, not only for the DE.
An useful population diversity (PD) metric was proposed in [64]. This metric is based on the square root of the sum of deviations (6.2) of an individual's components from their corresponding means (6.1).

$$\overline{\boldsymbol{x}_j} = \frac{1}{NP} \sum_{i=1}^{NP} \boldsymbol{x}_{j,i} \tag{6.1}$$

$$PD = \sqrt{\frac{1}{NP} \sum_{i=1}^{NP} \sum_{j=1}^{D} (\boldsymbol{x}_{j,i} - \overline{\boldsymbol{x}_j})^2} \tag{6.2}$$

Where i is the population member iterator and j is the component (dimension) iterator.
Mean Population Diversity (MPD) [66] is a proposed metric for computing the average population diversity over multiple optimization runs in the moment of the first cluster occurrence. This metric should reflect the population's potential to explore the search space after its part exploits a locally promising area.

6.2 Multi–chaotic framework for parent selection

The Multi–Chaotic (MC) framework is based on the idea of using chaotic maps as PRNGs [67]; these generators are used for parent selection with a probability based on their success in generating better offspring in previous generations. The next subsections describe the whole framework and its use along with experimental results.

6.2.1 Chaotic maps as PRNGs

The chaotic maps are systems generated from a single initial position by simple equations. Current coordinates of the system are generated from the previous ones, consequently creating a system extremely dependent on the initial position. The generated chaotic sequence varies for different initial positions. Therefore, the generation of the initial position is randomized to obtain unique chaotic sequences. PRNG with uniform distribution is used for its generating. Equations used for generating the chaotic sequence also incorporate control parameters, which can be used to change the chaotic behavior. Chaotic systems implemented in this framework, with their generating equations, control parameter values, and initial position generator settings based on [68], are depicted in Tab. 6.1.

Tab. 6.1 Chaotic maps, generating equations, control parameters and initial position ranges.

Chaotic maps	**Equations**	**Parameters**	**Initial position**
Burgers	$X_{n+1} = aX_n - Y_n^2$ $Y_{n+1} = bY_n + X_nY_n$	$a = 0.75$ $b = 1.75$	$X_0 = U[-0.1, -0.01]$ $Y_0 = U[0.01, 0.1]$
Delayed Logistic	$X_{n+1} = AX_n(1 - Y_n)$ $Y_{n+1} = X_n$	$A = 2.27$	$X_0 = Y_0 = U[0.8, 0.9]$
Dissipative	$X_{n+1} = X_n + Y_{n+1}\,(\mathrm{mod}2\pi)$ $Y_{n+1} = bY_n + k\sin X_nY_n\,(\mathrm{mod}2\pi)$	$b = 0.1$ $k = 8.$	$X_0 = Y_0 = U[0, 0.1]$
Lozi	$X_{n+1} = 1 - a\lvert X_n\rvert - bY_n$ $Y_{n+1} = X_n$	$a = 1.7$ $b = 0.5$	$X_0 = Y_0 = U[0, 0.1]$
Tinkerbell	$X_{n+1} = X_n + Y_n + aX_n + bY_n$ $Y_{n+1} = 2X_nY_n + cX_n + dY_n$	$a = 0.9$ $b = -0.6$ $c = 2$ $d = 0.5$	$X_0 = U[-0.1, -0.01]$ $Y_0 = U[0, 0.1]$

In order to use these maps as PRNGs, the transformation rule has to be developed. The process of obtaining the i-th random integer value $rndInt_i$ from the chaotic map is presented in (6.3).

$$rndInt_i = \mathrm{round}\left(\frac{\mathrm{abs}\,(X_i)}{\max\,(\mathrm{abs}\,(X_{i\in N}))} \cdot (maxRndInt - 1)\right) + 1 \qquad (6.3)$$

Where abs(X_i) is the absolute value of the i-th generated X coordinate from the chaotic sequence of length N, max(abs(X)$i \in N$)) is a maximum value of all absolute values of generated X coordinates in chaotic sequence. The function round() is a common rounding function, and *maxRndInt* is a constant to ensure that integers will be generated in the range [1, *maxRndInt*].

Each of the chaotic map based PRNGs has different probability distribution and unique sequencing. This may be beneficial for the parent selection process. The obtained parent vector combinations exhibit a different dynamic than that of parent vector combinations selected by a PRNG with uniform distribution.

6.2.2 Parent selection

MC framework for the parent selection process is based on the ranking selection of chaotic map based PRNGs. A list of chaotic PRNGs *Clist* has to be added to the algorithm and each chaotic PRNG is initialized with the same probability $pc_{init} = 1/Csize$, where *Csize* is the size of *Clist*. For example, for five chaotic PRNGs $Csize = 5$ and each of them will have the probability of selection $pc_{init} = 1/5 = 0.2 = 20\%$.
For each target vector $\boldsymbol{x}_{i,G}$ in generation G, the chaotic generator $PRNG_k$ is selected from the *Clist* according to its probability $\boldsymbol{pc}_k$, where k is the index of selected chaotic PRNG. This selected generator is then used to replace standard PRNG for the selection of parent vectors, and if the generated trial vector succeeds in the selection, the probabilities are adjusted. There is an upper boundary for the probability of selection $pc_{max} = 0.6 = 60\%$; if the selected chaotic PRNG reaches this probability, then no adjustment takes place. The whole process is depicted in (6.4).

$$
\begin{array}{ll}
\text{if } f\left(\boldsymbol{u}_{i,G}\right) \leq f\left(\boldsymbol{x}_{i,G}\right) \text{ and } \boldsymbol{pc}_k < pc_{max} & \boldsymbol{pc}_j = \begin{cases} \frac{\boldsymbol{pc}_j + 0.01}{1.01} & \text{if } j = k \\ \\ \frac{\boldsymbol{pc}_j}{1.01} & \text{otherwise} \end{cases} \\
\text{otherwise} & \boldsymbol{pc}_j = \boldsymbol{pc}_j
\end{array}
\tag{6.4}
$$

6.2.3 Results

This section provides basic statistics of the 51 independent runs of SHADE and MC–SHADE algorithms. Both algorithms use the same setting, and the evaluation is done according to the CEC 2015 benchmark set [53].

Tab. 6.2 shows median and mean values obtained by SHADE and MC–SHADE

Tab. 6.2 CEC 2015 benchmark set results of
SHADE and MC–SHADE algorithms in 10D.

	SHADE		**MC–SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
3	2,01E+01	1,85E+01	2,01E+01	1,73E+01	=
4	3,07E+00	2,78E+00	2,36E+00	2,44E+00	+
5	3,19E+01	4,51E+01	3,53E+01	4,85E+01	=
6	6,80E−01	6,04E+00	4,18E−01	5,53E+00	=
7	1,78E−01	2,09E−01	1,66E−01	1,96E−01	=
8	4,78E−01	4,73E−01	2,52E−01	2,77E−01	=
9	1,00E+02	1,00E+02	1,00E+02	1,00E+02	+
10	2,17E+02	2,17E+02	2,17E+02	2,18E+02	=
11	3,34E+00	1,19E+02	3,15E+00	1,31E+02	=
12	1,01E+02	1,01E+02	1,01E+02	1,01E+02	=
13	2,79E+01	2,74E+01	2,77E+01	2,74E+01	=
14	2,94E+03	4,27E+03	2,94E+03	3,86E+03	=
15	1,00E+02	1,00E+02	1,00E+02	1,00E+02	=

algorithms. The last column of the table shows the Wilcoxon rank–sum test result, with the significance level set to 5%. Whenever the MC–SHADE algorithm outperforms the SHADE algorithm, "+" is used, when there is a tie, "=" is used, and if the SHADE algorithm performed better, there would be "–".
It can be seen that the MC–SHADE algorithm significantly outperformed the SHADE algorithm on 2 test functions. Selected convergence graphs are shown in Fig. 6.2 and Fig. 6.3. However, even when the difference between the results is not statistically significant, the convergence graphs can show that one algorithm can reach lower values, as shown in Fig. 6.2.

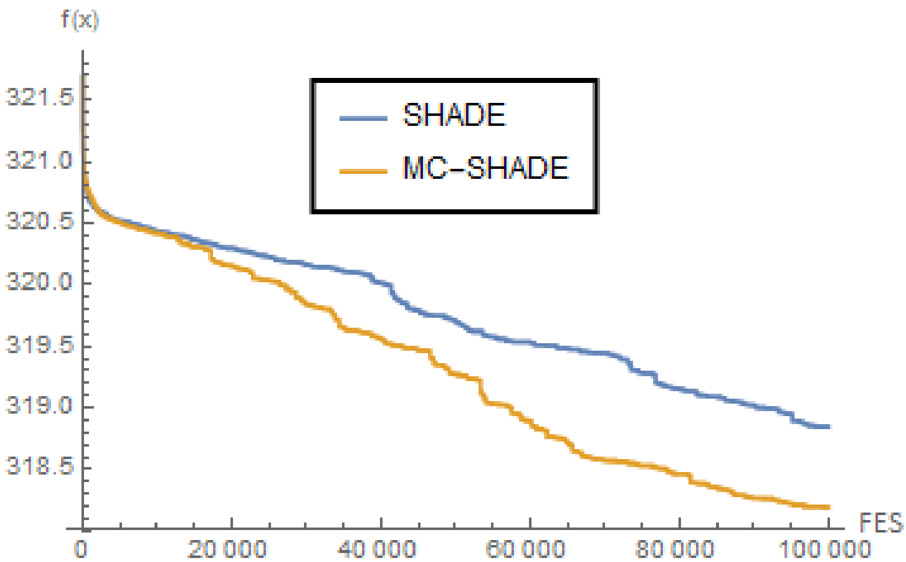

Fig. 6.2 Average convergence of SHADE and MC–SHADE algorithms on CEC 2015 benchmark, function 3, 10D.

The MC–SHADE algorithm was sent for the CEC 2016 competition and ranked 5^{th} out of 9 contestants – Tab. 6.3. The biggest strength of the algorithm was solving optimization problems in higher dimensions – 50D and 100D. Results of the MC–SHADE algorithm on CEC 2016 benchmark set are listed in Appendix E in tables E.1 – E.4.

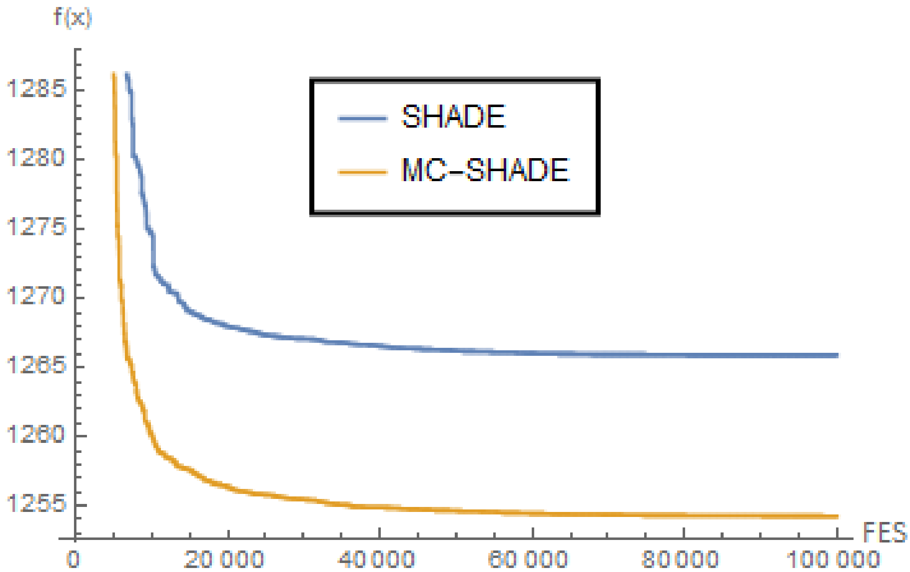

Fig. 6.3 Average convergence of SHADE and MC–SHADE algorithms on CEC 2015 benchmark, function 9, 10D.

Tab. 6.3 CEC 2016 competition ranking.

Algorithm	D = **10**	D = **30**	D = **50**	D = **100**	**Score**	**Rank**
LSHADE_EpSin [40]	1.51E+03	3.18E+03	5.88E+03	3.33E+04	4.38E+04	1
UMOEAII [57]	1.44E+03	4.38E+03	1.59E+04	2.96E+04	5.14E+04	2
SSEABC [69]	2.11E+03	7.68E+03	1.91E+04	3.06E+04	5.96E+04	3
iL–SHADE [43]	1.98E+03	5.32E+03	1.80E+04	2.23E+05	2.49E+05	4
MC–SHADE [41]	1.96E+03	1.06E+04	4.55E+04	1.96E+05	2.54E+05	**5**
AEPDJADE [70]	2.17E+03	8.36E+03	4.42E+04	2.77E+05	3.32E+05	6
LSHADE44 [42]	1.91E+03	5.97E+03	2.20E+04	3.76E+05	4.06E+05	7
SHADE4 [71]	1.83E+03	1.77E+04	1.65E+05	7.79E+05	9.64E+05	8
SPMGTLO [72]	8.64E+04	2.28E+06	3.87E+07	1.10E+08	1.51E+08	9

6.3 Distance based parameter adaptation

The distance based (Db) parameter adaptation was developed for SHADE–based [31] algorithms to overcome their problem with premature convergence to local optima. The original adaptation mechanism for scaling factor F and crossover rate CR values uses weighted forms of means (5.6) and (5.7), where weights are based on the improvement in objective function value (5.9). Such an approach promotes exploitation over exploration, and therefore, leads to premature convergence. This is a problem, especially when solving problems of higher dimensionality. The Db approach is based on the Euclidean distance between the trial and the target individual. Scaling factor F and crossover rate CR values connected with the individual that moved the furthest will have the highest weight (6.5). This approach slightly increases the algorithm's complexity by replacing the scalar subtraction for Euclidean distance computation.

$$\boldsymbol{w}_n = \frac{\sqrt{\sum_{j=1}^{D} \left(\boldsymbol{u}_{n,j,G} - \boldsymbol{x}_{n,j,G}\right)^2}}{\sum_{m=1}^{|\boldsymbol{S}_{CR}|} \sqrt{\sum_{j=1}^{D} \left(\boldsymbol{u}_{m,j,G} - \boldsymbol{x}_{m,j,G}\right)^2}} \tag{6.5}$$

The exploration ability is rewarded, leading to avoidance of premature convergence in higher dimensional objective spaces. Such an approach might also be useful for constrained problems, where constrained areas could be overcome by individual's increased movement in the search space.

6.3.1 Results

The proposed Db adaptation was implemented into SHADE [31] and L–SHADE [34] algorithms, and the resulting algorithm variants were named Db_SHADE and DbL_SHADE respectively. All versions (SHADE, Db_SHADE, L–SHADE and DbL_SHADE) were run on the CEC 2015 benchmark set, and their performance was assessed in accordance with the benchmark rules. The basic descriptive statistics and the results of the Wilcoxon rank–sum test in the same format as in section 6.2.3 are provided in Appendix F in the following tables F.1 – F.8.

As can be seen in tables F.1 and F.2, the Db adaptation is comparable to the original adaptation on $10D$ problems. There is a statistically significant difference in performance only in two cases – functions 6 and 12 (L–SHADE), resulting in the final score in Db wins/ties/loses of 0/15/0 (SHADE) and 1/13/1 (L–SHADE). Different results are visible on higher dimensional problems in tables F.3 – F.8. There, the Db adaptation outperforms the original versions with scored reported in Tab. 6.4.

Tab. 6.4 Wilcoxon rank-sum results in a form of *wins/ties/loses* from the perspective of Db adaptation enhanced algorithm - CEC 2015.

D	SHADE	L-SHADE
10	0/15/0	1/13/1
30	5/10/0	5/9/1
50	6/7/2	9/5/1
100	5/9/1	5/8/2
sum	16/41/3	20/35/5

The results show that the Db adaptation is beneficial for SHADE [31] and L–SHADE [34] algorithms when solving problems of higher dimensionality. Also, function 1 in the dataset is a unimodal rotated high conditioned elliptic function [53], and the DbL_SHADE algorithm does not perform well in 30, 50 and $100D$ on that problem. This is due to a slower convergence in comparison with L–SHADE with the original parameter adaptation. Understandably, the greedy approach is more suitable for unimodal functions since there is no need for exploration of the search space. The global optima of such functions can be found by simply following the gradient from each point in the search space.

6.3.2 Clustering analysis

The influence of the distance based parameter adaptation on the exploration and exploitation abilities of the algorithm was also tested on CEC 2015 benchmark set. A clustering and population diversity analysis described in section 6.1 were

performed and the results are available in Appendix G in tables G.1 – G.4 for SHADE and in tables G.5 – G.8 for L–SHADE.
It can be seen that the population's clustering occurs later for the algorithm variants with distance based parameter adaptation. This is mainly true for higher dimensional settings (30, 50, and 100D). Also, when numbers of cluster occurrence instances differ between Db and non-Db version, the Db version usually clusters in fewer cases. As for the population diversity, the characteristic feature is that when clusters occur in the population, diversity is similar regardless of the used adaptation scheme. It is important to note that in Db versions, clustering occurs later. Therefore, the population can explore the search space for a longer time, which proves to be beneficial for multimodal and complex objective function landscapes.

6.4 DISH

In the original version of the DE, there were three user–defined control parameters – population size NP, scaling factor F and crossover rate CR. These parameters' values are usually adapted during the optimization in the modern versions of the DE, and DISH is no exception. However, the mutation operator and adaptation mechanisms evolved via several successful algorithms. The evolution line from DE to DISH is described in the following steps:

1. DE from 1995 by Storn and Price [12].

2. JADE from 2009 [23] – algorithm created by Zhang and Sanderson proposed a novel mutation strategy – "current–to–$p$best/1" with an optional archive of inferior solutions.

3. SHADE from 2013 by Tanabe and Fukunaga [31] – built on the JADE algorithm with added memories for historically successful F and CR values and new adaptation mechanism for these parameters. This algorithm placed 3$^{\text{rd}}$ in the CEC 2013 competition.

4. The linear decrease of population size was introduced into SHADE and created L–SHADE algorithm [34], the winner of the CEC 2014 competition.

5. Improved L–SHADE algorithm titled iL–SHADE [43] was proposed for a CEC 2016 competition by Brest et al. This algorithm introduced changes to the historical memory update system and the initialization of the historical memories. It also proposed a new mechanism for treating F and CR parameters based on the ratio between current and maximum generation (phase of the optimization). This algorithm placed 4^{th} in the CEC 2016 competition.

6. Distance based parameter adaptation was proposed for SHADE based algorithms by Viktorin et al. in 2017 [73]. This novel adaptation mechanism based on the distance between solutions instead of the difference between objective function value was presented on SHADE and L–SHADE algorithms and shown its superiority over the original.

7. jSO algorithm was proposed by Brest et al. in 2017 [45]. The algorithm uses a novel "current–to–$p$best–w/1" mutation strategy and slightly changes fixed values for F and CR parameters. The jSO algorithm was 2^{nd} in the CEC 2017 bound constrained competition.

8. DISH algorithm was introduced in 2018 by Viktorin et al. and published in 2019 [47]. It incorporates the distance based parameter adaptation into the jSO algorithm to improve its performance.

The following subsections provide the details of DISH algorithm mechanisms. As a result of the above mentioned evolution some operations (e.g., selection) are equal or similar to those mentioned in the previous sections of this work, but for the sake of readability, those operations are also described in this section.

6.4.1 Initialization

The initial population $\boldsymbol{P}$ of solutions to the optimized problem is generated randomly. The size of the population is determined by the user via NP_{init} parameter (initial population size). Each individual solution $\boldsymbol{x}$ is a vector of length D, which is a dimension of the problem, and each vector component is generated within its lower $\boldsymbol{lo}$ and upper $\boldsymbol{up}$ bounds by an uniform PRNG $U[lo, up]$ (6.6).

$$\boldsymbol{x}_{j,i} = U\left[\boldsymbol{lo}_j,\ \boldsymbol{up}_j\right] \text{ for } j = 1,\ \ldots,\ D; i = 1,\ \ldots,\ NP_{init} \tag{6.6}$$

Other parameters and variables that have to be set in the initialization phase are:

1. Final population size - NP_f.
2. Stopping criterion - a maximum number of objective function evaluations $MAXFES$ in the most common case.
3. p_{max} and p_{min} parameters for mutation operator. $p_{max} = 0.25$ and $p_{min} = p_{max}/2 = 0.125$
4. External archive $\boldsymbol{A}$ is initialized empty. $\boldsymbol{A} = \emptyset$
5. Historical memory size H. $H = 5$
6. Historical memories for scaling factor $\boldsymbol{M}_F$ (6.7) and crossover rate $\boldsymbol{M}_{CR}$ (6.8).
7. Update historical memory index k. $k = 1$.

$$\boldsymbol{M}_{F,i} = 0.5 \text{ for } i = 1, \ldots, H-1,\ \boldsymbol{M}_{F,H} = 0.9 \tag{6.7}$$

$$\boldsymbol{M}_{CR,i} = 0.8 \text{ for } i = 1, \ldots, H-1,\ \boldsymbol{M}_{CR,H} = 0.9 \tag{6.8}$$

The following steps – mutation, crossover, and selection are repeated for each individual solution in the generation G, and these generations are repeated until the stopping criterion is met.

