Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions

Vasileios Charilogis † and Ioannis G. Tsoulos *,†

Department of Informatics and Telecommunications, University of Ioannina, 47150 Ioannina, Greece; v.charilog@uoi.gr
* Correspondence: itsoulos@uoi.gr
† These authors contributed equally to this work.

Abstract: The Particle Swarm Optimization (PSO) method is a global optimization technique based on the gradual evolution of a population of solutions called particles. The method evolves the particles based on both the best position of each of them in the past and the best position of the whole. Due to its simplicity, the method has found application in many scientific areas, and for this reason, during the last few years, many modifications have been presented. This paper introduces three modifications to the method that aim to reduce the required number of function calls while maintaining the accuracy of the method in locating the global minimum. These modifications affect important components of the method, such as how fast the particles change or even how the method is terminated. The above modifications were tested on a number of known universal optimization problems from the relevant literature, and the results were compared with similar techniques.

Keywords: optimization; evolutionary techniques; stochastic methods; termination rules

1. Introduction

The global optimization problem is usually defined as:

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

with $S$:

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

The function $f$ is considered a continuous and differentiable function, which is formulated as $f : S \rightarrow \mathbb{R}, S \subseteq \mathbb{R}^n$. This problem finds application in a variety of objective problems in the real world, such as problems of physics [1–3], chemistry [4–6], economics [7,8], medicine [9,10], etc. Global optimization methods are grouped into two broad categories: deterministic and stochastic methods. The most common methods of the first category are the so-called Interval methods [11,12], where the set $S$ is divided iteratively into subregions, and some subregions that do not contain the global solution are discarded using some pre-defined criteria. In stochastic methods, the finding of the global minimum is guided by randomness. In these methods, there is no guarantee to find the global minimum, but they constitute the vast majority of global optimization methods, which is mainly due to the simplicity in their implementation. There have been proposed many stochastic methods by various researchers such as Controlled Random Search methods [13–15], Simulated Annealing methods [16–18], Differential Evolution methods [19,20], Particle Swarm Optimization methods [21–23], Ant Colony Optimization [24,25], Genetic Algorithms [26–28], etc. In addition, many works have appeared utilizing the modern GPU processing units [29–31]. The reader can find some complete surveys about metaheuristic algorithms in some recent works [32–34].

The method of Particle Swarm Optimization is a method based on a population of candidate solutions that are also called particles. These particles have two basic characteristics: their position at any given time $\vec{x}$ and the speed $\vec{u}$ at which they moved. The
position \( x_i \) of every particle can be defined as the summation of the current position \( x_i \) and the associated velocity \( u_i \): 

\[
x_i = x_i + u_i,
\]

where the velocity \( u_i \) is defined as a combination of the old velocity and the best located values \( p_i \) and \( p_{\text{best}} \):

\[
u_i = \omega u_i + r_1 c_1 (p_i - x_i) + r_2 c_2 (p_{\text{best}} - x_i)
\]

(2)

where \( r_1 \) and \( r_2 \) are random numbers and \( c_1, c_2 \) are user-defined constants. The value \( \omega \) is called the inertia of the algorithm and usually is calculated using some efficient algorithm. The purpose of the method is to move the particles repetitively, and their next position is calculated as a function not only of their position but also of the best position they had in the past as well as the best position of the population. Various researchers have provided an analytic review of the PSO process such as the review of Jain et al. [35] where 52 papers have been reviewed, the work of Khare et al. [36] where a systematic review of the PSO algorithm is provided along with the application of PSO in solar photovoltaic systems. The method was successfully used in a variety of scientific and practical problems in areas such as physics [37,38], chemistry [39,40], medicine [41,42], economics [43], etc.

Many researchers have proposed a variety of modifications in the PSO method during the last few years; such methods aimed to estimate a more efficient calculation for the inertia parameter \( \omega \) of the speed calculation [44–46]. The majority of methods used to calculate the inertia value calculate the inertia as a time-varying parameter, and hence, there is lack of diversity, which could be used to better explore the search space of the objective function. This paper introduces a new scheme for the calculation of inertia which takes into account the speed of the algorithm to locate new local minima in the search space of the objective function.

Other modifications of the PSO are used to alter the calculation of the factor \( c_1, c_2 \), which usually are called acceleration factors: for example, the work of Ratnaweera et al. [47], where a time-varying mechanism is proposed for the calculation of these factors. In addition, the PSO algorithm has been combined with the mutation mechanism [48–50] in order to improve the diversity of the algorithm and to better explore the search space of the objective function. Engelbrecht [51] proposed a novel method for the initialization of the velocity vector, where the velocity is initialized randomly rather than with zero value. In addition, in the relevant literature, there have been hybrid techniques [52–54], parallel techniques [55–57], methods aim to improve the velocity calculation [58–60], etc.

The method of PSO has been integrated into other optimization techniques. For example, Bogdanova et al. [61] combined Grammatical Evolution [62] with swarm techniques such as PSO. Their work is divided in two phases: during the first phase, the PSO method is decomposed into a list of basic grammatical rules. During the second phase, instances of the PSO method are created using the previous rules. Another interesting combination with an optimization technique is the work of Pan et al. [63], who combined the PSO method with simulated annealing to improve the local search ability of the PSO method. In addition, Mughal et al. [64] used a hybrid technique of PSO and simulated annealing for photovoltaic cell parameter estimation. Similarly, the work of Lin et al. [65] utilized a hybrid method of PSO and differential evolution for numerical optimization problems. In addition, Epitropakis et al. [66] suggested a hybrid PSO method with differential evolution, where the at each step of PSO, a differential evolution is applied to the best particle of the population. Among others, Wang et al. proposed a new PSO method based on chaotic neighbourhood search [67].

The PSO method is an iterative process, during which a series of particles evolve through a process that involves updating the position of the particles and fitness computation, i.e., evaluation of the objective function. Variations of PSO that aim at the global minimum in a shorter time may include the use of a local optimization method in each iteration of the algorithm. Of course, the above process can be extremely time consuming, and depending on the termination method used and the number of local searches performed, it may require a long execution time.
This text introduces three distinct modifications to the original method, which drastically improve the time required to find the total minimum by reducing the required number of function evaluations. These modifications cover a large part of the method: the speed calculation, a new method of avoiding running local search methods and a new adaptive termination rule. The first modification is used to enable the method to explore the search space more efficiently by calculating the inertia parameter with respect to the ability of the PSO to discover the local minima of the objective function. The second modification is used to avoid finding the same local minimum over and over again by the method. This way, on the one hand, the algorithm will not be trapped in local minima, and on the other hand, calls to the objective function will not be wasted, which will lead with some certainty to minima that have already been found. The third modification introduces a novel termination rule based on asymptotic calculations. Termination of methods such as PSO is an important process, as efficient termination methods will terminate the method in time without wasting many calls on the objective function, but at the same time, they should guarantee the ability of the method to find the global minimum.

