Annales Univ. Sci. Budapest., Sect. Comp. 53 (2022) 175–199

TESTING VARIOUS NUMERICAL OPTIMIZATION METHODS ON A SERIES OF ARTIFICIAL TEST FUNCTIONS

Simret Goitom, Tibor Nagy and Tamás Turányi
(Budapest, Hungary)

Communicated by Imre Kátai
(Received September 20, 2022; accepted October 15, 2022)

Abstract. Features of several well-known global and local optimization methods were discussed and their robustness and efficiency were tested on several artificial test functions in Matlab environment. The tested local methods were the interior-point, the quasi-Newton method, Nelder–Mead simplex, the pattern search, the NEWUOA and the BOBYQA methods. The global methods were the genetic algorithm (GA), the simulated annealing (SA), the particle swarm optimization (PSO), and the covariance matrix adaptation evolutionary strategy (CMA-ES) methods (see subsections 2.2 and 2.3 for their details). Furthermore, a novel global optimization method, called FOCusing robusT Optimization with Uncertainty-based Sampling (FOCTOPUS), which proved to be very efficient in the optimization of constrained and highly correlated parameters of combustion kinetic models, was also tested. The test functions were selected in such a way that they had a variety of features: uni-modal and multi-modal, differentiable and non-differentiable, separable and non-separable, low dimensional and high dimensional. The following test functions were used: the 20D Alpine function, the 4D Ackley function, the Cross-in-tray 2D function, the Hartmann-6D function, the Holder table 2D function, the 5D Rastrigin function, the 5D Rosenbrock function, the 4D modified Rosenbrock function, the 20D Zakharov function and a typical 2D multi-modal function. The general conclusion here is that, the global methods performed well on the multi-modal and high-dimensional test functions while the local methods were superior in the case of low-dimensional and uni-modal test functions. For the highly multi-modal test functions, the GA was better than all the other methods. The FOCTOPUS method proved to be inferior to GA for most of the test functions, thus its application cannot be generally recommended.

Key words and phrases: Local optimization methods, global optimization methods, artificial test functions.
2010 Mathematics Subject Classification: 65K10.
The Project is supported by the Hungarian National Research, Development and Innovation Office via grants K132109 (T.T.) and FK 134332 (T.N.).
https://doi.org/10.71352/ac.53.175

1. Introduction

In a recent study [4], efficient local optimization algorithms: the simplex, the NEWUOA, the Hooke-Jeeves, the Powel and the UNIRANDI [17] methods were compared on some benchmark functions including: Ackley (5D), Rastrigin (4D), Rosenbrock (5D, 40D), Hartmann-6D and Zakharov (5D, 40D and 60D). The methods were compared in terms of the number of function evaluations and success rates. The results indicated that, the simplex method converged to the global optima of the Ackley (5D), Rastrigin (4D), Rosenbrock (5D and 40D), Hartmann-6D and Zakharov (5D, 40D and 60D) functions with a success rate of 0%, 47%, (100%, 0%), 100% and (100%, 0%, 0%) respectively. However, the NEWUOA converged with a success rate of 100%, 20%, (100%, 100%), 100% and (100%, 100%, 100%) in the same order. In this paper, the benchmark study was extended by a few additional local and several global methods and by further test functions.

In a recent paper of Goitom et al. [9], several local and global optimization methods were tested on a combustion kinetic parameter fitting problem to optimize parameter sets of different sizes (with 5, 11 and 29 parameters) by fitting to large number of experimental data points. In the smallest problem (fitting 5 parameters to 732 experimental data points), all the local and global methods resulted in identical optimal value but different speed and number of function evaluations. In the case of fitting a large parameter set (fitting 29 parameters to 2307 data points), the FOCTOPUS method, which is a novel global optimization method developed specifically for combustion kinetic problems, found the best optimal value, though it needed a high number of function evaluations. FOCTOPUS is the acronym for FOCusing robusT Optimization with Uncertainty-based Sampling. The method was first published in [39] and later modified in [41] and [26], and described in the most detailed way in [9]. The BOBYQA method [32] also performed well in all the test cases. The second purposes of the present paper, is to assess and compare the performance of the novel FOCTOPUS and the other tested methods on a wider range of artificial, but challenging problems.

1.1. Mathematical formulations of optimization problems

Optimization is a technique that refers to minimization or maximization of a function output. The problems of optimization are everywhere in academic research and in real-world applications such as engineering, industry, finance, different scientific areas and so on. In a real-life optimization problem, the first step is to properly formulate the optimization problem using a mathematical model. This mathematical model contains the objective function to be minimized/maximized, the design variables that represent the arguments of the objective function and the conditions for these design variables called con-

straint functions. Based on the nature of the design variables, the constraint functions and the objective function, one can classify optimization problems as continuous and discrete, constrained and unconstrained, linear and nonlinear, respectively. A general form of n-variable continuous, constrained minimization problem is:

$$\begin{aligned} &\min_{\mathbf{x}\in X\subseteq\mathbb{R}^n} && f(\mathbf{x}), \\ &\text{subject to} && g_i(\mathbf{x}) = 0,\ i \in I_1, \\ &&& h_j(\mathbf{x}) \le 0,\ j \in I_2, \\ &&& a_k \le x_k \le b_k,\ \forall k = 1, 2, ..., n, \end{aligned} \tag{1.1}$$

where f, g_i and h_j are real-valued functions from $\mathbb{R}^n$ to $\mathbb{R}$, $\mathbf{x}$ is a vector of design variables, I_1 and I_2 are set of indices, $\mathbf{a}$ and $\mathbf{b}$ are the vectors of the lower and upper limits of the variables, respectively. The equality and inequality constraints (g_i and h_j) and the lower and upper bounds ($\mathbf{a}$ and $\mathbf{b}$) determine the feasible set of points ($X \subseteq \mathbb{R}^n$) in the n-variable optimization problem. In the above equation, if both I_1 and I_2 are empty sets, the problem is called *unconstrained* provided $\mathbf{a}$ and $\mathbf{b}$ are negative and positive infinity, respectively. However, if vectors $\mathbf{a}$ and $\mathbf{b}$ have finite values, it is called *bound constrained* problem.

2. Summary of tested local and global optimization methods

In this chapter, a short description of the local and global optimization methods, the general procedures and their implementations will be discussed.

2.1. Local and global optimizer points

Usually, two types of local search methods are considered: methods which rely on derivative information and those which are based only on function evaluations. Let $f : S \mapsto \mathbb{R}$ be a function defined on a set S (domain of search space). A point $\mathbf{x}^* \in S$ is called **local optimizer** point of $f(.)$, if there exists a neighborhood $N_{\mathbf{x}^*}$ of $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \le f(\mathbf{x})$ for all $\mathbf{x} \in N_{x^*}$ and the value of $f(\mathbf{x}^*)$ is called the **local optimum** value of f. On the other hand, a point $\mathbf{x}^{**} \in S$ is called a **global optimizer** point of $f(.)$, if $f(\mathbf{x}^{**}) \le f(\mathbf{x})$ for all $\mathbf{x} \in S$. The value $f(\mathbf{x}^{**})$ is called the **global optimum** value of f located at $\mathbf{x}^{**}$.