6.4.2 Mutation

The mutation operator used in DISH is a jSO's "current–to–$p$best–w/1", which combines a greedy approach in the first difference and the explorative factor in the second difference (6.9).

$$\boldsymbol{v}_i = \boldsymbol{x}_i + F_{w,i} \left(\boldsymbol{x}_{pBest} - \boldsymbol{x}_i \right) + F_i \left(\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2} \right) \tag{6.9}$$

The $\boldsymbol{v}_i$ is the i–th mutated vector created from target vector $\boldsymbol{x}_i$, randomly selected one of the $100p\%$ best solutions in the population $\boldsymbol{x}_{pBest}$ where p is determined by (6.10), a random solution from the population $\boldsymbol{x}_{r1}$ and random solution from the union of the population and external archive $\boldsymbol{x}_{r2}$. It is also important to note that all individuals are mutually different - $\boldsymbol{x}_i \neq \boldsymbol{x}_{pBest} \neq \boldsymbol{x}_{r1} \neq \boldsymbol{x}_{r2}$. The differences are scaled by two scaling factor parameters, the scaling factor F_i (6.11), and the weighted scaling factor $F_{w,i}$ (6.13).

$$p = FES_{ratio} \cdot (p_{max} - p_{min}) + p_{min} \tag{6.10}$$

Where FES_{ratio} stands for the ratio between the current number of objective function evaluations FES and the maximum number of objective function evaluations $MAXFES$ ($FES_{ratio} = FES/MAXFES$). Therefore, parameter p increases linearly with objective function evaluations.

$$F_i = C\left[\boldsymbol{M}_{F,r}, 0.1\right] \tag{6.11}$$

The scaling factor value F_i is generated from Cauchy distribution with the location parameter $\boldsymbol{M}_{F,r}$ and scale parameter value of 0. The index r is randomly generated from the range [1, H]. If the generated value F_i is smaller or equal to 0, it is generated again and if it is higher than 1, it is set to 1. Also, the scaling factor F_i is influenced by the FES_{ratio} in order to truncate its value in the exploration phase of the algorithm run (6.12).

$$F_i = 0.7, \; FES_{ratio} < 0.6 \text{ and } F_i > 0.7 \tag{6.12}$$

$$F_{w,i} = \begin{cases} 0.7 \cdot F_i, & FES_{ratio} < 0.2 \\ 0.8 \cdot F_i, & FES_{ratio} < 0.4 \\ 1.2 \cdot F_i, & \text{otherwise} \end{cases} \tag{6.13}$$

The weighted scaling factor $F_{w,i}$ is based on the optimization phase given by the FES_{ratio}. The next step after the mutation is the crossover.

6.4.3 Crossover

The crossover operator in DISH algorithm is binomial and based on the crossover rate value CR_i generated from the normal distribution (6.14) with a mean parameter value $\boldsymbol{M}_{CR,r}$ selected from the crossover rate historical memory and standard deviation value of 0.1.

$$CR_i = N\left[\boldsymbol{M}_{CR,r}, 0.1\right] \tag{6.14}$$

The CR_i value is also bounded between 0 and 1 and whenever it is generated outside these bounds, it is truncated to the nearest bound. The crossover rate value is also a subject to the optimization phase given by FES_{ratio} (6.15).

$$CR_i = \begin{cases} \max(CR_i, 0.7), & FES_{ratio} < 0.25 \\ \max(CR_i, 0.6), & FES_{ratio} < 0.5 \\ CR_i, & \text{otherwise} \end{cases} \tag{6.15}$$

Finally, the binomial crossover is depicted in (6.16).

$$\boldsymbol{u}_{j,i} = \begin{cases} \boldsymbol{v}_{j,i} & \text{if } U\left[0,1\right] \leq CR_i \text{ or } j = j_{rand} \\ \boldsymbol{x}_{j,i} & \text{otherwise} \end{cases} \tag{6.16}$$

Where $\boldsymbol{u}_i$ is called a trial vector and j_{rand} is an index of one component that has to be taken from the mutated vector $\boldsymbol{v}_i$. The j_{rand} index ensures that at least one vector component of the target vector $\boldsymbol{x}_i$ will be replaced. Thus in the following selection step, the tested trial vector will provide new information.

6.4.4 Selection

In the selection step, a quality of the trial solution vector $\boldsymbol{u}_i$ is compared to the quality of the original solution vector $\boldsymbol{x}_i$. The quality is given by the objective function value of these solutions. And since the selection operator is elitist, the trial solution has to have at least equal objective function value as the target solution to proceed into the next generation $G+1$ (6.17).

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G} & \text{if } f\left(\boldsymbol{u}_{i,G}\right) \leq f\left(\boldsymbol{x}_{i,G}\right) \\ \boldsymbol{x}_{i,G} & \text{otherwise} \end{cases} \tag{6.17}$$

Where $f()$ depicts the objective function value, and in this case, the objective is to minimize it.

The mutation, crossover, and selection operators are repeated for each individual solution in the population, and after the population is exhausted, the algorithm proceeds to the next generation. However, before processing each individual solution of the next generation, two essential mechanisms are incorporated into the algorithm – a linear decrease of the population size and the update of historical memories. These two mechanisms are described in the following subsections.

6.4.5 Linear decrease of the population size

The population size is decreased during the algorithm run to provide more time for exploration in the later phase of optimization. Thus, the smaller population of individual solutions will have more time to exploit the promising areas of the objective function landscape.

The mechanism used in the DISH algorithm is a linear decrease of population size, which uses the information of current objective function evaluations to shrink the population of solutions. A new population size NP_{new} is calculated as follows (6.18).

$$NP_{new} = \text{round}\left(NP_{init} - FES_{ratio} \cdot \left(NP_{init} - NP_f\right)\right) \tag{6.18}$$

The size of an external archive $\boldsymbol{A}$ is connected to the size of the population, and therefore, after decreasing the population size, the archive size is reduced as well. Whereas when decreasing the population size, the worst individual solutions are discarded from the population, in the archive, solutions to discard are selected randomly.

6.4.6 Update of historical memories

Historical memories $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ store historically successful values of scaling factors F and crossover rates CR that were helpful in the production of better trial individual solutions. Therefore, these memories have to be updated during the optimization in order to store recently used values. After each generation, one cell of both memories is updated, and for that, the algorithm uses index k to remember, which cell will be updated. The index is initialized to 1, and therefore, after the first generation, the first memory cell will be updated. The index is increased by one after each update, and when it overflows the memory size H, it starts from 1 again. There is one exception to the update, the last cell of both memories is never updated and still contains values 0.9 for both control parameters.

What will be stored in the k–th cell after the generation G is computed by a weighted Lehmer mean (6.19) of the corresponding generation control parameter arrays $\boldsymbol{S}_F$ and $\boldsymbol{S}_{CR}$. These arrays are filled during the generation by the values of control parameters when the trial solution succeeds in the selection step.

$$\text{mean}_{WL}\left(\boldsymbol{S}\right) = \frac{\sum_{n=1}^{|\boldsymbol{S}|} \boldsymbol{w}_n \cdot \boldsymbol{S}_n^2}{\sum_{n=1}^{|\boldsymbol{S}|} \boldsymbol{w}_n \cdot \boldsymbol{S}_n} \tag{6.19}$$

The $\text{mean}_{WL}()$ stands for weighted Lehmer mean and the computation is equal for both $\boldsymbol{S}_F$ and $\boldsymbol{S}_{CR}$, therefore, there is no subscript for $\boldsymbol{S}$ in the equation. The k–th memory cells of $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ are then updated according to (6.20) and (6.21).

$$\boldsymbol{M}_{F,k} = \begin{cases} \text{mean}_{WL}\left(\boldsymbol{S}_F\right) & \text{if } \boldsymbol{S}_F \neq \emptyset \text{ and } k \neq H \\ \boldsymbol{M}_{F,k} & \text{otherwise} \end{cases} \tag{6.20}$$

$$\boldsymbol{M}_{CR,k} = \begin{cases} \text{mean}_{WL}\left(\boldsymbol{S}_{CR}\right) & \text{if } \boldsymbol{S}_{CR} \neq \emptyset \text{ and } k \neq H \\ \boldsymbol{M}_{CR,k} & \text{otherwise} \end{cases} \tag{6.21}$$

The weights for the weighted Lehmer means (6.19) are in the case of the DISH algorithm computed as depicted in (6.22). This weighting was introduced as the distance based parameter adaptation [4]. It is titled like that, because in the original SHADE, L–SHADE, iL–SHADE and jSO algorithms, the weights were based on the difference between objective function values of trial individual $\boldsymbol{u}_i$ and its corresponding target individual $\boldsymbol{x}_i$, whereas in DISH, the weight is computed from the Euclidean distance between those two - $\boldsymbol{u}_i$ and $\boldsymbol{x}_i$.

$$\boldsymbol{w}_n = \frac{\sqrt{\sum_{j=1}^{D}\left(\boldsymbol{u}_{n,j,G} - \boldsymbol{x}_{n,j,G}\right)^2}}{\sum_{m=1}^{|\boldsymbol{S}_{CR}|}\sqrt{\sum_{j=1}^{D}\left(\boldsymbol{u}_{m,j,G} - \boldsymbol{x}_{m,j,G}\right)^2}} \tag{6.22}$$

This approach promotes exploration and tries to avoid the premature convergence of the algorithm into local optima.
Pseudo–code for the DISH algorithm is available in Appendix D.

6.4.7 Results

Median and mean obtained values on the CEC 2015 benchmark for the comparison between jSO and DISH are listed in Appendix H in tables H.1, H.2, H.3 and H.4. The algorithm was also tested on CEC 2017 benchmark. Results are available in Appendix H in tables H.5, H.6, H.7 and H.8. A quick summary of the results in the same format (wins/ties/losses according to Wilcoxon rank–sum test) as in section 6.3.1 is presented in Table 6.5.

Tab. 6.5 Wilcoxon rank-sum results in a form of *wins/ties/loses* from the perspective of DISH - CEC 2015 and CEC 2017.

D	CEC 2015	CEC 2017
10	0/15/0	0/29/1
30	3/12/0	3/26/1
50	7/8/0	14/16/0
100	6/7/2	19/10/1
sum	16/42/2	36/81/3

Once again, it is perceivable from the results that distance based parameter adaptation is beneficial for higher dimensional problems and the algorithm variant implementing it (DISH) is able to outperform the original algorithm without it (jSO).

The results of clustering and population diversity analysis on the CEC 2015 benchmark set are provided in Appendix I in the following tables I.1, I.2, I.3 and I.4. As it was stated in the previous chapter 6.3, the mean cluster occurrence for the algorithm variant with distance based parameter adaptation (DISH in this case) is mostly higher, therefore clusters emerge later in the optimization phase and the mean population diversity is similar during that time. This supports the initial idea of prolonging the exploration phase of the algorithm.

In order to present the algorithm to the scientific community, DISH algorithm was submitted for the CEC 2019 competition – 100–Digit Challenge [50]. The results are presented in Tab. 6.6 [60]. There were two variants in the competition – DISHchain 1e+12 by Zamuda [49] and DISH by Viktorin et al. [47]. The difference between these versions was in the larger initial population and more computing resources in the case of DISHchain 1e+12. It was shown, that the DISH algorithm is capable of obtaining competitive results and ended on joined 1$^{\text{st}}$ (DISHchain 1e+12) and 7$^{\text{th}}$ place out of 18 contestants.

Tab. 6.6 CEC 2019 competition ranking. *results presented after the original deadline of the competition

Algorithm	**Total Score**	**Ranking**
jDE100 [48]	100.00	1
DISHchain 1e+12 [49]	97.12 *(100.00)	1
HyDE–DF [61]	93.00	2
SOMA T3A [62]	93.00	2
ESHADE–USM [74]	85.52	3
SOMA Pareto [75]	85.04	4
rCIPDE [76]	85.00	5
Co–Op [77]	84.56	6
DISH [47]	83.92	7
rjDE [78]	83.52	8
mL–SHADE [79]	78.20	9
GADE [80]	75.44	10
CMEAL [81]	73.44	11
HTPC [82]	73.36	12
UMDE–MS [83]	70.40	13
DLABC [84]	67.88	14
MiLSHADE–LSP [85]	60.72	15
ESP–SOMA [86]	51.92	16

7 THE CONTRIBUTION TO SCIENCE AND PRACTICE

This part focuses on the benefits of proposed techniques in both scientific and application fields. Author believes that one of the most important aspects of developing new heuristic optimization techniques is a proper analysis of their behaviour and identification of problem domains in which the algorithm performs adequately. Therefore, the population dynamic analysis might be a helpful tool for researchers, and can determine some of the key features of their evolutionary algorithms – described in the next section 7.1. The section 7.2 describes implementation possibilities of the proposed distance based parameter adaptation and section 7.3 is devoted to an example of practical application of DISH–based algorithm on the problem of sustainable waste–to–energy facility location.

7.1 Population dynamic analysis

As its name suggests, population dynamic analysis is a tool that helps to analyze and understand the collective behavior of the evolutionary algorithm during the optimization run. This is very important for the development of new ideas, mainly in the adaptive evolutionary computation field. Since most of the proposed algorithms in this area aim to balance exploration and exploitation abilities, population dynamic analysis is a great tool to unveil possible issues and confirm the intended influence of the proposed changes to the evolutionary algorithm.
The main advantages of using proposed clustering and population diversity analysis can be summarized as follows:

- **Optimization phase detection** – by combining the information from cluster and diversity analysis with additional information from the algorithm, the exploration, stall, and convergence phases can be detected.
- **Premature convergence detection** – forming of early clusters in the

population is a good pointer towards premature convergence.

- **Sub–optimal computational budget** – when clusters are not formed during the whole optimization run and the population is still evolving, it is a good sign of underestimated computational budget. On the other hand, when clusters are formed, and the population diversity remains the same, it suggests an overestimated computational budget because the algorithm is most likely not going to converge any further.
- **Population size advisor** – forming of clusters that leave out only a couple of individuals might call for a larger population size. However, forming of one large cluster suggests that there are more individuals in the same area than needed.
- **Population management tool** – when managing the population size during the optimization run, clustering information can be used to select potential candidates for removal and refine the area of new individual generation.

7.2 Distance based parameter adaptation

Distance based parameter adaptation mechanism can be implemented into various evolutionary algorithms that use a greedy approach for any weighted parameter adaptation. Thus, it may be a tool for achieving a longer exploration phase in environments suitable for it. However, it is important to consider the additional computational complexity when implementing distance based parameter adaptation. Especially the large-scale optimization and the use of Euclidean distance, may suffer from the curse of dimensionality [87].

7.3 Practical applications of DISH

DISH algorithm can be used for any optimization task with continuous parameters and can be considered a good choice for problems, which have over 30 optimized parameters since it works best with problems of higher dimensionality. One of the real–world applications is described in the next section.

7.3.1 Sustainable waste–to–energy facility location

The problem of waste–to–energy facility location with reduced energy sales and unutilized capacity of plants [88, 89, 90, 91, 92, 93, 94, 95] leads to a mixed–integer non–linear model, which can be solved quite efficiently for small and medium–sized instances by traditional commercial solvers. However, for larger instances, the computational complexity becomes an issue. Therefore, a modified version of DISH was proposed and tested in [96].

The case study described in the paper deals with finding a location for waste–to–energy facilities in the Czech Republic and the problem can be summarized as follows:

- **Existing waste–to–energy facilities** – 4 (locations and capacities: Praha – 310 kt, Brno – 240 kt, Liberec – 96 kt and Plzeň – 95 kt).
- **The capacity of Praha and Brno facilities can be extended** – one additional option for each facility (430 kt Praha and 360 kt Brno).
- **Limited number of potential new facility locations** – 36.
- **Each potential new facility has multiple capacity options** – ranging from 2 to 27.
- **Waste limitation** – facilities cannot process more waste than their capacity.
- **Waste producers** – 206 locations and each of them has to be processed in a facility.
- **Transportation cost** – based on the traveled distance and the amount of transported waste.
- **Gate fee** – based on the capacity of the facility.

- **Unutilized capacity penalization** – non–linear penalization for unused capacity of a facility.

In order to use the DISH algorithm for solving mixed–integer non–linear problems, the algorithm had to be slightly adapted and led to the emergence of the Distance Random DISH (DR_DISH) algorithm. DR_DISH algorithm combines DISH with distance–based clustering–inspired allocation of waste producers to waste–to–energy facilities with a random sequence of producer processing and can be described in a three–step process [96]:

1. **Location** – DISH algorithm determines whether or not to build a facility in each potential location (dimension of the problem is based on the number of potential new facilities, and each optimized parameter is simplified to a binary decision 1 = build, 0 = do not build).
2. Repeat N–times
   (a) **Allocation** - Randomly iterate through producers and assign them to the nearest existing facility (determined in the first step). If the nearest facility does not have enough capacity (maximum capacity is lower than the sum of waste would be), the next nearest facility with adequate capacity is selected.
   (b) **Capacities** - for each waste–to–energy facility a closest larger capacity than the sum of its waste is selected.
   (c) Evaluation of the solution quality.
3. Out of N solutions, the best is selected and returned.

The DR_DISH algorithm was tested on 14 test cases dealing with instances of the problem from the smallest (only one considered region) to the largest (all 14 regions of the Czech Republic). The results are provided in Table 7.1, where the conventional solver DICOPT [97] is incorporated as a baseline. An example of the proposed solution by DR_DISH algorithm for the whole Czech Republic is

Tab. 7.1 DICOPT and DR_DISH solving the sustainable waste–to–energy facility location.

# of regions [–]	Cost [M€]		# of facilities [–]		Computational time [h:mm:ss]	
	DICOPT	DR_DISH	DICOPT	DR_DISH	DICOPT	DR_DISH
1	21.0	21.0	1	1	0:00:04	0:01:48
2	46.3	47.3	5	2	0:00:15	0:03:38
3	61.6	70.0	4	4	0:00:28	0:05:31
4	94.5	102.4	9	4	0:01:15	0:08:22
5	105.5	111.5	6	4	0:01:39	0:09:46
6	119.7	127.2	10	5	0:10:09	0:12:50
7	138.5	146.3	10	5	0:02:14	0:14:54
8	159.8	162.1	12	6	3:55:32	0:17:09
9	211.0	211.9	14	8	5:54:08	0:22:21
10	–	241.9	–	9	–	0:23:44
11	–	252.3	–	10	–	0:26:19
12	–	268	–	11	–	0:31:58
13	–	292.4	–	12	–	0:38:01
14	–	301.7	–	12	–	0:40:53

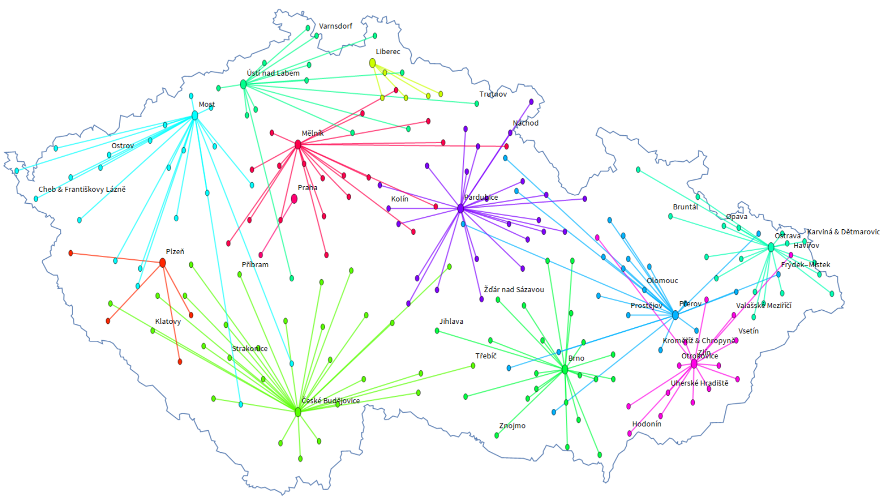

Fig. 7.1 DR_DISH solution for the sustainable waste–to–energy facility location - 14 regions.

shown in Fig. 7.1.
As can be seen in Table 7.1, DICOPT solver was able to provide better results up to 9 regions. However, the computational complexity became too high for 10 and more regions, and the commercial solver was unable to provide a feasible solution under given experimental conditions. It is also apparent that DR_DISH provides solutions that use a smaller number of waste–to–energy facilities with higher capacity. This is important for the possible real–world implementation of the solution since it is easier to guarantee a sufficient waste supply for larger facilities. Therefore, their economic sustainability is easier [96]. Moreover, the perception of waste–to–energy facilities amongst the general public is still bad, even though the currently used technologies are ensuring clean incineration. Thus, the solution provided by DR_DISH algorithm is more likely to be implemented in practice.
Several other approaches worth mentioning were considered during the design of the DR_DISH algorithm [96]:

- **Small incinerator** – locations of facilities determined by DISH, but each facility is initialized with the smallest available capacity. Producers are iterated over randomly and processed in the nearest facility with sufficient capacity. If none of the facilities has sufficient capacity, the nearest one that can be enlarged to accommodate the waste from the currently evaluated producer is enlarged. This approach seems to be more viable for smaller instances (1 – 4 regions with an average improvement of 5.84% in comparison with DR_DISH) but increases the cost for larger instances (5 – 14 regions with 3.47% average increase).