The proposed modifications were applied to a number of problems from the relevant literature and compared with similar techniques, and the results are presented in a separate section. From the experiments performed, it was shown that the proposed modifications significantly reduce the required execution time of the method by drastically reducing the number of function calls required to find the total minimum. In addition, these modifications can be applied either alone or in combination with the same positive results. This means that they are quite general and can be included in other techniques based on PSO.

The rest of this article is divided as follows: in Section 2, the initial method and the proposed modifications are discussed, in Section 3, the experiments are listed, and finally, in Section 4, some conclusions and guidelines for future improvements are presented.

2. Method Description

The base algorithm of PSO and the proposed modifications are outlined in detail in the following subsections. The discussion starts with a new mechanism that calculates the velocity of the populations, continues with a discarding procedure used to minimize the number of local searches performed, and ends with a discussion about the new stopping rule proposed here.

2.1. The Base Algorithm

The base Algorithm 1 is listed below with the periodical application of the local search method in order to enhance the estimation of the global minimum, i.e., at every iteration, a decision with probability $p_l$ is made in order to apply a local search procedure to some of the particles. Usually, this probability is small, for example 0.05 (5%).

The current work modifies the above algorithm in three key points:

1. In Step 2, a new termination rule based on asymptotic considerations is introduced.
2. In Step 3b, the algorithm calculates the new position of the particles. The proposed methodology modifies the position of the particles based on the average speed of the algorithm to discover new minimums.
3. In Step 3c, a method based on gradient calculations will be used to prevent the PSO method from executing unnecessary local searches.

2.2. Velocity Calculation

The algorithm of Section 2.1 calculates at every iteration the new position $x_i$, which is calculated using the old position $x_i$, and the associated velocity $u_i$ according to the scheme:

$$x_i = x_i + u_i$$

(3)
Algorithm 1: The base algorithm of PSO and the proposed modifications

1. Initialization.
   (a) Set $\text{iter} = 0$ (iteration counter).
   (b) Set the number of particles $m$.
   (c) Set the maximum number of iterations allowed $\text{itermax}$.
   (d) Set the local search rate $p_l \in [0, 1]$.
   (e) Initialize randomly the positions of the $m$ particles $x_1, x_2, \ldots, x_m$, with $x_i \in S \subseteq R^n$.
   (f) Initialize randomly the velocities of the $m$ particles $u_1, u_2, \ldots, u_m$, with $u_i \in S \subseteq R^n$.
   (g) For $i = 1..m$ do $p_i = x_i$. The $p_i$ vectors are the best located values for every particle $i$.
   (h) Set $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$.

2. Termination Check. Check for termination. If termination criteria are met, then stop.

3. For $i = 1..m$ Do
   (a) Update the velocity $u_i$ as a function of $u_i$, $p_i$ and $p_{\text{best}}$.
   (b) Update the position $x_i = x_i + u_i$.
   (c) Set $r \in [0, 1]$ a random number. If $r \leq p_m$ then $x_i = \text{LS}(x_i)$, where LS($x$) is a local search procedure.
   (d) Evaluate the fitness of the particle $i$, $f(x_i)$.
   (e) If $f(x_i) \leq f(p_i)$ then $p_i = x_i$.
4. End For
5. Set $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$.
6. Set $\text{iter} = \text{iter} + 1$.
7. Goto Step 2.

Typically, the velocity is calculated as a combination of the old velocity and the best located values $p_i$ and $p_{\text{best}}$ and may be defined as:

$$u_i = \omega u_i + r_1 c_1 (p_i - x_i) + r_2 c_2 (p_{\text{best}} - x_i) \quad (4)$$

where

1. The parameters $r_1$, $r_2$ are random numbers with $r_1 \in [0, 1]$ and $r_2 \in [0, 1]$.
2. The constant numbers $c_1$, $c_2$ are in the range $[1, 2]$.
3. The variable $\omega$ is called inertia, with $\omega \in [0, 1]$.

The inertia was proposed by Shi and Eberhart [21]. They argued that high values of the inertia coefficient cause better exploration of the search area, while small values of this variable are used when we want to achieve better local research around promising areas for the global minimum. The value of the inertia factor generally starts with large values and decreases with repetition. This article proposes a new adaptive technique for the inertia parameter, and this mechanism is compared against three others from the relevant literature.

2.2.1. Random Inertia

The inertia calculation is proposed in [68], and it is defined as

$$\omega_{\text{iter}} = 0.5 + \frac{r}{2} \quad (5)$$

where $r$ is a random number with $r \in [0, 1]$. This inertia calculation will be called $\Pi_1$ in the tables with the experimental results.
2.2.2. Linear Time-Varying Inertia (Min Version)

This inertia schema has been proposed and used in various studies [68–70], and it is defined as:

\[
\omega_{\text{iter}} = \frac{\text{iter}_{\text{max}} - \text{iter}}{\text{iter}_{\text{max}}} (\omega_{\text{max}} - \omega_{\text{min}}) + \omega_{\text{min}}
\]  

(6)

where \( \omega_{\text{min}} \) is the minimum value of inertia and \( \omega_{\text{max}} \) the maximum value for inertia. This inertia calculation will be called I2 in the tables with the experimental results.

2.2.3. Linear Time-Varying Inertia (Max Version)

This method is proposed in [71,72], and it is defined as:

\[
\omega_{\text{iter}} = \frac{\text{iter}_{\text{max}} - \text{iter}}{\text{iter}_{\text{max}}} (\omega_{\text{min}} - \omega_{\text{max}}) + \omega_{\text{max}}
\]  

(7)

This inertia calculation will be called I3 in the tables with the experimental results.

2.2.4. Proposed Technique

This calculation of inertia involves the number of iterations where the method manages to find a new minimum. In the first iterations and when the method has to do more exploration of the research area, the inertia will be great. When the method should focus on a minimum, then the inertia will decrease. For this reason, at every iteration \( \text{iter} \), the quantity

\[
\delta(\text{iter}) = \left\| \sum_{i=1}^{m} f_i(\text{iter}) \right\| - \left\| \sum_{i=1}^{m} f_i(\text{iter}-1) \right\|
\]  

(8)

is measured. In the first steps of the algorithm, this quantity will change from repetition to repetition at a fast pace, and at some point, it will no longer change at the same rate or will be zero.

