NeuralMinimizer, a novel method for global optimization that incorporates machine learning

Ioannis G. Tsoulos[(1)*], Alexandros Tzallas[(1)], Evangelos Karvounis[(1)], Dimitrios Tsalikakis[(2)]

[(1)]Department of Informatics and Telecommunications, University of Ioannina, 47100 Arta, Greece
[(2)]University of Western Macedonia, Department of Engineering Informatics and Telecommunications, Greece

Abstract

The problem of finding the global minimum of multidimensional functions is often applied to a wide range of problems. An innovative technique for finding the global minimum is proposed in this paper. In this technique, an estimation of the objective function is constructed using an RBF network. The network constantly updates the estimation for the objective function with information it gathers from performing local minimizations. In addition, the proposed technique uses as a termination criterion a widely used criterion from the relevant literature which in fact evaluates it after each execution of the local minimization. The proposed technique was applied to a number of well-known problems from the relevant literature, and the comparative results with respect to modern global minimization techniques are extremely promising.

1 Introduction

An innovative method for finding the global minimum of multidimensional functions is presented here. The functions considered are continuous and differentiable function and defined as $f : S \to R, S \subset R^n$. The problem of locating the global optimum is usually formulated as:

$$x^* = \arg\min_{x \in S} f(x) \tag{1}$$

with S:

$$S = [a_1, b_1] \otimes [a_2, b_2] \otimes \ldots [a_n, b_n]$$

*Corresponding author. Email: itsoulos@uoi.gr

A variety of problems in the physical world can be represented as global minimum problems, such as problems from physics [?, ?, ?], chemistry [?, ?, ?], economics [?, ?], medicine [?, ?] etc. During the past years many methods, especially stochastic one, have been proposed to tackle the problem of equation ??, such as Controlled Random Search methods [?, ?, ?] , Simulated Annealing methods [?, ?, ?], Differential Evolution methods [?, ?], Particle Swarm Optimization (PSO) methods [?, ?, ?], Ant Colony Optimization [?, ?], Genetic algorithms [?, ?, ?] etc. A systematic review of global optimization methods can also be found in the work of Floudas et al [?]. In addition, during the last few years, a variety of work has been proposed on combinations and modifications to some global optimization methods to more efficiently find the global minimum, such as methods that combine PSO with other methods [?, ?, ?], methods aimed to discover all the local minima of functions [?, ?, ?], new stopping rules to efficiently terminate the global optimization techniques [?, ?, ?] etc. Also, due to the massive use of parallel processing techniques, several methods have been proposed that take full advantage of parallel processing, such as parallel techniques [?, ?, ?], methods that utilize the GPU architectures [?, ?] etc.

In this paper, a new multi-start method is proposed that uses a machine learning model, which is trained in parallel with the evolution of the optimization process. Although multistart methods are considered the basis for more modern optimization techniques, they have been successfully used in several problems such as the Traveling Salesman Problem (TSP) [?, ?, ?], the maximum clique problem [?, ?], the vehicle routing problem [?, ?], scheduling problems [?, ?] etc. In the new technique, a Radial Basis Function (RBF) network [?] is used to construct an approximation of the objective function. This construction is carried out in parallel with the execution of the optimization. A limited number of samples from the objective function and the local minima discovered during the optimization are used to construct the approximation function. During the execution of the method, the samples needed to start local minimizers are taken from the approximation function that is constructed by the neural network. The RBF network was used as an approximation tool as it has been successfully used in a wide range of problems in the field of artificial intelligence [?, ?, ?, ?] and its training procedure is very fast, if compared to artificial neural networks for example. In addition, for more efficient termination of the method, a termination method proposed by Tsoulos is used [?], but this termination method is applied after each execution of the local minimization procedure. The proposed technique was applied to a series of problems from the relevant literature and the results are extremely promising even compared to established global optimization techniques.

