



A Collaborative Auto-Diversified Optimization Scheme

Besma Hezili

(MISC Laboratory, University of Abdelhamid Mehri Constantine 2, Constantine, Algeria
 <https://orcid.org/0000-0003-3845-2507>, besma.hezili@univ-constantine2.dz)

Hichem Talbi

(MISC Laboratory, University of Abdelhamid Mehri Constantine 2, Constantine, Algeria
 <https://orcid.org/0000-0001-7320-5312>, hichem.talbi@univ-constantine2.dz)

Abstract: We present a Collaborative Auto-Diversified Optimization Scheme (CADOS) for solving continuous and combinatorial optimization problems. CADOS aims to explore the synergy of various optimization algorithms and enhance their effectiveness and efficiency, particularly in higher-dimensional problems. It incorporates an enhanced version of the previously proposed approach Auto-Diversified Ameliorated MultiPopulation-based Ensemble Differential Evolution (AD-AMPEDE), Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES), and a local search (LS) algorithm. AD-AMPEDE has demonstrated good performance in solving continuous optimization problems. However, its competitiveness wanes in higher dimensions. CADOS improves AD-AMPEDE's detection/re-diversification processes and parameters adaptation, making it effective for higher-dimensional problem classes. To explore nearby regions during stagnation, a trust-region local search is employed. For re-diversification, CADOS utilizes both CMA-ES, known for its efficiency in complex fitness landscapes, and the Auto-Enhanced Population Diversity (AEPD) technique. We tested CADOS on the COmparing Continuous Optimizers (COCO) platform and the results demonstrated excellent performance of CADOS. In addition, to show the proposed scheme's efficacy in tackling real-world issues, we employed it to optimize the design of water distribution networks (WDS). The results we obtained underscore the remarkable competitiveness of our strategy when compared to widely recognized existing algorithms.

Keywords: Auto-Diversified Algorithms, Covariance Matrix Adaptation Evolutionary Strategy, Differential Evolution, Local Search, Multi-Population Algorithms, Premature Convergence, Re-diversification, Stagnation, Trust-Region, Water distribution systems.

Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1

DOI: 10.3897/jucs.116480

1 Introduction

Numerous real-world problems could be cast as optimization problems. An optimization problem aims to determine the optimal decision variables values that minimize or maximize an objective function while satisfying any expressed constraints [Talbi and Draa 2020]. Various types of optimization problems exist in the literature, including continuous, discrete, and combinatorial optimization.

In continuous optimization [Munoz *et al.* 2013], the decision variables vary over a defined continuous (non-discrete) search space. Many difficulties can be faced when solving instances with non-linear, non-convex, multimodal, and non-separable objective functions. In fact, the high-dimensionality of the search space, the numerous local optima,

and the sensitivity to initial conditions make it difficult to get to optimal solutions [Karaboğa *et al.* 2004, Nomanet and Iba 2005].

To overcome these hurdles, several methods have been proposed. They can be divided into two primary classes: exact and approximate methods. Exact methods ensure that the solution obtained is the best possible one. However, they can be computationally expensive and may even be impractical for higher-dimensional problems. In contrast, approximate methods provide acceptable, but not necessarily optimal solutions, in a reasonable time [Talbi and Draa 2020]. Local Search (LS) algorithms, Differential Evolution (DE) algorithms, Genetic Algorithms (GA), and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) with its variants are among the most famous approximate methods used for the continuous optimization problems.

DE algorithm is a population-based stochastic optimization method [Draa *et al.* 2011]. It generates new solutions based on mutation, crossover, and selection operations [Das *et al.* 2016]. The mutation is the main operation in DE [Draa *et al.* 2011], which produces potential solutions (vectors) by moving an existing solution using the difference between some selected solutions multiplied by a scaling factor. Roughly speaking, it makes for each individual a move following the same direction leading from one given solution to another. The algorithm's performance can be notably improved by adapting the control parameters (scaling factor F , crossover rate Cr , and population size NP). The main advantages of DE are its relatively easy implementation, the fewer control parameters, and its possible adaptability for solving integer and discrete optimization problems [Karaboğa *et al.* 2004]. In addition, experimental results demonstrate its remarkable performance compared to other widely recognized evolutionary algorithms. Despite these advantages, it has some limitations, such as premature convergence, stagnation, and slower convergence speed, especially with higher dimension problems [Nomanet and Iba 2005].

The CMA-ES algorithm has gained significant interest as a method for solving unconstrained real-parameter optimization problems [Hansen *et al.* 2003]. CMA-ES uses a combination of the covariance matrix adaptation, the step size, and the mean of the best individuals to generate new individuals [Hansen *et al.* 2003]. It is a powerful and robust method [Hansen and Ostermeier. 2001] that is capable of finding high-quality solutions. Despite that, the time and space complexities of CMA-ES increases quadratically with problem dimensionality [Varelas *et al.* 2018], which can limit its practical use for high-dimensional optimization problems.

The LS algorithms are designed to concentrate search in the vicinity of the current solution for discovering the best nearby solution that enhances the objective function value [Nomanet and Iba 2005]. However, these algorithms very often get stuck in local optima because of their weak search space exploratory potential.

It is worth noting that several methods have been proposed in the field of algorithm combinations, such as iDEaSm (Improved Differential Evolution algorithm with Surrogate model) [Awad *et al.* 2018], which presented an effective surrogate model to assist the differential evolution algorithm in generating competitive solutions during the search process. CoBiDE (Covariance Matrix Learning and Bimodal Distribution Parameter Setting Differential Evolution) [Wang *et al.* 2014], which incorporates covariance matrix learning and bimodal distribution parameter setting to enhance the performance of DE in solving global optimization problems. The goal of these combinations is to increase the effectiveness of these methods by allowing them to inspire and influence each other.

Inside this realm, we seek in the current research to develop a Collaborative Auto-Diversified Optimization Scheme (CADOS) involving dynamically and in a coherent way three methods in order to benefit from the strengths of each and effectively tackle

all classes of optimization problems.

CADOS includes a modified version of our previous variant of DE (AD-AMPEDE) [Hezili and Talbi 2022], CMA-ES, and a local search algorithm to address continuous optimization problems, especially in higher dimensions. This collaboration seeks to leverage the benefits inherent in these methods.

The AD-AMPEDE (Auto-Diversified Ameliorated MultiPopulation-based Ensemble Differential Evolution) was used to solve stagnation and premature convergence problems. It has been tested on BBOB (Black-Box Optimization Benchmarking) functions. AD-AMPEDE [Hezili and Talbi 2022] has given good results when compared to other state-of-the-art algorithms.

To improve its performance, especially in higher dimensions, we have made changes to its detection/re-diversification process. These changes are summarized as follows: in case of stagnation, we execute a local search process having the local optimum of AD-AMPEDE as a starting point. This collaboration would thus give a better local optimum. After ensuring the presence of stagnation and premature convergence using the same strategy employed in AD-AMPEDE, we execute either the CMA-ES algorithm or the diversification technique utilized in [Hezili and Talbi 2022] according to a probability value to diversify the population. The CMA-ES is used to generate new individuals starting from subpopulations best solutions, while the diversification technique allows the algorithm to produce newly-formed individuals with an increased likelihood of being in proximity to the present location. In addition, it was important to understand the role of the mutation factor and the impact of its alteration on the effectiveness of the optimization process and its calibrating. A careful analysis of the collaborative scheme's behavior was conducted and we chose two different strategies for the mutation factor generation to obtain more diversity through its values and better results, especially in higher dimension multimodal functions.

To validate our approach, we first apply it to a set of 24 BBOB functions. The COmparing Continuous Optimizers (COCO) platform [Hansen et al. 2016] is adopted to test and compare our algorithm on different instances of the 24 functions with eight state-of-the-art algorithms.

In addition, to consolidate the validation of CADOS, we opted to employ our algorithm to solve a real world optimization problem: the optimal design of water distribution networks (WDNs).

WDNs problem could be considered as a combinatorial optimization problem [Suribabu 2010]. The objective is to find the best combination of discrete decision variables including flow rates, pipe diameters, and pressure heads from a large set of possibilities, while satisfying head constraints and water demands at each node [Zheng *et al.* 2011].

This paper is consequently structured as follows: Section 2 provides an overview of DE, CMA-ES, the LS Algorithm and some pre-existing hybrid algorithms. In Section 3, we present the proposed approach. Section 4 offers a detailed presentation of experimental results and conducts comprehensive comparisons with other state-of-the-art algorithms. Section 5 applies CADOS to solve a real-world problem. Finally, Section 6 provides a conclusion and suggests some future directions..

2 Background

In this section, we first introduce the fundamental principles of DE, CMA-ES, and Local Search algorithms, followed by a comprehensive discussion of some existing hybrid algorithms.

2.1 Fundamental Algorithms

2.1.1 DE

DE [Storn and Price 1997] is a flexible algorithm that can be employed for finding the global optimum in continuous optimization problems. It generates at first N random candidate solutions within the bounds of the search space as shown below:

$$\mathbf{X}_j^i = \mathbf{X}_{minj} + rand(0, 1) * (\mathbf{X}_{maxj} - \mathbf{X}_{minj}) \quad (1)$$

After the initialization step, it proceeds with the iterative optimization process, where at each iteration, it applies one of the mutation strategies for each individual in the population. The ones listed afterward are the most commonly used strategies:

“DE/rand/1” [Storn 1996] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) \quad (2)$$

“DE/Best/2” [Storn 1996] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{best,G} + F \cdot (\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}) + F \cdot (\mathbf{X}_{r_3,G} - \mathbf{X}_{r_4,G}) \quad (3)$$

“DE/current-to-Best/2” [Storn 1996] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot (\mathbf{X}_{best,G} - \mathbf{X}_{r_2,G}) + F \cdot (\mathbf{X}_{r_3,G} - \mathbf{X}_{r_4,G}) \quad (4)$$

“DE/target-to-best/1” [Lezama *et al.* 2018] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{i,G} + F(\mathbf{X}_{best,G} - \mathbf{X}_{i,G}) + F(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}) \quad (5)$$

$r_1, r_2, r_3,$ and r_4 are randomly selected individuals indices from the population of size N , these indices are distinct from i . $\mathbf{X}_{best,G}$ represents the best solution found in generation G . F is a scalar value randomly chosen within the range $[0,1]$.

Mutation operation offers new solutions (mutant vectors), which will undergo the crossover operation to produce trial vectors as follows:

$$\mathbf{U}_{i,G}^j = \begin{cases} \mathbf{V}_{i,G}^j & \text{if } (rand_j[0, 1] \leq Cr) \text{ or } j = k \\ \mathbf{X}_{i,G}^j & \text{otherwise} \end{cases} \quad (6)$$

$rand_j[0, 1]$ is a uniform number. Cr represents the crossover probability in DE. k is a randomly selected index. It ensures the inclusion of at least one component of $\mathbf{V}_{i,G}$ into $\mathbf{U}_{i,G}$.

Based on their fitness function value, the algorithm chooses between the trial and parent vectors, and the better one will participate in the next generation. The selection operation is determined as:

$$\mathbf{X}_{i,G+1} = \begin{cases} \mathbf{U}_{i,G} & \text{if } f(\mathbf{U}_{i,G}) \leq f(\mathbf{X}_{i,G}) \\ \mathbf{X}_{i,G} & \end{cases} \quad (7)$$

2.1.2 CMA-ES

CMA-ES [Hansen and Ostermeier 1996] was proposed by Hansen and Ostermeier. It uses the multivariate normal distribution over the covariance matrix to produce the next

generation's population. Sample points x_i in the population composed of λ individuals can be determined as shown in the equation 8:

$$x_i^{t+1} \sim m^t + \sigma^t N(0, C^t), \quad i = 1, \dots, \lambda \tag{8}$$

Where m represents the weighted mean of the μ best individuals, σ is the step size, and C denotes the covariance matrix. The initial value of m is set to 1 and it is updated as in equation 9:

$$m^g = \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i x_{i:\lambda}^g \tag{9}$$

The weight w_i assigned to the i^{th} selected solution is calculated based on the difference between the logarithm of the average rank and the logarithm of the individual ranks. It is typically computed using the following formula:

$$w_i = \ln \left(\frac{\lambda + 1}{2} \right) - \ln(i) \tag{10}$$

λ is the offspring population size. i is the rank of the solution, with $i=1, 2, \dots, \lambda$.