Hence, we define a metric to model the changes in \( \delta(\text{iter}) \) as

\[
\zeta(\text{iter}) = \begin{cases} 1, & \delta(i) = 0 \\ 0, & \text{otherwise} \end{cases}
\]  

(9)

Using this observation, two additional metrics are created, \( S_{\delta}(\text{iter}) \) and \( C_{\delta}(\text{iter}) \). The metric \( S_{\delta}(\text{iter}) \) is given by

\[
S_{\delta}(\text{iter}) = \sum_{i=1}^{\text{iter}} \zeta(i)
\]  

(10)

and the metric \( C_{\delta} \) is given by:

\[
C_{\delta}(\text{iter}) = \frac{S_{\delta}(\text{iter})}{\text{iter}}
\]  

(11)

The following definition for the inertia calculation is proposed:

\[
\omega_{\text{iter}} = \omega_{\text{max}} - \frac{\text{iter}}{C_{\delta}(\text{iter})} (\omega_{\text{max}} - \omega_{\text{min}})
\]  

(12)

This mechanism will be called IP in the tables with the experimental results.

2.3. The Discarding Procedure

The method in each iteration performs under the condition of a series of local searches. However, these searches will often either lead to local minima already found or locate values far below the global minimum. This means that much of the computing time will be wasted on actions that could have been avoided. In order to be able to group points that
would lead by local search to the same local minimum, we introduce the concept of cluster, which refers to a set of points that are believed, under some asymptotic considerations, to belong to the same region of attraction of the function. The region of attraction for a local minimum $x^*$ is defined as:

$$ A(x^*) = \{ x : x \in S \subset R^n, \text{LS}(x) = x^* \} $$

where LS$(x)$ is a local search procedure that starts from a given point $x$ and terminates when a local minimum is discovered. The discarding procedure suggested here prevents the method from starting a local search from a point $x$ if that point belongs to the same region of attraction with other points. This procedure is composed by two two major parts:

1. The first part is the so-called typical distance, which is a measure calculated after every local search, and it is given by

$$ r_C = \frac{1}{M} \sum_{i=1}^{M} \| x_i - x_{iL} \| $$

where the local search procedure LS$(x_i)$ initiates from $x_i$ and $x_{iL}$ is the outcome of LS$(x_i)$. This measure has been used also in [73]. If a point $x$ is close enough to an already discovered local minima, then it is highly possible that the point belongs to the so-called region of attraction of the minima.

2. The second part is a check using the gradient values between a candidate starting point and an already discovered local minimum. The function value $f(x)$ near to some local minimum $z$ can be calculated using:

$$ f(x) \approx f(z) + \frac{1}{2} (x - z)^T B (x - z) $$

where $B$ is the Hessian matrix at the minimum $z$. By taking gradients in both sides of Equation (15), we obtain:

$$ \nabla f(x) \approx B(x - z) $$

Of course, Equation (16) holds for any other point $y$ near to $z$

$$ \nabla f(y) \approx B(y - z) $$

By subtracting Equation (17) from Equation (16) and by multiplying with $(x - y)^T$, we have the following equation:

$$ (x - y)^T (\nabla f(x) - \nabla f(y)) \approx (x - y)^T B(x - y)^T > 0 $$

Hence, a candidate start point $x$ can be rejected if

$$ \| x - z \| \leq r_C \text{ AND } (x - y)^T (\nabla f(x) - \nabla f(z)) $$

for any already discovered local minimum $z$.

2.4. Stopping Rule

A common used way to terminate a global optimization method is to utilize a maximum number of allowed iterations iter$_{max}$, i.e., stop when iter $\geq$ iter$_{max}$. Although it is a simple criterion, it is not an efficient one, since if iter$_{max}$ is too small, then the algorithm will terminate without locating the global optimum. In addition, when iter$_{max}$ is too high, the optimization algorithm will spend computation time in unnecessary function calls. In this paper, a new termination rule is proposed to terminate the PSO process, and it is compared against two other methods used in various optimization methods.
2.4.1. Ali’s Stopping Method

A method is proposed in the work of Ali and Kaelo [74] where at every generation, the measure
\[ \alpha = |f_{\text{max}} - f_{\text{min}}| \] (20)
is calculated. The \( f_{\text{max}} \) stands for the maximum function value of the population and \( f_{\text{min}} \) represents the lowest function value of the population. The method will terminate when
\[ \alpha \leq \epsilon \] (21)
where \( \epsilon \) is a predefined small positive value, for example \( \epsilon = 10^{-3} \).

2.4.2. Double Box Method

The second method utilized is a method initially proposed by [75]. In this method, we denote with \( \sigma(\text{iter}) \) the variance of \( f_{\text{min}} \) calculated at iteration \( \text{iter} \). If the algorithm cannot locate a new lower value for \( f_{\text{min}} \) for a number of iterations, then the global minimum has already located, and the algorithm should terminate when
\[ \sigma(\text{iter}) \leq \frac{\sigma(\text{iter}_{\text{last}})}{2} \] (22)
where \( \text{iter}_{\text{last}} \) stands for the last iteration where a new lower value for \( f_{\text{min}} \) was discovered.

2.4.3. Proposed Technique

In the proposed termination technique, in each iteration \( k \), the difference between the current best value and the previous best value is measured, i.e., the difference \( |f^{(k)}_{\text{min}} - f^{(k-1)}_{\text{min}}| \). If this difference is zero for a predefined number of iterations \( k_{\text{max}} \), then the method terminates.

3. Experiments

To measure the effect of the proposed modifications on the original method, a series of experiments were performed on test functions from the relevant literature [76,77], and they have been used widely by various researchers [74,78–80]. The experiments evaluated both the effect of the new method of calculating inertia as well as the criterion for avoiding local minima as well as the new termination rule. The experiments were recorded in separate tables, so that it is more possible to understand the effect of each modification separately.

3.1. Test Functions

The definition of the test functions used are given below

- **Bf1** (Bohachevsky 1) function:
  \[ f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10} \]
  with \( x \in [-100, 100]^2 \).

- **Bf2** (Bohachevsky 2) function:
  \[ f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10} \]
  with \( x \in [-50, 50]^2 \).

- **Branin** function: 
  \[ f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \]
  with \( -5 \leq x_1 \leq 10, 0 \leq x_2 \leq 15. \) with \( x \in [-10, 10]^2 \).
• **CM function:**

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

where \( x \in [-1, 1]^n \). In the current experiments, we used \( n = 4 \).

• **Camel function:**

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

• **Easom function:**

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

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

• **Exponential function**, defined as:

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

In the current experiments, we used this function with \( n = 2, 4, 8, 16, 32 \).