In an optimization problem, if objective function f has only one local optimizer point in the closure of S, it is also a global optimizer. All convex functions have the property that a local optimum is also a global optimum and they are categorized as uni-modal. However, if the objective function has

more than one local optimizer point (their exact number might not be known) it is classified as multi-modal. These type of test functions are used to test the ability of numerical optimization methods to escape from the valley of a local optimum. Numerical optimization methods that are designed specifically to search for either a nearest local optimizer or the global optimizer point are called as local and global optimization methods, respectively.

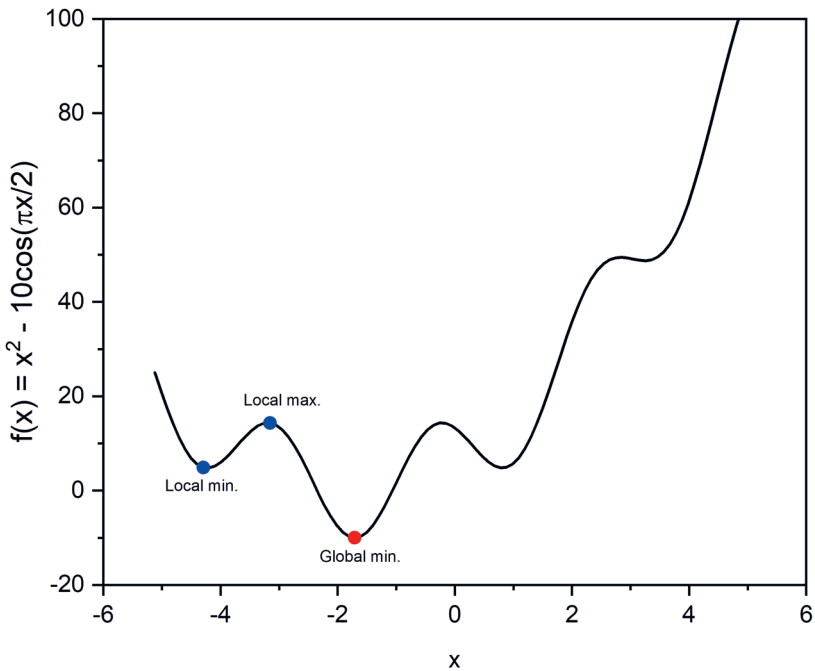

Figure 1. Local and global optimizer points and the corresponding optimum values.

Most of the local and global optimization methods considered in this paper were used from the MATLAB Optimization Toolbox [6] and the MATLAB Global Optimization Toolbox, respectively. The following methods were not present in the MATLAB Toolboxes: FOCTOPUS [39, 40, 9], NEWUOA, BOBYQA [32] and the CMA-ES [8]. In this work, most of the tuning parameters of the selected methods, were used with their default values recommended in the MATLAB Optimization Toolboxes. It is worth mentioning that, optimizing the control parameters of the algorithms may improve their performance, however, this is beyond the scope of this paper. The parameter settings used for the different methods will be discussed in detail in Section 2.4.

2.2. Local optimization methods

Local optimization methods are designed to iteratively search for extrema of a function starting from some initial guess of the variables. In literature, several derivative-based and derivative-free local optimization methods are available in different implementations. In this paper, we investigated the most common local optimization methods from MATLAB Optimization Toolbox and MATLAB implementations of some other popular methods were taken from web pages.

(a) The *Nelder-Mead simplex method*: it was first published by J.A. Nelder and R. Mead [27] for real-valued function minimization tasks. The search is based on a simplex and on some transformations (reflection, expansion, and contraction) in order to explore the search space. It has become one of the most widely used methods for nonlinear unconstrained optimization. This method is a very efficient and robust local search method, especially for small-scale problems [12]. The name of the corresponding Matlab function is fminsearch, and it uses the simplex search method of Lagarias et al. [21]. In this paper we will refer to it as the *simplex* method.

(b) The *pattern search algorithm*: This algorithm belongs to the family of derivative-free optimization techniques that are used to solve non-smooth constrained or unconstrained multivariate optimization problems. The pattern search is a larger search in the improving direction also called pattern direction. The Powell's method tries to discard one direction in each iteration step by replacing it with the pattern direction. One important feature of these algorithms is that they can follow easily the contour lines of the problems having narrow, turning valley-like shapes. Well-known variants of the method are the Hooke–Jeeves algorithm [15] and the NOMAD [22] method based on the Mesh Adaptive Direct Search. The MATLAB codename is patternsearch and we will use the name *pattern search* method.

(c) The *constrained optimization method*: it uses sequential quadratic programming (SQP) that relies on the BFGS formula [28] when updating the Hessian of the Lagrangian. The interior-point variant of the method is a widely used procedure to solve the quadratic programming (QP) sub-problem [28] and it was the default algorithm in the Matlab Optimization Toolbox. The toolbox contains it under the name fmincon for nonlinearly constrained optimization problems and we will refer to it as the *interior-point* method.

(d) The *unconstrained optimization method*: this algorithm is designed specifically for unconstrained optimization problems and it has several variants. Its quasi-Newton variant uses the BFGS formula for updating the

approximation of the Hessian matrix. If the gradient of the objective function is not supplied by the user, the method calculates it by using the finite-difference method. MATLAB provides it as the fminunc routine for solving continuous problems without constraints. fminunc with the BFGS updated procedure has been extensively tested [12] and is the default algorithm in the Matlab Optimization Toolbox. In this paper it will be referred to as the *quasi-Newton* method.

(e) The *New Unconstrained Optimization with Quadratic Approximation* (NEWUOA) method [31] is a derivative-free unconstrained optimization method that iteratively exploits the trust-region technique. The trust region technique is opposite to the line search technique, as the former first chooses a step size, then determines the direction, whereas the latter first determines the direction then the step size. In each iteration, the algorithm minimizes the objective function value within the trust region. One important feature of NEWUOA is the Frobenius norm [10] updating procedure that is applied during the quadratic interpolation process of the model function. In many papers (see e.g. Hansen et al. [12], Rios and Sahinidis [34], Pál [29]) NEWUOA appeared as a reference algorithm and it was considered to be a state-of-the-art solver. NEWUOA was obtained from the NLopt open source nonlinear optimization package [18] through the OPTI Toolbox [6] and it was called through a MATLAB Mex interface.

(f) The *Bound Optimization BY Quadratic Approximation* (BOBYQA) method [32] is a constrained optimization solver without derivatives. It is a bound-constrained variant of the NEWUOA algorithm. Similar to the NEWUOA method, BOBYQA constructs the quadratic models by the Frobenius norm updating process. BOBYQA is suitable for high-dimensional functions and on problems where the function evaluation is expensive. The method appeared as a reference algorithm in many relevant articles and studies (Hansen et al. [12], Rios and Sahinidis [34]). In this paper BOBYQA was tested using the implementation from the NLopt optimization package through the OPTI Toolbox and it was called through a MATLAB Mex interface.