The step size is a crucial parameter that impacts the exploration-exploitation trade-off during the search process. It regulates how much of a mutation is applied to the solution variables, which affects the algorithm's search behavior and rate of convergence. It is calculated as described below:

$$\ln \sigma^{g+1} = \ln \sigma^g + \frac{c_\sigma}{d_\sigma} \left(\frac{\| P_\sigma^{g+1} \|}{E \| N(0, I) \|} - 1 \right) \tag{11}$$

Where P_σ is the evolution path initialised to 0, $E \| N(0, I) \|$ is the expectation of the Euclidean norm of a $N(0, I)$ distributed random vector, $d_\sigma \geq 1$ is the damping parameter, c_σ is the learning rate for updating the evolution path of the covariance matrix, and $\| P_\sigma^{g+1} \|$ refers to the length of the evolution path vector at $g+1$.

In addition, the covariance matrix C is modified based on the changes in the evolution path according to the equation 12.

$$C^{(g+1)} = (1 - c_{cov}).C^{(g)} + c_{cov}.P_c^{(g+1)} (P_c^{(g+1)})^T \tag{12}$$

Where P_c is the evolution path, c_{cov} is the change rate of C . Its value is within $[0,1]$.

2.1.3 Local Search Algorithms

A Local Search (LS) algorithm is an optimization method that iteratively refines a given solution by exploring its neighborhood and progressing towards better solutions within that vicinity [Pál 2013]. Various LS algorithms have been proposed, including the Quasi-Newton Method [Broyden 1965], Interior Point Algorithm [Pál 2013], and Nelder-Mead Simplex [Pál 2013]. Despite the diversity in these approaches, all of them share the following fundamental principle:

- **Step1:** Choose an initial solution and evaluate its fitness using the problem-specific objective function.

- **Step2:** Explore the neighborhood of the current solution by making a move or transformation according to a given heuristic or rule.
- **Step3:** Evaluate the fitness of the obtained solution and update the current best solution if the new one is better.
- **Step4:** Repeat Steps 2-3 until a stopping criterion is met:
 - Maximum number of iterations is achieved.
 - A solution is found that satisfies a predetermined fitness threshold.
 - No more enhancement is possible from the current solution.

The most known issue when using local search is getting stuck at a locally optimal solution. Among the ideas employed to fix this problem, we could mention:

- Restarting the algorithm from different initial solutions, in order to explore different regions of the search space.
- Using some metaheuristics such as taboo search [Glover 1989] or simulated annealing [Laarhoven *et al.* 1987] to escape the local minimum trap.

2.2 Related Work

Recently, many hybrid algorithms have been introduced, by combining different optimization approaches including evolutionary and non-evolutionary algorithms, to enhance the optimization performance. For instance, in [Xu *et al.* 2019], the authors proposed a multi-population algorithm that combines TW-PSO (Time-Window Particle Swarm Optimization), CMA-ES, and lbest PSO, where lbest refers to the local best topology. TW-PSO is utilized to find the likely promising areas, whereas CMA-ES is used to improve the exploitation. However, the lbest PSO is working as a bridge between them.

Mohamed *et al.* [Mohamed *et al.* 2017] introduced LSHADE-SPACMA. It hybridizes a new variant of LSHADE (Linear Population Size Reduction SHADE), named LSHADE-SPA, and a modified version of CMA-ES. LSHADE is an extension of the SHADE algorithm (Success-History Adaptive Differential Evolution). LSHADE-SPA employs new semi-parameter adaptation methods to effectively adapt the scaling factor value of the DE algorithm. The modified version of CMA-ES includes the crossover operation to enhance the exploration capability of the algorithm. In LSHADE-SPACMA, the two algorithms will operate concurrently on the same population. However, during the optimization process, more individuals will be gradually assigned to the algorithm that demonstrates superior performance.

A multi-population based memetic algorithm CDELS (Competitive Differential Evolution with Local Search) has been proposed in [Mandal *et al.* 2011]. It aims to solve some real world optimization problems by combining a competitive variant of DE with Solis Wet's algorithm. DE is utilized for better exploration, while Solis Wet's is used as local search to improve exploitation.

For solving nonlinear-regression problems a hybrid evolutionary, a clustering process, and a local-search algorithm has been proposed in [Martínez-Estudillo *et al.* 2006]. The clustering procedure enables the selection of individuals who represent various areas of the search space. These individuals are the ones who are susceptible to local optimization.

This increases the likelihood that the optimised individuals will converge on several local optima.

In [Okulewicz and Zaborski 2021], the authors evaluated the SHADE-LM algorithm that hybridizes the SHADE algorithm with a model-based optimization. In this collaboration, SHADE is granted access to valuable data samples thoughtfully generated by the model-based optimization technique. The key to this synergy lies in the employment of mathematical models meticulously tailored to capture the intricate nuances of square functions, which are intelligently fitted to the current population.

A new variant of a hyper-heuristic framework: Generalized Self-Adapting Particle Swarm Optimization (PSO) with samples archive (M-GAPSO) has been introduced in [Okulewicz *et al.* 2022]. M-GAPSO incorporates PSO, DE and model based optimizers. The proposed hybrid method's effectiveness in addressing black-box optimization has been evaluated on 24 continuous benchmark functions from the COCO test set and functions from the CEC-2017 test set.

A hybrid genetic algorithm-particle swarm optimization (GA-PSO) with fuzzy adaptive acceleration coefficients was introduced in [Noronha 2022]. The proposed approach aimed to provide an efficient optimization method that achieves fast and non-premature convergence in the search process by performing a parametric adaptation of the acceleration coefficients of PSO using a fuzzy system.

In [Boks *et al.* 2020], the authors introduced a novel hybridization method called PSODE, which integrates particle swarm optimization (PSO) and differential evolution (DE) algorithms. This hybrid approach involves creating two populations with variation operators from PSO and DE and selecting individuals from these populations. The efficiency of PSODE is evaluated based on the value of the best objective function (bOFV) and the execution time (eTime).

As we have seen above, various hybrid algorithms have emerged to augment the performance of foundational optimization techniques. Inside this realm, CADOS stands out as a novel approach designed to foster effective collaboration. It emphasizes auto-diversification by leveraging the strengths inherent to each constituent algorithm. This strategic integration enhances the overall performance of the scheme, particularly when addressing a broader range of BBOB instances in higher dimensions.

3 Proposed Approach

3.1 Auto-Diversified Ameliorated Multi-Population-based Ensemble Differential Evolution

Numerous studies have been presented to improve the evolutionary process of DE by dividing the whole population into sub-populations, each of which has its mutation strategy. In this field, we introduced a multi-population differential evolution with an automatic re-diversification process (AD-AMPEDE). AD-AMPEDE [Hezili and Talbi 2022] aims to establish an effective collaboration between sub-populations using our previously proposed approach (AMPEDE) [Hezili and Talbi 2021] and to deal with stagnation and premature convergence issues using a detection/re-diversification process. To balance exploitation and exploration, we have used $DE/rand$, $DE/target-to-best/1$, and $DE/current-to-mpbest/1$ mutation strategies. $DE/current-to-mpbest/1$ represents a novel mutation strategy introduced in this work to complement existing approaches and achieve a more balanced trade-off between exploitation and exploration in the evolutionary process. The $DE/current-to-mpbest/1$ strategy is described as follows [Hezili and Talbi 2021]:

$$\mathbf{V}_{i,G} = \mathbf{X}_{i,G} + F(\text{mpbest}_G - \mathbf{X}_{i,G}) + F(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}) \quad (13)$$

Where mpbest_G denotes the mean of the best solutions (pbest), corresponding to the top 20% of the entire population.

The objective function values' mean is employed to detect whether the population has stagnated in a non-optimal region or not, as shown in the following:

$$UM = \begin{cases} UM + 1, & \text{if } |\text{mean}_g - \text{mean}_{g-1}| < \epsilon, \quad \epsilon = 10^{-8} \\ UM & \text{otherwise} \end{cases} \quad (14)$$

After confirming the stagnation, we have employed the coefficient of variance (CV) to verify the loss of diversity within the population, indicating with a high probability that the population is being trapped in a local optimum. The CV is calculated as in equation (15):

$$CV = \frac{\sigma}{\mu} \quad (15)$$

(σ) denotes the standard deviation of the population, and (μ) is the population's mean. Based on experimental results, it has been observed that AD-AMPEDE provides a significant performance enhancement when compared to AMPEDE, especially for multimodal functions with weak global structure.

3.2 Collaborative Auto-Diversified Optimization Scheme (CADOS)

To enhance the efficiency of AD-AMPEDE in higher dimensions, we opted to utilize a collaborative scheme, including a modified version of AD-AMPEDE, CMA-ES, and LS. Each one works in specific conditions and communicates the result of its work to another. The evolution process of CADOS starts with executing the AD-AMPEDE steps until stagnation detection. At that moment, the local search trust-region algorithm [Conn *et al.* 2000] is launched with the top-performing individual as a starting point to effectively explore its nearby region. If the local search returns a better individual than the one given by AD-AMPEDE, we consider it as the new local optimum. Otherwise, we keep the old one. The main idea behind the trust-region methods is to use a quadratic model to approximate the objective function within a specific area. The trust-region method then solves the subproblem as follow:

$$\min m_k(p_k) = f_k + g_k^T p_k + 1/2 p_k^T B_k p_k \quad (16)$$

Where f , g , and B represent respectively the objective function, its gradient, and its Hessian at x_k .

The step p_k is then updated as defined in equation 17:

$$p_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \quad (17)$$

Where $m_k(0)$, and $m_k(p_k)$ denote respectively the model function value at x_k and $x_k + p_k$. If p is greater than a certain threshold, the trust region will be widened in the next iteration. Otherwise, its size will be reduced in the next iteration. The trust-region method is detailed in Algorithm 1.

Algorithm 1 Trust-region method

```

input: starting point  $x_0$ 
for each iteration  $k$  do
  solve the sub-problem of equation (13) to get the improvement step
  evaluate  $p_k$  using equation (14)
  if  $p_k > \text{threshold}$  then
     $\lambda_{k+1} \leftarrow t_1 \cdot \lambda_k$  //  $t_1$  is a factor  $> 1$ ,  $\lambda_k$  is the radius of the region
    if  $\lambda_{k+1} > \lambda_M$  then //  $\lambda_M$  is the upper bound of the trust region size
       $\lambda_{k+1} \leftarrow \lambda_M$ 
    end if
  else
     $\lambda_{k+1} \leftarrow t_2 \cdot \lambda_k$  //  $t_2$  is a reduction factor  $< 1$ 
  end if
   $x_0 \leftarrow x_k + p_k$ 
end for

```

Upon confirming stagnation, we should ensure the loss of population diversity, through the calculation of the coefficient of variation as in AD-AMPEDE. Once affirmed stagnation and premature convergence, a re-diversification process is called. We have adopted two different manners to do the re-diversification according to a probability value.

1. The first one aims to create new solutions around every subpopulation's best solution, by executing the CMA-ES algorithm. Before running the CMA-ES, we call first the trust-region method to possibly improve the subpopulations' current best solutions. These local solutions are combined with randomly selected individuals from the whole population to form the initial solution vector for CMA-ES. In our proposal, the weighted mean of the u best individuals is initiated to the weighted mean of the solution vector instead of 1. When CMA-ES budget is consumed (budget=100), the solutions are sorted based on their objective function values. The three best-performing individuals are sent to AD-AMPEDE and will replace the subpopulations best solutions. The other CMA-ES solutions will replace random individuals of the AD-AMPEDE population. The implementation of CMA-ES is described in Algorithm 2.