• **Goldstein and Price function**

\[ f(x) = \left[1 + (x_1 + x_2 + 1)^2 \right. \\
(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \cdot \\
\left. (30 + (2x_1 - 3x_2)^2) \right] \cdot \\
\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2 \right) \]

with \( x \in [-2, 2]^2 \).

• **Griewank 2 function:**

\[ f(x) = 1 + \frac{1}{200} \sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2} \cos(x_i) \sqrt{i}, \quad x \in [-100, 100]^2 \]

• **Gkls function.** \( f(x) = \text{Gkls}(x, n, w) \), is a function with \( w \) local minima, which was described in [81] with \( x \in [-1, 1]^n \) and \( n \) being a positive integer between 2 and 100. The value of the global minimum is -1, and in our experiments, we have used \( n = 2, 3 \) and \( w = 50, 100 \).

• **Hansen function:**

\[ f(x) = \sum_{i=1}^{5} i \cos([i-1]x_1 + i] \sum_{j=1}^{5} j \cos([j+1]x_2 + j], \]

\( x \in [-10, 10]^2 \).

• **Hartman 3 function:**

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

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

**Hartman 6 function:**

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

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 

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

**Potential function.** The molecular conformation corresponding to the global minimum of the energy of $N$ atoms interacting via the Lennard–Jones potential \[82\] is used as a test function here, and it is defined by:

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

(23)

For our experiments, we used: $N = 3, 4, 5$.

**Rastrigin function.**

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

**Rosenbrock function.**

\[
f(x) = \sum_{i=1}^{n-1} \left( 100 \left( x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right), \quad -30 \leq x_i \leq 30.
\]

In our experiments, we used the values $n = 4, 8, 16$.

**Shekel 7 function.**

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

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

**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}, \quad 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}, \quad 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:**

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

The case of \( n = 4, 8, 16, 32 \) and \( z = \frac{\pi}{6} \) was used in the experimental results.

- **Test2N function:**

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

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

- **Test30N function:**

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

with \( x \in [-10,10], \) with \( 30^n \) local minima in the search space. For our experiments, we used \( n = 3, 4. \)

### 3.2. Experimental Setup

All the experiments have been performed 30 times using a different seed for the random number generator each. The code has been implemented in ANSI C++, and the well-known function drand48() was used to produce random numbers. The local search method used was the BFGS method [83]. All the parameters used in the conducted experiments are listed in Table 1.
3.3. Experimental Results

For every stopping rule, two tables are listed: the first one contains experiments with the relevant stopping rule without the gradient discarding procedure, and the second table contains experiments with the gradient discarding procedure enabled. The numbers in table cells stand for average function calls. The fraction in parentheses denotes the fraction of runs where the global optimum was found. The absence of these fractions means that the global minimum was discovered in every run (100% success). The experimental results using the stopping rule of Equation (21) are listed in Tables 2 and 3. The experimental results for the Double Box stopping rule of Equation (22) are listed in Tables 4 and 5. Finally, for the proposed stopping rule, the results are listed in Tables 6 and 7. In addition, the boxplots for the proposed stopping rules without and with the gradient check of Equation (19) are illustrated in the Figures 1 and 2 respectively. The above results lead to a number of observations:

1. The PSO method is a robust method, and this is evident by the high success rate in finding the global minimum, although the number of particles used was relatively low (100).
2. The proposed inertia calculation method as defined in Equation (12) achieves a significant reduction in the number of calls between 11 and 25% depending on the termination criterion used. However, the presence of the gradient check mechanism of Equation (19) nullifies any gain of the method, as the rejection criterion significantly reduces the number of calls regardless of the inertia calculation mechanism used.
3. The local optimization avoidance mechanism of the gradient check drastically reduces the required number of calls for each termination criterion while maintaining the success rate of the method at extremely high levels.
4. The proposed termination criterion is significantly superior to the other two with which the comparison was made. In addition, if the termination criterion is combined with the mechanism for avoiding local optimizations, then the gain in the number of calls grows even more.

To show the effect of the proposed termination method, an additional experiment was performed in which the dimension of the sinus problem increased from 2 to 32, and in each case, all three termination techniques were tested. The result of this experiment is graphically represented in Figure 3. This graph shows that the double box method is significantly superior to the Ali method but, of course, the new proposed method further reduces the required number of function calls. In addition, the effect of the application of the rejection mechanism based on gradients of Equation (19) is illustrated graphically in Figure 4, where the proposed termination rule is applied on a series of test functions with the gradient check and without the gradient check.