- **Cost oriented** – DISH algorithm solves the sequence of processing producers. Producers waste is transported to the most cost–effective facility in terms of transport cost and gate fees. This approach leads to a higher number of small facilities, but the solution's cost is increased by 15.5% on average in comparison with DR_DISH.

- **Heuristic sequence** – DISH algorithm solves both – facility location and the sequence of processing producers. This leads to similar results as for

DR_DISH algorithm (only 0.22% cost increase), but the dimensionality of the problem quickly increases and thus is not suitable for larger instances.

- **Iterative deterministic approach** – This is an approach without heuristic. The algorithm starts with all potential facilities. Waste producers are allocated to their nearest facility with sufficient maximum capacity. Then the algorithm cyclically tries to remove each facility and reallocate its waste producers. This continues while the overall cost decreases. This approach is due to its deterministic nature faster but led to the average increase in the cost of 5.63%.

8 DISSERTATION GOAL FULFILLMENT

This section describes steps that were taken in order to fulfill the dissertation goal. This was to investigate current trends in adaptive DE design and utilize datamining techniques to improve the understanding of the population dynamic and possibly use this information to develop more robust and performance-wise better algorithm variants.

- **State–of–the–art review** – current trends and ideas in the field of adaptive DE were studied and analyzed for possible deficiencies in the algorithm design [98, 99, 100].
- **Proposal of novel adaptive DE variants** – based on the knowledge gained from the first step, the proposed methods highlight understanding of the population dynamic by addressing the problem of premature convergence and fast clustering of the population [41, 44].
- **Comparison with state of the art methods** – proposed methods were implemented into the state–of–the–art algorithms (SHADE, L–SHADE and jSO [73, 47]) and compared with their canonical forms on CEC benchmark sets. The proposed algorithms were also participating in CEC competitions in 2016 (5^{th} place) [52] and 2019 (joined 1^{st}, and 7^{th} place) [101].
- **Analysis** – an analysis of the results [66, 102] was executed to understand the benefits and drawbacks of the proposed methods. The next step in this research direction is to utilize the analysis findings for the development of refined adaptive methods and their implementation.

9 CONCLUSION

This chapter summarizes the results of the doctoral thesis. Methods proposed in this work can be used in both – research and practice.

Researchers might find clustering and population diversity analysis useful for better understanding of the collective behavior of their evolutionary algorithm's population. Thus, it can help with the development, implementation, and mainly evaluation of new ideas in the field of adaptive parameter control. Practitioners may use the analysis results to identify potential problems with incorrect computational budget or population size selection.

Following the findings of clustering and population diversity analysis of state-of-the-art DE algorithms, a distance based parameter adaptation was proposed. This led to the development of the DISH algorithm, which is a good choice for single–objective optimization problems in the continuous domain with a higher number of optimized parameters. The algorithm is reasonably easy to implement, and its implementation in Java is already available on Github [103]. It may also serve for researchers as a baseline for comparison of their proposed algorithms.

As for the distance based parameter adaptation scheme, it is possible to implement it into other population–based evolutionary algorithms to affect their balance between exploration and exploitation and help with the algorithm's performance aspect.

The author would like to utilize the knowledge gained in his doctoral studies and dedicate his future research time and capacity to the development of an analysis framework for continuous single–objective population–based optimization techniques. Such a framework should help analyze the behavior of an algorithm during the optimization phase and serve as a guide for the refinement of developed techniques.

The incremental development of new evolutionary algorithms is an ongoing process that gradually improves the quality of the heuristic optimization field [104]. Therefore, in author's opinion, this type of research should be encouraged, but with a great emphasis on good research practices.

REFERENCES

[1] Gropp, W. and Moré, J. Optimization Environments and the NEOS Server. Approximation Theory and Optimization, MD Buhmann and A. Iserles, eds, 1997.

[2] Cavazzuti, M. Deterministic Optimization. In *Optimization Methods.* : Springer Berlin Heidelberg, September 2012. pp. 77–102. doi: 10.1007/978-3-642-31187-1_4. Available at: <https://doi.org/10.1007/978-3-642-31187-1_4>.

[3] Darwin, C. *On the Origin of Species by Means of Natural Selection Or the Preservation of Favoured Races in the Struggle for Life.* H. Milford; Oxford University Press, 1859.

[4] Jesus, D. A. R. and Rivera, W. Application of Evolutionary Algorithms for Load Forecasting in Smart Grids. In *Proceedings of the International Conference on Foundations of Computer Science (FCS)*, pp. 40–44. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2018.

[5] Sathiya, P., Aravindan, S., Haq, A. N. and Paneerselvam, K. Optimization of friction welding parameters using evolutionary computational techniques. *Journal of Materials Processing Technology.* 2009, vol. 209, no. 5, pp. 2576 – 2584. ISSN 0924-0136. doi: https://doi.org/10.1016/j.jmatprotec.2008.06.030.

[6] Zamuda, A., Sosa, J. D. H. and Adler, L. Constrained differential evolution optimization for underwater glider path planning in sub-mesoscale eddy sampling. *Applied Soft Computing.* 2016, vol. 42, pp. 93 – 118. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2016.01.038.

[7] Gobeyn, S., Mouton, A. M., Cord, A. F., Kaim, A., Volk, M. and Goethals, P. L. Evolutionary algorithms for species distribution modelling: A review in the context of machine learning. *Ecological Modelling.* 2019, vol. 392, pp. 179 – 195. ISSN 0304-3800. doi: https://doi.org/10.1016/j.ecolmodel.2018.11.013.

[8] Sun, L., Lin, L., Li, H. and Gen, M. Large scale flexible scheduling optimization by a distributed evolutionary algorithm. *Computers & Industrial Engineering*. 2019, vol. 128, pp. 894 – 904. ISSN 0360-8352. doi: https://doi.org/10.1016/j.cie.2018.09.025.

[9] Yeguas-Bolivar, E., Muñoz-Salinas, R., Medina-Carnicer, R. and Carmona-Poyato, A. Comparing evolutionary algorithms and particle filters for Markerless Human Motion Capture. *Applied Soft Computing*. 2014, vol. 17, pp. 153 – 166. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2014.01.007.

[10] Devi, R. V., Sathya, S. S. and Coumar, M. S. Evolutionary algorithms for de novo drug design – A survey. *Applied Soft Computing*. 2015, vol. 27, pp. 543 – 552. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2014.09.042.

[11] Wolpert, D. H. and Macready, W. G. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*. 1997, vol. 1, no. 1, pp. 67–82.

[12] Price, K. and Storn, R. Differential Evolution-a simple and efficient adaptive scheme for global optimization over continuous space. Technical report, Technical Report, International Computer Science Institute, 1995.

[13] Sörensen, K. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*. 2015, vol. 22, no. 1, pp. 3–18. doi: https://doi.org/10.1111/itor.12001. Available at: <https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.12001>.

[14] Storn, R. and Price, K. Differential Evolution - A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*. 1997, vol. 11, no. 4, pp. 341–359. ISSN 1573-2916. doi: 10.1023/A:1008202821328. Available at: <https://doi.org/10.1023/A:1008202821328>.

[15] Gämperle, R., Müller, S. D. and Koumoutsakos, P. A parameter study for differential evolution. *Advances in intelligent systems, fuzzy systems, evolutionary computation.* 2002, vol. 10, no. 10, pp. 293–298.

[16] Liu, J. On setting the control parameter of the differential evolution method. In *Proceedings of the 8th international conference on soft computing (MENDEL 2002)*, pp. 11–18, 2002.

[17] Ronkkonen, J., Kukkonen, S. and Price, K. Real-Parameter Optimization with Differential Evolution. In *2005 IEEE Congress on Evolutionary Computation.* IEEE, 2005. doi: 10.1109/cec.2005.1554725. Available at: <https://doi.org/10.1109/cec.2005.1554725>.

[18] Liu, J. and Lampinen, J. A Fuzzy Adaptive Differential Evolution Algorithm. *Soft Computing.* 2005, vol. 9, no. 6, pp. 448–462. ISSN 1433-7479. doi: 10.1007/s00500-004-0363-x. Available at: <https://doi.org/10.1007/s00500-004-0363-x>.

[19] Omran, M. G. H., Salman, A. and Engelbrecht, A. P. Self-adaptive Differential Evolution. In *Computational Intelligence and Security.* : Springer Berlin Heidelberg, 2005. pp. 192–199. doi: 10.1007/11596448_28. Available at: <https://doi.org/10.1007/11596448_28>.

[20] Qin, A. and Suganthan, P. Self-adaptive Differential Evolution Algorithm for Numerical Optimization. In *2005 IEEE Congress on Evolutionary Computation.* IEEE, 2005. doi: 10.1109/cec.2005.1554904. Available at: <https://doi.org/10.1109/cec.2005.1554904>.

[21] Brest, J., Greiner, S., Boskovic, B., Mernik, M. and Zumer, V. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Transactions on Evolutionary Computation.* December 2006, vol. 10, no. 6, pp. 646–657. doi: 10.1109/tevc.2006.872133. Available at: <https://doi.org/10.1109/tevc.2006.872133>.

[22] Qin, A., Huang, V. and Suganthan, P. Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization.

IEEE Transactions on Evolutionary Computation. April 2009, vol. 13, no. 2, pp. 398–417. doi: 10.1109/tevc.2008.927706. Available at: <https://doi.org/10.1109/tevc.2008.927706>.

[23] Zhang, J. and Sanderson, A. C. JADE: adaptive differential evolution with optional external archive. *IEEE Transactions on evolutionary computation*. 2009, vol. 13, no. 5, pp. 945–958.

[24] Mallipeddi, R., Suganthan, P., Pan, Q. and Tasgetiren, M. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*. 2011, vol. 11, no. 2, pp. 1679 – 1696. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2010.04.024. The Impact of Soft Computing for the Progress of Artificial Intelligence.

[25] Wang, Y., Cai, Z. and Zhang, Q. Differential Evolution With Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Transactions on Evolutionary Computation*. February 2011, vol. 15, no. 1, pp. 55–66. doi: 10.1109/tevc.2010.2087271. Available at: <https://doi.org/10.1109/tevc.2010.2087271>.

[26] Gong, W., Fialho, Cai, Z. and Li, H. Adaptive strategy selection in differential evolution for numerical optimization: An empirical study. *Information Sciences*. 2011, vol. 181, no. 24, pp. 5364 – 5386. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2011.07.049.

[27] Elsayed, S. M., Sarker, R. A. and Essam, D. L. Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems. In *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, June 2011. doi: 10.1109/cec.2011.5949732. Available at: <https://doi.org/10.1109/cec.2011.5949732>.

[28] Tvrdik, J. and Polakova, R. Competitive differential evolution applied to CEC 2013 problems. In *2013 IEEE Congress on Evolutionary Computation*. IEEE, June 2013. doi: 10.1109/cec.2013.6557759. Available at: <https://doi.org/10.1109/cec.2013.6557759>.

[29] Zamuda, A., Brest, J. and Mezura-Montes, E. Structured Population Size Reduction Differential Evolution with Multiple Mutation Strategies on CEC 2013 real parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*. IEEE, June 2013. doi: 10.1109/cec.2013.6557794. Available at: <https://doi.org/10.1109/cec.2013.6557794>.

[30] Zhu, W., Tang, Y., Fang, J. and Zhang, W. Adaptive population tuning scheme for differential evolution. *Information Sciences*. 2013, vol. 223, pp. 164 – 191. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2012.09.019.

[31] Tanabe, R. and Fukunaga, A. Success-history based parameter adaptation for Differential Evolution. In *2013 IEEE Congress on Evolutionary Computation (CEC)*, pp. 71–78. IEEE, 2013.

[32] Yu, W.-J., Shen, M., Chen, W.-N., Zhan, Z.-H., Gong, Y.-J., Lin, Y., Liu, O. and Zhang, J. Differential Evolution With Two-Level Parameter Adaptation. *IEEE Transactions on Cybernetics*. July 2014, vol. 44, no. 7, pp. 1080–1099. doi: 10.1109/tcyb.2013.2279211. Available at: <https://doi.org/10.1109/tcyb.2013.2279211>.

[33] Falco, I. D., Cioppa, A. D., Maisto, D., Scafuri, U. and Tarantino, E. An adaptive invasion-based model for distributed Differential Evolution. *Information Sciences*. 2014, vol. 278, pp. 653 – 672. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2014.03.083.

[34] Tanabe, R. and Fukunaga, A. S. Improving the search performance of SHADE using linear population size reduction. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1658–1665, 2014.

[35] Wu, G., Mallipeddi, R., Suganthan, P., Wang, R. and Chen, H. Differential evolution with multi-population based ensemble of mutation strategies. *Information Sciences*. 2016, vol. 329, pp. 329 – 345. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2015.09.009. Special issue on Discovery Science.

[36] Mallipeddi, R. and Lee, M. An evolving surrogate model-based differential evolution algorithm. *Applied Soft Computing*. 2015, vol. 34, pp. 770 – 787. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2015.06.010.

[37] Díaz-Manríquez, A., Toscano-Pulido, G. and Gómez-Flores, W. On the selection of surrogate models in evolutionary optimization algorithms. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pp. 2155–2162, 2011. doi: 10.1109/CEC.2011.5949881.

[38] Draa, A., Bouzoubia, S. and Boukhalfa, I. A sinusoidal differential evolution algorithm for numerical optimisation. *Applied Soft Computing*. 2015, vol. 27, pp. 99 – 126. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2014.11.003.

[39] Guo, S.-M., Tsai, J. S.-H., Yang, C.-C. and Hsu, P.-H. A self-optimization approach for L-SHADE incorporated with eigenvector-based crossover and successful-parent-selecting framework on CEC 2015 benchmark set. In *2015 IEEE congress on evolutionary computation (CEC)*, pp. 1003–1010. IEEE, 2015.

[40] Awad, N. H., Ali, M. Z., Suganthan, P. N. and Reynolds, R. G. An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems. In *2016 IEEE congress on evolutionary computation (CEC)*, pp. 2958–2965. IEEE, 2016.

[41] Viktorin, A., Pluhacek, M. and Senkerik, R. Success-history based adaptive differential evolution algorithm with multi-chaotic framework for parent selection performance on CEC2014 benchmark set. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7744404. Available at: <https://doi.org/10.1109/cec.2016.7744404>.

[42] Polakova, R., Tvrdik, J. and Bujok, P. Evaluating the performance of L-SHADE with competing strategies on CEC2014 single parameter-operator test suite. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7743921. Available at: <https://doi.org/10.1109/cec.2016.7743921>.

[43] Brest, J., Maučec, M. S. and Bošković, B. iL-SHADE: Improved L-SHADE Algorithm for Single Objective Real-Parameter Optimization. In *2016 IEEE World Congress on Computational Intelligence (IEEE WCCI 2016), Vancouver, Canada*, pp. 1188–1195, 2016.

[44] Viktorin, A., Senkerik, R., Pluhacek, M., Kadavy, T. and Zamuda, A. Distance based parameter adaptation for differential evolution. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, November 2017. doi: 10.1109/ssci.2017.8280959. Available at: <https://doi.org/10.1109/ssci.2017.8280959>.

[45] Brest, J., Maučec, M. S. and Bošković, B. Single objective real-parameter optimization: algorithm jSO. In *2017 IEEE congress on evolutionary computation (CEC)*, pp. 1311–1318. IEEE, 2017.

[46] Stanovov, V., Akhmedova, S. and Semenkin, E. LSHADE Algorithm with Rank-Based Selective Pressure Strategy for Solving CEC 2017 Benchmark Problems. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2018. doi: 10.1109/cec.2018.8477977. Available at: <https://doi.org/10.1109/cec.2018.8477977>.

[47] Viktorin, A., Senkerik, R., Pluhacek, M., Kadavy, T. and Zamuda, A. Distance based parameter adaptation for success-history based differential evolution. *Swarm and Evolutionary Computation*. 2019, vol. 50, pp. 100462.

[48] Brest, J., Maučec, M. S. and Bošković, B. The 100-Digit Challenge: Algorithm jDE100. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 19–26, 2019. doi: 10.1109/CEC.2019.8789904.

[49] Zamuda, A. Function Evaluations Upto 1e+12 and Large Population Sizes Assessed in Distance-Based Success History Differential Evolution for 100-Digit Challenge and Numerical Optimization Scenarios (DISHchain 1e+12): A Competition Entry for "100-Digit Challenge, and Four Other Numerical Optimization Competitions" at the Genetic

and Evolutionary Computation Conference (CECCO) 2019. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, pp. 11–12, New York, NY, USA, 2019. Association for Computing Machinery. doi: 10.1145/3319619.3326751. Available at: <https://doi.org/10.1145/3319619.3326751>. ISBN 9781450367486.

[50] Price, K., Awad, N., Ali, M. and Suganthan, P. Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. In *Technical Report.* : Nanyang Technological University, 2018.

[51] Liang, J., Qu, B. and Suganthan, P. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.* 2013, vol. 635.

[52] Liang, J., Qu, B. and Suganthan, P. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.* 2013, vol. 635.

[53] Liang, J., Qu, B., Suganthan, P. and Chen, Q. Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization. *Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.* 2014, vol. 29, pp. 625–640.

[54] Awad, N., Ali, M. Z. and Reynolds, R. G. A differential evolution algorithm with success-based parameter adaptation for CEC2015 learning-based optimization. In *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, May 2015. doi: 10.1109/cec.2015.7257012. Available at: <https://doi.org/10.1109/cec.2015.7257012>.

[55] SALLAM, K. M., SARKER, R. A., ESSAM, D. L. and ELSAYED, S. M. Neurodynamic differential evolution algorithm and solving CEC2015 competition problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, May 2015. doi: 10.1109/cec.2015.7257003. Available at: <https://doi.org/10.1109/cec.2015.7257003>.

[56] RUEDA, J. L. and ERLICH, I. Testing MVMO on learning-based real-parameter single objective benchmark optimization problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, May 2015. doi: 10.1109/cec.2015.7257002. Available at: <https://doi.org/10.1109/cec.2015.7257002>.

[57] ELSAYED, S., HAMZA, N. and SARKER, R. Testing united multi-operator evolutionary algorithms-II on single objective optimization problems. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7744164. Available at: <https://doi.org/10.1109/cec.2016.7744164>.

[58] AWAD, N. H., ALI, M. Z. and SUGANTHAN, P. N. Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pp. 372–379. IEEE, 2017.

[59] MOHAMED, A. W., HADI, A. A., FATTOUH, A. M. and JAMBI, K. M. LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In *2017 IEEE Congress on evolutionary computation (CEC)*, pp. 145–152. IEEE, 2017.

[60] PRICE, K., AWAD, N., ALI, M. and SUGANTHAN, P. The 2019 100-Digit Challenge on Real-Parameter, Single Objective Optimization: Analysis of Results. Technical report, Technical Report 2019. Available online: https:// www.ntu.edu.sg /home /epnsugan /index_files /CEC2019 /CEC2019.htm, 2019.