The rest of this article is organized as follows: in section ?? the description of the proposed method is provided, in section ?? the used experimental functions as well as the experimental results and comparisons are listed and finally in section ?? some conclusions and final thoughts are given.

2 Method Description

The proposed technique generates an estimation of the objective function during the optimization using an RBF network. This estimation is initially generated from some samples from the objective function and gradually local minima that will have been discovered during the optimization are added to it. In this way, the estimation of the objective function will be continuously improved to approximate the true function as much as possible. At every iteration, several samples are then taken from the estimated function and sorted in ascending order. Those with the lowest value will be starting points of the local minimization method. The local optimization method used here is a BFGS variant of Powell [?]. This process has the effect of drastically reducing the total number of function calls that are made and, at the same time, the points used as initiators of the local minimization technique approach the global minimum of the objective function. Also, the proposed method checks the termination rule after the application of every local search method. That way, if the absolute minimum has already been discovered with some certainty, no more function calls will be wasted finding it.

In the following subsections, the training procedure of RBF networks as well as the proposed method are fully described.

2.1 Rbf preliminaries

An RBF network can be defined as:

$$N\left(\overrightarrow{x}\right)=\sum_{i=1}^{k}w_i\phi\left(\left\|\overrightarrow{x}-\overrightarrow{c_i}\right\|\right) \tag{2}$$

Where

1. The vector $\overrightarrow{x}$ is called the input pattern to the equation.
2. The vectors $\overrightarrow{c_i},\ i=1,..,k$ are called the center vectors.
3. The vector $\overrightarrow{w}$ stands for the the output weight of the RBF network.

In most cases the function $\phi(x)$ is a Gaussian function:

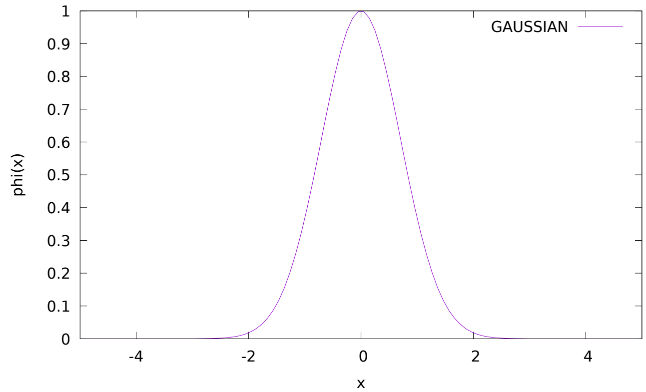
$$\phi(x)=\exp\left(-\frac{(x-c)^2}{\sigma^2}\right) \tag{3}$$

A plot for this function with $c=0,\ \sigma=1$ is shown in Figure **??**. The training error for the RBF network on a set $T=\left\{\left(x_1,y_1\right),\left(x_2,y_2\right),\ldots,\left(x_M,y_M\right)\right\}$ points is estimated as

$$E\left(N\left(\overrightarrow{x}\right)\right)=\sum_{i=1}^{M}\left(N\left(x_i\right)-y_i\right)^2 \tag{4}$$

In most approaches, the equation **??** is minimized with respect to the parameters of the RBF network using a two phase procedure:

Figure 1: A plot for the Gaussian function.



1. During the first phase, the k centers and the corresponding variances are calculated using the K-Means algorithm [**?**].

2. During the second phase, the weights $\overrightarrow{w} = (w_1, w_2, \ldots, w_k)$ are calculated by solving a linear system of equations with the following procedure:

   (a) **Set** $W = w_{kj}$

   (b) **Set** $\Phi = \phi_j(x_i)$

   (c) **Set** $T = \{t_i = f(x_i), i = 1, .., M\}$**.**

   (d) The system to be solved is identified as:

   $$\Phi^T \left(T - \Phi W^T\right) = 0 \tag{5}$$

   With solution:

   $$W^T = \left(\Phi^T \Phi\right)^{-1} \Phi^T T = \Phi^\dagger T \tag{6}$$

2.2 The main algorithm

The main steps of the proposed algorithm have as follows:

1. **Initialization** step.