2.3. Global optimization methods

Global optimization methods are designed to search the best possible solution out of all feasible solutions. Depending on the domain of the search space, there could be multiple, even an infinite number of optimal solutions to an optimization problem. In this case, the task of any good global optimization algorithm is to find globally optimal solution or at least a local minimum with similarly low function value.

(a) The FOCusing robusT Optimization with Uncertainty-based Sampling (FOCTOPUS) is a global optimization method that was developed by Tibor Nagy and first published in [39] by Turányi *et al.* This method is an iterative technique based on a sampling distribution, a search domain defined by a prior covariance matrix and a focusing procedure (shrinking or enlarging the search volume). While previously the method was used to minimize the weighted mean-square deviation of model results from experimental data, it is straightforward to apply to any test function in this study.

(b) The *genetic algorithm* (GA) [36] is a heuristic population-based search procedure that mimics the natural selection process. The method relies on mutation, crossover and selection operations, in order to find a solution for multivariate functions without the usual mathematical requirements of strict continuity, differentiability, convexity, and other properties. Das et al. [7] presented a survey of the state-of-art GA methods together with other evolutionary techniques.

(c) The *simulated annealing* (SA) [20] is a stochastic optimization technique inspired by the thermodynamic process of cooling molten metals (initially at high temperature) to attain the lowest free energy state. The method was successfully applied as a tool for single and multiobjective optimization problems. The reader can find detailed descriptions about simulated annealing variants in the paper of Suman and Kumar [37].

(d) The *particle swarm optimization* (PSO) method is a stochastic population based optimization method, first published by Kennedy and Eberhart [19]. The standard particle swarm optimizer includes a swarm of particles that represent the potential solutions to the problem on hand. PSO belongs to the most studied evolutionary techniques. A review of the method was provided by Barrera and Coello [5].

(e) The *covariance matrix adaptation evolutionary strategy* (CMA-ES) method was proposed by Hansen and Ostermeier [11] for nonlinear, non-convex function optimization. It is considered as a state-of-the-art evolutionary method, and it has been applied in many different research fields. The primary feature of the CMA-ES is the adaptation of the covariance matrix through an iterative procedure so that the obtained covariance matrix is similar to the inverse Hessian of the objective function. Detailed benchmarking results of the CMA-ES algorithm were published by Hansen et al. [12]. The MATLAB implementation was obtained from the GitHub site of the method [8].

2.4. Parameter settings

Generally speaking, optimization algorithms have several control parameters (i.e. tuning parameters or settings) and their performance depends on the user-selected values of these parameters. Well-tuned control parameters provide optimal optimization performance, but how to find these well-tuned settings for an algorithm is an open question for researchers. Optimization of tuning parameters of optimization methods in practical applications were discussed in [42, 14, 1].

In this work, the tested numerical optimization methods were taken from the MATLAB Optimization Toolbox and some Matlab codes were taken from the authors' web sites. The algorithm parameters were pretuned by the authors so as to get optimal efficiency for most problems. Nevertheless, one can optionally redefine these parameter settings using the optimOption keyword.

In this paper, the local optimization methods were tested using their default settings unless they are specified. The *fminunc* and *fmincon* methods (in this case, interior-point and quasi-Newton solvers) usually stop if the termination tolerance on x (TolX) or the termination tolerance on the function value (TolFun) is smaller than 1e-6. Furthermore, the maximum number of function evaluations allowed is 100*NumberOfVariables. The maximum number of iterations for the interior-point algorithm is 1000 and for all the other algorithms is 400. The fminsearch (simplex method) algorithm stops when it satisfies both TolFun and TolX which we decreased to 1e-8 from their default value of 1e-4. The maximum number of function evaluations allowed is 100*NumberOfVariables. The pattern search solver stops when the mesh size and the TolX or TolFun are smaller than 1e-6. The maximum number of function evaluations allowed is 2000*NumberOfVariables and the maximum number of iterations is 100*NumberOfVariables. The NEWUOA and BOBYQA methods stop when the trust region radius becomes smaller than 1e-6 or the number of function evaluations reaches 100.

Using the FOCTOPUS method, the NumberOfSamples parameter was set to 100. The stopping criterium was achieved when the investigated volume of the parameter space was reduced to 1/10000 of the starting volume. It is achieved with 100 samples when the FocusParameter = 2*NumberOfParameters. We set the ForwardStep and BackwardSteps to 1 and 2, respectively. The NumberOfFunctionEvaluation = NumberOfSamples*NumberOfIterations.

The other global optimization algorithms had many settings. In our test, we used the default settings provided by the MATLAB Global Optimization Toolbox, nevertheless, we list the most important ones here. The genetic algorithm (GA) used a population size of 50 or 200 depending on the number of variables. The crossover rate was 0.8 and the maximal number of generations was 100*NumberOfVariables. The algorithm stopped if the average relative

change in the best fitness function value over 50 generations was less than or equal to 1e-6.

For the simulated annealing (SA) method, the maximum number of objective function evaluations was 3000*NumberOfVariables, while no limit was set on the number of iterations. The termination tolerance on function value was set to 1e-6.

The particle swarm optimization (PSO) algorithm, we used a swarm size of 100 in all cases. The termination tolerance on function value was 1e-6. The algorithm also stopped when it achieved 200*NumberOfVariables iterations.

3. Artificial test functions

Test functions are important to validate and compare the performances of local and global optimization methods. They include artificial and real-life problem types in which the second is obtained from applications in science, engineering and economics. In the work of Goitom [9], several local and global optimization methods were tested on a real-life optimization problem. However, in the present work, only artificial test functions that have a known optimal global minimum (or minima) are selected for the test.

In the literature, several test functions with different characterizations are available. The characterizations that include: modality, basins, valleys, separability, dimensionality, convexity, linearity, differentiability determine the optimization complexity. Convex test functions are among the easiest problems to test any optimization methods.

A collection of several unconstrained test functions with different characterizations from different sources were reviewed and compiled in [16]. An extended form of some standard test functions are available in [2]. More challenging test problems in science, engineering and economics can be found in [23]. In references [38, 24, 16], several analytic test functions that are traditionally used for testing local and global optimization methods are available. In the present study, the following test functions were used for benchmarking the optimization algorithms. For those functions, where a 2-variable variant exist, a graphical representations of it is shown in the following figures to visualize the distribution and nature of their local minima. For the more complex test functions, we used a low-dimensional variant, because otherwise the computational cost would be overly high and most methods would probably fail to converge.