[61] LEZAMA, F., SOARES, J. a., FAIA, R. and VALE, Z. Hybrid-Adaptive Differential Evolution with Decay Function (HyDE-DF) Applied to the 100-Digit Challenge Competition on Single Objective Numerical Optimization.

In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO ’19, pp. 7–8, New York, NY, USA, 2019. Association for Computing Machinery. doi: 10.1145/3319619.3326747. Available at: <https://doi.org/10.1145/3319619.3326747>. ISBN 9781450367486.

[62] Diep, Q. B., Zelinka, I., Das, S. and Senkerik, R. SOMA T3A for Solving the 100-Digit Challenge. In Zamuda, A., Das, S., Suganthan, P. N. and Panigrahi, B. K. (Ed.) *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, pp. 155–165, Cham, 2020. Springer International Publishing. ISBN 978-3-030-37838-7.

[63] Ester, M., Kriegel, H.-P., Sander, J., Xu, X. and others. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, vol. 96, pp. 226–231, 1996.

[64] Poláková, R., Tvrdík, J., Bujok, P. and Matoušek, R. Population-size adaptation through diversity-control mechanism for differential evolution. In *MENDEL, 22th International Conference on Soft Computing*, pp. 49–56, 2016.

[65] Deza, M. M. and Deza, E. Encyclopedia of Distances. In *Encyclopedia of Distances.* : Springer, 2009. pp. 1–583.

[66] Viktorin, A., Senkerik, R., Pluhacek, M. and Kadavy, T. Clustering Analysis of the Population in Db_SHADE Algorithm. *MENDEL*. Jun. 2018, vol. 24, no. 1, pp. 9–16. doi: 10.13164/mendel.2018.1.009. Available at: <https://mendel-journal.org/index.php/mendel/article/view/13>.

[67] PLUHACEK, M. *Modern methods of development and modification of evolutionary computational techniques*. Doctoral Thesis, Tiomas Bata University in Zlín, Faculty of Applied Informatics, Zlín, 2015.

[68] Sprott, J. C. and Sprott, J. C. *Chaos and time-series analysis*. Oxford University Press, 2003. ISBN 9780198508403.

[69] Yavuz, G., Aydin, D. and Stutzle, T. Self-adaptive search equation-based artificial bee colony algorithm on the CEC 2014 benchmark functions. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7743920. Available at: `<https://doi.org/10.1109/cec.2016.7743920>`.

[70] Yang, M., Guan, J. and Li, C. Differential evolution with auto-enhanced population diversity: The experiments on the CEC'2016 competition. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7744402. Available at: `<https://doi.org/10.1109/cec.2016.7744402>`.

[71] Bujok, P., Tvrdik, J. and Polakova, R. Evaluating the performance of SHADE with competing strategies on CEC 2014 single-parameter test suite. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 5002–5009. IEEE, July 2016. doi: 10.1109/cec.2016.7748322. Available at: `<https://doi.org/10.1109/cec.2016.7748322>`.

[72] Kommadath, R., Sivadurgaprasad, C. and Kotecha, P. Single phase multi-group teaching learning algorithm for computationally expensive numerical optimization (CEC 2016). In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016. doi: 10.1109/cec.2016.7744167. Available at: `<https://doi.org/10.1109/cec.2016.7744167>`.

[73] Viktorin, A., Senkerik, R., Pluhacek, M., Kadavy, T. and Zamuda, A. Distance based parameter adaptation for differential evolution. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–7. IEEE, 2017.

[74] Kumar, A., Misra, R. K., Singh, D. and Das, S. Testing A Multi-Operator based Differential Evolution Algorithm on the 100-Digit Challenge for Single Objective Numerical Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 34–40, 2019. doi: 10.1109/CEC.2019.8789907.

[75] Truong, T. C., Diep, Q. B., Zelinka, I. and Senkerik, R. Pareto-Based Self-organizing Migrating Algorithm Solving 100-Digit Challenge.

In ZAMUDA, A., DAS, S., SUGANTHAN, P. N. and PANIGRAHI, B. K. (Ed.) *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, pp. 13–20, Cham, 2020. Springer International Publishing. ISBN 978-3-030-37838-7.

[76] ZHANG, S. X., SHING CHAN, W., TANG, K. S. and YONG ZHENG, S. Restart based Collective Information Powered Differential Evolution for Solving the 100-Digit Challenge on Single Objective Numerical Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 14–18, 2019. doi: 10.1109/CEC.2019.8790279.

[77] ZHANG, G., SHI, Y. and STEED HUANG, J. Cooperative Optimization Algorithm for the 100-Digit Challenge. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 376–380, 2019. doi: 10.1109/CEC.2019.8790272.

[78] ALIĆ, A., BERKOVIČ, K., BOŠKOVIĆ, B. and BREST, J. Population Size in Differential Evolution. In ZAMUDA, A., DAS, S., SUGANTHAN, P. N. and PANIGRAHI, B. K. (Ed.) *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, pp. 21–30, Cham, 2020. Springer International Publishing. ISBN 978-3-030-37838-7.

[79] YEH, J., CHEN, T. and CHIANG, T. Modified L-SHADE for Single Objective Real-Parameter Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 381–386, 2019. doi: 10.1109/CEC.2019.8789991.

[80] EPSTEIN, A., ERGEZER, M., MARSHALL, I. and SHUE, W. GADE with Fitness-based Opposition and Tidal Mutation for Solving IEEE CEC2019 100-Digit Challenge. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 395–402, 2019. doi: 10.1109/CEC.2019.8790159.

[81] BUJOK, P. and ZAMUDA, A. Cooperative Model of Evolutionary Algorithms Applied to CEC 2019 Single Objective Numerical Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 366–371, 2019. doi: 10.1109/CEC.2019.8790317.

[82] Xu, P., Luo, W., Lin, X., Qiao, Y. and Zhu, T. Hybrid of PSO and CMA-ES for Global Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 27–33, 2019. doi: 10.1109/CEC.2019.8789912.

[83] Fu, Y. and Wang, H. A Univariate Marginal Distribution Resampling Differential Evolution Algorithm with Multi-Mutation Strategy. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1236–1242, 2019. doi: 10.1109/CEC.2019.8790358.

[84] Lu, J., Zhou, X., Ma, Y., Wang, M., Wan, J. and Wang, W. A Novel Artificial Bee Colony Algorithm with Division of Labor for Solving CEC 2019 100-Digit Challenge Benchmark Problems. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 387–394, 2019. doi: 10.1109/CEC.2019.8790252.

[85] Molina, D. and Herrera, F. Applying Memetic algorithm with Improved L-SHADE and Local Search Pool for the 100-digit challenge on Single Objective Numerical Optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 7–13, 2019. doi: 10.1109/CEC.2019.8789916.

[86] Kadavy, T., Pluhacek, M., Senkerik, R. and Viktorin, A. The Ensemble of Strategies and Perturbation Parameter in Self-organizing Migrating Algorithm Solving CEC 2019 100-Digit Challenge. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 372–375, 2019. doi: 10.1109/CEC.2019.8790012.

[87] Bellman, R. *Dynamic Programming.* Dover Books on Computer Science Series. Dover Publications, 2003. Available at: <https://books.google.cz/books?id=fyVtp3EMxasC>. ISBN 9780486428093.

[88] Šomplák, R., Kudela, J., Smejkalová, V., Nevrlý, V., Pavlas, M. and Hrabec, D. Pricing and advertising strategies in conceptual waste management planning. *Journal of Cleaner Production.* 2019, vol. 239, pp. 118068. doi: 10.1016/j.jclepro.2019.118068.

[89] Gregor, J., Šomplák, R. and Pavlas, M. Transportation cost as an integral part of supply chain optimisation in the field of waste management. *Chemical Engineering Transactions*. 2017, vol. 56, pp. 1927–1932. doi: 10.3303/CET1756322.

[90] Alçada-Almeida, L., Coutinho-Rodrigues, J. and Current, J. A multiobjective modeling approach to locating incinerators. *Socio-Economic Planning Sciences*. 2009, vol. 43, pp. 111–120. doi: 10.1016/j.seps.2008.02.008.

[91] Ferdan, T., Šomplák, R., Zavíralová, L., Pavlas, M. and Frýba, L. A waste-to-energy project: A complex approach towards the assessment of investment risks. *Applied Thermal Engineering*. 2015, vol. 89, pp. 1127–1136. doi: 10.1016/j.applthermaleng.2015.04.005.

[92] Ferdan, T., Šomplák and Pavlas, M. Improved feasibility analysis under volatile conditions: Case of waste-to-energy. *Chemical Engineering Transactions*. 2014, vol. 39, pp. 691–696. doi: 10.3303/CET1439116.

[93] Hu, C., Liu, X. and Lu, J. A bi-objective two-stage robust location model for waste-to-energy facilities under uncertainty. *Decision Support Systems*. 2017, vol. 99, pp. 37–50. doi: 10.1016/j.dss.2017.05.009.

[94] Kyriakis, E., Psomopoulos, C., Kokkotis, P., Bourtsalas, A. and N., T. A step by step selection method for the location and the size of a waste-to-energy facility targeting the maximum output energy and minimization of gate fee. *Environmental Science and Pollution Research*. 2018, vol. 25, pp. 26715–26724. doi: 10.1007/s11356-017-9488-1.

[95] Malinauskaite, J. et al. Municipal solid waste management and waste-to-energy in the context of a circular economy and energy recycling in Europe. *Energy*. 2017, vol. 141, pp. 2013–2044. doi: 10.1016/j.energy.2017.11.128.

[96] Hrabec, D., Šomplák, R., Nevrlý, V., Viktorin, A., Pluháček, M. and Popela, P. Sustainable waste-to-energy facility location: Influence

of demand on energy sales. *Energy*. 2020, vol. 207, pp. 118257. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2020.118257.

[97] Grossmann, I. E., Viswanathan, J., Vecchietti, A., Raman, R., Kalvelagen, E. and others. GAMS/DICOPT: A discrete continuous optimization package. *GAMS Corporation Inc.* 2002, vol. 37, pp. 55.

[98] Viktorin, A., Senkerik, R., Pluhacek, M. and Kadavy, T. The Influence of Archive Size to SHADE. In Silhavy, R., Senkerik, R., Kominkova Oplatkova, Z., Prokopova, Z. and Silhavy, P. (Ed.) *Artificial Intelligence Trends in Intelligent Systems*, pp. 517–527, Cham, 2017. Springer International Publishing. ISBN 978-3-319-57261-1.

[99] Viktorin, A., Pluhacek, M., Senkerik, R. and Kadavy, T. Detecting Potential Design Weaknesses in SHADE Through Network Feature Analysis. In Pisón, F. J., Urraca, R., Quintián, H. and Corchado, E. (Ed.) *Hybrid Artificial Intelligent Systems*, pp. 662–673, Cham, 2017. Springer International Publishing. ISBN 978-3-319-59650-1.

[100] Viktorin, A., Senkerik, R., Pluhacek, M. and Kadavy, T. Analysing knowledge transfer in SHADE via complex network. *Logic Journal of the IGPL*. 09 2018, vol. 28, no. 2, pp. 153–170. ISSN 1367-0751. doi: 10.1093/jigpal/jzy042. Available at: <https://doi.org/10.1093/jigpal/jzy042>.

[101] Price, K., Awad, N., Ali, M. and Suganthan, P. Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. In *Technical Report.* : Nanyang Technological University, 2018.

[102] Viktorin, A., Senkerik, R., Pluhacek, M. and Kadavy, T. Analyzing Control Parameters in DISH. In Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R. and Zurada, J. M. (Ed.) *Artificial Intelligence and Soft Computing*, pp. 519–529, Cham, 2019. Springer International Publishing. ISBN 978-3-030-20912-4.

[103] Viktorin, A. *TBU–AILab: DISH_java*, 2020. Available at: <https://github.com/TBU-AILab/DISH_java>.

[104] Piotrowski, A. P. and Napiorkowski, J. J. Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure? *Swarm and Evolutionary Computation*. 2018, vol. 43, pp. 88 – 108. ISSN 2210-6502. doi: https://doi.org/10.1016/j.swevo.2018.03.007. Available at: <http://www.sciencedirect.com/science/article/pii/S2210650217306600>.

PUBLICATIONS OF THE AUTHOR

Journals with IF

[P.1] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas, ZAMUDA, Ales. Distance based parameter adaptation for Success-History based Differential Evolution. Swarm and Evolutionary Computation, 2019, 2019, vol. 50, pp. 1-17. ISSN 2210-6502.

[P.2] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Analysing knowledge transfer in SHADE via complex network. LOGIC JOURNAL OF THE IGPL, 2020, vol. 28, no. 2, pp. 153-170. ISSN 1367-0751.

[P.3] HRABEC, Dusan, SOMPLAK, Radovan, NEVRLY, Vlastimir, VIKTORIN, Adam, PLUHACEK, Michal, POPELA, Pavel. Sustainable waste-to-energy facility location: Influence of demand on energy sales. Energy, 2020, vol. 207, pp. 1-15. ISSN 0360-5442.

[P.4] PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman, KADAVY, Tomas, ZELINKA, Ivan. Extended experimental study on PSO with partial population restart based on complex network analysis. LOGIC JOURNAL OF THE IGPL, 2020, vol. 28, no. 2, pp. 211-225. ISSN 1367-0751.

Journals in Scopus

[P.5] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Modified progressive random walk with chaotic PRNG. International Journal of Parallel, Emergent and Distributed Systems, 2017, neuveden, neuveden, pp. 1-10. ISSN 1744-5760.

[P.6] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Clustering analysis of the population in Db_SHADE algorithm. Mendel, 2018, vol. 24, no. 1, pp. 9-16. ISSN 1803-3814.

[P.7] SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan, PLUHACEK, Michal, KADAVY, Tomas, KOMINKOVA

OPLATKOVA, Zuzana, BHATEJA, Vikrant, CHANDRA SATAPATHY, Suresh. Differential Evolution And Deterministic Chaotic Series: A Detailed Study. Mendel, 2018, vol. 24, no. 2, pp. 61-68. ISSN 1803-3814.

[P.8] PLUHACEK, Michal, ZELINKA, Ivan, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Particle swarm optimization with distance based repulsivity. Mendel, 2018, 2018, vol. 24, no. 2, pp. 81-86. ISSN 1803-3814.

Conference papers

[P.9] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal. Hybridization of Chaotic Systems and Success-History Based Adaptive Differential Evolution. In HYBRID METAHEURISTICS. Cham : Springer International Publishing Switzerland, 2016, pp. 145-156. ISSN 0302-9743. ISBN 978-3-319-39635-4.

[P.10] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas, ZAMUDA, Ales. Distance Based Parameter Adaptation for Differential Evolution. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. New Jersey, Piscataway : IEEE, 2017, pp. 2612-2618. ISBN 978-1-5386-2725-9.

[P.11] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Analyzing Control Parameters in DISH. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2019, pp. 519-529. ISSN 03029743. ISBN 978-3-030-20911-7.

[P.12] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman. Multichaotic System Induced Success-History Based Adaptive Differential Evolution. In ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING, ICAISC 2016. Cham : Springer International Publishing Switzerland, 2016, pp. 517-527. ISSN 0302-9743. ISBN 978-3-319-39377-3.

[P.13] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal. Simulating the Effect of Adaptivity on Randomization. In Proceedings of 2016 9th EUROSIM Congress on Modelling and Simulation. New York : IEEE, 2016, pp. 471-476. ISBN 978-1-5090-4119-0.

[P.14] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas, JASEK, Roman. A lightweight SHADE-based algorithm for global optimization - liteSHADE. In Lecture Notes in Electrical Engineering. Berlin : Springer Verlag, 2020, pp. 197-206. ISSN 1876-1100. ISBN 978-3-030-14906-2.

[P.15] VIKTORIN, Adam, PLUHACEK, Michal, KOMINKOVA OPLATKOVA, Zuzana, SENKERIK, Roman. Analytical Programming with Extended Individuals. In Proceedings - 30th European Conference on Modelling and Simulation ECMS 2016. Nottingham : EUROPEAN COUNCIL MODELLING & SIMULATION, SCHOOL COMPUTING & MATHEMATICS, 2016, pp. 237-244. ISBN 978-0-9932440-2-5.

[P.16] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman. Synthetic objective function to improve the performance of de - Initial study. In AIP Conference Proceedings. Maryland : American Institute of Physics Inc., 2017, pp. nestrankovano. ISSN 0094-243X. ISBN 978-073541538-6.

[P.17] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Archive analysis in SHADE. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 688-699. ISSN 0302-9743. ISBN 978-3-319-59059-2.

[P.18] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. The Influence of Archive Size to SHADE. In Artificial Intelligence Trends in Intelligent Systems, CSOC2017, VOL 1 Book Series: Advances in Intelligent Systems and Computing. Cham : Springer International Publishing AG, 2017, pp. 517-527. ISSN 2194-5357. ISBN 978-3-319-57260-4.

[P.19] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Enhanced archive for SHADE. In Advances in Intelligent Systems and Computing, Volume 837. Berlin : Springer Verlag, 2019, pp. 40-55. ISSN 21945357. ISBN 978-331997887-1.

[P.20] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. On the prolonged exploration of distance based parameter adaptation in SHADE. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10841. Berlin : Springer Verlag, 2018, pp. 561-571. ISSN 03029743. ISBN 978-331991252-3.

[P.21] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Cluster Occurrence in the DbL-SHADE Population. In 2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2018, pp. 1-8. ISBN 978-150906017-7.

[P.22] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Analysing knowledge transfer in SHADE via complex network. 2018,

[P.23] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman. Lozi Map Generated Initial Population in Analytical Programming. In Artificial Intelligence Perspectives in Intelligent Systems: Proceedings of the 5th computer science on-line conference 2016, vol. 1. Heidelberg : Springer-Verlag Berlin, 2016, pp. 297-306. ISSN 2194-5357. ISBN 978-3-319-33623-7.

[P.24] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman. Success-History Based Adaptive Differential Evolution Algorithm with Multi-Chaotic Framework for Parent Selection Performance on CEC2014 Benchmark Set. In 2016 IEEE Congress on Evolutionary Computation, CEC 2016. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2016, pp. 4797-4803. ISBN 978-1-5090-0622-9.

[P.25] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. L-SHADE Algorithm with Distance Based Parameter Adaptation. In Lecture Notes in Electrical Engineering. Berlin : Springer Verlag, 2017, pp. 69-80. ISSN 1876-1100. ISBN 978-331969813-7.

[P.26] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Towards better population sizing for differential evolution through active population analysis with complex network. In Advances in Intelligent Systems and Computing. Berlin : Springer Verlag, 2017, pp. 225-235. ISSN 2194-5357. ISBN 978-331961565-3.

[P.27] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman, KADAVY, Tomas. Detecting potential design weaknesses in shade through network feature analysis. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 662-673. ISSN 0302-9743. ISBN 978-3-319-59649-5.

[P.28] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Shade mutation strategy analysis via dynamic simulation in complex network. In Proceedings - 31st European Conference on Modelling and Simulation, ECMS 2017. Madrid : European Council for Modelling and Simulation, 2017, pp. 299-305. ISBN 978-099324404-9.

[P.29] VIKTORIN, Adam, HRABEC, Dusan, PLUHACEK, Michal. Multi-Chaotic Differential Evolution for Vehicle Routing Problem with Profits. In Proceedings - 30th European Conference on Modelling and Simulation ECMS 2016. Nottingham : EUROPEAN COUNCIL MODELLING & SIMULATION, SCHOOL COMPUTING & MATHEMATICS, 2016, pp. 245-251. ISBN 978-0-9932440-2-5.

[P.30] VIKTORIN, Adam, PLUHACEK, Michal, SENKERIK, Roman. Network Based Linear Population Size Reduction in SHADE. In 2016 8TH INTERNATIONAL CONFERENCE ON INTELLIGENT NET-

WORKING AND COLLABORATIVE SYSTEMS (INCOS). New York : IEEE, 2016, pp. 86-93. ISSN 2470-9166. ISBN 978-1-5090-4123-7.

[P.31] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, ZAMUDA, Ales. Steady Success Clusters in Differential Evolution. In 2016 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2016, pp. 1-8. ISBN 978-1-5090-4240-1.

[P.32] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. SHADE Algorithm Dynamic Analyzed Through Complex Network. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2017, pp. 666-677. ISSN 0302-9743. ISBN 978-331962388-7.