### Table 1. Values for the experimental parameters.

| Parameter | Value |
|-----------|-------|
| $m$       | 100   |
| $\text{iter}_{\text{max}}$ | 100   |
| $p_1$     | 0.05  |
| $c_1$     | 1.0   |
| $c_2$     | 1.0   |
| $\omega_{\text{min}}$ | 0.4   |
| $\omega_{\text{max}}$ | 0.9   |
| $\epsilon$ | 0.001 |
| $k_{\text{max}}$ | 15    |
| FUNCTION | I1  | I2  | I3  | IP   |
|----------|-----|-----|-----|------|
| BF1      | 24,929 | 22,874 | 18,739 | 22,088 |
| BF2      | 24,043 | 22,254 | 17,172 | 20,743 |
| BRANIN   | 17,691 | 16,205 | 13,397 | 12,471 |
| CM4      | 20,117 | 22,568 | 26,867 | 14,941 |
| CAMEL    | 19,474 | 17,813 | 14,461 | 13,492 |
| EASOM    | 13,327 | 13,106 | 9969  | 9212  |
| EXP2     | 6339  | 8243 | 7853 | 3501  |
| EXP4     | 7816  | 10,066 | 10,900 | 4458  |
| EXP8     | 8667  | 10,937 | 13,126 | 4761  |
| EXP16    | 8748  | 11,402 | 15,754 | 5098  |
| EXP32    | 9567  | 12,323 | 18,189 | 5471  |
| GKLS250  | 10,907 | 12,562 | 9673  | 8552  |
| GKLS100  | 12,960 | 13,403 | 9930  | 9541  |
| GKLS350  | 15,410(0.97) | 14,722 | 10,542 | 9298  |
| GKLS3100 | 16,639 | 14,495 | 10,412 | 13,075(0.97) |
| GOLSTEIN | 20,437 | 22,877 | 16,410 | 8935  |
| GRIEVANK2| 27,620 | 24,230 | 18,473 | 20,133 |
| HANSEN   | 21,513 | 20,279 | 16,326 | 15,046 |
| HARTMAN3 | 16,233 | 17,152 | 12,305 | 6511  |
| HARTMAN6 | 47,038 | 48,947 | 46,852 | 23,431 |
| POTENTIAL3 | 31,684 | 32,175 | 36,930 | 24,463 |
| POTENTIAL4 | 184,602 | 181,231 | 168,962 | 129,267 |
| POTENTIAL5 | 74,508 | 70,519 | 76,890 | 54,042 |
| RASTRIGIN | 23,574 | 20,865 | 15,596 | 16,198 |
| ROSENBROCK4 | 145,178 | 161,136 | 160,341 | 129,891 |
| ROSENBROCK8 | 95,290 | 97,035 | 96,687 | 80,408 |
| ROSENBROCK16 | 118,614 | 116,454 | 115,122 | 97,004 |
| SHEKEL5  | 27,458 | 27,088 | 25,927 | 18,036 |
| SHEKEL7  | 27,521 | 27,271 | 25,967 | 18,805 |
| SHEKEL10 | 29,699(0.97) | 28,082 | 25,511 | 20,823 |
| TEST2N4  | 26,740 | 27,050 | 22,905 | 19,495 |
| TEST2N5  | 20,243(0.97) | 20,290 | 17,729 | 16,024(0.97) |
| TEST2N6  | 33,118 | 33,366 | 30,118 | 25,235(0.93) |
| TEST2N7  | 23,266(0.90) | 22,804 | 21,294 | 18,218(0.90) |
| SINU4    | 17,035 | 20,487 | 18,971 | 11,079 |
| SINU8    | 22,827 | 27,176 | 27,732 | 12,379 |
| SINU16   | 31,055 | 35,998 | 42,984 | 15,692 |
| SINU32   | 44,736(0.97) | 51,624 | 82,114 | 25,991 |
| TEST30N3 | 18,733 | 20,119 | 17,803 | 17,543 |
| TEST30N4 | 20,348 | 22,191 | 20,679 | 20,085 |
| TOTAL    | 1,365,704(0.99) | 1,399,419 | 1,367,612 | 1,001,436(0.99) |
Table 3. Experiments with the Ali stopping rule with the gradient check enabled.

| FUNCTION     | I1    | I2    | I3    | IP    |
|--------------|-------|-------|-------|-------|
| BF1          | 9709  | 8918  | 9531  | 10,932|
| BF2          | 10,196| 9588  | 9089  | 10,730|
| BRANIN       | 10,718| 9597  | 9813  | 9501  |
| CM4          | 6242  | 7503  | 12,531| 6985  |
| CAMEL        | 10,422| 9306  | 8491  | 9624  |
| EASOM        | 11,565| 11,366| 8497  | 8196  |
| EXP2         | 3364  | 4443  | 4558  | 1926  |
| EXP4         | 3558  | 4767  | 6023  | 2122  |
| EXP8         | 3716  | 4787  | 7753  | 2186  |
| EXP16        | 3784  | 5076  | 9696  | 2211  |
| EXP32        | 4137  | 5698  | 11,379| 2323  |
| GKL250       | 5917  | 7080  | 7517  | 5273  |
| GKL2100      | 6843  | 8261  | 7449  | 7296  |
| GKL350       | 6845  | 8076  | 7833  | 5881(0.97)|
| GKL3100      | 10,290(0.93)| 10,187| 7828  | 8066(0.97)|
| GOLDSTEIN    | 7977  | 9035  | 8505  | 4381  |
| GRIEWANK2    | 12,567| 12,222| 12,000| 12,037|
| HANSEN       | 13,441| 13,360| 11,876| 10,818|
| HARTMAN3     | 9758  | 9548  | 8123  | 4114  |
| HARTMAN6     | 12,893(0.90)| 12,889(0.93)| 22,309| 10,126(0.93)|
| POTENTIAL13  | 17,912| 16,420| 21,904| 15,969|
| POTENTIAL4   | 73,629| 64,886| 95,707| 67,084|
| POTENTIAL5   | 40,585| 35,239| 47,807| 33,661|
| RASTRIGIN    | 11,305| 10,101| 11,141| 10,046|
| ROSENROCK4   | 18,115| 21,919| 38,407| 43,093|
| ROSENROCK8   | 12,869| 14,192| 31,923| 25,405|
| ROSENROCK16  | 12,096| 13,023| 38,486| 23,165|
| SHEKEL5      | 10,347| 11,466| 14,446| 11,802|
| SHEKEL7      | 11,511| 10,521| 13,944| 10,399|
| SHEKEL10     | 10,834| 10,842| 13,785| 12,253|
| TEST2N4      | 11,133| 10,869| 12,161| 11,546|
| TEST2N5      | 10,923(0.97)| 10,315| 10,868| 11,072(0.97)|
| TEST2N6      | 12,331(0.97)| 12,345| 14,123| 15,652|
| TEST2N7      | 11,342(0.93)| 11,354| 12,118| 12,370(0.93)|
| SINU4        | 7724  | 9845  | 12,294| 6575  |
| SINU8        | 8468  | 10,969| 18,122| 5382  |
| SINU16       | 9334  | 13,213| 31,589| 9294  |
| SINU32       | 13,290| 17,502(0.97)| 63,111(0.97)| 14,959|
| TEST30N3     | 12,675| 12,954| 12,472| 12,482|
| TEST30N4     | 13,964| 14,903| 14,999| 15,389|
| TOTAL        | 494,119(0.99)| 504,585(0.99)| 720,208(0.99)| 502,326(0.99)|
Table 4. Experiments with the double box stopping rule without gradient check.