- The *Alpine function* [33] is a continuous, non-differentiable, separable and multi-modal function, whose d-variable variant is defined by:

$$f(\mathbf{x}) = \sum_{i=1}^{d} |x_i \sin(x_i) + 0.1x_i| \tag{3.1}$$

subject to $-10 \leq x_i \leq 10$ for all $i = 1, 2, ..., d$. The global minimum is $f(\mathbf{x}^*) = 0$ which is located at $\mathbf{x}^* = (0, 0, ..., 0)$. In our study, we used the 20-dimensional (20D) Alpine function (i.e. $d = 20$).

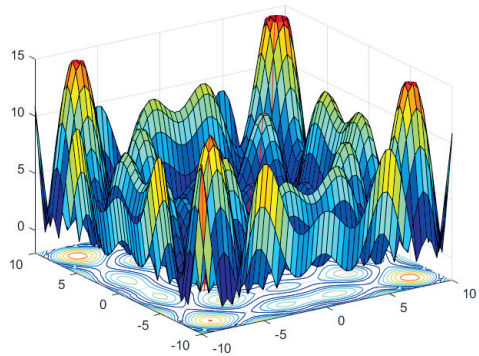

Figure 2. Surface plot of the 2D Alpine function.

- The *Ackley function* [3] in d-variables is given by the equation:
(3.2)

$$f(\mathbf{x}) = 20 + \exp(1) - 20\exp\left[-\frac{1}{5}\left(\frac{\sum_{i=1}^{d} x_i^2}{d}\right)^{0.5}\right] - \exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right]$$

where $\mathbf{x} \in [-32.768,\ 32.768]^{\mathrm{d}}$. This function is continuous, differentiable, non-separable and multi-modal. It has a single global minimum at $\mathbf{x}^* = = (0, 0, ..., 0)$ with a value $f(\mathbf{x}^*) = 0$. In our study, we used the 4D Ackley function.

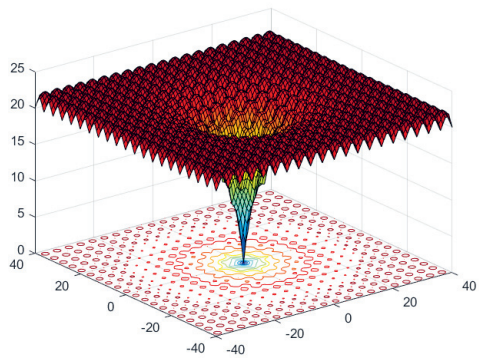

Figure 3. Surface plot of the 2D Ackley function.

- *Cross-in-tray function* [24] is a 2-variable test function that has several local and four global minimum points with identical objective function values. It is a continuous, non-differentiable, non-separable and multi-modal function. This function is given by the formula:

$$f(\mathbf{x}) = -0.0001 \left[\left| \sin(x_1)\sin(x_2)\mathrm{e}^{\left|100-\left(x_1^2+x_2^2\right)^{0.5}/\pi\right|} \right| + 1 \right]^{0.1}. \tag{3.3}$$

It is subject to $-10 \leq x_i \leq 10$. The four global minimum points are located at $\mathbf{x}^* \approx (\pm 1.3494, \pm 1.3494)$ with $f(\mathbf{x}^*) \approx -2.0626$.

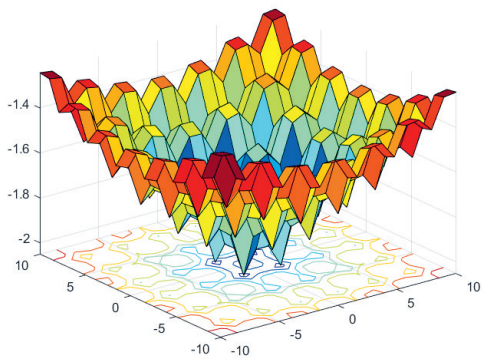

Figure 4. Surface plot of the cross-in-tray (2D) function.

- The *Hartmann 6-D function* [13] is a 6-variable, continuous, differentiable, non-separable and multi-modal function, defined by:

$$f(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij} \left(x_j - p_{ij} \right)^2 \right], \tag{3.4}$$

and it is subject to $0 \leq x_j \leq 1$, $j = \{1, 2, ..., 6\}$ with constants c_i, a_{ij} and p_{ij} are given as:

$$\mathbf{c}_i = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}, \mathbf{a}_{ij} = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 4 \end{pmatrix}, \tag{3.5}$$

$$\mathbf{p}_{ij} = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5586 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}. \tag{3.6}$$

This function has six local minima with a single global minimum value of $f(\mathbf{x}^*) \approx -3.32237$ located at

$$\mathbf{x}^* \approx (0.201690, 0.150011, 0.476874, 0.275332, 0.311652, 0.657300). \tag{3.7}$$

- The *Holder table function* [24] is a 2-variable test function that has several local and four global minimum points with identical objective function values. It is continuous, differentiable, separable and multi-modal. This function is given as:

$$f(\mathbf{x}) = -\left|\sin(x_1)\cos(x_2)\mathrm{e}^{\left|1-(x_1^2+x_2^2)^{0.5}/\pi\right|}\right| \tag{3.8}$$

subject to $-10 \leq x_i \leq 10$. The four global minima points are located at $\mathbf{x}^* \approx (\pm 8.055, \pm 9.665)$ with $f(\mathbf{x}^*) \approx -19.2085$

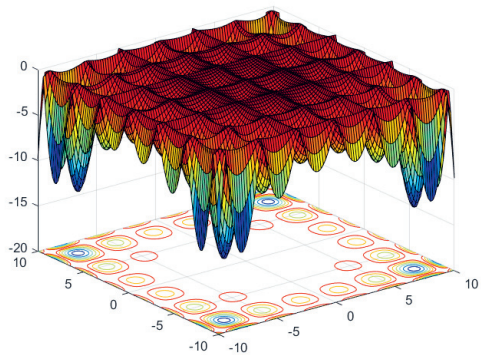

Figure 5. Surface plot of the Holder table function (2D).

- The *Rastrigin function* [25] is a typical multi-modal test function, whose d-dimensional ($d = 1, 2, ...$) variant is defined as:

$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left(x_i^2 - 10\cos\left(2\pi x_i\right)\right) \tag{3.9}$$

Rastrigin functions have a global minimum value of $f(\mathbf{x}^*) = 0$ corresponding to point $\mathbf{x}^* = (0, 0, ..., 0)$. Its search domain is defined as: $-5.12 \leq x_i \leq 5.12$ ($i = 1, 2, ..., d$). It is a difficult test function for most global optimization methods.

- The *Rosenbrock function* [35] is an uni-modal valley-shaped function defined for $d \geq 2$ dimensions by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^{d-1} 100\left[\left(x_i^2 - x_{i-1}\right)^2 + \left(x_i - 1\right)^2\right]. \tag{3.10}$$

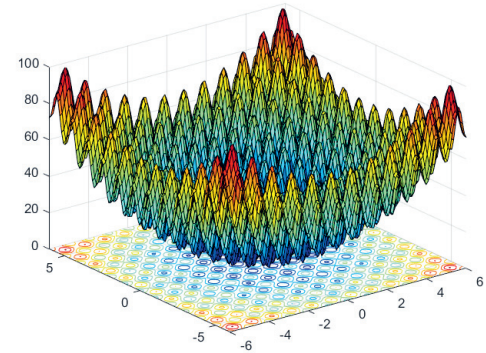

Figure 6. Surface plot of the 2D Rastrigin function.