[P.33] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Addressing Premature Convergence with Distance based Parameter Adaptation in SHADE. In International Conference on Systems, Signals, and Image Processing. Los Alamitors : IEEE Computer Society, 2018, pp. nestrankovano. ISSN 21578672. ISBN 978-153866979-2.

[P.34] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. How distance based parameter adaptation affects population diversity. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10835. Berlin : Springer Verlag, 2018, pp. 307-319. ISSN 03029743. ISBN 978-331991640-8.

[P.35] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. COMPARATIVE STUDY OF THE DISTANCE/IMPROVEMENT BASED SHADE. In Proceedings - 32nd European Conference on Modelling and Simulation, ECMS 2018. Madrid : European Council for Modelling and Simulation, 2018, pp. 163-169. ISSN 2522-2414. ISBN 978-0-9932440-6-3.

[P.36] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas. Distance vs. Improvement based parameter adaptation in SHADE. In Advances in Intelligent Systems and Computing, Volume 764. Berlin : Springer Verlag, 2019, pp. 455-464. ISSN 2194-5357. ISBN 978-331991188-5.

[P.37] VIKTORIN, Adam, SENKERIK, Roman, PLUHACEK, Michal, KADAVY, Tomas, ZAMUDA, Ales. DISH Algorithm Solving the CEC 2019 100-Digit Challenge. In 2019 IEEE Congress on Evolutionary Computation (CEC). Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 1-6. ISBN 978-1-72812-153-6.

[P.38] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, JANOSTIK, Jakub, DAVENDRA, Donald. On the influence of different randomization and complex network analysis for differential evolution. In 2016 IEEE Congress on Evolutionary Computation, CEC 2016. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2016, pp. 3346-3353. ISBN 978-1-5090-0622-9.

[P.39] SENKERIK, Roman, ZELINKA, Ivan, PLUHACEK, Michal, VIKTORIN, Adam, JANOSTIK, Jakub, KOMINKOVA OPLATKOVA, Zuzana. Randomization and Complex Networks for Meta-Heuristic Algorithms. In Evolutionary Algorithms, Swarm Dynamics and Complex Networks. Heidelberg : Springer-Verlag GmbH, 2018, pp. 177-194. ISBN 978-3-662-55661-0.

[P.40] SENKERIK, Roman, KADAVY, Tomas, VIKTORIN, Adam, PLUHACEK, Michal. Ensemble of Strategies and Perturbation Parameter Based SOMA for Constrained Technological Design Optimization Problem. In 2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 2872-2877. ISBN 978-1-72812-153-6.

[P.41] SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, PLUHACEK, Michal, ZELINKA, Ivan. Insight into adaptive differential evolution variants with unconventional randomization schemes. In Communications in Computer and Information Science. Londyn : Springer Nature, 2020, pp. 177-188. ISSN 1865-0929. ISBN 978-3-030-37837-0.

[P.42] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, JANOSTIK, Jakub. On The Application Of Complex Network Analysis For Metaheuristics. In BIOINSPIRED OPTIMIZATION METHODS AND THEIR APPLICATIONS: Proceedings of the Seventh International Conference on Bioinspired Optimization Methods and their Applications, BIOMA 2016. Ljubljana : Jozef Stefan Institute, 2016, pp. 201-213. ISBN 978-961-264-093-4.

[P.43] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, KADAVY, Tomas. On the Randomization of Indices Selection for Differential Evolution. In Artificial Intelligence Trends in Intelligent Systems, CSOC2017, VOL 1 Book Series: Advances in Intelligent Systems and Computing. Cham : Springer International Publishing AG, 2017, pp. 537-547. ISSN 2194-5357. ISBN 978-3-319-57260-4.

[P.44] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, ZELINKA, Ivan. Differential evolution driven analytic programming for prediction. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 676-687. ISSN 0302-9743. ISBN 978-3-319-59059-2.

[P.45] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal. On the Transforming of the Indices Selection Mechanism inside Differential Evolution into Complex Network. In 2016 8TH INTERNATIONAL CONFERENCE ON INTELLIGENT NETWORKING AND COLLABORATIVE SYSTEMS (INCOS). New York : IEEE, 2016, pp. 186-192. ISSN 2470-9166. ISBN 978-1-5090-4123-7.

[P.46] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas. Population diversity analysis for the chaotic based selection of individuals in differential evolution. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10835. Berlin : Springer Verlag, 2018, pp. 283-294. ISSN 03029743. ISBN 978-331991640-8.

[P.47] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, KADAVY, Tomas, JANOSTIK, Jakub, KOMINKOVA OPLATKOVA, Zuzana. A Review On The Simulation of Social Networks Inside Heuristic Algorithms. In Proceedings - 32nd European Conference on Modelling and Simulation, ECMS 2018. Madrid : European Council for Modelling and Simulation, 2018, pp. 176-182. ISSN 2522-2414. ISBN 978-0-9932440-6-3.

[P.48] SENKERIK, Roman, PLUHACEK, Michal, ZELINKA, Ivan, VIKTORIN, Adam, JANOSTIK, Jakub. Extended Study on the Randomization and Sequencing for the Chaos Embedded Heuristic. In ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING, ICAISC 2016. Cham : Springer International Publishing Switzerland, 2016, pp. 493-504. ISSN 0302-9743. ISBN 978-3-319-39377-3.

[P.49] SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, PLUHACEK, Michal, KAZIKOVA, Anezka, DIEP, Quoc Bao, ZELINKA, Ivan. Population Diversity Analysis in Adaptive Differential Evolution Variants with Unconventional Randomization Schemes. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2019, pp. 506-518. ISSN 03029743. ISBN 978-3-030-20911-7.

[P.50] SENKERIK, Roman, ZELINKA, Ivan, PLUHACEK, Michal, VIKTORIN, Adam. Study on the development of complex network for evolutionary and swarm based algorithms. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2017, pp. 151-161. ISSN 0302-9743. ISBN 978-3-319-62427-3.

[P.51] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, KADAVY, Tomas, KOMINKOVA OPLATKOVA, Zuzana. Randomization of individuals selection in differential evolution. In Advances in Intelligent Systems and Computing, Volume 837. Berlin : Springer Verlag, 2019, pp. 180-191. ISSN 21945357. ISBN 978-331997887-1.

[P.52] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, KOMINKOVA OPLATKOVA, Zuzana. Differential Evolution and Chaotic Series. In International Conference on Systems, Signals, and Image Processing. Los Alamitors : IEEE Computer Society, 2018, pp. nestrankovano. ISSN 21578672. ISBN 978-153866979-2.

[P.53] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, ZELINKA, Ivan. How unconventional chaotic pseudo-random generators influence population diversity in differential evolution. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10841. Berlin : Springer Verlag, 2018, pp. 524-535. ISSN 03029743. ISBN 978-331991252-3.

[P.54] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, KOMINKOVA OPLATKOVA, Zuzana. On The Simulation Of Complex Chaotic Dynamics For Chaos Based Optimization. In Proceedings - 30th European Conference on Modelling and Simulation ECMS 2016. Nottingham : EUROPEAN COUNCIL MODELLING & SIMULATION, SCHOOL COMPUTING & MATHEMATICS, 2016, pp. 258-264. ISBN 978-0-9932440-2-5.

[P.55] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, ZELINKA, Ivan. Differential Evolution for Constrained Industrial Optimization. In Lecture Notes in Electrical Engineering. neuveden : Springer Nature, 2018, pp. 123-132. ISSN 18761100. ISBN 978-331969813-7.

[P.56] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, KOMINKOVA OPLATKOVA, Zuzana. On the applicability of random and the best solution driven metaheuristics for analytic programming and time series regression. In Advances in Intelligent Systems and Computing, Volume 764. Berlin : Springer Verlag, 2019, pp. 489-498. ISSN 2194-5357. ISBN 978-331991188-5.

[P.57] SENKERIK, Roman, PLUHACEK, Michal, ZELINKA, Ivan, VIKTORIN, Adam. Comparison Of Pso And De In The Task Of Optimal Control Of Chaotic Lozi Map. In Proceedings of the European Modelling & Simulation Symposium 2016. Genova : DIME University of Genoa, 2016, pp. 303-308. ISBN 978-88-97999-76-8.

[P.58] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, ZELINKA, Ivan. Hybridization of analytic programming and differential evolution for time series prediction. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 686-698. ISSN 0302-9743. ISBN 978-3-319-59649-5.

[P.59] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas, KOMINKOVA OPLATKOVA, Zuzana. Performance Comparison of Differential Evolution Driving Analytic Programming for Regression. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. New Jersey, Piscataway : IEEE, 2017, pp. 2358-2365. ISBN 978-1-5386-2725-9.

[P.60] SENKERIK, Roman, ZELINKA, Ivan, PLUHACEK, Michal, VIKTORIN, Adam. Comparison of swarm and evolutionary based algorithms for the stabilization of chaotic oscillations. In AETA 2016: Recent Advances in Electrical Engineering and Related Sciences Book Series: Lecture Notes in Electrical Engineering. Cham : Springer International Publishing AG, 2017, pp. 63-73. ISSN 1876-1100. ISBN 978-3-319-50903-7.

[P.61] SENKERIK, Roman, VIKTORIN, Adam, PLUHACEK, Michal, KADAVY, Tomas. On the Population Diversity for the Chaotic Differential Evolution. In 2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2018, pp. nestrankovano. ISBN 978-150906017-7.

[P.62] SENKERIK, Roman, PLUHACEK, Michal, VIKTORIN, Adam, KOMINKOVA OPLATKOVA, Zuzana, KADAVY, Tomas. Simulation of chaotic dynamics for chaos based optimization-an extended study. In Proceedings - 31st European Conference on Modelling and Simulation, ECMS 2017. Madrid : European Council for Modelling and Simulation, 2017, pp. 319-325. ISBN 978-099324404-9.

[P.63] SPOLAOR, Simone, GRIBAUDO, Marco, IACONO, Mauro, KADAVY, Tomas, KOMINKOVA OPLATKOVA, Zuzana, MAURI, Giancarlo, PILANA, Sabri, SENKERIK, Roman, STOJANOVIC, Natalija, TURUNEN, Esko, VIKTORIN, Adam, VITABILE, Salvatore, ZAMUDA, Ales, NOBILE, Marco Salvatore. Towards Human Cell Simulation. In High-Performance Modelling and Simulation for Big Data Applications. Cham : Springer, 2019, pp. 221-249. ISBN 978-3-030-16271-9.

[P.64] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. PSO with Attractive Search Space Border Points. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 665-675. ISSN 0302-9743. ISBN 978-3-319-59059-2.

[P.65] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Why Simple Population Restart Does Not Work in PSO. In Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 770-776. ISBN 978-1-5386-9276-9.

[P.66] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan. Single Swarm and Simple Multi-Swarm PSO Comparison. In Proceedings of 2016 9th EUROSIM Congress on Modelling and Simulation. New York : IEEE, 2016, pp. 498-502. ISBN 978-1-5090-4119-0.

[P.67] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. UNCOVERING COMMUNICATION DENSITY IN PSO USING COMPLEX NETWORK. In Proceedings - 31st European Conference on Modelling and Simulation, ECMS 2017. Madrid : European Council for Modelling and Simulation, 2017, pp. 306-312. ISBN 978-099324404-9.

[P.68] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Particle Swarm Optimization with Single Particle Repulsivity for Multi-modal Optimization. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10841. Berlin : Springer Verlag, 2018, pp. 486-494. ISSN 03029743. ISBN 978-331991252-3.

[P.69] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, ZELINKA, Ivan. A Review of Real-world Applications of Particle Swarm Optimization Algorithm. In Lecture Notes in Electrical Engineering. neuveden : Springer Nature, 2018, pp. 115-122. ISSN 1876-1100. ISBN 978-331969813-7.

[P.70] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, ZELINKA, Ivan. Chaos Driven PSO with Attractive Search Space Border Points. In 2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2018, pp. 1147-1152. ISBN 978-150906017-7.

[P.71] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Self-Organizing Migrating Algorithm with non-binary perturbation. In Communications in Computer and Information Science. Londyn : Springer Nature, 2020, pp. 43-57. ISSN 1865-0929. ISBN 978-3-030-37837-0.

[P.72] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, ZELINKA, Ivan. PSO with Partial Population Restart Based on Complex Network Analysis. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 183-192. ISSN 0302-9743. ISBN 978-3-319-59649-5.

[P.73] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Exploring the Shortest Path in PSO Communication Network. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. New Jersey, Piscataway : IEEE, 2017, pp. 1494-1499. ISBN 978-1-5386-2725-9.

[P.74] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas, ZELINKA, Ivan. USING COMPLEX NETWORK VISUALIZATION AND ANALYSIS FOR UNCOVERING THE INNER DYNAMICS OF PSO ALGORITHM. In Mendel. Brno : VUT Brno, 2017, pp. 87-94. ISSN 1803-3814.

[P.75] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan. Multi-chaotic Approach for Particle Acceleration in PSO. In HYBRID METAHEURISTICS. Cham : Springer International Publishing Switzerland, 2016, pp. 75-86. ISSN 0302-9743. ISBN 978-3-319-39635-4.

[P.76] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam. On the Impact of Cognitive Factor in PSO – Testing on Selected Functions from CEC 15 Benchmark. In AIP Conference Proceedings. Maryland : American Institute of Physics Inc., 2017, ISSN 0094-243X. ISBN 978-073541538-6.

[P.77] PLUHACEK, Michal, HRDY, Michal, VIKTORIN, Adam, KADAVY, Tomas, SENKERIK, Roman. Spiral extrusion die design using modified differential evolution algorithm. In Mendel. Brno : Brno University of Technology, 2019, pp. 121-130. ISSN 1803-3814.

[P.78] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan. Chaos Enhanced Repulsive MC-PSO/DE Hybrid. In ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING, ICAISC 2016. Cham : Springer International Publishing Switzerland, 2016, pp. 466-475. ISSN 0302-9743. ISBN 978-3-319-39377-3.

[P.79] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, JANOSTIK, Jakub, DAVENDRA, Donald. Complex network analysis in PSO as an fitness landscape classifier. In 2016 IEEE Congress on Evolutionary Computation, CEC 2016. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2016, pp. 3332-3337. ISBN 978-1-5090-0622-9.

[P.80] PLUHACEK, Michal, KADAVY, Tomas, SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan. Comparing Selected PSO Modifications on CEC 15 Benchmark Set. In 2016 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2016, pp. "nestrankovano". ISBN 978-1-5090-4240-1.

[P.81] PLUHACEK, Michal, SENKERIK, Roman, JANOSTIK, Jakub, VIKTORIN, Adam, ZELINKA, Ivan. Study on swarm dynamics converted into complex network. In Proceedings - 30th European Conference on Modelling and Simulation ECMS 2016. Nottingham : EUROPEAN COUNCIL MODELLING & SIMULATION, SCHOOL COMPUTING & MATHEMATICS, 2016, pp. 252-257. ISBN 978-0-9932440-2-5.

[P.82] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, ZELINKA, Ivan. Creating Complex Networks Using Multi-swarm PSO. In 2016 8TH INTERNATIONAL CONFERENCE ON INTELLIGENT NETWORKING AND COLLABORATIVE SYSTEMS (INCOS). New York : IEEE, 2016, pp. 180-185. ISSN 2470-9166. ISBN 978-1-5090-4123-7.

[P.83] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. How Chaotic Sequences and Generator Sequencing Affect the Particle Trajectory in PSO. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. New Jersey, Piscataway : IEEE, 2017, pp. 2570-2577. ISBN 978-1-5386-2725-9.

[P.84] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. STUDY ON VELOCITY CL AMPING IN PSO USING CEC'13 BENCHMARK. In Proceedings - 32nd European Conference on Modelling and Simulation, ECMS 2018. Madrid : European Council for Modelling and Simulation, 2018, pp. 150-155. ISSN 2522-2414. ISBN 978-0-9932440-6-3.

[P.85] PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam, KADAVY, Tomas. Complex Networks in Particle Swarm. In Evolutionary Algorithms, Swarm Dynamics and Complex Networks. Heidelberg : Springer-Verlag GmbH, 2018, pp. 145-159. ISBN 978-3-662-55661-0.

[P.86] KOMINKOVA OPLATKOVA, Zuzana, VIKTORIN, Adam, SENKERIK, Roman, URBANEK, Tomas. Different Approaches for constant estimation in analytic programming. In Proceedings - 31st European Conference on Modelling and Simulation, ECMS 2017. Madrid : European Council for Modelling and Simulation, 2017, pp. 326-332. ISBN 978-099324404-9.

[P.87] KOMINKOVA OPLATKOVA, Zuzana, SENKERIK, Roman, VIKTORIN, Adam. Differential Evolution and Analytic Programming in the case of Trigonometric Identities Discovery. In International Conference on Systems, Signals, and Image Processing. Los Alamitors : IEEE Computer Society, 2018, pp. nestrankovano. ISSN 21578672. ISBN 978-153866979-2.

[P.88] KOMINKOVA OPLATKOVA, Zuzana, VIKTORIN, Adam, SENKERIK, Roman. Pseudo Neural Networks via Analytic Programming with Direct Coding of Constant Estimation. In Proceedings - 32nd European Conference on Modelling and Simulation, ECMS

2018. Madrid : European Council for Modelling and Simulation, 2018, pp. 143-149. ISSN 2522-2414. ISBN 978-0-9932440-6-3.

[P.89] KOMINKOVA OPLATKOVA, Zuzana, VIKTORIN, Adam, SENKERIK, Roman. Comparison of three novelty approaches to constants (Ks) handling in analytic programming powered by SHADE. In Advances in Intelligent Systems and Computing, Volume 837. Berlin : Springer Verlag, 2019, pp. 134-145. ISSN 21945357. ISBN 978-331997887-1.

[P.90] KAZIKOVA, Anezka, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. New Running Technique for the Bison Algorithm. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10841. Berlin : Springer Verlag, 2018, pp. 417-426. ISSN 03029743. ISBN 978-331991252-3.

[P.91] KAZIKOVA, Anezka, PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam. Proposal of a new swarm optimization method inspired in bison behavior. In Advances in Intelligent Systems and Computing, Volume 837. Berlin : Springer Verlag, 2019, pp. 146-156. ISSN 21945357. ISBN 978-331997887-1.

[P.92] KADAVY, Tomas, PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam. On the Performance Significance of Boundary Strategies for Firefly Algorithm. In Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018. Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 765-769. ISBN 978-1-5386-9276-9.

[P.93] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Firework algorithm dynamics simulated and analyzed with the aid of complex network. In Proceedings - 31st European Conference on Modelling and Simulation, ECMS 2017. Madrid : European Council for Modelling and Simulation, 2017, pp. 313-318. ISBN 978-099324404-9.

[P.94] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Orthogonal Learning Firefly Algorithm. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2018, pp. 315-326. ISSN 03029743. ISBN 978-331992638-4.

[P.95] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Comparing border strategies for roaming particles on single and Multi-Swarm PSO. In Artificial Intelligence Trends in Intelligent Systems, CSOC2017, VOL 1 Book Series: Advances in Intelligent Systems and Computing. Cham : Springer International Publishing AG, 2017, pp. 528-536. ISSN 2194-5357. ISBN 978-3-319-57260-4.

[P.96] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Comparing Boundary Control Methods for Firefly Algorithm. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10835. Berlin : Springer Verlag, 2018, pp. 163-173. ISSN 03029743. ISBN 978-331991640-8.

[P.97] KADAVY, Tomas, KOVAR, Stanislav, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Evolutionary Algorithms Applied to a Shielding Enclosure Design. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Berlin : Springer Verlag, 2019, pp. 445-455. ISSN 03029743. ISBN 978-3-030-20911-7.

[P.98] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Hypersphere universe boundary method comparison on hclpso and PSO. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 173-182. ISSN 0302-9743. ISBN 978-3-319-59649-5.

[P.99] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Multi-swarm Optimization Algorithm Based on Firefly and Particle Swarm Optimization Techniques. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Volume 10841. Berlin : Springer Verlag, 2018, pp. 405-416. ISSN 03029743. ISBN 978-331991252-3.

[P.100] KADAVY, Tomas, PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam. The Ensemble of Strategies and Perturbation Parameter in Self-organizing Migrating Algorithm Solving CEC 2019 100-Digit Challenge. In 2019 IEEE Congress on Evolutionary Computation (CEC). Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 372-375. ISBN 978-1-72812-153-6.