Algorithm 2 CMA-ES

```

1: Input: Xbest, Xrand // 3 best-performing individuals and  $\lambda - 3$  randomly selected
   individuals;
2: Set parameters:  $\sigma = 1$ ;  $min_\sigma = 10^{-1}$ ;  $\lambda = 13$ ;
3: Calculate weights based on ranks:  $w_i = \log \frac{\lambda+1}{2} - \log(i)$ ;
4: // Parameter Adaptation:
5:  $cc = \frac{4}{N+4}$ ;  $ccov = \frac{2}{(N+\sqrt{2})^2}$ ;  $cs = \frac{4}{N+4}$ ;  $damp = \frac{1}{cs} + 1$ ;
6:  $\chi_N = \sqrt{N} \left(1 - \frac{1}{4N} + \frac{1}{21N^2}\right)$ 
7: //Initialize dynamic strategy parameters and constants:
8:  $B = I_N$ ;  $D = I_N$ ; //Identity matrices
9:  $BD = B * D$ ;  $C = (BD)(BD)^t$ ;
10: // Population Initialization:
11: Concatenate Xbest and Xrand to form X;
12: Evaluate the fitness of each individual in  $X_k$  using the function  $f(X_k) =$ 
   cocoEvaluateFunction( $f, X_k$ );
13: // Evolution:
14: Sort the population based on fitness;
15: Calculate the weighted mean of the best-performing individuals:  $xmeanw$ ;
16: Generate new offspring: Y by adding random noise to the mean(X);
17: Update the population with the new offspring: Y;
18: //Evaluate the fitness of the new population:
19:  $f(Y_k) = \text{cocoEvaluateFunction}(f, Y_k)$ ;
20: //Adaptation of Strategy Parameters:
21: Calculate ymeanw; // mean of the best-performing individuals;
22: //Update pc and ps based on ymeanw:
23:  $pc = (1 - cc) * pc + \sqrt{cc * (2 - cc)} * cw * (BD * ymeanw)$ ;
24:  $ps = (1 - cs) * ps + \sqrt{cs * (2 - cs)} * cw * (B * ymeanw)$ ;
25: //Update  $\sigma$  based on  $ps$  and  $\chi_N$ :
26:  $\sigma = \sigma * e^{\frac{\|ps\| - \chi_N}{\chi_N}} / damp$ ;
27: Ensure symmetry of C:
28:  $C = (1 - ccov) * C + ccov * pc * pc^t$ ;
29: //Perform eigen decomposition of C to get B and D:
30:  $[B, D] = \text{eig}(C)$ ;
31: //Ensure numerical stability by scaling C and D:
32:  $tmp = \frac{\max(\text{diag}(D))}{10^{14}} - \min(\text{diag}(D))$ ;
33:  $C = C + tmp * I_N$ ;  $D = D + tmp * I_N$ ;
34:  $D = \sqrt{\text{diag}(D)}$ ;
35: //Update BD:
36:  $BD = B * D$ ;
37: // Termination Condition:
38: if  $(\sigma * \min(\text{diag}(D))) < min_\sigma$  then increase  $\sigma$ ;
39: Repeat the evolution until the condition  $counteval < 100$  is met;
40: // Results Extraction:
41: Sort the final population based on fitness;
42: Extract the three best-performing individuals:  $Vbest$  and their fitness values:
    $VFbest$ ;
43: Extract the random individuals:  $Xrand$  and their fitness values:  $Vrand$ ;

```

2. The second manner consists in using a re-diversification strategy employed in AD-AMPEDE, which enables the algorithm to exploit the local area more extensively if it has not been thoroughly investigated, while also exploring other promising regions. The detailed AEPD re-diversification method is described in Algorithm 3, while Algorithm 4 determines the diversification strategy for our population.

Algorithm 3 The AEPD re-diversification strategy

```

1: Generate a random index  $nb$  between 1 and  $NP$ ;
2: for  $i = 1$  to  $nb$  do
3:   Regenerate  $X_{i,G+1}$  of the next generation;
4: end for
5: Calculate the fitness value of  $X_{i,G+1}$ ;
6: if the fitness value of the regenerated  $X_{i,G+1} \leq$  the fitness value of the original
    $X_{i,G+1}$  then
7:   Substitute the original  $X_{i,G+1}$  by the regenerated  $X_{i,G+1}$ , and the fitness value
   of the original  $X_{i,G+1}$  by the fitness value of the regenerated  $X_{i,G+1}$ 
8: else
9:   Use the original  $X_{i,G+1}$  for the next generation;
10: end if

```

Algorithm 4 Diversification technique

```

1: Compute CV of the entire population using the equation (12);
2: Generate a random number  $p \sim U(0, 1)$ ;
3: if  $CV \leq 0.15$  then
4:   if  $p < 0.5$  then
5:     Execute the CMAES algorithm (Algorithm 2);
6:   else
7:     Execute the AEPD re-diversification strategy (Algorithm 3);
8:   end if
9: end if

```

The Cr parameter is generated following a normal distribution as in [Hezili and Talbi 2021]. However, the F parameter is generated either by following a Cauchy distribution or by selecting a random value within the range of $[0.3, 0.63]$. This deliberate approach enhances the diversity in F 's values and subsequently yields improved outcomes, particularly when addressing Adequate Global Structure for multimodal functions with higher dimensions.

Algorithm 5 describes the collaborative scheme based on the algorithms described above. In addition, Figure 1 displays an overview illustrating the CADOS methodology, providing a visual understanding of the employed approach.

Algorithm 5 pseudo code of CADOS

```

1: Set  $\mu CR_j = 0.5$ ,  $\mu F_j = 0.5$ ,  $\Delta f_j = 0$  and  $\Delta f_{es_j} = 0$  for each  $j = 1, \dots, 4$ ;
2: Initialize,  $NP$ ,  $ng$ , for each  $j = 1, \dots, 4$ ;
3: Initialize, the  $pop$  randomly distributed in the solution space;
4: Initial  $\lambda_j$  and set,  $NP_j = \lambda_j \cdot NP$ ;
5: Randomly partition  $pop$  into  $pop_1$ ,  $pop_2$ ,  $pop_3$  and  $pop_4$  with respect to their sizes.;
6: Randomly select a sub-population  $pop_j$  ( $j = 1, 2, 3$ ) and combine  $pop_j$  with  $pop_4$ . Let  $pop_j = pop_j \cup pop_4$  and  $NP_j = NP_j + NP_4$ ;
7:  $moy = \text{mean}(pop)$ ;
8: Set  $g = 0$ ;
9: while  $g \leq MaxG$  do
10:    $g = g + 1$ ;
11:    $PM = moy$ ;
12:   for  $j=1 \rightarrow 3$  do
13:     Calculate  $\mu CR_j$  and  $\mu F_j$ ;
14:     Calculate  $CR_{i,j}$  and  $F_{i,j}$  for each individual  $X_i$  in  $pop_j$ ;
15:      $nb = \text{rand}()$ ;
16:     if  $nb < 0.5$  then
17:       Calculate  $F_{i,j}$  for each individual  $X_i$  in  $pop_j$  using a Cauchy distribution;
18:     else
19:       Calculate  $F_{i,j}$  for each individual  $X_i$  in  $pop_j$ , where  $F_{i,j} = \text{rand}()/3 + 0.3$ ;
20:     end if
21:     Perform the  $j$  th mutation strategy and related crossover operators over subpopulation  $pop_j$ ;
22:     Set  $SCR_{i,j} = \emptyset$ ;  $SF_{i,j} = \emptyset$ ;
23:     end for
24:     for  $i=1 \rightarrow NP$  do
25:       if  $f(X_{i,g}) \leq f(u_{i,g})$  then
26:          $X_{i+1,g} = X_{i,g}$ ;
27:       else
28:          $X_{i+1,g} = u_{i,g}$ ;  $CR_{i,j} \rightarrow SCR_{i,j}$ ;  $F_{i,j} \rightarrow SF_{i,j}$ ;
29:       end if
30:     end for
31:      $pop = \bigcup_{j=1 \dots 3} pop_j$ ;
32:     if  $\text{mod}(g, ng) = 0$  then
33:        $k = \arg \left( \max_{1 < j \leq 3} \left( \frac{\Delta f_j}{ng \cdot NP_j} \right) \right)$ ;
34:        $\Delta f_j = 0$ ;
35:       Randomly partition  $pop$  into  $pop_1$ ,  $pop_2$ ,  $pop_3$  and  $pop_4$ ;
36:       Let  $pop_k = pop_k \cup pop_4$  and  $NP_k = NP_k + pop_4$ ;
37:     end if
38:      $moy = \text{mean}(X)$ ;
39:     if  $moy = PM$  then
40:        $UM = UM + 1$ ;
41:     end if
42:     if  $UM = NP$  then
43:       execute the trust-region method (Algorithm 1);
44:     if the Trust-region method returns better individual than the one given by AD-AMPEDE
45:     then
46:       Substitute the local optimum with the one derived from the Trust-region method.
47:     end if
48:     execute the diversification process using in Algorithm 4;
49:   end if
50: end while

```

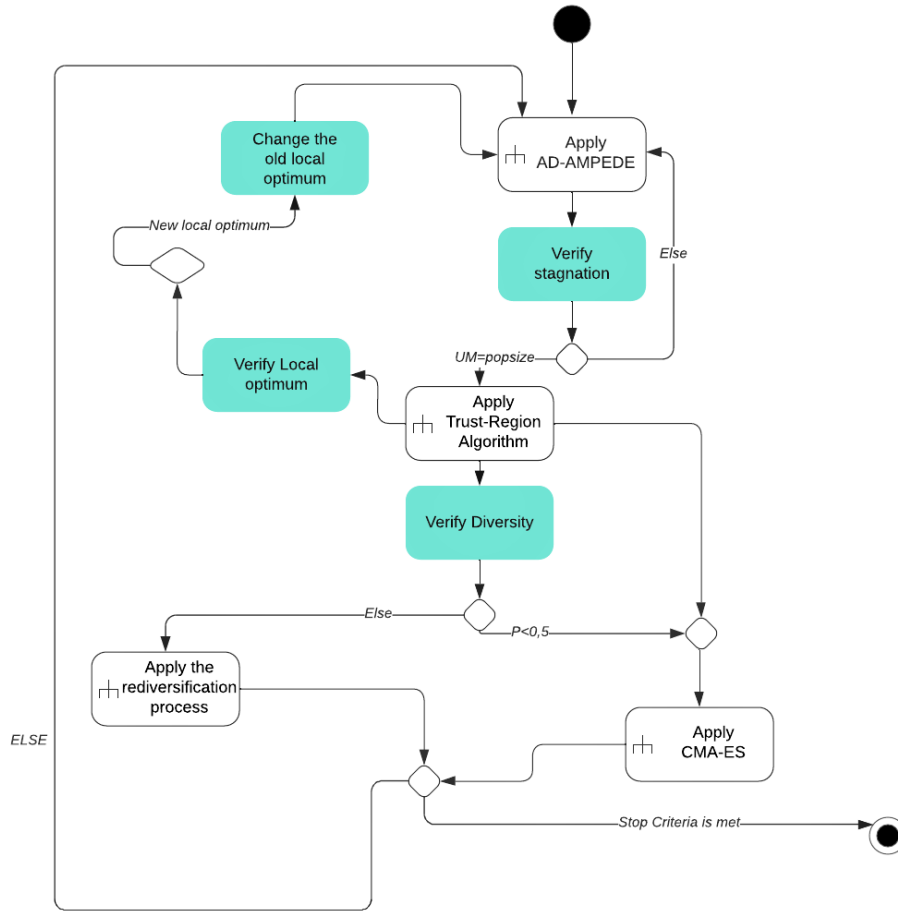


Figure 1: CADOS overview

3.3 Algorithm’s Complexity

The complexity of DE can vary depending on several factors, including the dimensionality of the problem, population size, and the number of generations. Initially, the algorithm randomly generates candidate solutions within the search space, a step typically with a time complexity of $O(\text{population size} * \text{dimensionality of the problem space})$. Subsequently, DE executes its main loop for a fixed number of generations (max-generations), wherein each generation involves mutation, crossover, and selection operations. The time complexity of each generation depends on the population size and the complexity of the objective function. Assuming the objective function evaluation dominates the computational cost, the time complexity per generation can be approximated as $O(\text{population size} * \text{complexity of objective function})$.

As the dimensionality increases, the computational cost of DE grows significantly faster, making it less efficient for high-dimensional problems. That is primarily due to

the increased number of function evaluations required to explore the entire search space, resulting in a higher computational burden.

CMA-ES involves the parameters adaptation strategy, such as the covariance matrix, the evolution path, and the step size, along with the generation of new candidate solutions. The primary computational effort in CMA-ES arises from the iterative process of generating new candidate solutions, evaluating their fitness, and updating the mean and covariance matrix accordingly. In higher dimensional problems, the covariance matrix becomes more complex, leading to increased computational complexity. Furthermore, the computational complexity of function evaluation for each individual, generated from a multivariate Gaussian distribution determined by the mean vector and covariance matrix, also increases.