   (a) **Set** k the number of weights in RBF network.

   (b) **Set** N_S the initial samples that will be taken from the function $f(x)$.

   (c) **Set** N_T the number of samples, that will used in every iteration as starting points for the local optimization procedure.

   (d) Set N_R the number of samples that will be drawn from the RBF network at every iteration with $N_R > N_T$

   (e) **Set** N_G the maximum number of allowed iterations.

(f) **Set** Iter=0, the iteration number.

(g) **Set** (x^*, y^*) as the global minimum. Initially $y^* = \infty$

2. **Creation** Step.

   (a) **Set** $T = \emptyset$, the training set for the RBF network.

   (b) **For** $i = 1, .., N_S$ do

      i. **Take** a new sample $x_i \in S$

      ii. **Calculate** $y_i = f(x_i)$

      iii. $T = T \cup (x_i, y_i)$

   (c) **EndFor**

   (d) **Train** the RBF network on the training set T.

3. **Sampling** Step

   (a) **Set** $T_R = \emptyset$

   (b) **For** $i = 1, .., N_R$ **do**

      i. **Take** a random sample (x_i, y_i) from the RBF network.

      ii. **Set** $T_R = T_R \cup (x_i, y_i)$

   (c) **EndFor**

   (d) **Sort** T_R according to the y values in ascending order.

4. **Optimization** Step.

   (a) **For** $i = 1, .., N_T$ do

      i. **Take** the next sample (x_i, y_i) from T_R.

      ii. $y_i = \mathrm{LS}(x_i)$. Where LS(x) is a predefined local search method.

      iii. $T = T \cup (x_i, y_i)$, this step updates the training set of the RBF network.

      iv. **Train** the RBF network on set T.

      v. **If** $y_i \leq y^*$ then $x^* = x_i, y^* = y_i$

      vi. **Check** the termination rule as suggested in [**?**]. If it holds report (x^*, y^*) as the located global minimum and terminate.

   (b) **EndFor**

5. **Set** iter=iter+1

6. **Goto** to Sampling step.

3 Experiments

3.1 Test functions

To estimate the efficiency of the new technique a number of functions from the relevant literature were used [?, ?]. These functions are provided in the following.

- ***Bent Cigar function*** The function is

$$f(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$$

with the global minimum $f(x^*) = 0$. For the conducted experiments the value $n = 10$ was used.

- **Bf1** function. The function Bohachevsky 1 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$$

with $x \in [-100, 100]^2$.

- **Bf2** function. The function Bohachevsky 2 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$$

with $x \in [-50, 50]^2$.

- **Branin** function. The function is defined by $f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$ with $-5 \le x_1 \le 10,\ 0 \le x_2 \le 15$. The value of global minimum is 0.397887.with $x \in [-10, 10]^2$.

- **CM** function. The Cosine Mixture function is given by the equation

$$f(x) = \sum_{i=1}^{n} x_i^2 - \frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$$

with $x \in [-1, 1]^n$. For the conducted experiments the value $n = 4$ was used.

- **Camel** function. The function is given by

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, \quad x \in [-5, 5]^2$$

The global minimum has the value of $f(x^*) = -1.0316$

- **Discus function** The function is defined as

$$f(x) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$$

with global minimum $f(x^*) = 0$. For the conducted experiments the value $n = 10$ was used.

- **Easom** function The function is given by the equation

$$f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2-\pi)^2 - (x_1-\pi)^2\right)$$

with $x \in [-100, 100]^2$ and global minimum -1.0

- **Exponential** function. The function is given by

$$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \le x_i \le 1$$

The global minimum is located at $x^* = (0, 0, ..., 0)$ with value -1. In our experiments we used this function with $n = 4, 16, 64$ and the corresponding functions are denoted by the labels EXP4, EXP16, EXP64.