It has a single local minimum (i.e. also global) with a value of $f(\mathbf{x}^*) = 0$ located at $\mathbf{x}^* = (1, 1, ..., 1)$. Its search domain is: $-5 \leq x_i \leq 10$, $(i = 1, 2, ..., d)$. It is a continuous, differentiable, non-separable, non-convex and uni-modal function. The function is best known for having very slow convergence in the neighborhood of the minimum point.

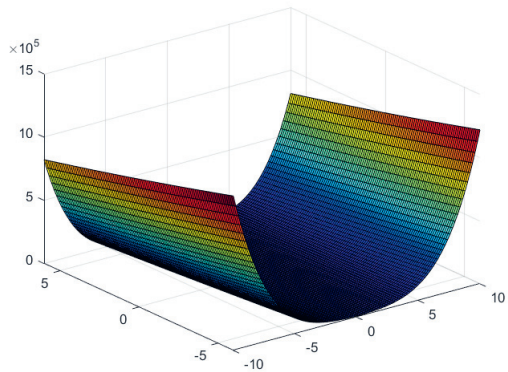

Figure 7. Surface plot of the 2D Rosenbrock function.

- *Modified Rosenbrock function* [30] is an alternative form of the standard 4D Rosenbrock function defined on the domain $[0, 1]^4$. It's defined as: (3.11)

$$f(\mathbf{x}) = \frac{1}{3.755 \times 10^5} \sum_{i=1}^{3} \left[100 \left(\bar{x}_{i+1} - \bar{x}_i^2 \right)^2 + \left(1 - \bar{x}_i \right)^2 - 3.827 \times 10^5 \right]$$

where $\bar{x}_i = 15x_i - 5$ for $i = 1, 2, 3, 4$. This function has a global minimum value of $f(\mathbf{x}^*) \approx -1.0192$.

- The *Zakharov function* [33] is a plate-shaped function, defined in d dimensions ($d = 1, 2, ...$) with the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2 + \left[\sum_{i=1}^{d} i x_i/2\right]^2 + \left[\sum_{i=1}^{d} i x_i/2\right]^4 . \tag{3.12}$$

It is defined in the search domain $-5 \leq x_i \leq 10$; $(i = 1, 2, ..., d)$ and it has a single minimum $f(\mathbf{x}^*) = 0$ located at $\mathbf{x}^* = (0, 0, ..., 0)$.

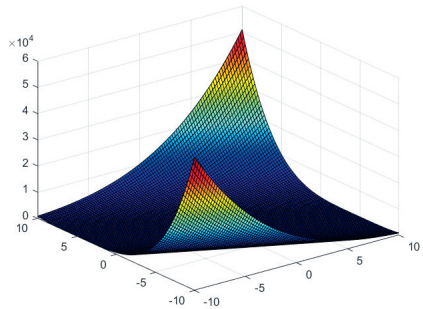

Figure 8. Surface plot of the 2D Zakharov function.

- A *typical multi-modal test function* [24] is a 2-dimensional non-convex function defined by:

$$f(\mathbf{x}) = -\cos(x_1)\cos(x_2)\exp\left(-\frac{1}{4}\sqrt{x_1^2 + x_2^2}\right). \tag{3.13}$$

This function has a global minimum value of $f(\mathbf{x}^*) = -1$, located at $\mathbf{x}^* = (0, 0)$.

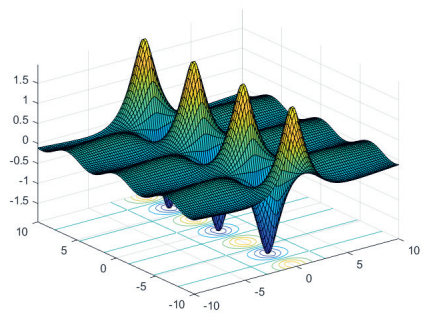

Figure 9. Surface plot of the 2D multi-modal function.

4. Benchmark of the optimization methods

In this section, the performance of the discussed numerical optimization methods on the artificial functions is compared. In each case (i.e. method and function combination), function minimization runs were carried out starting from 50 randomly selected initial points in the search space of the test function. The comparisons were carried on the following metrics: average and median function value (f), mean CPU time, average number of function evaluation ($feval$) and the success rate of the 50 optimizations. For each test function, the metrics obtained for the 11 optimization methods are summarized in Tables 1–10. An optimization starting from a given set of initial conditions is considered to be successfully convergent to the global optimum if $|f(\mathbf{x}) - f(\mathbf{x}^*)| \leq 10^{-4}$ where $f(\mathbf{x}^*)$ and $f(\mathbf{x})$ are the known exact global minimum and lowest found value of the objective function, respectively. Otherwise, the optimization is considered to be unsuccessful. For fully unsuccessful cases (i.e. success rate is 0%), the average $feval$ is not reported.

4.1. Results

Table 1 shows the results obtained for the 20D Alpine function. In this test, the pattern search and the PSO methods always converged to the global minimum, the GA and the quasi-Newton algorithms converged in 90% and 74%, respectively, the interior-point and CMA-ES had very low success rates and finally, the simplex, NEWUOA, BOBYQA, FOCTOPUS and the SA methods always failed to converge.

Table 1. Result statistics of 50 minimization runs of the 20D Alpine function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** $feval$	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.27	3846	0.058	0.0085	14
Quasi-Newton	1.2E-4	5019	0.11	3.6E-5	74
Simplex	0.034	-	13	12	0
Pattern search	0.34	10785	2.2E-4	2.2E-4	100
NEWUOA	0.32	-	5.1	3.5	0
BOBYQA	0.11	-	11	10	0
FOCTOPUS	0.32	-	0.54	0.48	0
GA	0.68	28907	4.2E-4	2.9E-4	90
SA	3.0	-	11	11	0
PSO	0.15	16063	4.7E-6	5.5E-7	100
CMA-ES	2.9	36085	0.077	0.0027	28

Table 2 shows the results obtained for the 4D Ackley function. The pattern search and the GA methods had converged successfully to the global optimum, whereas the interior-point, the quasi-Newton, the simplex and the SA methods

failed to converge in all runs. The FOCTOPUS, the PSO and the CMA-ES methods converged in 88%, 98% and 68% of the total runs, respectively, whereas the NEWUOA and BOBYQA methods had very low success rates.