The complexity of the gradient-based local search method (fmincon) [Pál 2013] depends on various factors such as the problem dimensionality, the complexity of the objective function, and the chosen optimization method (for instance, interior-point, trust-region reflective, etc.). Each iteration in fmincon requires several steps, such as updating the trust region radius, solving the optimization subproblem, and maybe changing other parameters like the step size. The complexity of solving the optimization subproblem depends on the choice of optimization algorithm and the problem dimensionality.

After considering the complexities of DE, CMA-ES, and fmincon, it becomes evident that each algorithm has its complexity factors and weaknesses, particularly in high-dimensional optimization problems. By combining AD-AMPEDE (a variant of DE) to efficiently explore the search space, CMA-ES to address challenges with the complexity of the covariance matrix, and fmincon to provide efficient local search capabilities, our proposed algorithm (CADOS) effectively mitigates the individual weaknesses of each method while capitalizing on their strengths. To improve the efficiency of AD-AMPEDE and reduce computational complexity, CADOS strategically employs fmincon only in stagnation scenarios to achieve better convergence. Additionally, CADOS utilizes CMA-ES with fewer iterations, particularly during loss of diversity scenarios. This approach helps mitigate the complexity of fmincon, the computational burden associated with the covariance matrix in CMA-ES, and reduces the number of function evaluations in AD-AMPEDE, thereby enhancing overall performance.

4 Experimental Results

In this section, we initially introduce the COCO platform, which serves as the evaluation ground for our approach. Then, we outline our parameter settings to provide context for our experiments. Finally, we present our results and engage in a detailed discussion, comparing and analyzing our findings.

4.1 COCO platform

The efficiency of our algorithm is assessed within the COmparing Continuous Optimizers (COCO) platform [Hansen et al. 2016]. The execution is realized on the BBOB test suite, which includes twenty-four noiseless test functions [Talbi and Draa 2020]. Additionally, the COCO platform provides tools for processing, displaying and comparing data produced by one or more optimizers. Its efficacy is evaluated based on the runtime required to achieve particular targets. A target represents the algorithm's best solution that gets closer to the problem's best-known solution with a precision equal to 10^{-8} .

4.2 Parameter setting

In the experiments, the following parameter values were empirically chosen after many series of tests:

- Feasible solutions are within the range $[-5,5]$ for each solution.
- The population size was equal to 150.
- The mean value of the crossover probability (μCR_j)=0.5.
- The mean value of the scale parameter (μF_j)=0.5.
- The generation gap (ng)=30.
- The ratio between indicator population and the total population ($\lambda_1 = \lambda_2 = \lambda_3$)=0.3.

In order to get a fair comparison, each algorithm runs over 15 instances across different dimensions until a stop criterion is met: either attaining the maximum number of function evaluations ($10^5 \cdot D$, D is the problem dimension) or getting a value less than 10^{-8} close to the target value.

4.3 Results and Discussion

CADOS has been evaluated against a set of the finest state-of-the-art algorithms:

- The basic DE [Pošík and . 2012].
- The PSO [El-Abd *et al.* 2009].
- The hybrid DE-BFGS (Broyden-Fletcher-Goldfarb-Shanno) [Voglis *et al.* 2012].
- The CMA-ES [Hansen *et al.* 2019]
- The adaptive differential evolution with optional external archive (JADE) [Zhang *et al.*2009].
- The AD-AMPEDE [Hezili and Talbi 2022]
- The GA [Nicolau 2009]
- The fmincon [Pál 2013]

The performance of the different algorithms has been tested for dimensions 10D and 20D. Graphical results are provided with the Empirical Cumulative Distribution Function (ECDF) of simulated runtimes in Figures 2 and 3. The ECDFs represent the comparison of convergence rates and precision through the runtime distributions. The numerical results are shown in Figure 4 as the average Runtime (aRT). The average Runtime is calculated by dividing the total number of function evaluations by the number of successes to get the best value recorded during the BBOB 2009 competition.

From Figure 2, it can be observed that CADOS demonstrates superior performance when compared to all other algorithms across all 24 functions in 10D. It achieves the target function value in approximately 89% of instances. It surpasses all other algorithms in dealing with multimodal functions (f15-f19), where it obtained the desired function value in roughly 78% of the cases.

CADOS achieved a success rate of 100% in solving the first three categories(f1-f14) in 10D.

We notice from Figure 3 that our CADOS algorithm has reached the optimal function value in about 76% of the 20D cases. In addition, CADOS has excellent performance compared to CMAES in separable functions and low or moderate conditioning functions, where it solved all the instances of these two categories (it reached a 100% success rate). For 20D, CADOS has achieved a better convergence rate than DE-BFGS, DE, PSO, AD-AMPEDE, fmincon, and GA in separable function, low or moderate conditioning functions, unimodal for high conditioning functions, and weak global structure for multimodal functions.

The numerical results presented in Figure 4 confirm the good performance of our CADOS algorithm in higher dimensions (20D). In separable functions (especially f3 and f4), CADOS outperformed AD-AMPEDE, CMAES, DE, PSO, fmincon, and GA. Moreover, in the context of adequate global structure with multimodal, specifically in the case of function f15, CADOS has achieved convergence to a solution with a precision of 10^0 , surpassing all other state-of-the-art methods. Additionally, in weak global structure for multimodal functions (namely f23) CADOS converges towards a solution with a precision of 10^{-2} , outperforming all other state-of-the-art methods. For the remaining functions, numerical results showing the competitiveness of the proposed approach are provided in the appendix.

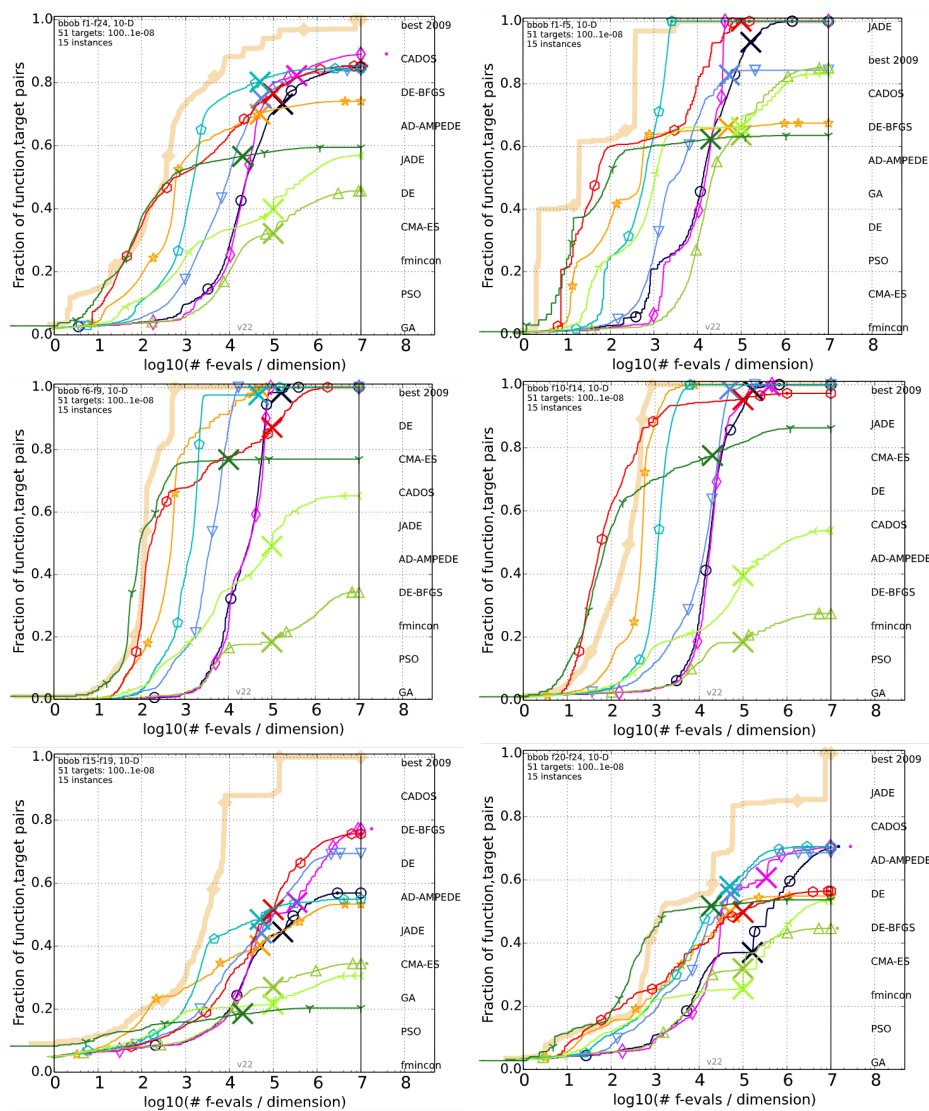


Figure 2: Convergence rate of CADOS compared with other state-of-the-art algorithms in 10-D

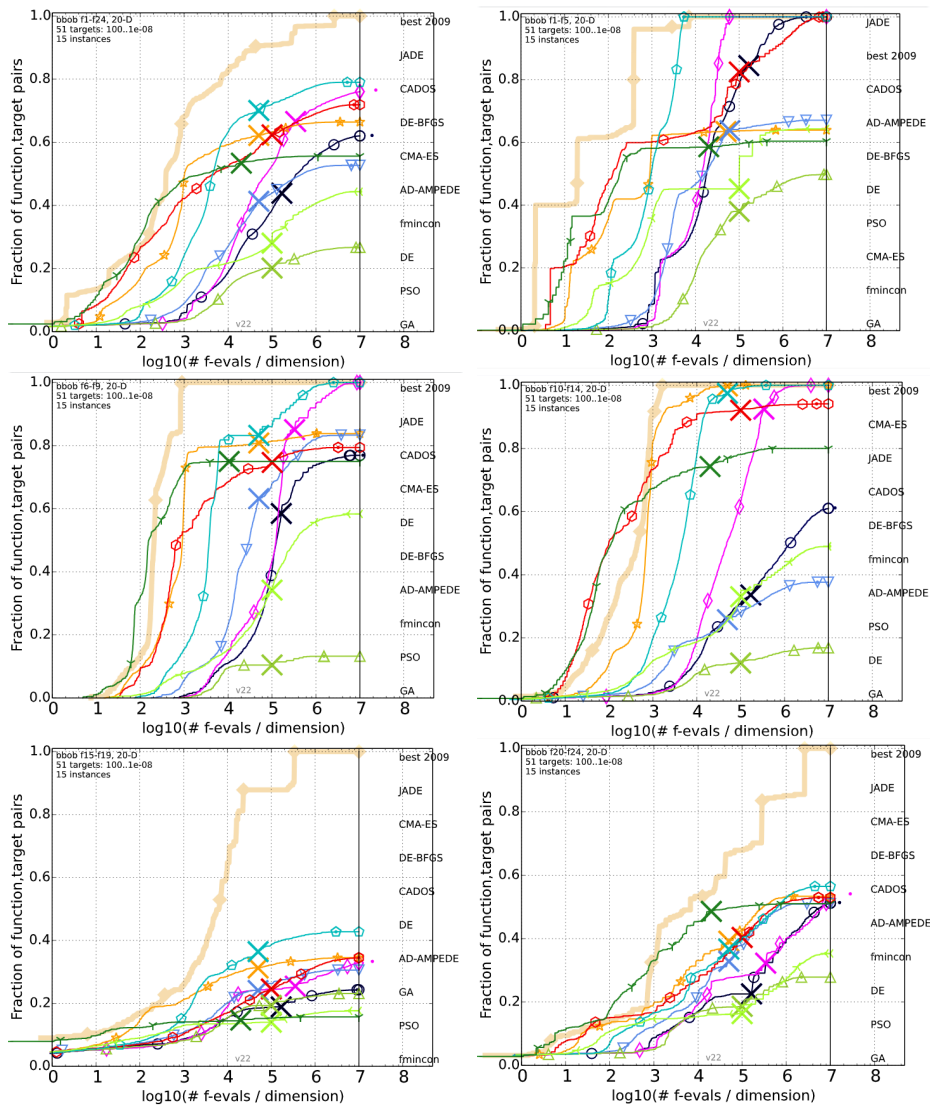


Figure 3: Convergence rate of CADOS compared with other state-of-the-art algorithms in 20-D