- **Griewank10** function. The function is given by the equation

$$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

with $n = 10$.

- **Hansen** function. $f(x) = \sum_{i=1}^{5} i\cos\left[(i-1)x_1 + i\right]\sum_{j=1}^{5} j\cos\left[(j+1)x_2 + j\right]$, $x \in [-10, 10]^2$. The global minimum of the function is -176.541793.

- **Hartman 3** function. The function is given by

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$$

with $x \in [0, 1]^3$ and $a = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}$$

The value of global minimum is -3.862782.

- **Hartman 6** function.

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

with $x \in [0,1]^6$ and $a = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$

and

$$p = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}$$

the value of global minimum is -3.322368.

- **High Conditioned Elliptic** function, defined as

$$f(x) = \sum_{i=1}^{n} \left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$$

with global minimum $f(x^*) = 0$ and the value $n = 10$ was used in the conducted experiments

- **Potential** function. The molecular conformation corresponding to the global minimum of the energy of N atoms interacting via the Lennard-Jones potential[**?**] is used as a test case here. The function to be minimized is given by:

$$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{7}$$

In the current experiments two different cases were studied: $N = 3,\ 5$

- **Rastrigin** function. The function is given by

$$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2), \quad x \in [-1,1]^2$$

- **Shekel 7** function.

$$f(x) = -\sum_{i=1}^{7} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{pmatrix}$.

- **Shekel 5** function.

$$f(x) = -\sum_{i=1}^{5} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{pmatrix}$.

- **Shekel 10** function.

$$f(x) = -\sum_{i=1}^{10} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.6 \end{pmatrix}$.

- **Sinusoidal** function. The function is given by

$$f(x) = -\left(2.5 \prod_{i=1}^{n} \sin(x_i - z) + \prod_{i=1}^{n} \sin(5(x_i - z))\right), \quad 0 \le x_i \le \pi.$$

  The global minimum is located at $x^* = (2.09435, 2.09435, ..., 2.09435)$ with $f(x^*) = -3.5$. For the conducted experiments the cases of $n = 4, 8$ and $z = \frac{\pi}{6}$ were studied.

- **Test2N** function. This function is given by the equation

$$f(x) = \frac{1}{2} \sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i, \quad x_i \in [-5,5].$$

  The function has 2^n in the specified range and in our experiments we used $n = 4, 5, 6, 7$.

- **Test30N** function. This function is given by

$$f(x) = \frac{1}{10} \sin^2(3\pi x_1) \sum_{i=2}^{n-1} \left((x_i - 1)^2 \left(1 + \sin^2(3\pi x_{i+1})\right)\right) + (x_n - 1)^2 \left(1 + \sin^2(2\pi x_n)\right)$$

  with $x \in [-10,10]$. The function has 30^n local minima in the specified range and we used $n = 3, 4$ in the conducted experiments.

Table 1: Experimental settings.

PARAMETER	MEANING	VALUE
k	Number of weights	10
N_S	Start samples	50
N_T	Number of samples used as starting points	100
N_R	Number of samples that will be drawn from the RBF network	$10 \times N_T$
N_C	Chromosomes or Particles	100
N_G	Maximum number of iterations	200

3.2 Experimental results

The proposed technique was tested on the previously presented test functions and the results produced were compared with those given by a simple genetic algorithm or a PSO method. In order to have fairness in the comparison of the results, both in the case of genetic algorithms and in the case of the PSO method, the same local minimization method has been used as that of the proposed method. In addition, the number of chromosomes in the genetic algorithm and the number of particles in the PSO method are identical to the parameter N_T of the proposed procedure. The values for the used parameters are listed in Table **??**. The conducted experiments were executed 30 times for each test function and averages were taken. In each execution, different seed for the random generator was used. The proposed method is implemented as the method with the name *NeuralMinimizer* in the *OPTIMUS* global optimization environment, which is freely available from `https://github.com/itsoulos/OPTIMUS`. All the experiments were conducted on an AMD Ryzen 5950X with 128GB of RAM and the Debian Linux operating system.