| FUNCTION    | I1  | I2  | I3  | IP  |
|-------------|-----|-----|-----|-----|
| BF1         | 6807| 6866| 6712| 6757|
| BF2         | 6102| 6150| 6057| 6207|
| BRANIN      | 4551| 4596| 4470| 4435|
| CM4         | 9814| 10,101| 9580| 9342|
| CAMEL       | 5055| 5202| 4897| 5004|
| EASOM       | 2975| 2788| 3014| 3000|
| EXP2        | 4436| 4541| 4377| 4543|
| EXP4        | 5443| 5562| 5331| 5290|
| EXP8        | 5682| 5754| 5614| 5504|
| EXP16       | 5707| 5799| 5638| 5526|
| EXP32       | 5871| 5797| 5769| 5659|
| GKLS250     | 3973| 3906| 3971| 3921|
| GKLS2100    | 4009| 3862| 4073| 3958|
| GKLS350     | 4558| 3965| 4525(0.97)| 4266|
| GKLS100     | 4701(0.87)| 4266| 4361(0.90)| 4465|
| GOLDSTEIN   | 10,259| 9145| 7945| 7625|
| GRIEWANK2   | 5932| 6194| 5700| 5915|
| HANSEN      | 6386| 6260| 5688(0.97)| 5874|
| HARTMAN3    | 4681| 4694| 4625| 4675|
| HARTMAN6    | 14,245| 14,091| 13,793| 13,825|
| POTENTIAL4  | 7219| 7206| 7532| 7234|
| POTENTIAL5  | 38,053| 37,924| 38,421| 38,897|
| POTENTIAL8  | 15,196| 14,459| 15,708| 15,358|
| RASTRIGIN   | 5915| 5797| 5944(0.83)| 5844|
| ROSENBROCK4 | 91,574| 101,485| 117,512| 76,367|
| ROSENBROCK8 | 66,648| 61,974| 58,831| 41,591|
| ROSENBCROK16 | 62,029| 54,550| 63,406| 55,800|
| SHEKEL5     | 9119| 10,271(0.97)| 8975| 8538|
| SHEKEL7     | 9197| 9831| 9638| 8732|
| SHEKEL10    | 10,417| 10,449| 9373| 9721(0.90)|
| TEST2N4     | 8512| 8272| 8884| 7992|
| TEST2N5     | 5793| 5704| 5511(0.90)| 5515|
| TEST2N6     | 9797(0.93)| 9731| 9657(0.83)| 9666(0.97)|
| TEST2N7     | 6435(0.80)| 6659| 6713(0.77)| 5990(0.87)|
| SINU4       | 7567| 7774| 7334| 7063|
| SINU8       | 9882| 10,083| 9643| 9331|
| SINU16      | 12,750| 12,947| 12,569| 12,207|
| SINU32      | 20,164| 21,112| 19,684(0.90)| 19,239|
| TEST30N3    | 6388| 7942| 5934| 5855|
| TEST30N4    | 7611| 9251| 6385| 8284|
| **TOTAL**   | 531,453(0.99)| 532,690(0.99)| 543,794(0.98)| 475,015(0.99)|
Table 5. Experiments with the double box stopping rule with the gradient check enabled.

| FUNCTION     | I1     | I2     | I3     | IP    |
|--------------|--------|--------|--------|-------|
| BF1          | 3296   | 3038   | 3063   | 3003  |
| BF2          | 2922   | 2762   | 2863   | 2845  |
| BRANIN       | 2562   | 2641   | 2538   | 2564  |
| CM4          | 3569   | 4277   | 2944   | 3230  |
| CAMEL        | 2646   | 2854   | 2467   | 2577  |
| EASOM        | 2490   | 2390   | 2479   | 2464  |
| EXP2         | 2377   | 2489   | 2261   | 2315  |
| EXP4         | 2456   | 2669   | 2282   | 2389  |
| EXP8         | 2429   | 2671   | 2268   | 2385  |
| EXP16        | 2358   | 2569   | 2227   | 2326  |
| EXP32        | 2337   | 2533   | 2248   | 2312  |
| GKLS250      | 2394   | 2535   | 2274   | 2321  |
| GKLS2100     | 2364   | 2511   | 2267   | 2333  |
| GKLS350      | 2492   | 2410   | 2212   | 2339  |
| GKLS3100     | 2800(0.90) | 2708   | 2684(0.83) | 2571 |
| GOLDSTEIN    | 3161   | 3701   | 3166   | 2799  |
| GRIEWANK2    | 3910   | 4520   | 3543   | 3641  |
| HANSEN       | 4409   | 4268   | 3755   | 4325  |
| HARTMAN3     | 2423   | 2518   | 2374   | 2425  |
| HARTMAN6     | 3913   | 4390   | 4199(0.93) | 3700 |
| POTENTIAL3   | 3951   | 4093   | 4482   | 4021  |
| POTENTIAL4   | 18,555 | 19,559 | 19,506 | 18,691 |
| POTENTIAL5   | 8771   | 8397   | 9677   | 9154  |
| RASTRIGIN    | 3111   | 3244(0.97) | 3031   | 3146  |
| ROSENROCK4   | 9729   | 12,980 | 11,453 | 8587  |
| ROSENROCK8   | 4987   | 6738   | 4688   | 5512  |
| ROSENROCK16  | 4410   | 5939   | 4553   | 4002  |
| SHEKEL5      | 3906   | 4095   | 3203   | 3495  |
| SHEKEL7      | 3119   | 3965   | 2950   | 3528  |
| SHEKEL10     | 3497(0.97) | 4464   | 3142(0.97) | 3353 |
| TEST2N4      | 3468(0.97) | 4059   | 4167   | 3881(0.93) |
| TEST2N5      | 3318(0.97) | 3786   | 2926(0.90) | 3157(0.97) |
| TEST2N6      | 4523(0.93) | 5046(0.93) | 5537(0.83) | 4066(0.97) |
| TEST2N7      | 3364(0.80) | 4191(0.90) | 4183(0.80) | 3315(0.87) |
| SINU4        | 3173   | 3807   | 2610   | 3004  |
| SINU8        | 3055   | 3742   | 2592   | 2857  |
| SINU16       | 3160   | 3746   | 3854   | 3290  |
| SINU32       | 6613   | 7377   | 6327   | 6450  |
| TEST3N3      | 5129   | 6367   | 5605   | 4451  |
| TEST3N4      | 5649   | 6441   | 6074   | 5543  |
| TOTAL        | 162,816(0.99) | 182,490(0.99) | 164,638(0.98) | 158,367(0.99) |
Table 6. Experiments with the proposed stopping rule without the gradient check.