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f3, 20-D	5066	7626	7635	7637	7643	7646	7651	15/15
AD-AMPEDE	89 (54)	166 (165)	218 (79)	257 (152)	355 (242)	463 (239)	743 (428)	7/15
CADOS	53 (6)	62 (8)	73 (13)	74 (5)	74 (18)	74 (10)	74 (12)	15/15
CMA-ES	638 (597)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	99 (35)	2497 (5356)	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	56 (20)	132 (73)	181 (264)	181 (110)	181 (141)	181 (82)	191 (137)	13/15
GA	29 (7)	3709 (2231)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	6.4 (0.3)*4	6.0 (0.2)*4	6.8 (0.3)*4	7.6 (0.2)*4	8.3 (0.2)*4	10 (0.2)*4	11 (0.1)*4	15/15
PSO	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f4, 20-D	4722	7628	7666	7686	7700	7758	1.4e5	9/15
AD-AMPEDE	253 (351)	369 (447)	489 (490)	602 (405)	901 (992)	1165 (1961)	113 (101)	3/15
CADOS	72 (7)	79 (21)	106 (36)	106 (48)	106 (37)	105 (45)	5.8 (2)	15/15
CMA-ES	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	146 (76)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	143 (114)	589 (381)	3893 (5616)	3883 (2540)	3876 (6827)	3848 (2904)	212 (163)	1/15
GA	65 (5)	3751 (5313)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	8.0 (0.3)*4	7.0 (0.1)*4	8.0 (0.2)*4	8.8 (0.3)*4	10 (0.1)*4	11 (0.2)*4	0.71 (9e-3)*4	15/15
PSO	5940 (3494)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f15, 20-D	30378	1.5e5	3.1e5	3.2e5	3.2e5	4.5e5	4.6e5	15/15
AD-AMPEDE	∞	∞	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	206 (266)	341 (254)	∞	∞	∞	∞	$\infty 7e6$	0/15
CMA-ES	37 (44)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	28 (16)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	39 (43)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
PSO	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f23, 20-D	3.0	1614	67457	3.7e5	4.9e5	8.1e5	8.4e5	15/15
AD-AMPEDE	1.6 (2)	754 (2037)	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	2.0 (2)	94 (4)	451 (1234)	281 (383)	∞	∞	$\infty 7e6$	0/15
CMA-ES	3.1 (3)	571 (530)	63 (46)	∞	∞	∞	$\infty 1e6$	0/15
DE	2.2 (1)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	2.3 (3)	7.0 (8)	54 (40)	∞	∞	∞	$\infty 2e6$	0/15
GA	1.8 (2)	4570 (4650)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	2.2 (3)	131 (48)	∞	∞	∞	∞	$\infty 1e6$	0/15
PSO	2.4 (1)	1554 (1582)	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	11 (18)	4.4 (6)	90 (121)	∞	∞	∞	$\infty 4e5$	0/15

Figure 4: Numerical results in 20D

Δf denotes the difference to the optimal function value. #succ is the number of trials that reached the final target $f_{opt} + 10^{-8}$. Best results are in **bold**. The second row (labelled with the function and the dimension) gives, for each target $f_{opt} + 10^n, n \in \{1, 0, -1, \dots, -7\}$, the best *aRT* obtained for the considered function/dimension by the

one among all the solvers participating in the BBOB 2009 competition. The semi-interdecile range (half the difference between the 10 and 90%-tile) of runtimes is given in brackets as a dispersion measure.

5 Case Study

In this section, we present the water distribution networks (WDNs) optimal design problem. In addition, we share our experimental results and compare how well our method works compared to other known algorithms.

5.1 The optimal design of WDN

WDNs are a major component in the urban infrastructure system that requires significant expenditure for construction. It is a hydraulic system composed of pipes, reservoirs, tanks, pumps, valves, etc. An optimal design of the WDN is recommended for any agency. However, it is a very complex task because of the discontinuous nature of pipe diameters' availability and the nonlinear correlation between head loss and flow. The flow is mathematically determined as follows:

$$\sum_{i \in in, n} Q_i = \sum_{j \in out, n} Q_j + ND_n \quad \forall n \in NN \quad (18)$$

Q denotes the flow within the pipes, ND_n represents the demand at node n . in, n and out, n represent the sets of pipes entering and exiting node n . NN is the node set.

The Hazen-Williams head loss for the pipe i with connected nodes j and k is calculated using the following equation:

$$H_j - H_k = \frac{\alpha L_i Q_i |Q_i|^{0.852}}{C_{HW,i}^{1.852} D_i^{4.87}} \quad \forall j \in NP \quad (19)$$

where NP is the number of pipes, D_i represents the diameter of the pipe i , $C_{HW,i}$ is the Hazen-Williams coefficient, L_i denotes the length of pipe i , and α is the conversion factor related to the units employed for computation (in this context, $\alpha = 10.667$).

In addition, the objective of optimal design for the WDN is to minimize the network cost while meeting the necessary water requirements and head restrictions at each node. Numerous optimization methods have been introduced to tackle this issue, including deterministic optimization techniques and evolutionary algorithms. Due to the combinatorial nature of the WDN design, deterministic optimization algorithms provide typically local optimal solutions only, and deal hardly with large-size instances.

Metaheuristic algorithms (MAs) have become a good alternative for optimizing the design of WDN [Zheng *et al.* 2011]. Furthermore, it has been observed that MAs outperform deterministic approaches since they directly address the discrete search space, increasing the likelihood that they will find the global optimal solution [Zheng *et al.* 2011]. Many EAs have been presented to optimize the design of WDN, for instance GA [Simpson *et al.* 1994], Shuffled Frog Leaping Algorithms (SFLA) [Eusuff *et al.* 2003], Simulated Annealing (SA) [Cunha *et al.* 2001], MultiObjective Simulated Annealing with new Generation and Reannealing procedures (MOSA-GR) [Cunha and Marques 2020], Non-dominated Sorting Genetic Algorithm II (NSGA-II) [Wang *et al.* 2019], and DE [Suribabu 2010]. In this field, we aim to employ our approach linked with the EPANET-Matlab Toolkit [Sabzkouhi *et al.* 2022] to test its efficiency in dealing with the

optimal design for the WDN as a real-world optimization problem. EPANET is utilised to examine the network and evaluate the pressure at each node that must satisfy specific nodal pressure requirements.

Pipe diameter values are considered candidate solutions during the evolution process of CADOS. However, due to its discrete nature and the continuous search space of our algorithm, we have to design a discretization method. We have chosen to keep manipulating real values in all steps until the selection one. The individuals pipe diameters real values are converted to discrete commercial sizes.

The commercial size could be deterministically obtained through rounding, but in our approach, we utilized a probabilistic technique to enhance diversity in our research while converting, i.e. the same real value vector could generate different pipe sizes at different iterations. To do so, a normalized Euclidean distance is firstly calculated between the continuous pipe value x and the closest lower commercial pipe size x_l using the closest upper commercial pipe size x_u . The probability to select one pipe size among the two is proportional to the calculated distance.

In practice, the normalized distance is $dl = (x - x_l)/(x_u - x_l)$. We generate a random value $r \in [0, 1]$. If $r < dl$ then we convert the continuous pipe diameter into x_l . Otherwise, we convert it to x_u .

The optimal design for the WDN is a constrained optimization problem. Therefore, we should combine CADOS with a constraint-handling method. Constraint tournament selection is utilized in our approach to tackle pressure constraints. As in [Deb *et al.* 2000], we devise the fitness function based on feasible and infeasible rules as follows:

$$F(\vec{x}) = \begin{cases} f(\vec{x}) & \text{if } g(\vec{x}) \geq 0 \\ p + \|g(\vec{x})\| & \text{Otherwise} \end{cases} \quad (20)$$

Where p is a penalty term set empirically to 10^7

In a feasible solution (for which the constraint vector $g(\vec{x}) \geq 0$), the fitness function is equal to its objective function value $f(\vec{x})$ (cost of the network). Furthermore, computing the objective function value for an infeasible solution is useless; thus, its fitness value is calculated using the constraint violation vector ($g(\vec{x})$) and the penalty term p to ensure that an infeasible solution will never be better than a feasible one in the current population.

5.2 Experimental Results

CADOS is utilized on a widely recognized network: the Hanoi network, presented in Figure 5. the Hanoi network is a three-loop network composed of 34 links, 32 nodes, and a reservoir. Six different pipe diameters 12, 16, 20, 24, 30, 40 inches are defined to design this network. The minimum pressure head requirement is set at 30 m for all nodes. The entire search space comprises a total of 6^{34} possible combinations. More details on this network are found in Fujiwara and Khang (1990) [Fujiwara *et al.* 1990]. The dither mutation and dither crossover factors presented by Karaboga and Okdem [Pan *et al.* 2011] were used in our work. These factors improve the effectiveness and robustness of optimization algorithms in tackling complex and multifaceted problems [Karaboğa and Ökdem 2004]. In addition, they were randomized as follows:

$$F_{dither} = F_L + U_i^F (F_h - F_L) \quad (21)$$

$$Cr_{dither} = Cr_L + U_i^{Cr} (Cr_h - Cr_L) \quad (22)$$

In our experiments, F_h and F_L are set to 0.5 and 1, respectively. Cr_h and Cr_L are set to 0.8 and 1, respectively. Fifty (50) runs have been performed using a population of 60 individuals.

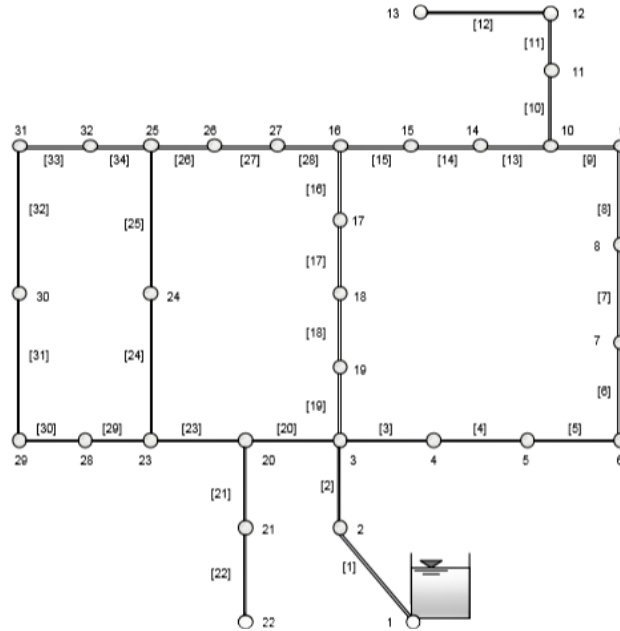


Figure 5: The layout of the Hanoi problem network

To evaluate the performance of CADOS, we conducted a comprehensive comparative study against several state-of-the-art algorithms, employing a diverse set of statistical metrics. Table 1 presents the performance comparison of CADOS with HD-DDS [Tolson *et al.* 2020], GA, PSO, DE-Best, and DE-rand.

We adopted statistical measures consistent with those delineated in [Zheng *et al.* 2011]. The "Number of different runs" denotes the count of independent evaluations executed for each algorithm, serving as an indicator of its performance stability. The "Best solution (\$M)" column signifies the best solution (in terms of cost, measured in \$M) achieved by each algorithm across all runs. Noting that the current best-known solution for the Hanoi problem is characterized by a cost of \$6.081 million [Reca and Martínez 2006].

The "Times with the best solution found" (percent of trials) illustrates the frequency and the proportion of instances each algorithm achieved its optimal solution, with higher percentages denoting more reliable performance. The "Average cost (\$M)" provides the mean cost across all trials for each algorithm, where diminished values typically signify superior performance. The "Maximum allowable evaluations" denotes the ceiling on the number of evaluations permitted for each algorithm. Additionally, we present the "Average evaluations required to find the best solutions," reflecting the mean evaluations necessary for each algorithm to converge to its optimal solution. Smaller values in this

metric imply faster convergence rates.

From the data illustrated in Table 1, CADOS found the best-known solution solution 17 times out of 50 runs, representing a 34% success rate. This is better than DE-Best (4% success), GA (0% success), PSO (0% success), and HD-DDS (8% success). CADOS has an average cost of \$6.170 million, which is better than HD-DDS, DE-Best, PSO, GA, and relatively close to the average cost achieved by DE-rand (\$6.088 million). CADOS is limited to 90,000 evaluations, which is less than DE-Best and DE-rand that can go up to 300,000 evaluations. In addition, CADOS took an average of 39,674 evaluations to find its best solution. This is lower than DE-rand (74,584 evaluations), HD-DDS (100,000 evaluations). However, the average evaluations to find the best solution for GA and PSO is not available.