Table 2. Result statistics of 50 minimization runs of the 4D Ackley function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.039	-	19	19	0
Quasi-Newton	0.0094	-	19	20	0
Simplex	0.0052	-	19	20	0
Pattern search	0.044	627	7.2E-5	6.9E-5	100
NEWUOA	0.0016	201	12	19	20
BOBYQA	0.0015	222	15	19	8
FOCTOPUS	0.090	42301	0.22	2.9E-5	88
GA	0.14	6340	3.2E-6	2.9E-6	100
SA	0.50	-	5.4	5.5	0
PSO	0.042	5396	0.037	2.2E-8	98
CMA-ES	0.91	1701	4.4	3.4E-11	76

Table 3 shows the results obtained for the Cross-in-tray 2D function. The pattern search, the FOCTOPUS, the GA and the PSO methods converged successfully in all runs. The CMA-ES and SA converged in 94% and 74% of the total runs, and the remaining methods converged with a very low success rate.

Table 3. Result statistics of 50 minimization runs of the Cross-in-tray 2D function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.028	49	-1.7459	-1.7155	14
Quasi-Newton	0.0098	36	-1.7427	-1.7155	12
Simplex	0.0033	133	-1.7202	-1.7155	12
Pattern search	0.028	207	-2.0626	-2.0626	100
NEWUOA	6.1E-4	36	-1.7938	-1.7155	24
BOBYQA	0.0018	35	-1.7990	-1.7155	22
FOCTOPUS	0.16	136201	-2.0626	-2.0626	100
GA	0.073	2709	-2.0626	-2.0626	100
SA	0.24	1766	-2.0611	-2.0625	74
PSO	0.020	2076	-2.0626	-2.0626	100
CMA-ES	0.58	541	-2.0433	-2.0626	94

Table 4 shows the results of obtained for the Hartman-6D function. In this test, all the local methods and the CMA-ES had a good success rate, but the

PSO and the FOCTOPUS methods converged only in 36% and 14%, whereas the GA and SA methods failed to converge in all runs.

Table 4. Result statistics of 50 minimization runs of the Hartmann-6D function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.050	291	-3.2914	-3.3224	74
Quasi-Newton	0.027	266	-3.2914	-3.3224	74
Simplex	0.021	824	-3.2914	-3.3224	74
Pattern search	0.10	1307	-3.2961	-3.3224	78
NEWUOA	0.0051	120	-3.2914	-3.3224	74
BOBYQA	0.0082	208	-3.2890	-3.3224	72
FOCTOPUS	1.4	97741	-3.1920	-3.2009	14
GA	0.12	-	-2.6484	-2.8122	0
SA	0.88	-	-3.2939	-3.2977	0
PSO	0.13	6606	-3.2461	-3.2032	36
CMA-ES	0.95	1641	-3.2842	-3.3224	68

Table 5 shows the results obtained for the Holder table 2D function. In this calculation, the GA and the PSO converged successfully in all runs, but the FOCTOPUS failed to converge in all runs. The SA and the pattern search methods converged respectively in 98% and 72%, the CMA-ES converged in half of the cases, whereas the other remaining methods converged with very low success rates.

Table 5. Result statistics of 50 minimization runs of the Holder table 2D function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.032	46	-8.3672	-8.0951	10
Quasi-Newton	0.0074	15	-7.2838	-4.7125	12
Simplex	0.0031	122	-7.6160	-4.7125	14
Pattern search	0.027	191	-16.0777	-19.2085	72
NEWUOA	0.0014	33	-12.1372	-9.5047	14
BOBYQA	0.0022	51	-13.8551	-16.2678	28
FOCTOPUS	0.18	-	-1.9797	-1.7330	0
GA	0.079	2936	-19.2085	-19.2085	100
SA	0.26	1969	-19.2084	-19.2085	98
PSO	0.020	2150	-19.2085	-19.2085	100
CMA-ES	0.69	646	-17.6479	-19.2085	52

Table 6 shows the results obtained for the 5D Rastrigin function. The pattern search and the GA methods successfully converged to the global opti-

mum in all runs. However, the PSO converged only in 20%, whereas all other methods failed to converge to the global optimum in all runs.

Table 6. Result statistics of 50 minimization runs of the 5D Rastrigin function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.059	-	44	39	0
Quasi-Newton	0.012	-	43	37	0
Simplex	0.0053	-	45	41	0
Pattern search	0.057	975	4.7E-9	4.6E-9	100
NEWUOA	0.0012	-	39	37	0
BOBYQA	0.0018	-	39	37	0
FOCTOPUS	0.10	-	5.2	5.0	0
GA	0.14	6613	2.4E-7	9.5E-8	100
SA	0.63	-	4.7	5.0	0
PSO	0.054	7670	1.6	1.0	20
CMA-ES	1.9	-	6.5	6.0	0

Table 7 shows results obtained for the 5D Rosenbrock function. The GA method had the highest success rate (96%) followed by the CMA-ES, quasi-Newton, simplex, NEWUOA, BOBYQA, the interior-point and the FOCTOPUS (in the order of 94%,88%, 78%, 76%, 72%, 72% and 56%). However, the SA method converged only in 2% of the calculations, the PSO and pattern search methods had also fairly low success rates.

Table 7. Result statistics of 50 minimization runs of the 5D Rosenbrock function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.079	390	1.1	1.4E-11	72
Quasi-Newton	0.020	558	0.47	3.7E-10	88
Simplex	0.0091	1169	0.86	4.2E-17	78
Pattern search	1.7	35427	1.8	0.098	42
NEWUOA	0.0064	575	1.1	3.7E-12	72
BOBYQA	0.010	997	0.86	1.7E-11	76
FOCTOPUS	1.5	967941	1.7	4.2E-5	56
GA	6.9	376053	0.0020	1.9E-5	96
SA	1.1	7900	1.8	0.97	2
PSO	1.6	295290	0.19	0.0020	40
CMA-ES	1.1	2942	0.39	5.3E-16	90

Table 8 shows the results obtained for the 4D modified Rosenbrock function. All the methods successfully converged to the global minimum except the

GA, which converged in 94% of the runs. The FOCTOPUS method required high number of function evaluations, whereas the local methods converged very quickly with a very low number of function evaluations.

Table 8. Result statistics of 50 minimization runs of the 4D modified Rosenbrock function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.094	492	-1.0192	-1.0192	100
Quasi-Newton	0.025	415	-1.0192	-1.0192	100
Simplex	0.0089	629	-1.0192	-1.0192	100
Pattern search	1.7	30911	-1.0192	-1.0192	100
NEWUOA	0.0053	345	-1.0192	-1.0192	100
BOBYQA	0.0063	447	-1.0192	-1.0192	100
GA	0.12	2818	-1.0192	-1.0192	94
FOCTOPUS	0.44	99961	-1.0191	-1.0192	100
SA	0.55	3855	-1.0192	-1.0192	100
PSO	0.036	2516	-1.0192	-1.0192	100
CMA-ES	1.6	1838	-1.0192	-1.0192	100

Table 9 shows the results obtained for the 20D Zakharov function. The interior-point, the NEWUOA, the FOCTOPUS and the GA methods successfully converged to the global optimum in all the runs. The simplex, the pattern search and the SA methods failed to converge in all runs, whereas the other remaining methods had very low success rates.