[P.101] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Boundary Strategies For Firefly Algorithm Analysed Using CEC'17 Benchmark. In Proceedings - 32nd European Conference on Modelling and Simulation, ECMS 2018. Madrid : European Council for Modelling and Simulation, 2018, pp. 170-175. ISSN 2522-2414. ISBN 978-0-9932440-6-3.

[P.102] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Firefly Algorithm: Enhanced Version with Partial Population Restart Using Complex Network Analysis. In Lecture Notes in Electrical Engineering. neuveden : Springer Nature, 2018, pp. 59-68. ISSN 18761100. ISBN 978-331969813-7.

[P.103] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Firefly Algorithm Enhanced by Orthogonal Learning. In Advances in Intelligent Systems and Computing, Volume 764. Berlin : Springer Verlag, 2019, pp. 477-488. ISSN 2194-5357. ISBN 978-331991188-5.

[P.104] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Dynamic of firework algorithm analyzed with complex network. In Mendel. Brno : VUT Brno, 2017, pp. 79-86. ISSN 1803-3814.

[P.105] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Partial Population Restart of Firefly Algorithm Using Complex Network Analysis. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) Proceedings. New Jersey, Piscataway : IEEE, 2017, pp. 2505-2511. ISBN 978-1-5386-2725-9.

[P.106] KADAVY, Tomas, PLUHACEK, Michal, VIKTORIN, Adam, SENKERIK, Roman. Comparing strategies for search space boundaries violation in PSO. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Heidelberg : Springer-Verlag Berlin, 2017, pp. 655-664. ISSN 0302-9743. ISBN 978-3-319-59059-2.

[P.107] KADAVY, Tomas, PLUHACEK, Michal, SENKERIK, Roman, VIKTORIN, Adam. Introducing Self-Adaptive Parameters to Self-organizing Migrating Algorithm. In 2019 IEEE Congress on Evolutionary Computation (CEC). Piscataway, New Jersey : Institute of Electrical and Electronics Engineers Inc., 2019, pp. 2908-2914. ISBN 978-1-72812-153-6.

[P.108] HRABEC, Dusan, VIKTORIN, Adam, SOMPLAK, Radovan, PLUHACEK, Michal, POPELA, Pavel. A heuristic approach to the facility location problem for waste management: A case study. In Mendel. Brno : Brno University of Technology, 2016, pp. 61-66. ISSN 1803-3814.

CURRICULUM VITAE

Personal Information:	
Name	Adam Viktorin
Date of birth	18 November 1989
Present address	Druzstevni 1764 Vsetin 755 01
Contact	phone: +420 605 062 931 email: aviktorin@utb.cz
Education:	
2010 - 2013	Tomas Bata University in Zlín, Faculty of Applied Informatics, Information and Control Technologies (Bc.)
2013 - 2015	Tomas Bata University in Zlín, Faculty of Applied Informatics, Computer and Communication Systems (Ing.)
2015 - present	Tomas Bata University in Zlín, Faculty of Applied Informatics, Engineering Informatics (Ph.D.)
Language knowledge:	
Czech	Native
English	Intermediate (Doctoral exam)
Professional achievements:	
Best paper award	AETA 2018
Others	Nominee for Joseph Fourier Prize 2017 Short term scientific mission - University of Maribor, Slovenia, 2018 Erasmus+ - Stellenbosh Univeristy, ZAR, 2020
Grant activities	Technology Agency of the Czech Republic, FW01010381 (member of the team) Grant Agency of the Czech Republic, GA 20-0009IY (member of the team)

LIST OF APPENDICES

APPENDIX A: DE PSEUDO–CODE

Algorithm 1 DE

Set *NP, CR, F, D* and stopping criterion;
$G = 1$, $\boldsymbol{x}_{best} = \{\}$;
Randomly initialize (3.1) population $\boldsymbol{P} = (\boldsymbol{x}_{1,G}, \ldots, \boldsymbol{x}_{NP,G})$;
$\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
while stopping criterion not met **do**
    for $i = 1$ to NP **do**
        $\boldsymbol{x}_{i,G} = \boldsymbol{P}[i]$;
        $\boldsymbol{v}_{i,G}$ by mutation (3.2);
        $\boldsymbol{u}_{i,G}$ by crossover (3.3);
        $\boldsymbol{x}_{i,G+1}$ by selection (3.4);
        $\boldsymbol{x}_{i,G+1} \rightarrow \boldsymbol{P}_{new}$;
    end for
    $\boldsymbol{P} = \boldsymbol{P}_{new}$, $\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
end while
return $\boldsymbol{x}_{best}$ as the best found solution;

APPENDIX B: SHADE PSUEDO–CODE

Algorithm 2 SHADE

Set NP, H, D, and $MAXFES$ (stopping criterion);
$G = 1$, $\boldsymbol{x}_{best} = \{\}$, $k = 1$, $p_{min} = 2/NP$, $\boldsymbol{A} = \emptyset$;
Randomly initialize population $\boldsymbol{P} = (\boldsymbol{x}_{1,G}, \ldots, \boldsymbol{x}_{NP,G})$ (3.1);
$FES+ = NP$;
Set all values in $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ according to (5.1);
$\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
while stopping criterion not met **do**
  $\boldsymbol{S}_F = \emptyset$, $\boldsymbol{S}_{CR} = \emptyset$;
  for $i = 1$ to NP **do**
    $\boldsymbol{x}_{i,G} = \boldsymbol{P}[i]$;
    $r = U[1, H]$, $p_i = U[p_{min}, 0.2]$;
    Set F_i by (5.3) and CR_i by (5.5);
    $\boldsymbol{v}_{i,G}$ by mutation (5.2);
    $\boldsymbol{u}_{i,G}$ by binomial crossover (5.4);
    if $f(\boldsymbol{u}_{i,G}) \leq f(\boldsymbol{x}_{i,G})$ **then**
      $\boldsymbol{x}_{i,G+1} = \boldsymbol{u}_{i,G}$;
      $\boldsymbol{x}_{i,G} \rightarrow \boldsymbol{A}$;
      $F_i \rightarrow \boldsymbol{S}_F$, $CR_i \rightarrow \boldsymbol{S}_{CR}$;
    else
      $\boldsymbol{x}_{i,G+1} = \boldsymbol{x}_{i,G}$;
    end if
    if $|\boldsymbol{A}| > NP$ **then**
      Randomly delete $|\boldsymbol{A}| - NP$ individuals from $|\boldsymbol{A}|$;
    end if
    $\boldsymbol{x}_{i,G+1} \rightarrow \boldsymbol{P}_{new}$;
  end for
  if $\boldsymbol{S}_F \neq \emptyset$ **and** $\boldsymbol{S}_{CR} \neq \emptyset$ **then**
    Update $\boldsymbol{M}_{F,k}$ (5.6) and $\boldsymbol{M}_{CR,k}$ (5.6);
    $k++$;
    if $k > H$ **then**
      $k = 1$;
    end if
  end if
  $\boldsymbol{P} = \boldsymbol{P}_{new}$, $\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
end while
return $\boldsymbol{x}_{best}$ as the best–found solution;

APPENDIX C: L–SHADE PSUEDO–CODE

Algorithm 3 L–SHADE

Set NP, H, D, and $MAXFES$ (stopping criterion);
$G = 1$, $\boldsymbol{x}_{best} = \{\}$, $k = 1$, $p_{min} = 2/NP$, $\boldsymbol{A} = \emptyset$;
Randomly initialize population $\boldsymbol{P} = (\boldsymbol{x}_{1,G}, \ldots, \boldsymbol{x}_{NP,G})$ (3.1);
$FES+ = NP$;
Set all values in $\boldsymbol{M}_F$ and $\boldsymbol{M}_{CR}$ according to (5.1);
$\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
while stopping criterion not met **do**
  $\boldsymbol{S}_F = \emptyset$, $\boldsymbol{S}_{CR} = \emptyset$;
  for $i = 1$ to NP **do**
    $\boldsymbol{x}_{i,G} = \boldsymbol{P}[i]$;
    $r = U[1, H]$, $p_i = U[p_{min}, 0.2]$;
    Set F_i by (5.3) and CR_i by (5.5);
    $\boldsymbol{v}_{i,G}$ by mutation (5.2);
    $\boldsymbol{u}_{i,G}$ by binomial crossover (5.4);
    if $f(\boldsymbol{u}_{i,G}) \leq f(\boldsymbol{x}_{i,G})$ **then**
      $\boldsymbol{x}_{i,G+1} = \boldsymbol{u}_{i,G}$;
      $\boldsymbol{x}_{i,G} \rightarrow \boldsymbol{A}$;
      $F_i \rightarrow \boldsymbol{S}_F$, $CR_i \rightarrow \boldsymbol{S}_{CR}$;
    else
      $\boldsymbol{x}_{i,G+1} = \boldsymbol{x}_{i,G}$;
    end if
    if $|\boldsymbol{A}| > NP$ **then**
      Randomly delete $|\boldsymbol{A}| - NP$ individuals from $|\boldsymbol{A}|$;
    end if
    $\boldsymbol{x}_{i,G+1} \rightarrow \boldsymbol{P}_{new}$;
  end for
  if $\boldsymbol{S}_F \neq \emptyset$ **and** $\boldsymbol{S}_{CR} \neq \emptyset$ **then**
    Update $\boldsymbol{M}_{F,k}$ (5.6) and $\boldsymbol{M}_{CR,k}$ (5.6);
    $k++$;
    if $k > H$ **then**
      $k = 1$;
    end if
  end if
  $\boldsymbol{P} = \boldsymbol{P}_{new}$, $\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
end while
return $\boldsymbol{x}_{best}$ as the best–found solution;

APPENDIX D: DISH PSUEDO–CODE

Algorithm 4 DISH

Set NP_{init}, NP_f, D, and $MAXFES$ (stopping criterion);
$NP = NP_{init}$, $H = 5$, $G = 1$, $\boldsymbol{x}_{best} = \{\}$, $k = 1$, $p_{max} = 0.25$, $p_{min} = 0.125$, $\boldsymbol{A} = \emptyset$, $FES = 0$;
Randomly initialize population $\boldsymbol{P} = (\boldsymbol{x}_{1,G}, \dots, \boldsymbol{x}_{NP,G})$ (6.6);
$FES+ = NP$;
Set all values in $\boldsymbol{M}_F$ to 0.5 and $\boldsymbol{M}_{CR}$ to 0.8;
$\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$;
while stopping criterion not met **do**
    $\boldsymbol{S}_F = \emptyset$, $\boldsymbol{S}_{CR} = \emptyset$;
    for $i = 1$ to NP **do**
        $r = U[1, H]$;
        if $r = H$ **then**
            $\boldsymbol{M}_{F,r} = 0.9$;
            $\boldsymbol{M}_{CR,r} = 0.9$;
        end if
        $CR_{i,G} = N(\boldsymbol{M}_{CR,r}, 0.1)$;
        if $CR_{i,G} < 0$ **then**
            $CR_{i,G} = 0$;
        else if $CR_{i,G} > 1$ **then**
            $CR_{i,G} = 1$;
        end if
        $F_{i,G} = C(\boldsymbol{M}_{F,r}, 0.1)$;
        while $F_{i,G} \leq 0$ **do**
            $F_{i,G} = C(\boldsymbol{M}_{F,r}, 0.1)$;
        end while
        if $F_{i,G} > 1$ **then**
            $F_{i,G} = 1$;
        end if
        $FES_{ratio} = FES/MAXFES$;
        if $FES_{ratio} < 0.6$ **and** $F_{i,G} > 0.7$ **then**
            $F_{i,G} = 0.7$;
        end if
        if $FES_{ratio} < 0.25$ **then**
            $CR_{i,G} = \max(CR_{i,G}, 0.7)$;
        else if $FES_{ratio} < 0.5$ **then**
            $CR_{i,G} = \max(CR_{i,G}, 0.6)$;
        end if

$\boldsymbol{x}_{i,G} = \boldsymbol{P}[i]$;
$p_i = p_{min} + FES_{ratio} * (p_{max} - p_{min})$ (6.10);
if $FES_{ratio} < 0.2$ **then**
$F_{w,i,G} = 0.7F_{i,G}$;
else if $FES_{ratio} < 0.4$ **then**
$F_{w,i,G} = 0.8F_{i,G}$;
else
$F_{w,i,G} = 1.2F_{i,G}$;
end if
$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F_{w,i,G}(\boldsymbol{x}_{pBest} - \boldsymbol{x}_{i,G}) + F_{i,G}(\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2})$ (6.9);
$\boldsymbol{u}_{i,G}$ by binomial crossover (6.16);
if $f(\boldsymbol{u}_{i,G}) \leq f(\boldsymbol{x}_{i,G})$ **then**
$\boldsymbol{x}_{i,G+1} = \boldsymbol{u}_{i,G}$;
$\boldsymbol{x}_{i,G} \rightarrow \boldsymbol{A}$;
$F_i \rightarrow \boldsymbol{S}_F$, $CR_i \rightarrow \boldsymbol{S}_{CR}$;
else
$\boldsymbol{x}_{i,G+1} = \boldsymbol{x}_{i,G}$;
end if
if $|\boldsymbol{A}| > NP$ **then**
Randomly delete $|\boldsymbol{A}| - NP$ individuals from $|\boldsymbol{A}|$;
end if
$\boldsymbol{x}_{i,G+1} \rightarrow \boldsymbol{P}_{new}$;
end for
$NP_{new} = \text{round}(NP_{init} - FES_{ratio} * (NP_{init} - NP_f))$ (6.18);
if $NP_{new} < NP$ **then**
Sort individuals in $\boldsymbol{P}$ according to their objective function values and remove $NP - NP_{new}$ worst ones;
$NP = NP_{new}$;
end if
if $|\boldsymbol{A}| > NP$ **then**
Randomly delete $|\boldsymbol{A}| - NP$ individuals from $|\boldsymbol{A}|$;
end if
if $\boldsymbol{S}_F \neq \emptyset$ **and** $\boldsymbol{S}_{CR} \neq \emptyset$ **then**
Update $\boldsymbol{M}_{F,k}$ (6.20) and $\boldsymbol{M}_{CR,k}$ (6.21) with Lehmer mean computed by (6.19) with distance based weights from (6.22);
$k++$;
if $k > H$ **then**
$k = 1$;
end if
end if
$\boldsymbol{P} = \boldsymbol{P}_{new}$, $\boldsymbol{P}_{new} = \{\}$, $\boldsymbol{x}_{best}$ = best from population $\boldsymbol{P}$, $G++$;
end while
return $\boldsymbol{x}_{best}$ as the best–found solution;

APPENDIX E: MC–SHADE - RESULT TABLES

Tab. E.1 CEC 2016 benchmark set results of
MC–SHADE algorithm in $10D$.

Func.	Best	Worst	Median	Mean	Std
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
4	0.00E+00	3.48E+01	3.48E+01	2.47E+01	1.58E+01
5	3.74E+00	2.01E+01	2.00E+01	1.54E+01	6.03E+00
6	0.00E+00	8.95E−01	0.00E+00	3.93E−02	1.77E−01
7	0.00E+00	3.44E−02	2.35E−03	4.47E−03	5.65E−03
8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
9	1.19E+00	4.90E+00	2.24E+00	2.47E+00	7.93E−01
10	0.00E+00	1.87E−01	0.00E+00	2.20E−02	4.11E−02
11	1.31E+01	2.72E+02	4.83E+01	6.32E+01	5.18E+01
12	9.43E−02	2.66E−01	1.89E−01	1.85E−01	3.75E−02
13	5.65E−02	1.12E−01	8.09E−02	8.07E−02	1.36E−02
14	5.50E−02	2.18E−01	1.13E−01	1.17E−01	3.91E−02
15	2.94E−01	6.97E−01	5.09E−01	4.92E−01	8.61E−02
16	8.40E−01	2.21E+00	1.49E+00	1.50E+00	2.47E−01
17	0.00E+00	1.21E+02	9.95E−01	9.42E+00	2.85E+01
18	7.04E−03	1.36E+00	5.88E−02	1.72E−01	2.42E−01
19	9.61E−02	1.04E+00	1.50E−01	2.08E−01	2.08E−01
20	7.92E−02	4.41E−01	2.25E−01	2.33E−01	8.23E−02
21	4.52E−05	1.13E+00	2.52E−01	2.88E−01	2.84E−01
22	7.64E−02	6.83E−01	1.63E−01	1.77E−01	8.84E−02
23	3.29E+02	3.29E+02	3.29E+02	3.29E+02	0.00E+00
24	1.00E+02	1.11E+02	1.08E+02	1.08E+02	2.12E+00
25	1.09E+02	2.01E+02	1.17E+02	1.29E+02	2.85E+01
26	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.73E−02
27	1.21E+00	4.00E+02	1.81E+00	6.81E+01	1.37E+02
28	3.57E+02	4.95E+02	3.72E+02	4.10E+02	5.30E+01
29	1.27E+02	2.25E+02	2.22E+02	2.20E+02	1.33E+01
30	4.54E+02	5.97E+02	4.63E+02	4.75E+02	2.44E+01

Tab. E.2 CEC 2016 benchmark set results of MC–SHADE algorithm in 30D.

Func.	Best	Worst	Median	Mean	Std
1	6.83E−01	1.09E+04	2.09E+03	2.92E+03	2.72E+03
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
4	0.00E+00	3.99E+00	0.00E+00	7.82E−02	5.58E−01
5	2.01E+01	2.02E+01	2.02E+01	2.02E+01	2.39E−02
6	0.00E+00	9.15E+00	4.56E−02	1.23E+00	2.37E+00
7	0.00E+00	1.48E−02	0.00E+00	2.90E−04	2.07E−03
8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
9	1.26E+01	2.65E+01	1.92E+01	1.95E+01	2.95E+00
10	0.00E+00	4.16E−02	0.00E+00	1.10E−02	1.46E−02
11	1.10E+03	1.94E+03	1.59E+03	1.57E+03	1.91E+02
12	1.31E−01	2.73E−01	2.16E−01	2.10E−01	2.98E−02
13	1.43E−01	2.74E−01	2.04E−01	2.06E−01	3.00E−02
14	1.56E−01	2.85E−01	2.14E−01	2.19E−01	3.59E−02
15	1.83E+00	3.71E+00	3.00E+00	3.00E+00	3.91E−01
16	7.93E+00	1.03E+01	9.39E+00	9.42E+00	4.43E−01
17	4.44E+02	2.15E+03	1.20E+03	1.24E+03	3.93E+02
18	2.03E+01	2.25E+02	7.57E+01	7.81E+01	3.68E+01
19	2.88E+00	6.96E+00	4.40E+00	4.50E+00	8.29E−01
20	2.98E+00	8.12E+01	1.83E+01	1.99E+01	1.27E+01
21	1.30E+01	6.90E+02	3.06E+02	3.15E+02	1.58E+02
22	2.87E+01	2.97E+02	1.50E+02	1.37E+02	6.64E+01
23	3.15E+02	3.15E+02	3.15E+02	3.15E+02	0.00E+00
24	2.22E+02	2.36E+02	2.24E+02	2.25E+02	2.02E+00
25	2.03E+02	2.09E+02	2.05E+02	2.05E+02	1.80E+00
26	1.00E+02	2.00E+02	1.00E+02	1.02E+02	1.40E+01
27	3.00E+02	4.38E+02	3.00E+02	3.40E+02	4.76E+01
28	6.85E+02	8.63E+02	7.97E+02	8.00E+02	3.02E+01
29	5.31E+02	8.22E+02	7.38E+02	7.35E+02	3.86E+01
30	5.67E+02	3.47E+03	1.44E+03	1.56E+03	6.43E+02

Tab. E.3 CEC 2016 benchmark set results of MC–SHADE algorithm in $50D$.