Overall, CADOS shows competitive performance compared to the other algorithms. It offers a balanced approach, achieving good results with fewer maximum allowable evaluations compared to the other algorithms.

Table 1: Comparison of CADOS with other state-of-the-art algorithms

Algorithms	Number of different runs	Best Solution (\$M)	Times with best solution found (percent of trials)	Average cost (\$M)	Maximum allowable evaluations	Average evaluations required to find the best solutions
CADOS	50	6.081	17 (34%)	6.170	90,000	39,674
DE-Best	50	6.081	2 (4%)	6.240	300,000	6,660
DE-rand	50	6.081	43 (86%)	6.088	300,000	74,584
HD-DDS	50	6.081	4 (8%)	6.252	100,000	100,000
GA	30	6.167	0 (0%)	6.277	300,000	NA
PSO	30	6.373	0 (0%)	6.483	300,000	NA

6 Conclusion

In this paper, we have presented a collaborative auto-diversified optimization scheme. It is based on a modified version of our previously proposed approach AD-AMPEDE, CMAES, and a LS method using trust regions.

CADOS retains the advantages of these three algorithms. Firstly, effective interaction between sub-populations and parameter value adaptation using AD-AMPEDE. Secondly, efficient exploitation of the local area of the best-performing individual using a trust-region algorithm that could allow the search to find a better local optimum in case of stagnation. Finally, a good diversification using either the ability of CMAES to produce highly qualified solutions around each subpopulation's best individual or the AEPD diversification process employed in AD-AMPEDE that could generate new promising solutions. The new solutions replace randomly selected individuals.

This collaborative effort facilitated an enhanced algorithmic solution for a broader range of BBOB instances. By capitalizing on the strengths of each algorithm and strategically integrating their capabilities, our approach achieved remarkable results. CADOS

was first evaluated on the COCO platform. The experimental outcomes underscore the competitive and superior performance of CADOS, particularly in higher dimensions (10D and 20D), where in 10D, the obtained results showed its superiority over state-of-the-art algorithms, especially in Adequate Global Structure for higher dimension multimodal functions. Furthermore, meticulous analysis of parameter adaptation significantly contributed to the optimization of results. In 20D, our proposed approach showed an excellent rate compared to AD-AMPEDE, CMA-ES, DE-BFGS, DE, fmincon, GA, and PSO.

In addition, CADOS was compared to other state-of-the-art algorithms in solving the optimal design of WDS. Based on the experimental results, CADOS has demonstrated the ability to achieve the current best-known solution within a reasonable timeframe. These findings suggest that our approach holds promise as a viable alternative for solving combinatorial optimization problems.

However, despite the excellent performance of CADOS, there is still a need to minimize any side effects on its performance, particularly concerning choosing appropriately the moments when to check stagnation, for efficiently handling a broad range of BBOB functions within a reasonable timeframe.

In our future work, we plan to propose a collaborative scheme involving deep learning models to improve the adaptability of CADOS through better decision-making based on previous knowledge. That would allow CADOS to deal efficiently with complex situations.

References

- [Awad *et al.* 2018] Awad, N.H., Ali, M.Z., Mallipeddi, R., & Suganthan, P.N.: “An improved differential evolution algorithm using efficient adapted surrogate model for numerical optimization”. *Information Sciences* 451, 326–347 (2018).
- [Boks *et al.* 2020] Boks, R., Wang, H., & Bäck, T.: “A modular hybridization of particle swarm optimization and differential evolution”. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, (2020).
- [Broyden 1965] Broyden, C. G.: “A class of methods for solving nonlinear simultaneous equations”. *Mathematics of computation* 19.92 (1965): 577-593.
- [Conn *et al.* 2000] Conn, A.R., Gould, N.I., & Toint, P.L.: “Trust region methods”. SIAM (2000).
- [Cunha and Marques 2020] Cunha, M., Marques, J.: “A new multiobjective simulated annealing algorithm—mosa-gr: Application to the optimal design of water distribution networks”. *Water Resources Research* 56(3), 2019–025852 (2020).
- [Cunha *et al.* 2001] Cunha, M.d.C., Sousa, J.: “Hydraulic infrastructures design using simulated annealing”. *Journal of Infrastructure Systems* 7(1), 32–39 (2001).
- [Das *et al.* 2016] Das, S., Mullick, S.S., & Suganthan, P.N.: “Recent advances in differential evolution an updated survey”. *Swarm and evolutionary computation* 27, 1–30 (2016).
- [Deb *et al.* 2000] Deb, K.: “An efficient constraint handling method for genetic algorithms”. *Computer methods in applied mechanics and engineering* 186(2-4), 311–338 (2000).
- [Draa *et al.* 2011] Draa, A., Meshoul, S., Talbi, H., & Batouche, M.: “A quantum-inspired differential evolution algorithm for solving the n-queens problem”. *neural networks* 1(2), 21–27 (2011).

- [El-Abd *et al.* 2009] El-Abd, M., Kamel, M.S.: “Black-box optimization benchmarking for noiseless function testbed using particle swarm optimization”. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, pp. 2269–2274 (2009).
- [Eusuff *et al.* 2003] Eusuff, M.M., Lansey, K.E.: “Optimization of water distribution network design using the shuffled frog leaping algorithm”. *Journal of Water Resources planning and management* 129(3), 210–225 (2003).
- [Fujiwara *et al.* 1990] Fujiwara, O., Khang, D.B.: “A two-phase decomposition method for optimal design of looped water distribution networks”. *Water resources research* 26(4), 539–549 (1990).
- [Glover 1989] Glover, F.: “Tabu search—part I”. *ORSA Journal on computing* 1(3), 190–206 (1989).
- [Hansen and Ostermeier 1996] Hansen, N., Ostermeier, A.: “Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation”. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 312–317 (1996). IEEE.
- [Hansen and Ostermeier. 2001] Hansen, N., Ostermeier, A.: “Completely derandomized self-adaptation in evolution strategies”. *Evolutionary computation* 9(2), 159–195 (2001).
- [Hansen *et al.* 2016] Hansen, N., Auger, A., Brockhoff, D., Tušar, D., & Tušar, T.: “Coco: performance assessment”. arXiv preprint arXiv:1605.03560 (2016).
- [Hansen *et al.* 2003] Hansen, N., Müller, S.D., & Koumoutsakos, P.: “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)”. *Evolutionary computation* 11(1), 1–18 (2003).
- [Hansen *et al.* 2019] Hansen, N.: “A global surrogate assisted cma-es”. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 664–672 (2019).
- [Hezili and Talbi 2021] Hezili, B., Talbi, H.: “Improvement of differential evolution with multipopulation-based ensemble of mutation strategies”. in TACC2021, 534–545 (2021)
- [Hezili and Talbi 2022] Hezili, B., Talbi, H.: “Auto-diversified ameliorated multipopulation-based ensemble differential evolution”. In: International Symposium on Modelling and Implementation of Complex Systems, pp. 177–191 (2022).
- [Karaboğa and Ökdem 2004] Karaboğa, D., & Ökdem, S.: “A simple and global optimization algorithm for engineering problems: differential evolution algorithm”. *Turkish Journal of Electrical Engineering and Computer Sciences* 12(1), 53–60 (2004).
- [Karaboğa *et al.* 2004] Karaboğa, D., Derviş, & Ökdem, S.: “A simple and global optimization algorithm for engineering problems: differential evolution algorithm”. *Turkish Journal of Electrical Engineering and Computer Sciences* 12(1), 53–60 (2004).
- [Laarhoven *et al.* 1987] Laarhoven, P. J. M. and Aarts, E. H. L.: *Simulated annealing: theory and applications*. Kluwer Academic Publishers (1987).
- [Lezama *et al.* 2018] Lezama, F., Soares, J., Faia, R., Pinto, T., & Vale, Z.: “A new hybrid-adaptive differential evolution for a smart grid application under uncertainty”. 2018 IEEE Congress on Evolutionary Computation (CEC). IEEE, (2018).
- [Mandal *et al.* 2011] Mandal, A., Das, A.K., Mukherjee, P., Das, S., & Suganthan, P.N.: “Modified differential evolution with local search algorithm for real world optimization”. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 1565–1572 (2011). IEEE.
- [Martínez-Estudillo *et al.* 2006] Martínez-Estudillo, A.C., Hervás-Martínez, C., Martínez-Estudillo, F.J., & García-Pedrajas, N.: “Hybridization of evolutionary algorithms and local search by means of a clustering method”. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 36(3), 534–545 (2006).

- [Mohamed *et al.* 2017] Mohamed, A.W., Hadi, A.A., Fattouh, A.M., & 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 (2017). IEEE.
- [Munoz *et al.* 2013] Munoz, M.A., Kirley, M., & Halgamuge, S.K.: “The algorithm selection problem on the continuous optimization domain”. In: Computational Intelligence in Intelligent Data Analysis, pp. 75–89. Springer (2013).
- [Nicolau 2009] Nicolau, M.: “Application of a simple binary genetic algorithm to a noiseless testbed benchmark”. Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers. (2009).
- [Nomanet and Iba 2005] Noman, N., Iba, H.: “Enhancing differential evolution performance with local search for high-dimensional function optimization”. In: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, pp. 967–974 (2005).
- [Noronha 2022] Noronha, R.P.: “Improvement of trade-off between global and local search in hybridization GA-PSO with fuzzy adaptive acceleration coefficients”. Ubiquitous Intelligent Systems: Proceedings of ICUIS 2021. Springer Singapore, (2022).
- [Okulewicz and Zaborski 2021] Okulewicz, M., Zaborski, M.: “Benchmarking shade algorithm enhanced with model based optimization on the bbob noiseless testbed”. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1259–1266 (2021).
- [Okulewicz *et al.* 2022] Okulewicz, M., Zaborski, M., & Mańdziuk, J.: “Self-adapting particle swarm optimization for continuous black box optimization”. Applied Soft Computing 131, 109722 (2022).
- [Pál 2013] Pál, L.: “Comparison of multistart global optimization algorithms on the BBOB noiseless testbed”. Proceedings of the 15th annual conference companion on Genetic and evolutionary computation (pp. 1153-1160) (2013).
- [Pan *et al.* 2011] Pan, I., Das, S., & Gupta, A.: “Handling packet dropouts and random delays for unstable delayed processes in ncs by optimal tuning of π l δ u controllers with evolutionary algorithms”. ISA transactions 50(4), 557–572 (2011).
- [Pošík and . 2012] Pošík, P., Klemš, V.: “Benchmarking the differential evolution with adaptive encoding on noiseless functions”. In: Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 189–196 (2012).
- [Reca and Martínez 2006] Reca, J., and Martínez, J.: “Genetic Algorithms for the design of looped irrigation water distribution networks”. Water Resources Research, 42, W05416, doi:10.1029/2005WR004383, 2006.
- [Sabzkouhi *et al.* 2022] Sabzkouhi, A.M., Lee, J., & Keck, J.: “Calibration and uncertainty analysis of hydraulic models“ (2022).
- [Simpson *et al.* 1994] Simpson, A.R., Dandy, G.C., & Murphy, L.J.: “Genetic algorithms compared to other techniques for pipe optimization”. Journal of water resources planning and management 120(4), 423–443 (1994).
- [Storn 1996] Storn, R.: “On the usage of differential evolution for function optimization”. In: Proceedings of North American Fuzzy Information Processing, pp. 519–523 (1996). IEEE.
- [Storn and Price 1997] Storn, R., Price, K.: “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces”. Journal of global optimization 11, 341–359 (1997).
- [Suribabu 2010] Suribabu, C.: “Differential evolution algorithm for optimal design of water distribution networks”. Journal of Hydroinformatics 12(1), 66–82 (2010).
- [Talbi and Draa 2020] Talbi, H., Draa, A.: “A continuous optimization scheme based on an enhanced differential evolution and a trust region method“. In Proceedings of the 8th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT’18), Vol. 1 (pp. 222-233). Springer International Publishing (2020).