Table 9. Result statistics of 50 minimization runs of the 20D Zakharov function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	8.5E-2	1170	1.9E-14	5.0E-15	100
Quasi-Newton	0.020	987	141	7.2E-2	20
Simplex	0.34	-	52	19	0
Pattern search	1.2	-	105	98	0
NEWUOA	0.14	4526	4.0E-9	2.2E-9	100
BOBYQA	0.49	8978	40	15	18
FOCTOPUS	0.11	358861	1.6E-8	1.5E-8	100
GA	1.1	46817	9.4E-5	8.5E-5	100
SA	3.7	-	4.4	3.8	0
PSO	0.18	18115	49	5.2E-2	44
CMA-ES	1.2	5006	1.9E-2	8.1E-3	10

Table 10 shows the results obtained for a typical multi-modal 2D function. This is a highly multi-modal function where, most of the methods failed to converge in all runs. However, the FOCTOPUS and the PSO methods converged successfully in all runs followed by the GA which converged in 80% of the total runs. The CMA-ES and the pattern search methods converged only in 14% and 6% of the runs, respectively.

Table 10. Result statistics of 50 minimization runs of a typical multi-modal 2D function obtained for the different numerical methods

Method	**Mean CPU time (sec)**	**Mean** *feval*	**Mean** f	**Median** f	**Success rate (%)**
Interior-point	0.042	-	-9.1E-4	-8.1E-11	0
Quasi-Newton	0.019	-	-9.1E-4	-8.1E-11	0
Simplex	0.0029	-	-9.1E-4	-2.0E-9	0
Pattern search	0.026	213	-0.15	-0.020	6
NEWUOA	5.3E-4	-	-2.5E-3	-9.7E-9	0
BOBYQA	7.0E-4	-	-9.1E-3	-9.7E-9	0
FOCTOPUS	0.030	22621	-1.0	-1.0	100
GA	0.085	3469	-0.86	-1.0	80
SA	0.22	-	-0.24	-0.18	0
PSO	0.031	3509	-1.0	-1.0	100
CMA-ES	0.62	947	-0.23	-0.0036	18

4.2. Discussions

Altogether 6 local and 5 global optimization methods were benchmarked on 10 test functions. The corresponding results were summarized in Tables 1 to 10. From the tabular results, significant differences can be observed in the metrics of success rate and number of function evaluations of the various optimization methods.

In comparison with the benchmark study in ref. [4], the conclusions of the present work are different regarding the performances of the simplex and NEWUOA methods for the common test function of the same dimensionality (e.g. Hartmann-6D, Rosenbrock 5D). In the case of the Ackley and Rastrigin functions, different dimensions (5D vs. 4D) were used in the two studies, hence, the results cannot be compared.

In the present paper, it was observed that, the local optimization methods had high failure rates on the highly multi-modal and high-dimensional test functions. Exceptions to this were the pattern search method which had a high success rate on the 20D Alpine, the 5D Rastrigin and the 4D Ackley functions, furthermore, the quasi-Newton method which had a high success rate on the 20D Alpine function, and also the NEWUOA method which had a high success rate on the 20D Zakharov function.

In the case of the global methods, the PSO and GA methods had high failure rates for the Hartmann-6D function, whereas the CMA-ES method had high success rates on the Hartmann-6D and valley-shaped Rosenbrock functions. The SA method converged successfully on most of the low-dimensional test functions, however, it failed in all runs on the 2D multi-modal function. The FOCTOPUS method failed to converge on some multi-modal functions and it required very high number of function evaluations on most functions. The GA, the PSO and the pattern search methods also had a high number of function evaluations for the valley-shaped Rosenbrock function.

5. Conclusion

In overall, the pattern search method converged well to the global optimum in all the functions except for the Zakharov (20D, valley-shaped) function and it had a low success rate on the 5D Rosenbrock (plate-shaped) function and the typical 2D multi-modal functions. The other local optimization methods either failed or had very low success rates on the multi-modal and high-dimensional test functions. The only exception is the NEWUOA method which converged successfully for the 20D Zakharov function.

The genetic algorithm (GA) usually had a high success rate on all the test functions except for the Hartmann-6D function. The particle swarm optimization (PSO) method had a low success rate on the Rastrigin and Hartmann-6D functions, a fairly good success rate on the Zakharov and Rosenbrock functions and always converged on the rest of the test functions. The SA converged successfully only on the low-dimensional (2D) test functions, whereas the CMA-ES method had good success rates only on the Rosenbrock and Hartmann-6D functions. The FOCTOPUS method failed to converge for the Alpine, Holder table and Rastrigin functions and had a low success rate for the Hartmann-6D function, an acceptable success rate for the Rosenbrock function, and successfully converged on the rest of the test functions. In terms of the function evaluation and run time, the FOCTOPUS method required high number of function evaluations and the largest amount of CPU time. Furthermore, the convergence of the GA method was also very slow on average. The PSO method had a positive success rate in all the benchmarks and it had a higher convergence speed than the other global methods in almost all benchmarks. This investigation made it clear that the FOCTOPUS method while being tailored for and proved to be a robust method for combustion kinetic problems, fails to converge efficiently to the global minimum value for several artificial test functions, thus its application cannot be recommended in general.

Acknowledgments. The authors are grateful to Drs. László Pál, István Gy. Zsély and Éva Valkó and Mr. Máté Papp and Mr. Márton Kovács for helpful discussions. The authors thank the financial support of grants K132109 (T.T.) and FK 134332 (T.N.) of the Hungarian National Research, Development and Innovation Office.

References

[1] **Al-Thanoon, N.A., O.S. Qasim and Z.Y. Algamal,** A new hybrid firefly algorithm and particle swarm optimization for tuning parameter estimation in penalized support vector machine with application in chemometrics. *Chemometrics And Intelligent Laboratory Systems*, **184** (2019), 142–152.

[2] **Andrei, N.,** An unconstrained optimization test functions collection. *Adv. Model. Optim*, **10** (2008), 147–161.

[3] **Bäck, T. and H. Schwefel,** An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, **1** (1993), 1–23.

[4] **Bánhelyi, B., T. Csendes, B. Lévai, L. Pál and D. Zombori,** *The Global Optimization Algorithm: Newly Updated with Java Implementation and Parallelization*, Springer, 2018.

[5] **Barrera, J. and C.A.C. Coello**, A review of particle swarm optimization methods used for multimodal optimization. *Innovations In Swarm Intelligence*, **248** (2009), 9–37.

[6] **Currie, J. and Wilson D.,** OPTI: Lowering the barrier between open source optimizers and the industrial MATLAB user. *Foundations Of Computer-aided Process Operations*, (2012), paper 24.

[7] **Das, S., S. Maity, B. Qu and P.N. Suganthan,** Real-parameter evolutionary multimodal optimization — A survey of the state-of-the-art. *Swarm And Evolutionary Computation*, **1** (2011), 71–88.