| FUNCTION   | I1    | I2    | I3    | IP    |
|------------|-------|-------|-------|-------|
| BF1        | 5305  | 5326  | 5240  | 5209  |
| BF2        | 4760  | 4841  | 4750  | 4856  |
| BRANIN     | 3599  | 3703  | 3520  | 3443  |
| CM4        | 7674  | 7835  | 7430  | 7057  |
| CAMEL      | 3996  | 4131  | 3864  | 3825  |
| EASOM      | 2370  | 2292  | 2425  | 2478  |
| EXP2       | 3528  | 3613  | 3455  | 3675  |
| EXP4       | 4292  | 4350  | 4178  | 4020  |
| EXP8       | 4579  | 4632  | 4515  | 4278  |
| EXP16      | 4576  | 4637  | 4505  | 4236  |
| EXP32      | 4692  | 4771  | 4588  | 4296  |
| GKLS250    | 3105  | 3065  | 3115  | 3024  |
| GKLS2100   | 3193  | 3049  | 3193  | 3099  |
| GKLS350    | 3308  | 3000  | 3560  | 3401  |
| GKLS3100   | 2935(0.97) | 2777 | 3158(0.83) | 3088 |
| GOLDSTEIN  | 5534  | 5595  | 5332  | 5265  |
| GRIEVANK2  | 4225  | 4332  | 4413  | 4489  |
| HANSEN     | 3865  | 3762  | 3824  | 3769  |
| HARTMAN3   | 3724  | 3770  | 3714  | 3705  |
| HARTMAN6   | 11,901(0.97) | 11,829(0.97) | 11,386 | 10,573 |
| POTENTIAL13 | 5910  | 5850  | 6134  | 6501  |
| POTENTIAL4 | 30,880 | 30,570 | 31,180 | 30,682 |
| POTENTIAL5 | 12,021 | 11,643 | 12,521 | 13,475 |
| RASTRIGIN  | 4583  | 4595  | 4625  | 4360  |
| ROSENBROCK4 | 58,299 | 61,266 | 55,759 | 35,517 |
| ROSENBROCK8 | 31,778 | 30,888 | 30,989 | 22,055 |
| ROSENBROCK16 | 32,719 | 30,503 | 30,957 | 24,478 |
| SHEKEL5    | 6806  | 7047(0.97) | 6636  | 6233  |
| SHEKEL7    | 6807  | 7001  | 6626  | 6270  |
| SHEKEL10   | 6774  | 6987  | 6583  | 6534  |
| TEST2N4    | 6111  | 6127  | 5909  | 5893  |
| TEST2N5    | 4455(0.97) | 4558 | 4372(0.97) | 4271(0.93) |
| TEST2N6    | 7446(0.97) | 7419 | 7218(0.87) | 7122(0.93) |
| TEST2N7    | 4992(0.90) | 5057 | 4888(0.83) | 4680(0.90) |
| SINU4      | 5948  | 6043  | 5750  | 5229  |
| SINU8      | 7965  | 8095  | 7778  | 6963  |
| SINU16     | 10,121 | 10,252 | 9968  | 9219  |
| SINU32     | 16,093 | 16,509 | 15,663 | 14,478 |
| TEST30N3   | 4331  | 4953  | 4230  | 3957  |
| TEST30N4   | 6290  | 6341  | 4288  | 4717  |
| TOTAL      | 361,490(0.99) | 363,013(0.99) | 352,239(0.99) | 310,420(0.99) |
Table 7. Experiments with the proposed stopping rule with the gradient check enabled.

| FUNCTION | I1   | I2   | I3   | IP   |
|----------|------|------|------|------|
| BF1      | 2276 | 2379 | 2266 | 2250 |
| BF2      | 2157 | 2274 | 2098 | 2191 |
| BRANIN   | 2132 | 2178 | 2051 | 2170 |
| CM4      | 3098 | 3717 | 2538 | 2791 |
| CAMEL    | 2198 | 2335 | 1974 | 2058 |
| EASOM    | 2007 | 2011 | 2031 | 2084 |
| EXP2     | 1952 | 2030 | 1842 | 1861 |
| EXP4     | 2046 | 2266 | 1877 | 1909 |
| EXP6     | 1990 | 2240 | 1849 | 1879 |
| EXP16    | 1944 | 2110 | 1828 | 1838 |
| EXP32    | 1953 | 2126 | 1859 | 1867 |
| GKLS250  | 1982 | 2079 | 1850 | 1900 |
| GKLS2100 | 1983 | 2064 | 1859 | 1891 |
| GKLS350  | 1882 | 1944 | 1793 | 1831 |
| GKLS3100 | 1898 | 1909(0.97) | 1850(0.83) | 1833(0.83) |
| GOLDSTEIN| 2523 | 2670 | 2110 | 2164 |
| GRIEWANK2| 2893 | 2885 | 2791 | 2681 |
| HANSEN   | 2766 | 2879 | 2731 | 2804 |
| HARTMAN3 | 1988 | 2093 | 1949 | 2015 |
| HARTMAN6 | 3366 | 3871(0.97) | 2767 | 3133 |
| POTENTIAL3| 3312 | 3487 | 3613 | 3892 |
| POTENTIAL4| 15,392 | 16,390 | 16,223 | 17,497 |
| POTENTIAL5| 7109 | 7104 | 7732 | 8477 |
| RASTRIGIN| 2591 | 2648 | 2474 | 2732 |
| ROSENROCK4| 8023 | 12,179 | 4433 | 6025 |
| ROSENROCK8| 4376 | 6081 | 2721 | 3314 |
| ROSENBCROK16| 3643 | 4954 | 2746 | 2485 |
| SHEKEL5 | 2849 | 3296 | 2274 | 2390 |
| SHEKEL7 | 2696 | 3294 | 2262 | 2283 |
| SHEKEL10| 2624 | 3251 | 2338(0.93) | 2359 |
| TEST2N4 | 2536 | 2637 | 2427 | 2782 |
| TEST2N5 | 2266(0.97) | 2336(0.97) | 2163(0.90) | 2342(0.90) |
| TEST2N6 | 2724(0.93) | 2832 | 2694(0.80) | 3133(0.90) |
| TEST2N7 | 2283(0.80) | 2370 | 2279(0.80) | 2585(0.90) |
| SINU4   | 2789 | 3245 | 2228 | 2436 |
| SINU8   | 2601 | 3151 | 2233 | 2348 |
| SINU16  | 2721 | 3086 | 2443 | 2624 |
| SINU32  | 4652 | 5135 | 4086 | 4089 |
| TEST30N3| 3031 | 3349 | 3007 | 2562 |
| TEST30N4| 3747 | 3797 | 3250 | 3237 |
| TOTAL   | 126,999(0.99) | 142,682(0.99) | 115,539(0.98) | 122,742(0.99) |
Figure 1. Standard deviation of function calls for the termination rules without the gradient check.

Figure 2. Standard deviation of function calls for the termination rules with the gradient check.

Figure 3. Experiments with the SINU function for a series of problem dimension from $n = 2$ to $n = 32$.

Figure 4. Experiments with the proposed stopping rule with the gradient check and without the gradient check.
4. Conclusions

In the current work, three new modifications of the PSO method for locating the global minimum of continuous and differentiable functions were presented. The first modification alters the population velocity calculation in an attempt to cause large changes in velocity when the method is in its infancy and constantly finds new local minima and small velocity changes when the method is to be centered around a promising area of a global minimum. The second modification limits the number of local searches performed by the method through an asymptotic criterion based on derivative computation. The third modification introduces a new termination criterion based on the observation that the method from some iteration onwards will not be able to detect a new minimum, and therefore, its termination should be considered. All of these modifications have low computational requirements.