Func.	Best	Worst	Median	Mean	Std
1	4.45E+03	7.11E+04	2.03E+04	2.33E+04	1.38E+04
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
3	0.00E+00	2.26E−02	1.90E−06	1.03E−03	4.05E−03
4	0.00E+00	9.81E+01	5.00E−04	1.27E+01	2.76E+01
5	2.02E+01	2.03E+01	2.02E+01	2.02E+01	2.16E−02
6	7.72E−01	7.92E+00	3.83E+00	3.87E+00	1.84E+00
7	0.00E+00	1.97E−02	0.00E+00	3.96E−03	5.89E−03
8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
9	3.17E+01	5.75E+01	4.59E+01	4.53E+01	6.96E+00
10	0.00E+00	5.00E−02	1.25E−02	1.30E−02	1.34E−02
11	2.74E+03	4.28E+03	3.57E+03	3.60E+03	3.89E+02
12	1.58E−01	2.64E−01	2.12E−01	2.11E−01	2.24E−02
13	2.28E−01	4.89E−01	3.26E−01	3.25E−01	5.30E−02
14	2.08E−01	4.68E−01	2.87E−01	2.94E−01	4.28E−02
15	5.09E+00	1.07E+01	6.86E+00	6.96E+00	1.01E+00
16	1.70E+01	1.84E+01	1.78E+01	1.78E+01	3.57E−01
17	1.23E+03	4.51E+03	2.63E+03	2.74E+03	7.89E+02
18	7.39E+01	2.69E+02	1.81E+02	1.73E+02	4.41E+01
19	6.20E+00	2.91E+01	1.21E+01	1.41E+01	5.57E+00
20	1.06E+02	4.46E+02	2.30E+02	2.39E+02	7.43E+01
21	8.03E+02	3.29E+03	1.50E+03	1.58E+03	5.66E+02
22	1.68E+02	7.00E+02	4.12E+02	4.18E+02	1.33E+02
23	3.44E+02	3.44E+02	3.44E+02	3.44E+02	0.00E+00
24	2.69E+02	2.79E+02	2.75E+02	2.75E+02	2.29E+00
25	2.06E+02	2.26E+02	2.18E+02	2.17E+02	5.59E+00
26	1.00E+02	2.00E+02	1.00E+02	1.04E+02	1.95E+01
27	3.04E+02	5.56E+02	4.51E+02	4.49E+02	5.02E+01
28	1.03E+03	1.28E+03	1.15E+03	1.15E+03	5.71E+01
29	7.80E+02	1.11E+03	8.71E+02	8.84E+02	6.52E+01
30	8.42E+03	1.18E+04	9.89E+03	9.87E+03	7.70E+02

Tab. E.4 CEC 2016 benchmark set results of MC–SHADE algorithm in $100D$.

Func.	Best	Worst	Median	Mean	Std
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
4	0.00E+00	3.48E+01	3.48E+01	2.47E+01	1.58E+01
5	3.74E+00	2.01E+01	2.00E+01	1.54E+01	6.03E+00
6	0.00E+00	8.95E−01	0.00E+00	3.93E−02	1.77E−01
7	0.00E+00	3.44E−02	2.35E−03	4.47E−03	5.65E−03
8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
9	1.19E+00	4.90E+00	2.24E+00	2.47E+00	7.93E−01
10	0.00E+00	1.87E−01	0.00E+00	2.20E−02	4.11E−02
11	1.31E+01	2.72E+02	4.83E+01	6.32E+01	5.18E+01
12	9.43E−02	2.66E−01	1.89E−01	1.85E−01	3.75E−02
13	5.65E−02	1.12E−01	8.09E−02	8.07E−02	1.36E−02
14	5.50E−02	2.18E−01	1.13E−01	1.17E−01	3.91E−02
15	2.94E−01	6.97E−01	5.09E−01	4.92E−01	8.61E−02
16	8.40E−01	2.21E+00	1.49E+00	1.50E+00	2.47E−01
17	0.00E+00	1.21E+02	9.95E−01	9.42E+00	2.85E+01
18	7.04E−03	1.36E+00	5.88E−02	1.72E−01	2.42E−01
19	9.61E−02	1.04E+00	1.50E−01	2.08E−01	2.08E−01
20	7.92E−02	4.41E−01	2.25E−01	2.33E−01	8.23E−02
21	4.52E−05	1.13E+00	2.52E−01	2.88E−01	2.84E−01
22	7.64E−02	6.83E−01	1.63E−01	1.77E−01	8.84E−02
23	3.29E+02	3.29E+02	3.29E+02	3.29E+02	0.00E+00
24	1.00E+02	1.11E+02	1.08E+02	1.08E+02	2.12E+00
25	1.09E+02	2.01E+02	1.17E+02	1.29E+02	2.85E+01
26	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.73E−02
27	1.21E+00	4.00E+02	1.81E+00	6.81E+01	1.37E+02
28	3.57E+02	4.95E+02	3.72E+02	4.10E+02	5.30E+01
29	1.27E+02	2.25E+02	2.22E+02	2.20E+02	1.33E+01
30	4.54E+02	5.97E+02	4.63E+02	4.75E+02	2.44E+01

APPENDIX F: DISTANCE BASED PARAMETER ADAPTATION - RESULT TABLES

Tab. F.1 CEC 2015 benchmark set results of SHADE and Db_SHADE algorithms in $10D$.

	SHADE		**Db_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.00E+01	1.89E+01	2.00E+01	1.92E+01	=
4	3.07E+00	2.97E+00	3.06E+00	2.98E+00	=
5	2.21E+01	3.42E+01	2.98E+01	4.52E+01	=
6	2.20E−01	2.97E+00	4.16E−01	8.08E−01	=
7	1.67E−01	1.88E−01	1.73E−01	1.91E−01	=
8	8.15E−02	2.69E−01	4.28E−02	2.06E−01	=
9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
10	2.17E+02	2.17E+02	2.17E+02	2.17E+02	=
11	3.00E+02	1.66E+02	3.00E+02	2.01E+02	=
12	1.01E+02	1.01E+02	1.01E+02	1.01E+02	=
13	2.78E+01	2.78E+01	2.79E+01	2.76E+01	=
14	2.94E+03	4.28E+03	2.98E+03	4.66E+03	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. F.2 CEC 2015 benchmark set results of
L–SHADE and DbL_SHADE algorithms in 10D.

	L–SHADE		**DbL_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
3	2,00E+01	1,87E+01	2,00E+01	1,89E+01	=
4	2,98E+00	2,58E+00	2,99E+00	2,95E+00	=
5	2,87E+01	6,05E+01	1,54E+01	4,23E+01	=
6	4,16E−01	2,84E+00	6,24E−01	7,74E−01	−
7	7,01E−02	1,31E−01	9,49E−02	1,89E−01	=
8	4,21E−01	4,13E−01	3,29E−01	3,44E−01	=
9	1,00E+02	1,00E+02	1,00E+02	1,00E+02	=
10	2,17E+02	2,17E+02	2,17E+02	2,17E+02	=
11	3,00E+02	1,83E+02	3,00E+02	1,95E+02	=
12	1,01E+02	1,01E+02	1,01E+02	1,01E+02	+
13	2,71E+01	2,66E+01	2,69E+01	2,69E+01	=
14	2,94E+03	4,19E+03	2,94E+03	4,77E+03	=
15	1,00E+02	1,00E+02	1,00E+02	1,00E+02	=

Tab. F.3 CEC 2015 benchmark set results of SHADE and Db_SHADE algorithms in $30D$.

	SHADE		**Db_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	3.73E+01	2.62E+02	2.12E+01	2.42E+02	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.01E+01	2.01E+01	2.01E+01	2.01E+01	=
4	1.41E+01	1.41E+01	1.32E+01	1.31E+01	=
5	1.55E+03	1.50E+03	1.54E+03	1.52E+03	=
6	5.36E+02	5.73E+02	3.37E+02	3.48E+02	+
7	7.17E+00	7.26E+00	6.81E+00	6.74E+00	+
8	1.26E+02	1.21E+02	5.27E+01	7.38E+01	+
9	1.03E+02	1.03E+02	1.03E+02	1.03E+02	=
10	6.27E+02	6.22E+02	5.29E+02	5.32E+02	+
11	4.53E+02	4.50E+02	4.10E+02	4.16E+02	+
12	1.05E+02	1.05E+02	1.05E+02	1.05E+02	=
13	9.52E+01	9.50E+01	9.47E+01	9.50E+01	=
14	3.21E+04	3.24E+04	3.22E+04	3.24E+04	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. F.4 CEC 2015 benchmark set results of
L–SHADE and DbL_SHADE algorithms in $30D$.

	L–SHADE		**DbL_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	1,60E+00	6,18E+00	3,86E+00	2,00E+01	−
2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	=
4	1,29E+01	1,39E+01	1,29E+01	1,29E+01	=
5	1,44E+03	1,39E+03	1,40E+03	1,41E+03	=
6	7,61E+02	7,71E+02	4,64E+02	4,74E+02	+
7	6,70E+00	6,48E+00	5,91E+00	5,62E+00	+
8	1,51E+02	1,47E+02	1,21E+02	1,14E+02	+
9	1,03E+02	1,03E+02	1,03E+02	1,03E+02	=
10	7,21E+02	7,75E+02	5,99E+02	5,85E+02	+
11	4,77E+02	4,68E+02	4,21E+02	4,33E+02	+
12	1,05E+02	1,05E+02	1,05E+02	1,05E+02	=
13	9,29E+01	9,24E+01	9,32E+01	9,25E+01	=
14	3,33E+04	3,29E+04	3,31E+04	3,25E+04	=
15	1,00E+02	1,00E+02	1,00E+02	1,00E+02	=

Tab. F.5 CEC 2015 benchmark set results of
SHADE and Db_SHADE algorithms in $50D$.

	SHADE		**Db_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	1.81E+04	2.14E+04	3.00E+04	3.27E+04	−
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.01E+01	2.01E+01	2.02E+01	2.02E+01	−
4	3.84E+01	3.92E+01	3.15E+01	3.27E+01	+
5	3.10E+03	3.09E+03	3.06E+03	3.01E+03	=
6	2.87E+03	3.56E+03	2.87E+03	3.91E+03	=
7	4.22E+01	4.25E+01	4.08E+01	4.12E+01	+
8	1.13E+03	1.12E+03	6.62E+02	6.68E+02	+
9	1.06E+02	1.06E+02	1.05E+02	1.05E+02	+
10	1.57E+03	1.59E+03	1.23E+03	1.24E+03	+
11	6.76E+02	6.81E+02	5.83E+02	5.85E+02	+
12	1.08E+02	1.08E+02	1.08E+02	1.08E+02	=
13	1.80E+02	1.80E+02	1.81E+02	1.80E+02	=
14	7.29E+04	6.66E+04	6.96E+04	6.51E+04	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. F.6 CEC 2015 benchmark set results of
L–SHADE and DbL_SHADE algorithms in 50D.

	L–SHADE		**DbL_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	4,37E+03	6,31E+03	1,17E+04	1,50E+04	−
2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	=
3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	=
4	3,68E+01	3,62E+01	3,09E+01	3,12E+01	+
5	3,07E+03	3,06E+03	2,93E+03	2,90E+03	+
6	2,74E+03	2,75E+03	2,48E+03	2,85E+03	=
7	4,29E+01	4,33E+01	4,20E+01	4,21E+01	+
8	1,15E+03	1,11E+03	8,25E+02	8,28E+02	+
9	1,06E+02	1,06E+02	1,05E+02	1,05E+02	+
10	1,60E+03	1,65E+03	1,41E+03	1,46E+03	+
11	6,93E+02	6,89E+02	5,98E+02	5,97E+02	+
12	1,08E+02	1,08E+02	1,08E+02	1,08E+02	+
13	1,78E+02	1,78E+02	1,79E+02	1,78E+02	=
14	7,30E+04	6,70E+04	5,92E+04	6,37E+04	+
15	1,00E+02	1,00E+02	1,00E+02	1,00E+02	=

Tab. F.7 CEC 2015 benchmark set results of SHADE and Db_SHADE algorithms in 100D.

	SHADE		**Db_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	2.00E+05	2.20E+05	2.00E+05	2.10E+05	=
2	7.80E-07	7.70E-03	7.00E-10	1.60E-08	+
3	2.00E+01	2.00E+01	2.00E+01	2.00E+01	−
4	1.60E+02	1.60E+02	1.30E+02	1.30E+02	+
5	9.60E+03	9.60E+03	9.40E+03	9.40E+03	=
6	3.50E+04	4.00E+04	3.50E+04	3.80E+04	=
7	1.20E+02	1.30E+02	1.40E+02	1.20E+02	=
8	1.30E+04	1.40E+04	1.10E+04	1.10E+04	=
9	1.10E+02	1.10E+02	1.10E+02	1.10E+02	+
10	4.20E+03	4.20E+03	4.00E+03	4.00E+03	=
11	1.90E+03	1.90E+03	1.70E+03	1.70E+03	+
12	1.20E+02	1.20E+02	1.20E+02	1.20E+02	=
13	3.90E+02	3.90E+02	3.90E+02	3.90E+02	=
14	1.10E+05	1.10E+05	1.10E+05	1.10E+05	=
15	1.10E+02	1.10E+02	1.00E+02	1.00E+02	+

Tab. F.8 CEC 2015 benchmark set results of
L–SHADE and DbL_SHADE algorithms in $100D$.

	L–SHADE		**DbL_SHADE**		
Func.	Median	Mean	Median	Mean	Result
1	9.14E+04	1.13E+05	1.35E+05	1.57E+05	−
2	1.20E−09	3.80E−09	2.90E−09	7.90E−09	=
3	2.00E+01	2.00E+01	2.00E+01	2.00E+01	=
4	1.49E+02	1.53E+02	1.30E+02	1.32E+02	+
5	9.46E+03	9.54E+03	9.19E+03	9.28E+03	+
6	2.76E+04	3.14E+04	3.28E+04	3.54E+04	−
7	1.14E+02	1.13E+02	1.11E+02	1.10E+02	+
8	8.86E+03	9.77E+03	1.02E+04	1.11E+04	=
9	1.13E+02	1.13E+02	1.12E+02	1.12E+02	+
10	4.45E+03	4.45E+03	4.29E+03	4.31E+03	=
11	1.92E+03	1.91E+03	1.69E+03	1.71E+03	+
12	1.19E+02	1.19E+02	1.19E+02	1.19E+02	=
13	3.85E+02	3.83E+02	3.87E+02	3.87E+02	=
14	1.09E+05	1.10E+05	1.09E+05	1.09E+05	=
15	1.04E+02	1.04E+02	1.02E+02	1.02E+02	+

APPENDIX G: DISTANCE BASED PARAMETER ADAPTATION - POPULATION ANALYSIS

Tab. G.1 CEC 2015 benchmark set clustering and population diversity analysis results of SHADE and Db_SHADE algorithms in $10D$.

	SHADE			**Db_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.01E+02	1.44E+01	51	1.16E+02	1.50E+01
2	51	6.25E+01	3.44E+01	51	7.03E+01	4.07E+01
3	0	–	–	3	6.47E+02	1.64E+02
4	0	–	–	1	5.48E+02	4.05E+01
5	0	–	–	0	–	–
6	47	6.42E+02	3.08E+01	49	6.56E+02	3.13E+01
7	1	7.01E+02	3.16E+01	0	–	–
8	51	5.07E+02	1.73E+01	51	5.97E+02	1.59E+01
9	0	–	–	0	–	–
10	51	1.63E+02	1.02E+01	51	1.96E+02	9.78E+00
11	33	1.82E+02	1.22E+01	42	2.03E+02	1.13E+01
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	8.47E+01	1.16E+01	51	7.77E+01	1.11E+01
15	51	5.42E+01	5.39E+00	51	6.15E+01	5.46E+00

Tab. G.2 CEC 2015 benchmark set clustering and population diversity analysis results of SHADE and Db_SHADE algorithms in $30D$.

	SHADE			Db_SHADE		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.74E+02	9.53E+00	51	2.51E+02	1.00E+01
2	51	7.63E+01	6.80E+00	51	9.50E+01	7.62E+00
3	0	–	–	0	–	–
4	0	–	–	0	–	–
5	0	–	–	0	–	–
6	51	2.62E+02	9.11E+00	51	4.90E+02	9.38E+00
7	0	–	–	2	1.40E+03	1.12E+01
8	51	5.45E+02	1.05E+01	51	8.91E+02	1.23E+01
9	0	–	–	0	–	–
10	51	3.65E+02	8.50E+00	51	5.01E+02	8.45E+00
11	51	1.21E+02	8.02E+00	51	1.57E+02	6.82E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	1.15E+02	6.96E+00	51	1.44E+02	6.82E+00
15	51	9.94E+01	6.07E+00	51	1.20E+02	6.18E+00

Tab. G.3 CEC 2015 benchmark set clustering and population diversity analysis results of SHADE and Db_SHADE algorithms in $50D$.

	SHADE			**Db_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	2.17E+02	9.70E+00	51	3.05E+02	9.57E+00
2	51	9.07E+01	7.01E+00	51	1.09E+02	6.94E+00
3	0	–	–	0	–	–
4	0	–	–	0	–	–
5	0	–	–	0	–	–
6	51	4.72E+02	8.28E+00	51	7.72E+02	8.19E+00
7	30	4.45E+02	8.95E+00	13	7.71E+02	9.58E+00
8	50	1.22E+03	1.15E+01	49	1.37E+03	1.14E+01
9	3	8.21E+02	7.93E+00	0	–	–
10	51	4.71E+02	7.89E+00	51	5.73E+02	7.89E+00
11	51	1.27E+02	7.62E+00	51	1.63E+02	7.62E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	1.58E+02	7.30E+00	51	1.98E+02	7.21E+00
15	51	1.59E+02	7.25E+00	51	1.72E+02	7.24E+00

Tab. G.4 CEC 2015 benchmark set clustering and population diversity analysis results of SHADE and Db_SHADE algorithms in $100D$.

	SHADE			**Db_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	2.40E+02	9.25E+00	51	3.07E+02	9.58E+00
2	51	1.92E+02	8.59E+00	51	1.96E+02	8.59E+00
3	0	–	–	0	–	–
4	0	–	–	2	7.25E+02	8.47E+00
5	0	–	–	0	–	–
6	51	2.64E+03	7.50E+00	51	4.02E+03	7.32E+00
7	37	3.87E+02	8.80E+00	26	7.45E+02	8.96E+00
8	33	6.99E+03	1.08E+01	7	7.64E+03	9.26E+00
9	10	5.70E+02	9.85E+00	0	–	–
10	51	1.05E+03	8.14E+00	51	1.04E+03	8.44E+00
11	51	1.50E+02	8.92E+00	51	1.96E+02	8.78E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	5.66E+02	7.74E+00	51	5.62E+02	7.80E+00
15	51	4.39E+02	8.85E+00	51	4.30E+02	8.71E+00

Tab. G.5 CEC 2015 benchmark set clustering and population diversity analysis results of L–SHADE and DbL_SHADE algorithms in 10D.

	L–SHADE			**DbL_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.02E+02	1.34E+01	51	1.23E+02	1.46E+01
2	51	6.01E+01	2.75E+01	51	7.01E+01	3.53E+01
3	0	–	–	5	8.84E+02	1.64E+02
4	2	3.45E+02	5.01E+01	4	3.23E+02	5.25E+01
5	0	–	–	0	–	–
6	48	5.15E+02	2.83E+01	49	5.70E+02	3.07E+01
7	0	–	–	0	–	–
8	51	4.52E+02	1.46E+01	51	5.26E+02	1.32E+01
9	0	–	–	2	8.97E+02	1.46E+01
10	51	1.71E+02	9.93E+00	51	2.15E+02	9.34E+00
11	35	1.60E+02	1.09E+01	39	1.75E+02	1.12E+01
12	11	1.47E+03	9.71E+00	12	1.54E+03	8.98E+00
13	0	–	–	0	–	–
14	51	7.06E+01	7.51E+00	51	7.48E+01	7.89E+00
15	51	5.76E+01	5.43E+00	51	6.65E+01	5.43E+00

Tab. G.6 CEC 2015 benchmark set clustering and population diversity analysis results of L–SHADE and DbL_SHADE algorithms in $30D$.

	L–SHADE			**DbL_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.51E+02	9.18E+00	51	2.22E+02	9.57E+00
2	51	7.06E+01	6.67E+00	51	9.05E+01	7.18E+00
3	0	–	–	0	–	–
4	0	–	–	0	–	–
5	0	–	–	0	–	–
6	51	2.00E+02	8.80E+00	51	3.65E+02	8.63E+00
7	7	4.21E+02	1.62E+01	12	6.66E+02	2.11E+01
8	51	4.26E+02	8.17E+00	51	6.43E+02	1.33E+01
9	2	5.82E+02	7.08E+00	0	–	–
10	51	3.18E+02	8.34E+00	51	4.75E+02	8.25E+00
11	51	1.13E+02	7.25E+00	51	1.39E+02	6.82E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	1.08E+02	6.82E+00	51	1.40E+02	7.03E+00
15	51	9.36E+01	6.13E+00	51	1.12E+02	6.12E+00

Tab. G.7 CEC 2015 benchmark set clustering and population diversity analysis results of L–SHADE and DbL_SHADE algorithms in $50D$.