[8] **Heris, M.K.,** CMA-ES in MATLAB (2020) https://github.com/smkalami/ypea108-cma-es

[9] **Goitom, S., M. Papp, M. Kovács, T. Nagy, I.G. Zsély, T. Turányi,** Efficient numerical methods for the optimization of large kinetic reaction mechanisms. *Combustion Theory And Modelling*, (2022) https://www.tandfonline.com/doi/full/10.1080/13647830.2022.2110945

[10] **Golub, G.H. and C.F. Van Loan,** *Matrix Computations (4th ed.)*, The Johns Hopkins University Press, Baltimore, 2013.

[11] **Hansen, N. and A. Ostermeier,** Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, **9** (2001), 159–195.

[12] **Hansen, N., A. Auger, R. Ros, S. Finck and P. Pošik,** Comparing Results of 31 Algorithms from the Black-Box Optimization Benchmarking BBOB-2009. *Proceedings Of The 12th Annual Conference Companion On Genetic And Evolutionary Computation*, (2010), 1689–1696.

[13] **Hartman, J.,** Some experiments in global optimization. *Naval Research Logistics Quarterly*, **20** (1973), 569–576.

[14] **Hong, Y.Y. and W. Wu,** A new approach using optimization for tuning parameters of power system stabilizers. *IEEE Transactions On Energy Conversion*, **14** (1999), 780–786.

[15] **Hooke, R. and T.A. Jeeves,** "Direct Search" Solution of Numerical and Statistical Problems. *Journal Of The ACM (JACM)*, **8** (1961), 212–229.

[16] **Jamil, M. and X. Yang,** A literature survey of benchmark functions for global optimisation problems. *International Journal Of Mathematical Modelling And Numerical Optimisation*, **4** (2013), 150–194.

[17] **Järvi, T.,** A random search optimizer with an application to a max-min problem: A special trajectory estimation problem, University of Turku, 1973.

[18] **Johnson, S.,** The NLopt nonlinear-optimization package (2021) https://github.com/stevengj/nlopt

[19] **Kennedy, J. and R. Eberhart,** Particle swarm optimization. *Proceedings Of ICNN'95 - International Conference On Neural Networks*, **4** (1995), 1942–1948.

[20] **Kirkpatrick, S., C.D. Gelatt and M.P. Vecchi,** Optimization by simulated annealing. *Science*, **220** (1983), 671–680.

[21] **Lagarias, J.C., J.A. Reeds, M.H. Wright and P.E. Wright,** Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM Journal On Optimization*, **9** (1998), 112–147.

[22] **Le Digabel, S.**, Algorithm 909: NOMAD: Nonlinear Optimization with the MADS Algorithm. *ACM Trans. Math. Softw.*, **37** (2011).

[23] **Mehta, D. and C. Grosan,** A collection of challenging optimization problems in science, engineering and economics. *2015 IEEE Congress On Evolutionary Computation*, (2015), 2697–2704.

[24] **Mishra, S.K.,** Performance of repulsive particle swarm method in global optimization of some important test functions: A fortran program. *Available At SSRN 924339*, (2006).

[25] **Mühlenbein, H., M. Schomisch and J. Born,** The parallel genetic algorithm as function optimizer. *Parallel Computing*, **17** (1991), 619–632.

[26] **Nagy, T., É. Valkó, I. Sedyó, I. G. Zsély, M. Pilling and T. Turányi,** Uncertainty of the rate parameters of several important elementary reactions of the H2 and syngas combustion systems. *Combustion And Flame*, **162** (2015), 2059–2076.

[27] **Nelder, J.A. and R. Mead,** A Simplex Method for Function Minimization. *The Computer Journal*, **7** (1965), 308–313.

[28] **Nocedal, J. and S.J. Wright,** *Numerical Optimization*, Springer, New York, 2006.

[29] **Pál, L.,** Empirical study of the improved UNIRANDI local search method. *Central European Journal Of Operations Research*, **25** (2017), 929–952.

[30] **Picheny, V., T. Wagner and D. Ginsbourger,** A benchmark of kriging-based infill criteria for noisy optimization. *Structural And Multidisciplinary Optimization*, **48** (2013), 607–626.

[31] **Powell, M.J.D.,** The NEWUOA software for unconstrained optimization without derivatives, in: G. Di Pillo, and M. Roma (Eds.) *Large-scale Nonlinear Optimization*, **83** (2006), 255–297.

[32] **Powell, M.J.D.,** The BOBYQA algorithm for bound constrained optimization without derivatives, Report DAMTP 2009/NA06, University Of Cambridge, (2009).

[33] **Rahnamayan, S., H.R. Tizhoosh and M.M.A. Salama,** A novel population initialization method for accelerating evolutionary algorithms. *Computers and Mathematics With Applications*, **53** (2007), 1605–1614.

[34] **Rios, L.M. and N.V. Sahinidis,** Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal Of Global Optimization*, **56** (2013), 1247–1293.

[35] **Rosenbrock, H.H.,** An automatic method for finding the greatest or least value of a function. *The Computer Journal*, **3** (1960), 175–184.

[36] **Salomon, R.,** Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. *BioSystems*, **39** (1996), 263–278.

[37] **Suman, B. and P. Kumar,** A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal Of The Operational Research Society*, **57** (2006(10)), 1143–1160.

[38] **Surjanovic S. and D. Bingham,** Virtual Library of Simulation Experiments: Test Functions and Datasets (2013)
http://www.sfu.ca/~ssurjano

[39] **Turányi, T., T. Nagy, I.G. Zsély, M. Cserháti, T. Varga, B. Szabó, I. Sedyó, P. Kiss, A. Zempléni and H. Curran,** Determination of rate parameters based on both direct and indirect measurements. *International Journal Of Chemical Kinetics*, **44** (2012), 284–302.

[40] **Varga, T., I.G. Zsély, T. Turányi, T. Bentz and M. Olzmann**, Kinetic analysis of ethyl iodide pyrolysis based on shock tube measurements. *International Journal Of Chemical Kinetics*, **46** (2014), 295–304.

[41] **Varga, T., C. Olm, T. Nagy, I.G. Zsély, É. Valkó, R. Pálvölgyi, H. Curran, T. Turányi**, Development of a joint hydrogen and syngas combustion mechanism based on an optimization approach. *International Journal Of Chemical Kinetics*, **48** (2016), 407–422.

[42] **Yang, X., S. Deb, M. Loomes and M. Karamanoglu**, A framework for self-tuning optimization algorithm. *Neural Computing And Applications*, **23** (2013), 2051–2057.

S. Goitom
Institute of Chemistry
Eötvös Loránd University
Institute of Mathematics
Eötvös Loránd University
Budapest
Hungary
simretab1222@gmail.com

T. Nagy
Institute of Materials and Environmental Chemistry
Research Centre for Natural Sciences
Budapest
Hungary
nagy.tibor@ttk.hu

T. Turányi
Institute of Chemistry
Eötvös Loránd University
Budapest
Hungary
turanyi@chem.elte.hu