The experimental results from the application of the proposed method and the Genetic algorithm and the PSO method are shown in Table **??**. The number show in the cells denote average function calls. The fraction in parentheses represents the fraction of executions where the global optimum was successfully found. When this number is absent, then the global minimum is computed for every execution (100% success). At the end of the table, an additional row named AVERAGE indicates the total number of function calls and the average success rate in locating the global minimum. In the experimental results, the superiority of the proposed technique over the other two methods in terms of the number of function calls is clear. The proposed technique requires an average of 90% fewer function calls than the other methods. In addition, the proposed technique appears to be more efficient than the other two as it finds more times on average the global minimum of most test functions in the experiments.

In addition, the table **??** presents the experimental results for the proposed method and for various values of the parameter N_S. As can be seen, the increase in this parameter does not cause a large increase in the total number of function calls, while at the same time it improves to some extent the ability of the proposed technique to find the global minimum.

Table 2: Comparison between the proposed method and the Genetic and PSO methods.

FUNCTION	GENETIC	PSO	PROPOSED
BF1	7150	9030(0.87)	1051
BF2	7504	6505(0.67)	921
BRANIN	6135	6865(0.93)	460
CAMEL	6564	5162	778
CIGAR10	11813	18803	1896
CM4	10537	11124	1877(0.87)
DISCUS10	20208	6039	478
EASOM	5281	2037	258
ELP10	20337	16731	2263
EXP4	10537	9155	750
EXP16	20131	14061	885
EXP64	20140	8958	948
GRIEWANK10	20151(0.10)	17497(0.03)	2697
POTENTIAL3	18902	9936	1192
POTENTIAL5	18477	12385	2399
HANSEN	10708	9104	2370(0.93)
HARTMAN3	8481	12971	642
HARTMAN6	17723(0.60)	15174(0.57)	883
RASTRIGIN	6744	7639(0.97)	1408(0.80)
ROSENBROCK4	20815(0.63)	11526	1619
ROSENBROCK8	20597(0.67)	16967	2444
SHEKEL5	14456(0.73)	15082(0.47)	2333(0.87)
SHEKEL7	16786(0.83)	14625(0.40)	1844(0.93)
SHEKEL10	15586(0.80)	12628(0.53)	2451
SINU4	11908	10659	802
SINU8	20115	13912	1500(0.97)
TEST2N4	13943	12948	878(0.93)
TEST2N5	15814	13936(0.90)	971(0.77)
TEST2N6	18987	15449(0.70)	997(0.70)
TEST2N7	20035	16020(0.50)	1084(0.30)
TEST30N3	13029	7239	1061
TEST30N4	12889	8051	854
AVERAGE	**472596(0.89)**	**368218(0.86)**	**42994(0.94)**

Table 3: Experimental results for different values of the parameter N_S and the proposed method.

FUNCTION	$N_S = 50$	$N_S = 100$	$N_S = 200$
BF1	1051	1116	1224
BF2	921	949	1058
BRANIN	460	506	599
CAMEL	778	676	739
CIGAR10	1896	1934	2042
CM4	1877(0.87)	1859(0.93)	1877(0.90)
DISCUS10	478	531	634
EASOM	258	307	450
ELP10	2263	2339	3130
EXP4	750	778	884
EXP16	885	932	1030
EXP64	948	998	1091
GRIEWANK10	2697	2647	2801
POTENTIAL3	1192	1228	1305
POTENTIAL5	2399	2417	2544
HANSEN	2370(0.93)	2602(0.93)	2578(0.97)
HARTMAN3	642	696	798
HARTMAN6	883	940	1038
RASTRIGIN	1408(0.80)	989(0.83)	1041
ROSENBROCK4	1619	1674	1751
ROSENBROCK8	2444	2499	2583
SHEKEL5	2333(0.87)	1267	1878(0.97)
SHEKEL7	1844(0.93)	1517(0.93)	1685(0.97)
SHEKEL10	2451	2695	1498
SINU4	802	821	901
SINU8	1500(0.97)	1216	1247
TEST2N4	878(0.93)	934	850(0.97)
TEST2N5	971(0.77)	941(0.80)	993
TEST2N6	997(0.70)	1087(0.77)	1098
TEST2N7	1084(0.30)	1160(0.53)	1313(0.57)
TEST30N3	1061	998	1320
TEST30N4	854	830	1108
AVERAGE	**42994(0.94)**	**42083(0.96)**	**45088(0.97)**