- [Tolson *et al.* 2020] Tolson, B. A., Asadzadeh, M., Maier, H. R., & Zecchin, A.: “Hybrid discrete dynamically dimensioned search (HD \square DDS) algorithm for water distribution system design optimization”. *Water Resources Research* 45.12, (2009).
- [Varelas *et al.* 2018] Varelas, K., Auger, A., Brockhoff, D., Hansen, N., ElHara, O.A., Semet, Y., Kassab, R., & Barbaresco, F.: “A comparative study of large-scale variants of cma-es”. In: *Parallel Problem Solving from Nature–PPSN XV: 15th International Conference, Coimbra, Portugal, September 8–12, 2018, Proceedings, Part I* 15, pp. 3–15 (2018). Springer
- [Voglis *et al.* 2012] Voglis, C., Piperagkas, G.S., Parsopoulos, K.E., Papageorgiou, D.G., & Lagaris, I.E.: “Mempso: an empirical assessment of local search algorithm impact on a memetic algorithm using noiseless testbed”. In: *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, pp. 245–252 (2012).
- [Wang *et al.* 2019] Wang, Q., Wang, L., Huang, W., Wang, Z., Liu, S., & Savić, D.A.: “Parameterization of nsga-ii for the optimal design of water distribution systems”. *Water* 11(5), 971 (2019).
- [Wang *et al.* 2014] Wang, Y., Li, H.-X., Huang, T., & Li, L.: “Differential evolution based on covariance matrix learning and bimodal distribution parameter setting”. *Applied Soft Computing* 18, 232–247 (2014).
- [Xu *et al.* 2019] Xu, P., Luo, W., Lin, X., Qiao, Y., & Zhu, T.: “Hybrid of pso and cma-es for global optimization”. In: *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 27–33 (2019). IEEE.
- [Zhang *et al.* 2009] Zhang, J., Sanderson, A.C.: “Jade: adaptive differential evolution with optional external archive”. *IEEE Transactions on evolutionary computation* 13(5), 945–958 (2009).
- [Zheng *et al.* 2011] Zheng, F., Simpson, A.R., & Zecchin, A.: “Performance study of differential evolution with various mutation strategies applied to water distribution system optimization”. In: *World Environmental and Water Resources Congress 2011: Bearing Knowledge for Sustainability*, pp. 166–176 (2011).