	L–SHADE			**DbL_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.90E+02	9.86E+00	51	3.04E+02	9.70E+00
2	51	8.48E+01	7.16E+00	51	1.06E+02	7.03E+00
3	0	–	–	1	8.86E+02	3.44E+02
4	1	4.86E+02	1.12E+01	0	–	–
5	0	–	–	0	–	–
6	51	3.02E+02	8.08E+00	51	3.69E+02	7.82E+00
7	41	2.76E+02	8.12E+00	35	5.87E+02	9.05E+00
8	51	6.35E+02	8.93E+00	51	8.43E+02	9.96E+00
9	11	5.72E+02	8.04E+00	1	9.29E+02	9.67E+00
10	51	3.78E+02	7.84E+00	51	5.60E+02	7.91E+00
11	51	1.17E+02	7.48E+00	51	1.56E+02	7.46E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	1.41E+02	7.14E+00	51	1.78E+02	7.10E+00
15	51	1.38E+02	7.08E+00	51	1.56E+02	7.06E+00

Tab. G.8 CEC 2015 benchmark set clustering and population diversity analysis results of L–SHADE and DbL_SHADE algorithms in $100D$.

	L–SHADE			**DbL_SHADE**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.90E+02	9.47E+00	51	2.79E+02	1.01E+01
2	51	1.56E+02	8.50E+00	51	1.75E+02	8.57E+00
3	1	1.48E+03	5.03E+02	1	1.34E+03	5.06E+02
4	0	–	–	1	4.44E+02	8.32E+00
5	0	–	–	0	–	–
6	51	2.02E+03	8.10E+00	51	3.50E+03	7.28E+00
7	47	2.62E+02	8.28E+00	45	5.35E+02	8.59E+00
8	51	4.95E+03	8.18E+00	51	7.19E+03	8.56E+00
9	7	7.65E+02	9.48E+00	5	7.57E+02	9.81E+00
10	51	6.13E+02	7.71E+00	51	8.20E+02	8.10E+00
11	51	1.34E+02	8.89E+00	51	1.85E+02	8.72E+00
12	0	–	–	0	–	–
13	0	–	–	0	–	–
14	51	3.88E+02	7.53E+00	51	4.34E+02	7.48E+00
15	51	3.20E+02	8.60E+00	51	3.39E+02	8.78E+00

APPENDIX H: DISH - RESULT TABLES

Tab. H.1 CEC 2015 benchmark set results of jSO and DISH algorithms in $10D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.00E+01	2.00E+01	2.00E+01	2.00E+01	=
4	2.00E+00	2.20E+00	2.00E+00	2.10E+00	=
5	1.00E+01	3.20E+01	1.50E+01	4.00E+01	=
6	2.00E+00	3.20E+00	1.40E+00	2.50E+00	=
7	2.90E−02	7.30E−02	2.90E−02	7.50E−02	=
8	4.00E−01	4.20E−01	5.00E−01	5.10E−01	=
9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
10	2.20E+02	2.20E+02	2.20E+02	2.20E+02	=
11	3.00E+02	1.70E+02	3.00E+02	2.10E+02	=
12	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
13	2.80E+01	2.80E+01	2.70E+01	2.70E+01	=
14	2.90E+03	4.70E+03	2.90E+03	4.20E+03	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. H.2 CEC 2015 benchmark set results of jSO and DISH algorithms in $30D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	8.80E−02	4.30E−01	7.20E−02	6.70E−01	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.10E+01	2.10E+01	2.10E+01	2.10E+01	=
4	1.40E+01	1.40E+01	1.40E+01	1.40E+01	=
5	1.50E+03	1.50E+03	1.60E+03	1.50E+03	=
6	2.30E+02	2.90E+02	1.80E+02	2.10E+02	+
7	2.80E+00	2.90E+00	2.60E+00	2.80E+00	=
8	5.10E+01	6.20E+01	3.60E+01	6.30E+01	=
9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	+
10	5.00E+02	5.00E+02	4.70E+02	4.70E+02	=
11	4.40E+02	4.40E+02	4.00E+02	4.20E+02	+
12	1.10E+02	1.10E+02	1.10E+02	1.00E+02	=
13	9.40E+01	9.40E+01	9.50E+01	9.50E+01	=
14	3.10E+04	3.20E+04	3.10E+04	3.20E+04	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. H.3 CEC 2015 benchmark set results of jSO and DISH algorithms in $50D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	7.70E+03	8.80E+03	8.70E+03	1.00E+04	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	2.10E+01	2.10E+01	2.00E+01	2.10E+01	=
4	4.10E+01	4.00E+01	3.40E+01	3.40E+01	+
5	3.30E+03	3.20E+03	3.30E+03	3.20E+03	=
6	2.00E+03	2.10E+03	1.80E+03	1.80E+03	+
7	4.10E+01	4.10E+01	4.10E+01	4.10E+01	=
8	6.20E+02	6.20E+02	5.00E+02	5.10E+02	+
9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
10	1.20E+03	1.10E+03	1.10E+03	1.10E+03	+
11	5.10E+02	5.10E+02	4.70E+02	4.80E+02	+
12	1.10E+02	1.10E+02	1.10E+02	1.10E+02	+
13	1.80E+02	1.80E+02	1.80E+02	1.80E+02	+
14	5.90E+04	6.00E+04	5.90E+04	6.20E+04	=
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=

Tab. H.4 CEC 2015 benchmark set results of jSO and DISH algorithms in 100D.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	9.10E+04	1.10E+05	1.30E+05	1.40E+05	−
2	5.00E−10	3.30E−09	7.00E−10	3.80E−09	=
3	2.00E+01	2.00E+01	2.00E+01	2.00E+01	=
4	1.80E+02	1.80E+02	1.50E+02	1.50E+02	+
5	10.00E+03	10.00E+03	1.00E+04	10.00E+03	=
6	2.50E+04	2.80E+04	3.20E+04	3.30E+04	−
7	1.10E+02	1.10E+02	1.00E+02	1.10E+02	=
8	7.50E+03	8.10E+03	7.20E+03	7.70E+03	=
9	1.10E+02	1.10E+02	1.10E+02	1.10E+02	+
10	3.90E+03	3.90E+03	3.70E+03	3.80E+03	=
11	1.40E+03	1.40E+03	1.20E+03	1.20E+03	+
12	1.20E+02	1.20E+02	1.20E+02	1.20E+02	+
13	4.00E+02	4.00E+02	4.00E+02	4.00E+02	=
14	1.10E+05	1.10E+05	1.10E+05	1.10E+05	+
15	1.00E+02	1.00E+02	1.00E+02	1.00E+02	+

Tab. H.5 CEC 2017 benchmark set results of jSO and DISH algorithms in $10D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
5	2.00E+00	1.80E+00	2.00E+00	1.80E+00	=
6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
7	1.20E+01	1.20E+01	1.20E+01	1.20E+01	=
8	2.00E+00	2.00E+00	2.00E+00	2.00E+00	=
9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
10	1.00E+01	3.60E+01	7.00E+00	4.40E+01	=
11	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
12	4.20E-01	2.70E+00	4.20E-01	3.30E-01	=
13	4.80E+00	3.00E+00	1.00E+00	2.10E+00	=
14	0.00E+00	5.90E-02	0.00E+00	1.20E-01	=
15	1.80E-01	2.20E-01	4.50E-01	3.10E-01	-
16	5.20E-01	5.70E-01	6.30E-01	5.60E-01	=
17	4.00E-01	5.00E-01	3.90E-01	4.40E−01	=
18	3.80E−01	3.10E−01	2.70E−01	2.70E−01	=
19	0.00E+00	1.10E−02	0.00E+00	9.20E−03	=
20	3.10E−01	3.40E−01	3.10E−01	3.40E−01	=
21	1.00E+02	1.30E+02	1.00E+02	1.40E+02	=
22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
23	3.00E+02	3.00E+02	3.00E+02	3.00E+02	=
24	3.30E+02	3.00E+02	3.30E+02	2.90E+02	=
25	4.00E+02	4.10E+02	4.00E+02	4.10E+02	=
26	3.00E+02	3.00E+02	3.00E+02	3.00E+02	=
27	3.90E+02	3.90E+02	3.90E+02	3.90E+02	=
28	3.00E+02	3.40E+02	3.00E+02	3.70E+02	=
29	2.30E+02	2.30E+02	2.40E+02	2.40E+02	=
30	4.00E+02	4.00E+02	4.00E+02	4.00E+02	=

Tab. H.6 CEC 2017 benchmark set results of jSO and DISH algorithms in $30D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
4	5.90E+01	5.90E+01	5.90E+01	5.90E+01	=
5	8.00E+00	8.60E+00	8.00E+00	8.20E+00	=
6	0.00E+00	6.00E−09	0.00E+00	1.30E−08	=
7	3.90E+01	3.90E+01	3.80E+01	3.80E+01	=
8	9.00E+00	9.10E+00	8.00E+00	8.40E+00	=
9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
10	1.50E+03	1.50E+03	1.50E+03	1.50E+03	=
11	2.00E+00	3.00E+00	2.00E+00	3.80E+00	=
12	1.40E+02	1.70E+02	1.20E+02	9.40E+01	+
13	1.60E+01	1.50E+01	1.70E+01	1.50E+01	=
14	2.10E+01	2.20E+01	2.20E+01	2.20E+01	=
15	7.80E−01	1.10E+00	9.10E−01	1.10E+00	=
16	2.60E+01	7.90E+01	2.50E+01	8.00E+01	=
17	3.50E+01	3.30E+01	3.50E+01	3.40E+01	=
18	2.10E+01	2.00E+01	2.10E+01	2.00E+01	=
19	4.10E+00	4.50E+00	3.50E+00	4.20E+00	=
20	2.90E+01	2.90E+01	2.80E+01	2.80E+01	=
21	2.10E+02	2.10E+02	2.10E+02	2.10E+02	=
22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	=
23	3.50E+02	3.50E+02	3.50E+02	3.50E+02	=
24	4.30E+02	4.30E+02	4.30E+02	4.30E+02	=
25	3.90E+02	3.90E+02	3.90E+02	3.90E+02	+
26	9.30E+02	9.20E+02	9.30E+02	9.40E+02	−
27	5.00E+02	5.00E+02	4.90E+02	4.90E+02	+
28	3.00E+02	3.10E+02	3.00E+02	3.00E+02	=
29	4.30E+02	4.30E+02	4.40E+02	4.30E+02	=
30	2.00E+03	2.00E+03	2.00E+03	2.00E+03	=

Tab. H.7 CEC 2017 benchmark set results of jSO and DISH algorithms in $50D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
4	2.90E+01	5.60E+01	2.90E+01	6.10E+01	=
5	1.60E+01	1.60E+01	1.40E+01	1.40E+01	+
6	3.10E−07	1.10E−06	4.80E−08	9.70E−08	+
7	6.70E+01	6.70E+01	6.40E+01	6.40E+01	+
8	1.70E+01	1.70E+01	1.30E+01	1.40E+01	+
9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=
10	3.20E+03	3.10E+03	3.30E+03	3.20E+03	=
11	2.90E+01	2.80E+01	2.30E+01	2.40E+01	+
12	1.70E+03	1.70E+03	1.20E+03	1.20E+03	+
13	3.70E+01	3.10E+01	1.60E+01	2.70E+01	=
14	2.40E+01	2.50E+01	2.40E+01	2.40E+01	=
15	2.30E+01	2.40E+01	2.00E+01	2.10E+01	+
16	4.80E+02	4.50E+02	4.70E+02	4.50E+02	=
17	2.60E+02	2.80E+02	2.90E+02	3.00E+02	=
18	2.40E+01	2.40E+01	2.20E+01	2.30E+01	+
19	1.40E+01	1.40E+01	1.10E+01	1.10E+01	+
20	1.10E+02	1.40E+02	1.10E+02	1.60E+02	=
21	2.20E+02	2.20E+02	2.20E+02	2.20E+02	+
22	1.00E+02	1.50E+03	1.00E+02	1.80E+03	=
23	4.30E+02	4.30E+02	4.30E+02	4.30E+02	+
24	5.10E+02	5.10E+02	5.10E+02	5.10E+02	=
25	4.80E+02	4.80E+02	4.80E+02	4.80E+02	+
26	1.10E+03	1.10E+03	1.10E+03	1.10E+03	=
27	5.10E+02	5.10E+02	5.10E+02	5.10E+02	+
28	4.60E+02	4.60E+02	4.60E+02	4.60E+02	=
29	3.60E+02	3.60E+02	3.60E+02	3.60E+02	+
30	5.90E+05	6.00E+05	5.90E+05	6.00E+05	=

Tab. H.8 CEC 2017 benchmark set results of jSO and DISH algorithms in $100D$.

	jSO		**DISH**		
Func.	Median	Mean	Median	Mean	Result
1	0.00E+00	0.00E+00	0.00E+00	8.20E−10	=
2	4.70E−07	8.90E+00	2.90E−07	1.00E+04	=
3	1.50E−06	2.40E−06	1.10E−05	1.60E−05	−
4	2.00E+02	1.90E+02	2.00E+02	2.00E+02	=
5	4.40E+01	4.40E+01	2.80E+01	2.80E+01	+
6	3.60E−05	2.00E−04	4.30E−06	5.70E−06	+
7	1.40E+02	1.50E+02	1.30E+02	1.30E+02	+
8	4.20E+01	4.20E+01	2.90E+01	2.90E+01	+
9	0.00E+00	4.60E−02	0.00E+00	3.50E−03	+
10	9.80E+03	9.70E+03	9.80E+03	9.80E+03	=
11	1.00E+02	1.10E+02	5.20E+01	5.80E+01	+
12	1.70E+04	1.80E+04	1.10E+04	1.20E+04	+
13	1.40E+02	1.50E+02	1.10E+02	1.20E+02	+
14	6.40E+01	6.40E+01	4.00E+01	4.00E+01	+
15	1.70E+02	1.60E+02	7.80E+01	8.90E+01	+
16	1.90E+03	1.90E+03	1.90E+03	1.80E+03	=
17	1.30E+03	1.30E+03	1.30E+03	1.30E+03	=
18	1.60E+02	1.70E+02	9.50E+01	9.90E+01	+
19	1.10E+02	1.10E+02	5.20E+01	5.30E+01	+
20	1.40E+03	1.40E+03	1.50E+03	1.40E+03	=
21	2.60E+02	2.60E+02	2.50E+02	2.50E+02	+
22	1.10E+04	1.00E+04	1.10E+04	1.10E+04	=
23	5.70E+02	5.70E+02	5.70E+02	5.70E+02	+
24	9.00E+02	9.00E+02	8.90E+02	8.90E+02	+
25	7.60E+02	7.40E+02	7.10E+02	7.20E+02	+
26	3.30E+03	3.30E+03	3.10E+03	3.10E+03	+
27	5.90E+02	5.90E+02	5.70E+02	5.70E+02	+
28	5.20E+02	5.30E+02	5.20E+02	5.20E+02	=
29	1.20E+03	1.30E+03	1.30E+03	1.30E+03	=
30	2.30E+03	2.30E+03	2.30E+03	2.30E+03	+

APPENDIX I: DISH - POPULATION ANALYSIS

Tab. I.1 CEC 2015 benchmark set clustering and population diversity analysis results of jSO and DISH algorithms in $10D$.

	jSO			DISH		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	1.22E+02	1.47E+01	51	1.27E+02	1.42E+01
2	51	6.55E+01	4.13E+01	51	6.72E+01	4.37E+01
3	2	1.68E+03	4.12E+00	1	2.11E+03	4.11E+00
4	51	1.54E+03	3.48E+01	51	1.48E+03	3.59E+01
5	50	1.91E+03	8.57E+01	50	2.01E+03	8.25E+01
6	51	1.10E+03	2.78E+01	51	1.07E+03	2.84E+01
7	51	1.46E+03	8.94E+00	51	1.55E+03	6.89E+00
8	51	7.20E+02	2.24E+01	51	7.04E+02	2.32E+01
9	0	–	–	0	–	–
10	51	2.06E+02	1.48E+01	51	2.17E+02	1.47E+01
11	51	2.91E+02	1.10E+01	51	2.68E+02	1.07E+01
12	0	–	–	0	–	–
13	47	2.01E+03	1.75E+01	49	2.12E+03	1.66E+01
14	51	8.35E+01	1.22E+02	51	7.83E+01	9.78E+01
15	51	5.88E+01	5.23E+00	51	6.04E+01	5.28E+00

Tab. I.2 CEC 2015 benchmark set clustering and population diversity analysis results of jSO and DISH algorithms in $30D$.

	jSO			DISH		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	2.61E+02	8.79E+00	51	2.93E+02	9.38E+00
2	51	1.13E+02	9.49E+00	51	1.31E+02	1.17E+01
3	0	–	–	0	–	–
4	51	4.02E+03	5.71E+01	51	4.06E+03	5.79E+01
5	50	5.43E+03	1.80E+02	51	5.54E+03	1.77E+02
6	51	1.93E+03	4.64E+01	51	2.40E+03	4.64E+01
7	51	4.21E+03	2.40E+01	51	4.39E+03	2.83E+01
8	51	3.26E+03	3.52E+01	51	3.30E+03	3.13E+01
9	23	4.82E+03	6.10E+00	23	4.86E+03	6.14E+00
10	51	1.22E+03	5.50E+01	51	1.25E+03	5.10E+01
11	51	1.99E+02	8.25E+00	51	2.24E+02	6.57E+00
12	0	–	–	0	–	–
13	43	6.83E+03	7.12E+01	40	7.10E+03	6.28E+01
14	51	1.94E+02	8.51E+00	51	2.21E+02	1.65E+01
15	51	1.36E+02	5.86E+00	51	1.53E+02	5.80E+00

Tab. I.3 CEC 2015 benchmark set clustering and population diversity analysis results of jSO and DISH algorithms in $50D$.

	jSO			**DISH**		
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	3.34E+02	9.11E+00	51	4.08E+02	9.21E+00
2	51	1.43E+02	6.93E+00	51	1.67E+02	7.15E+00
3	5	1.44E+04	1.15E+02	4	1.27E+04	2.49E+02
4	51	5.81E+03	6.37E+01	51	5.76E+03	6.28E+01
5	51	8.16E+03	2.40E+02	51	8.54E+03	2.43E+02
6	51	4.09E+02	1.15E+01	51	5.27E+02	1.41E+01
7	51	3.58E+03	1.51E+01	51	4.99E+03	1.99E+01
8	51	1.26E+03	2.31E+01	51	2.23E+03	3.66E+01
9	43	5.74E+03	7.16E+00	34	6.71E+03	6.86E+00
10	51	8.36E+02	9.10E+00	51	1.42E+03	1.67E+01
11	51	2.43E+02	7.35E+00	51	2.91E+02	7.35E+00
12	0	–	–	0	–	–
13	47	1.12E+04	9.15E+01	46	1.08E+04	1.00E+02
14	51	2.26E+02	6.87E+00	51	2.62E+02	8.73E+00
15	51	2.03E+02	6.74E+00	51	2.26E+02	6.69E+00

Tab. I.4 CEC 2015 benchmark set clustering and population diversity analysis results of jSO and DISH algorithms in 100D.

		jSO			**DISH**	
Func.	#runs	MCO	MPD	#runs	MCO	MPD
1	51	3.81E+02	9.84E+00	51	4.65E+02	9.85E+00
2	51	2.37E+02	8.12E+00	51	2.62E+02	8.13E+00
3	3	2.65E+04	3.25E+02	2	2.65E+04	2.86E+02
4	51	6.19E+03	2.99E+01	51	6.10E+03	2.59E+01
5	51	1.45E+04	3.49E+02	51	1.46E+04	3.48E+02
6	51	2.95E+03	7.53E+00	51	3.82E+03	7.09E+00
7	51	3.63E+03	9.48E+00	51	6.95E+03	1.26E+01
8	51	5.54E+03	8.07E+00	51	5.94E+03	8.17E+00
9	50	7.55E+03	8.58E+00	48	1.02E+04	8.35E+00
10	51	9.96E+02	7.83E+00	51	1.11E+03	7.73E+00
11	51	3.08E+02	8.51E+00	51	3.86E+02	8.46E+00
12	0	–	–	0	–	–
13	50	1.89E+04	1.73E+02	50	1.92E+04	1.71E+02
14	51	5.01E+02	7.23E+00	51	5.25E+02	7.21E+00
15	51	4.12E+02	8.39E+00	51	4.37E+02	8.36E+00