4 Conclusions

An innovative technique for finding the global minimum of multidimensional functions was presented in this work. This new technique is based on the multistart procedure, but also generates an estimation of the objective function through a machine learning model. The machine learning model constructs an estimation of the objective function using a small number of samples from the true function but also with the contribution of local minima discovered during the execution of the method. In this way, the estimation of the objective function is continuously improved and the sampling to perform local minimization is done from the estimated function rather than the actual one. This procedure combined with checking the termination criterion after each execution of the local minimization method led the proposed method to have excellent results both in terms of the speed of finding the global minimum and its efficiency. In addition, the method shows significant stability in its performance even in large changes of its parameters as presented in the experimental results section.

In the future, the use of the RBF network to construct an approximation of the objective function can be applied to more modern optimization techniques such as genetic algorithms. It would also be interesting to create a parallel implementation of the proposed method, in order to significantly speed up its execution and to be able to be used efficiently in optimization problems of higher dimensions.

References

[1] M. Honda, Application of genetic algorithms to modelings of fusion plasma physics, Computer Physics Communications **231**, pp. 94-106, 2018.

[2] X.L. Luo, J. Feng, H.H. Zhang, A genetic algorithm for astroparticle physics studies, Computer Physics Communications **250**, 106818, 2020.

[3] T.M. Aljohani, A.F. Ebrahim, O. Mohammed, Single and Multiobjective Optimal Reactive Power Dispatch Based on Hybrid Artificial Physics–Particle Swarm Optimization, Energies **12**, 2333, 2019.

[4] P.M. Pardalos, D. Shalloway, G. Xue, Optimization methods for computing global minima of nonconvex potential energy functions, Journal of Global Optimization **4**, pp. 117-133, 1994.

[5] A. Liwo, J. Lee, D.R. Ripoll, J. Pillardy, H. A. Scheraga, Protein structure prediction by global optimization of a potential energy function, Biophysics **96**, pp. 5482-5485, 1999.

[6] J. An, G.He, F. Qin, R. Li, Z. Huang, A new framework of global sensitivity analysis for the chemical kinetic model using PSO-BPNN, Computers & Chemical Engineering **112**, pp. 154-164, 2018.

[7] Zwe-Lee Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints, IEEE Transactions on **18** Power Systems, pp. 1187-1195, 2003.

[8] M. Basu, A simulated annealing-based goal-attainment method for economic emission load dispatch of fixed head hydrothermal power systems, International Journal of Electrical Power & Energy Systems **27**, pp. 147-153, 2005.

[9] Y. Cherruault, Global optimization in biology and medicine, Mathematical and Computer Modelling **20**, pp. 119-132, 1994.

[10] Eva K. Lee, Large-Scale Optimization-Based Classification Models in Medicine and Biology, Annals of Biomedical Engineering **35**, pp 1095-1109, 2007.

[11] W. L. Price, Global optimization by controlled random search, Journal of Optimization Theory and Applications **40**, pp. 333-348, 1983.

[12] Ivan Křivý, Josef Tvrdík, The controlled random search algorithm in optimizing regression models, Computational Statistics & Data Analysis **20**, pp. 229-234, 1995.