Appendix

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f1, 20-D	43	43	43	43	43	43	43	15/15
AD-AMPEDE	643 (188)	1460 (245)	2382 (500)	3370 (672)	4326 (848)	6270 (1246)	8351 (694)	15/15
CADOS	624 (96)	1298 (85)	2081 (104)	2818 (170)	3637 (186)	5243 (222)	7137 (435)	15/15
CMA-ES	7.5 (0.7)	13 (3)	20 (1)	26 (2)	33 (1)	45 (2)	58 (4)	15/15
DE	115 (16)	255 (52)	398 (38)	554 (69)	713 (59)	1002 (107)	1302 (87)	15/15
DE-BFGS	2.0 (0.2)	2.2 (0)	2.2 (0)	2.2 (0)	2.2 (0)*⁴	2.2 (0)*⁴	2.2 (0)*⁴	15/15
GA	876 (51)	1905 (71)	3205 (172)	1.2e4 (1e4)	3.1e4 (2e4)	6.7e5 (4e5)	∞ 2e6	0/15
JADE	47 (8)	94 (6)	143 (8)	191 (7)	240 (4)	340 (8)	437 (3)	15/15
PSO	22 (9)	3399 (2e4)	3446 (29)	3507 (24)	3563 (3e4)	3680 (37)	3808 (1e4)	14/15
fmincon	0.77 (0.2)*⁴	1.7 (0.5)*⁴	1.9 (0.1)*⁴	2.2 (0.2)	2.8 (0.2)	3.7 (0.6)	4.7 (0.2)	15/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f2, 20-D	385	386	387	388	390	391	393	15/15
AD-AMPEDE	590 (169)	692 (163)	782 (192)	868 (229)	957 (185)	1159 (77)	1359 (218)	15/15
CADOS	416 (10)	497 (17)	585 (26)	677 (38)	764 (25)	964 (58)	1151 (47)	15/15
CMA-ES	36 (3)	43 (3)	45 (2)	47 (2)	47 (2)	48 (2)	50 (1)	15/15
DE	62 (6)	79 (4)	95 (7)	112 (8)	127 (12)	157 (12)	186 (17)	15/15
DE-BFGS	5.7 (4)	6.1 (5)	6.2 (3)	6.4 (5)	6.5 (4)	7.1 (5)	10 (5)	15/15
GA	3228 (1360)	6769 (6486)	7.3e4 (8e4)	∞	∞	∞	∞ 2e6	0/15
JADE	28 (1)	34 (1)	39 (2)	44 (2)	50 (2)	61 (3)	71 (5)	15/15
PSO	4576 (9093)	4571 (1e4)	4565 (6465)	4560 (1e4)	4543 (5130)	4546 (7676)	4537 (6362)	8/15
fmincon	5.6 (2)	6.0 (1)	6.4 (2)	6.8 (3)	7.4 (2)	10 (3)	14 (8)	15/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f5, 20-D	41	41	41	41	41	41	41	15/15
AD-AMPEDE	503 (96)	545 (137)	556 (180)	556 (126)	556 (185)	556 (162)	556 (142)	15/15
CADOS	584 (73)	696 (39)	716 (76)	716 (107)	716 (94)	716 (92)	716 (78)	15/15
CMA-ES	5.1 (1)	6.1 (1.0)	6.1 (0.8)	6.1 (1)	6.1 (1)	6.1 (0.6)*²	6.1 (1)*⁴	15/15
DE	1407 (198)	3926 (306)	6625 (420)	9331 (592)	1.2e4 (789)	1.7e4 (997)	2.3e4 (834)	1/15
DE-BFGS	14 (9)	22 (12)	23 (14)	24 (18)	24 (15)	25 (20)	26 (14)	15/15
GA	2137 (121)	4548 (264)	7446 (243)	1.1e4 (464)	1.4e4 (552)	2.2e4 (823)	2.3e5 (3e5)	0/15
JADE	43 (6)	52 (9)	53 (7)	53 (6)	53 (8)	53 (6)	53 (9)	15/15
PSO	4.3e4 (1e5)	4.3e4 (6e4)	4.3e4 (7e4)	4.3e4 (1e5)	4.3e4 (2e4)	4.3e4 (4e4)	4.3e4 (1e5)	8/15
fmincon	3.6 (0)*²	5.1 (0)	5.7 (0)	6.2 (0)	6.7 (0)	7.2 (0)	102 (105)	4/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f6, 20-D	1296	2343	3413	4255	5220	6728	8409	15/15
AD-AMPEDE	178 (88)	283 (720)	237 (287)	223 (417)	214 (39)	218 (27)	264 (116)	12/15
CADOS	105 (18)	106 (27)	117 (89)	152 (235)	166 (25)	202 (30)	227 (30)	15/15
CMA-ES	1.7 (0.3)	1.3 (0.3)	1.2 (0.3)	1.2 (0.1)*	1.2 (0.1)*⁴	1.2 (0.1)*⁴	1.2 (0.1)*⁴	15/15
DE	67 (15)	79 (16)	85 (10)	94 (17)	98 (13)	110 (16)	209 (181)	4/15
DE-BFGS	6.8 (5)	8.5 (7)	17 (10)	25 (26)	32 (31)	96 (66)	295 (330)	2/15
GA	1955 (3111)	∞	∞	∞	∞	∞	∞ 2e6	0/15
JADE	9.4 (0.5)	7.8 (0.9)	7.3 (0.5)	7.4 (1.0)	7.2 (0.9)	7.4 (1.0)	7.4 (0.9)	15/15
PSO	1082 (1190)	1009 (1289)	705 (1195)	586 (488)	502 (667)	423 (186)	577 (483)	5/15
fmincon	1.6 (0.6)	1.4 (0.6)	1.4 (0.5)	1.6 (0.4)	1.7 (0.8)	2.0 (0.6)	2.4 (0.7)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f7, 20-D	1351	4274	9503	16523	16524	16524	16969	15/15
AD-AMPEDE	46 (5)	4947 (5288)	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	46 (10)	276 (36)	1048 (1036)	1623 (1791)	1623 (594)	1623 (1766)	1582 (2602)	3/15
CMA-ES	1.5 (0.2)* ³	191 (237)	754 (581)	∞	∞	∞	$\infty 1e6$	0/15
DE	19 (11)	28 (11)	207 (236)	352 (613)	352 (440)	352 (230)	343 (437)	3/15
DE-BFGS	502 (701)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
GA	77 (20)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	4.8 (1)	272 (293)	686 (947)	402 (199)	402 (318)	402 (277)	391 (339)	2/15
PSO	427 (1484)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 2e5$	0/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f8, 20-D	2039	3871	4040	4148	4219	4371	4484	15/15
AD-AMPEDE	426 (248)	745 (945)	766 (659)	781 (471)	791 (441)	816 (437)	834 (384)	11/15
CADOS	480 (138)	494 (94)	549 (110)	579 (130)	601 (112)	626 (188)	647 (69)	15/15
CMA-ES	3.9 (1)	4.3 (0.9)	4.7 (2)	4.8 (0.4)	4.8 (2)	4.9 (2)	4.9 (0.7)	15/15
DE	43 (12)	55 (26)	60 (52)	64 (4)	68 (23)	76 (43)	84 (47)	15/15
DE-BFGS	1.5 (1)	2.8 (4)	2.8 (3)	2.9 (3)	3.2 (2)	3.2 (4)	3.2 (3)	15/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	18 (0.8)	16 (0.4)	16 (0.5)	17 (0.5)	17 (0.5)	17 (0.6)	18 (0.5)	15/15
PSO	90 (81)	307 (429)	350 (401)	406 (146)	466 (350)	894 (849)	3276 (2481)	1/15
fmincon	0.84 (0.1)	0.84 (1.0)	0.85 (0.8)	0.85 (0.1)	0.85 (0.4)* ²	0.84 (0.1)* ²	0.83 (0.7)* ²	15/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f9, 20-D	1716	3102	3277	3379	3455	3594	3727	15/15
AD-AMPEDE	591 (186)	806 (81)	964 (298)	1409 (1279)	1773 (1501)	4696 (4264)	$\infty 3e6$	0/15
CADOS	800 (105)	951 (124)	1016 (107)	1038 (81)	1048 (222)	1046 (188)	1040 (213)	15/15
CMA-ES	4.5 (0.9)	5.0 (0.5)	5.4 (0.4)	5.5 (0.5)	5.5 (0.3)	5.5 (0.4)	5.5 (0.4)	15/15
DE	187 (41)	396 (86)	4537 (4120)	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	1.8 (2)	2.6 (3)	2.8 (1)	2.7 (0.9)	2.7 (1)	2.7 (3)	2.7 (2)	15/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	36 (2)	30 (3)	32 (2)	32 (2)	33 (2)	33 (2)	33 (3)	15/15
PSO	670 (320)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	0.17 (0.0)* ⁴	0.34 (0.0)* ⁴	0.38 (7e-3)* ⁴	0.39 (0.0)* ⁴	0.40 (0.0)* ⁴	0.40 (0.0)* ⁴	0.40 (0.0)* ⁴	15/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f10, 20-D	7413	8661	10735	13641	14920	17073	17476	15/15
AD-AMPEDE	1468 (1949)	5730 (4023)	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	79 (58)	135 (123)	160 (76)	165 (117)	179 (130)	220 (64)	315 (192)	9/15
CMA-ES	1.7 (0.1)	1.7 (0.2)	1.6 (0.1)	1.3 (0.0)	1.2 (0.1)	1.1 (0.0)	1.1 (0.0)	15/15
DE	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	6.1 (8)	5.3 (3)	4.3 (4)	3.4 (5)	3.1 (2)	2.7 (4)	2.6 (2)	15/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	12 (6)	15 (3)	15 (2)	14 (3)	15 (3)	15 (4)	18 (4)	15/15
PSO	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	0.15 (0.0)* ⁴	0.14 (0.1)* ⁴	0.12 (0.0)* ⁴	0.10 (0.0)* ⁴	0.10 (0.1)* ⁴	34 (25)	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f11, 20-D	1002	2228	6278	8586	9762	12285	14831	15/15
AD-AMPEDE	412 (832)	840 (1305)	626 (729)	1699 (1312)	4914 (5909)	3919 (3384)	3267 (3604)	1/15
CADOS	118 (36)	77 (24)	39 (14)	47 (56)	54 (33)	115 (192)	292 (388)	10/15
CMA-ES	11 (0.6)	5.3 (0.2)	2.0 (0.0)	1.5 (0.0)	1.4 (0.1)	1.2 (0.0)	1.0 (0.0)	15/15
DE	844 (370)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	0.20 (0.0)	0.11 (0.0)	0.05 (9e-3)	0.04 (0.0)	0.04 (0.0)* ³	0.04 (0.0)* ⁴	0.05 (0.0)* ⁴	15/15
GA	2.9e4 (2e4)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	92 (748)	46 (117)	18 (42)	15 (61)	15 (27)	15 (3)	15 (19)	14/15
PSO	143 (47)	186 (25)	109 (13)	114 (11)	132 (11)	148 (10)	2019 (1585)	0/15
fmincon	0.16 (0.0)	0.10 (0.0)	0.04 (8e-3)	0.04 (1e-2)	1.1 (2)	228 (253)	$\infty 4e5$	0/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f12, 20-D	1042	1938	2740	3156	4140	12407	13827	15/15
AD-AMPEDE	440 (434)	1156 (880)	2447 (1539)	7300 (1e4)	1.2e4 (2e4)	∞	$\infty 3e6$	0/15
CADOS	276 (12)	560 (685)	738 (612)	928 (689)	1021 (632)	537 (362)	643 (534)	7/15
CMA-ES	2.2 (0.1)	3.2 (3)	4.0 (3)	4.4 (4)	4.0 (2)	1.8 (0.4)	1.9 (0.7)	15/15
DE	119 (485)	200 (259)	1069 (1299)	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	1.4 (0.5)	1.4 (2)	1.6 (1.0)	1.6 (1)	1.7 (1)	1.1 (0.6)	4.6 (4)	15/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	19 (4)	20 (9)	28 (14)	30 (15)	28 (7)	13 (4)	14 (4)	15/15
PSO	1703 (2409)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	0.81 (0.7)	0.91 (0.4)	0.84 (0.4)	0.84 (0.2)	0.85 (0.7)	0.75 (0.6)	14 (3)	4/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f13, 20-D	652	2021	2751	3507	18749	24455	30201	15/15
AD-AMPEDE	374 (202)	521 (458)	893 (714)	1387 (561)	1305 (1136)	∞	$\infty 3e6$	0/15
CADOS	238 (27)	189 (88)	295 (156)	391 (352)	116 (56)	189 (213)	217 (120)	10/15
CMA-ES	6.3 (5)	5.1 (3)	4.5 (2)	4.4 (1)	1.9 (2)	4.6 (3)	8.4 (10)	12/15
DE	62 (13)	200 (177)	1077 (855)	2629 (1931)	493 (613)	782 (761)	$\infty 1e6$	0/15
DE-BFGS	1.3 (0.1)	0.59 (0.1)	0.53 (0.0)	0.50 (0.1)*	0.18 (0.1)	92 (184)	$\infty 2e6$	0/15
GA	5114 (1e4)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	17 (3)	14 (5)	15 (5)	14 (3)	3.6 (0.7)	4.8 (0.6)	9.0 (3)	15/15
PSO	6153 (7670)	6441 (8412)	1.0e4 (2e4)	7993 (8412)	1495 (2027)	∞	$\infty 2e6$	0/15
fmincon	1.1 (0.2)	0.58 (0.8)	0.79 (0.9)	1.3 (0.7)	0.55 (0.5)	∞	$\infty 4e5$	0/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f14, 20-D	75	239	304	451	932	1648	15661	15/15
AD-AMPEDE	188 (151)	240 (69)	366 (101)	453 (110)	552 (211)	∞	$\infty 3e6$	0/15
CADOS	209 (39)	225 (53)	297 (17)	302 (14)	217 (22)	338 (161)	196 (320)	8/15
CMA-ES	4.2 (1)	3.0 (0.4)	3.7 (0.3)	4.3 (0.7)	4.2 (0.3)	6.2 (0.4)	1.2 (0.1)* ⁴	15/15
DE	35 (15)	50 (13)	70 (7)	100 (14)	227 (54)	∞	$\infty 1e6$	0/15
DE-BFGS	1.7 (0.3)	0.84 (0.1)	0.90 (0.2)	0.93 (0.1)	0.62 (0.1)	0.56 (0.1)* ²	125 (166)	0/15
GA	277 (81)	320 (34)	477 (56)	2898 (3469)	∞	∞	$\infty 2e6$	0/15
JADE	18 (6)	18 (1)	23 (2)	25 (2)	20 (2)	38 (24)	62 (46)	5/15
PSO	6.7 (3)	12 (5)	20 (2)	27 (5)	54 (8)	∞	$\infty 2e6$	0/15
fmincon	0.74 (0.3)* ⁴	0.49 (0.0)* ⁴	0.65 (0.1)* ³	0.88 (0.1)	0.65 (0.1)	0.68 (0.1)	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f16, 20-D	1384	27265	77015	1.4e5	1.9e5	2.0e5	2.2e5	15/15
AD-AMPEDE	383 (1035)	∞	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	109 (37)	3709 (5467)	∞	∞	∞	∞	$\infty 7e6$	0/15
CMA-ES	1.9 (0.6)*³	2.7 (2)*⁴	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	595 (335)	242 (488)	∞	∞	∞	∞	$\infty 2e6$	0/15
GA	138 (31)	1037 (881)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	24 (7)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
PSO	111 (729)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	675 (656)	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f17, 20-D	63	1030	4005	12242	30677	56288	80472	15/15
AD-AMPEDE	51 (57)	90 (29)	5560 (3380)	∞	∞	∞	$\infty 3e6$	0/15
CADOS	54 (54)	79 (8)	47 (8)	407 (866)	∞	∞	$\infty 7e6$	0/15
CMA-ES	2.8 (0.8)	0.96 (0.2)*⁴	1.5 (2)*	3.7 (3)	8.1 (7)	∞	$\infty 1e6$	0/15
DE	12 (5)	21 (3)	16 (3)	16 (2)	27 (99)	265 (441)	$\infty 1e6$	0/15
DE-BFGS	17 (16)	117 (63)	327 (295)	1140 (1313)	952 (685)	∞	$\infty 2e6$	0/15
GA	57 (40)	92 (16)	7071 (1e4)	∞	∞	∞	$\infty 2e6$	0/15
JADE	7.8 (5)	7.4 (0.7)	4.4 (0.6)	2.4 (0.4)	1.7 (1)*²	7.2 (3)*³	23 (35)	2/15
PSO	3.2 (1)	2503 (4369)	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	21 (32)	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f18, 20-D	621	3972	19561	28555	67569	1.3e5	1.5e5	15/15
AD-AMPEDE	59 (20)	472 (438)	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	66 (15)	37 (4)	724 (542)	3442 (5276)	∞	∞	$\infty 7e6$	0/15
CMA-ES	1.1 (0.3)*³	2.0 (3)	4.2 (6)	493 (368)	∞	∞	$\infty 1e6$	0/15
DE	20 (6)	22 (8)	16 (5)	66 (127)	213 (307)	∞	$\infty 1e6$	0/15
DE-BFGS	69 (65)	350 (249)	1503 (2022)	∞	∞	∞	$\infty 2e6$	0/15
GA	76 (16)	311 (506)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	7.2 (1)	4.4 (1)	1.5 (0.5)	8.2 (6)*²	19 (18)	∞	$\infty 1e6$	0/15
PSO	236 (3)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f19, 20-D	1	1	3.4e5	4.7e6	6.2e6	6.7e6	6.7e6	15/15
AD-AMPEDE	3261 (2743)	6.6e5 (7e4)	43 (27)	∞	∞	∞	$\infty 3e6$	0/15
CADOS	7850 (3564)	1.4e5 (2e4)	∞	∞	∞	∞	$\infty 7e6$	0/15
CMA-ES	162 (52)	4.8e4 (3e4)	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	1961 (898)	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	3206 (2884)	3.8e4 (1e4)	5.4 (4)	∞	∞	∞	$\infty 2e6$	0/15
GA	1.4e4 (3222)	6.5e5 (6e5)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	856 (254)	7.0e5 (1e5)	∞	∞	∞	∞	$\infty 1e6$	0/15
PSO	382 (148)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	1 (0)*4	1 (0)*4	7.3e-4 (7e-7)*4	∞	∞	∞	$\infty 4e5$	0/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f20, 20-D	82	46150	3.1e6	5.5e6	5.5e6	5.6e6	5.6e6	14/15
AD-AMPEDE	194 (21)	204 (53)	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	297 (27)	5.0 (0.6)	16 (30)	19 (57)	19 (8)	19 (11)	19 (12)	1/15
CMA-ES	5.1 (1)	156 (211)	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	74 (26)	3.0 (2)	5.4 (3)	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	6.3 (1)	1.0 (1)	∞	∞	∞	∞	$\infty 2e6$	0/15
GA	502 (51)	2.8 (0.4)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	24 (2)	1.2 (0.2)	0.46 (0.5)	0.62 (0.9)	0.85 (0.6)	2.7 (3)	$\infty 1e6$	0/15
PSO	17 (3)	50 (54)	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	1.4 (0)*4	10 (6)	∞	∞	∞	∞	$\infty 4e5$	0/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f21, 20-D	561	6541	14103	14318	14643	15567	17589	15/15
AD-AMPEDE	65 (24)	755 (623)	353 (749)	351 (457)	345 (332)	330 (316)	297 (599)	6/15
CADOS	88 (19)	420 (548)	267 (604)	266 (251)	261 (324)	247 (228)	220 (176)	8/15
CMA-ES	5.4 (14)	9.4 (8)	5.5 (2)	5.4 (6)	5.3 (6)	5.0 (9)	4.5 (2)	15/15
DE	30 (42)	75 (130)	35 (75)	35 (25)	35 (54)	33 (42)	30 (48)	12/15
DE-BFGS	10 (0.3)	33 (44)	17 (27)	16 (21)	16 (25)	15 (10)	205 (189)	1/15
GA	90 (35)	625 (461)	398 (603)	397 (384)	904 (888)	∞	$\infty 2e6$	0/15
JADE	7.2 (2)	33 (43)	21 (11)	20 (35)	20 (36)	19 (15)	18 (26)	13/15
PSO	1785 (2674)	4281 (2981)	1986 (1950)	1956 (1083)	1913 (1571)	1799 (931)	1593 (1336)	1/15
fmincon	1.00 (1)	0.62 (1)*2	0.46 (0.4)*	0.46 (0.6)*	0.46 (0.1)*	0.46 (0.5)*	0.48 (0.5)*2	9/15
Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f22, 20-D	467	5580	23491	24163	24948	26847	1.3e5	12/15
AD-AMPEDE	1154 (33)	2349 (3352)	1948 (2218)	1901 (2863)	1845 (2056)	1728 (1485)	346 (707)	1/15
CADOS	133 (147)	8101 (1e4)	∞	∞	∞	∞	$\infty 7e6$	0/15
CMA-ES	3.8 (0.2)	42 (98)	142 (266)	138 (249)	133 (84)	124 (197)	25 (32)	4/15
DE	48 (47)	102 (45)	189 (194)	185 (184)	181 (214)	172 (150)	35 (42)	3/15
DE-BFGS	5.7 (10)	135 (237)	151 (175)	146 (129)	142 (135)	132 (80)	$\infty 2e6$	0/15
GA	110 (42)	1452 (2421)	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	25 (25)	261 (556)	638 (383)	620 (559)	601 (481)	559 (633)	$\infty 1e6$	0/15
PSO	5.0 (8)	411 (718)	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	4.2 (9)	3.1 (8)	4.8 (3)*3	4.7 (2)*3	4.5 (5)*3	4.3 (5)*3	1.3 (1)*3	5/15

Δf_{opt}	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f24, 20-D	1.3e6	7.5e6	5.2e7	5.2e7	5.2e7	5.2e7	5.2e7	3/15
AD-AMPEDE	∞	∞	∞	∞	∞	∞	$\infty 3e6$	0/15
CADOS	∞	∞	∞	∞	∞	∞	$\infty 4e6$	0/15
CMA-ES	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
DE-BFGS	4.3 (5)	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
GA	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
JADE	∞	∞	∞	∞	∞	∞	$\infty 1e6$	0/15
PSO	∞	∞	∞	∞	∞	∞	$\infty 2e6$	0/15
fmincon	∞	∞	∞	∞	∞	∞	$\infty 4e5$	0/15

Figure 6: Numerical results in 20D