[13] M.M. Ali, A. Törn, and S. Viitanen, A Numerical Comparison of Some Modified Controlled Random Search Algorithms, Journal of Global Optimization **11**,pp. 377–385,1997.

[14] S. Kirkpatrick, CD Gelatt, , MP Vecchi, Optimization by simulated annealing, Science **220**, pp. 671-680, 1983.

[15] L. Ingber, Very fast simulated re-annealing, Mathematical and Computer Modelling **12**, pp. 967-973, 1989.

[16] R.W. Eglese, Simulated annealing: A tool for operational research, Simulated annealing: A tool for operational research **46**, pp. 271-281, 1990.

[17] R. Storn, K. Price, Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization **11**, pp. 341-359, 1997.

[18] J. Liu, J. Lampinen, A Fuzzy Adaptive Differential Evolution Algorithm. Soft Comput **9**, pp.448–462, 2005.

[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.

[20] Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.

[21] Ioan Cristian Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.

[22] M. Dorigo, M. Birattari and T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine **1**, pp. 28-39, 2006.

[23] K. Socha, M. Dorigo, Ant colony optimization for continuous domains, European Journal of Operational Research 185, pp. 1155-1173, 2008.

[24] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.

[25] Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.

[26] S.A. Grady, M.Y. Hussaini, M.M. Abdullah, Placement of wind turbines using genetic algorithms, Renewable Energy **30**, pp. 259-270, 2005.

[27] C.A. Floudas, C.E. Gounaris, A review of recent advances in global optimization, J Glob Optim **45**, pp. 3–38, 2009.

[28] Y. Da, G. Xiurun, An improved PSO-based ANN with simulated annealing technique, Neurocomputing **63**, pp. 527-533, 2005.

[29] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing **10**, pp. 629-640, 2010.

[30] X. Pan, L. Xue, Y. Lu Y. et al, Hybrid particle swarm optimization with simulated annealing, Multimed Tools Appl **78**, pp. 29921–29936, 2019.

[31] M.M. Ali, C. Storey, Topographical multilevel single linkage, J. Global Optimization **5**, pp. 349–358,1994

[32] S. Salhi, N.M. Queen, A hybrid algorithm for identifying global and local minima when optimizing functions with many minima, European J. Oper. Res. **155**, pp. 51–67, 2004.

[33] I. G. Tsoulos and I. E. Lagaris, MinFinder: Locating all the local minima of a function, Computer Physics Communications **174**, pp. 166-179, 2006.

[34] B. Betro‘, F. Schoen, Optimal and sub-optimal stopping rules for the multistart algorithm in global optimization, Math. Program. **57**, pp. 445–458, 1992.

[35] W.E. Hart, Sequential stopping rules for random optimization methods with applications to multistart local search, Siam J. Optim. **9**, pp. 270–290, 1998.

[36] I.E. Lagaris and I.G. Tsoulos, Stopping Rules for Box-Constrained Stochastic Global Optimization, Applied Mathematics and Computation **197**, pp. 622-632, 2008.

[37] J. F. Schutte, J. A. Reinbolt, B. J. Fregly, R. T. Haftka, A. D. George, Parallel global optimization with the particle swarm algorithm, International journal for Numerical methods in Engineering **61**, pp. 2296-2315, 2004.

[38] J. Larson and S.M. Wild, Asynchronously parallel optimization solver for finding multiple minima, Mathematical Programming Computation **10**, pp. 303-332, 2018.

[39] I.G. Tsoulos, A. Tzallas, D. Tsalikakis, PDoublePop: An implementation of parallel genetic algorithm for function optimization, Computer Physics Communications **209**, pp. 183-189, 2016.

[40] R. Kamil, S. Reiji, An Efficient GPU Implementation of a Multi-Start TSP Solver for Large Problem Instances, Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 1441-1442, 2012.

[41] Van Luong T., Melab N., Talbi EG. (2011) GPU-Based Multi-start Local Search Algorithms. In: Coello C.A.C. (eds) Learning and Intelligent Optimization. LION 2011. Lecture Notes in Computer Science, vol 6683. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25566-3_24

[42] Li W., A Parallel Multi-start Search Algorithm for Dynamic Traveling Salesman Problem. In: Pardalos P.M., Rebennack S. (eds) Experimental Algorithms. SEA 2011. Lecture Notes in Computer Science, vol 6630. Springer, Berlin, Heidelberg, 2011.

[43] R. Martí, M.G.C. Resende, C.C. Ribeiro, Multi-start methods for combinatorial optimization, European Journal of Operational Research **226**, pp. 1-8, 2013.

[44] V. Pandiri, A. Singh, Two multi-start heuristics for the k-traveling salesman problem, OPSEARCH **57**, pp. 1164–1204, 2020.

[45] Q. Wu, J.K. Hao,An adaptive multistart tabu search approach to solve the maximum clique problem, J Comb Optim **26**, pp.86–108, 2013.

[46] Y. Djeddi, H.A. Haddadene, N. Belacel, An extension of adaptive multi-start tabu search for the maximum quasi-clique problem, Computers & Industrial Engineering **132**, pp. 280-292, 2019.

[47] Olli Bräysy, Geir Hasle, Wout Dullaert, A multi-start local search algorithm for the vehicle routing problem with time windows, European Journal of Operational Research **159**, pp. 586-605, 2004.

[48] J. Michallet, C. Prins, L. Amodeo, F. Yalaoui, G. Vitry, Multi-start iterated local search for the periodic vehicle routing problem with time windows and time spread constraints on services, Computers & Operations Research **41**, pp. 196-207, 2014.

[49] K. Peng, Q.K. Pan, L. Gao, X. Li, S. Das, B. Zhang, A multi-start variable neighbourhood descent algorithm for hybrid flowshop rescheduling, Swarm and Evolutionary Computation **45**, pp. 92-112, 2019.

[50] J.Y. Mao, Q.K. Pan, Z.H. Miao, L. Gao, An effective multi-start iterated greedy algorithm to minimize makespan for the distributed permutation flowshop scheduling problem with preventive maintenance, Expert Systems with Applications **169**, 114495, 2021.

[51] J. Park, I.W. Sandberg, Approximation and Radial-Basis-Function Networks, Neural Computation **5**, pp. 305-316, 1993.

[52] S.H. Yoo, S.K. Oh, W. Pedrycz, Optimized face recognition algorithm using radial basis function neural networks and its practical applications, Neural Networks **69,** pp. 111-125, 2015.

[53] G.B. Huang, P. Saratchandran, N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation, IEEE Transactions on Neural Networks **16**, pp. 57-67, 2005.

[54] Z. Majdisova, V. Skala, Radial basis function approximations: comparison and applications, Applied Mathematical Modelling **51**, pp. 728-743, 2017.

[55] B.C. Kuo, H.H. Ho, C. H. Li, C. C. Hung , J. S. Taur, A Kernel-Based Feature Selection Method for SVM With RBF Kernel for Hyperspectral Image Classification, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. **7**, pp. 317-326, 2014.

[56] I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation **203**, pp. 598-607, 2008.

[57] M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.

[58] MacQueen, J.: Some methods for classification and analysis of multivariate observations, in: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability **1**, pp. 281-297, 1967.

[59] M. Montaz Ali, Charoenchai Khompatraporn, Zelda B. Zabinsky, A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems, Journal of Global Optimization **31**, pp 635-672, 2005.

[60] C.A. Floudas, P.M. Pardalos, C. Adjiman, W. Esposoto, Z. Gümüs, S. Harding, J. Klepeis, C. Meyer, C. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, Dordrecht, 1999.

[61] J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.