The proposed modifications were applied to the PSO method either one by one or all together in combination. The purpose of the method is to find the total minimum of continuous functions using the smallest possible number of function calls. The experimental results showed that the modifications significantly reduce the number of function calls even when not used in combination. This means that they can be used individually and in other variations of PSO. The reduction in the number of function calls reaches up to 80%. In addition, the amendments did not reduce the ability of the PSO to find the total minimum of the objective function. In addition, the first modification reduces the number of required calls but only when the criterion for avoiding local minimization is not present.

Author Contributions: I.G.T. and V.C. conceived of the idea and methodology and supervised the technical part regarding the software. I.G.T. conducted the experiments, employing several datasets, and provided the comparative experiments. V.C. performed the statistical analysis and prepared the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The experiments of this research work were performed at the high-performance computing system established at Knowledge and Intelligent Computing Laboratory, Department of Informatics and Telecommunications, University of Ioannina, acquired with the project “Educational Laboratory equipment of TEI of Epirus” with MIS 5007094 funded by the Operational Programme “Epirus” 2014–2020, by ERDF and national funds.

Conflicts of Interest: All authors declare that they have no conflict of interest.

References
1. Yang, L.; Robin, D.; Sannibale, F.; Steier, C.; Wan, W. Global optimization of an accelerator lattice using multiobjective genetic algorithms. Nucl. Instrum. Methods Phys. Res. Accel. Spectrom. Detect. Assoc. Equip. 2009, 609, 50–57. [CrossRef]
2. Juliano, E. Global optimization of benchmark aerodynamic cases using physics-based surrogate models. Aerosp. Sci. Technol. 2017, 67, 273–286. [CrossRef]
3. Schneider, P.I.; Santiago, X.G.; Soltwisch, V.; Hammerschmidt, M.; Burger, S.; Rockstuhl, C. Benchmarking Five Global Optimization Approaches for Nano-optical Shape Optimization and Parameter Reconstruction. ACS Photonics 2019, 6, 2726–2733. [CrossRef]
4. Heiles, S.; Johnston, R.L. Global optimization of clusters using electronic structure methods. Int. J. Quantum Chem. 2013, 113, 2091–2109. [CrossRef]
5. Shin, W.H.; Kim, J.K.; Kim, D.S.; Seok, C. GalaxyDock2: Protein-ligand docking using beta-complex and global optimization. J. Comput. Chem. 2013, 34, 2647–2656. [CrossRef] [PubMed]
6. Marques, J.M.C.; Pereira, F.B.; Llanio-Trujillo, J.L.; Abreu, P.E.; Alberti, M.; Aguilar, A.; Bartolomei, F.P.F.M. A global optimization perspective on molecular clusters. Phil. Trans. R. Soc. A 2017, 375, 20160198. [CrossRef]
7. Aguilar-Rivera, R.; Valenzuela-Rendón, M.; Rodriguez-Ortiz, J.J. Genetic algorithms and Darwinian approaches in financial applications: A survey. Expert Syst. Appl. 2015, 42, 7684–7697. [CrossRef]
8. Hosseinnezhad, V.; Babaei, E. Economic load dispatch using θ-PSO. Int. J. Electr. Power Energy Syst. 2013, 49, 160–169. [CrossRef]
9. Lee, E.K. Large-Scale Optimization-Based Classification Models in Medicine and Biology. *Ann. Biomed. Eng.* 2007, 35, 1095–1109. [CrossRef]

10. Boutros, P.; Ewing, A.; Elliott, K. Global optimization of somatic variant identification in cancer genomes with a global community challenge. *Nat. Genet.* 2014, 46, 318–319. [CrossRef]

11. Wolfe, M.A. Interval methods for global optimization. *Appl. Math. Comput.* 1996, 75, 179–206.

12. Reinking, J. GNS-SNR water level estimation using global optimization based on interval analysis. *J. Geod.* 2016, 6, 80–92. [CrossRef]

13. Price, W.L. Global optimization by controlled random search. *J. Optim. Theory Appl.* 1983, 40, 333–348. [CrossRef]

14. Gupta, R.; Chandan, M. Use of “Controlled Random Search Technique for Global Optimization” in Animal Diet Problem. *Int. Emerg. Technol. Adv. Eng.* 2013, 3, 284–287.

15. Charilogis, V.; Tsoulos, I.; Tzallas, A.; Anastasopoulos, N. An Improved Controlled Random Search Method. *Symmetry* 2021, 13, 1981. [CrossRef]

16. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. *Science* 1983, 220, 671–680. [CrossRef]

17. Tavares, R.S.; Martins, T.C.; Tsuzuki, M.S.G. Simulated annealing with adaptive neighborhood: A case study in off-line robot path planning. *Expert Syst. Appl.* 2011, 38, 2951–2965. [CrossRef]

18. Geng, X.; Chen, Z.; Yang, W.; Shi, D.; Zhao, K. Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search. *Appl. Soft Comput.* 2011, 11, 3680–3689. [CrossRef]

19. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Glob. Optim.* 1997, 11, 341–359. [CrossRef]

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

21. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN’95—International Conference on Neural Networks, Perth, Australia, 1 December 1995; Volume 4, pp. 1942–1948.

22. Poli, R.; Kennedy, J.K.; Blackwell, T. Particle swarm optimization An Overview. *Swarm Intell.* 2007, 1, 33–57. [CrossRef]

23. Trelea, I.C. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.* 2003, 85, 317–325. [CrossRef]

24. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* 2006, 1, 28–39. [CrossRef]

25. Goldberg, D. Genetic Algorithms in Search, Optimization and Machine Learning; Addison-Wesley Publishing Company: Reading, MA, USA, 1989.

26. Hamblin, S. On the practical usage of genetic algorithms in ecology and evolution. *Methods Ecol. Evol.* 2013, 4, 184–194. [CrossRef]

27. Grady, S.A.; Hussaini, M.Y.; Abdullah, M.M. Placement of wind turbines using genetic algorithms. *Renew. Energy* 2005, 30, 259–270. [CrossRef]

28. Zhou, Y.; Tan, Y. GPU-based parallel particle swarm optimization. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway, 18–21 May 2009; pp. 1493–1500.

29. Dawson, L.; Stewart, I. Improving Ant Colony Optimization performance on the GPU using CUDA. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; pp. 1901–1908.

30. Kamilaris, C.; Gergel, V. Parallel global optimization on GPU. *J. Glob. Optim.* 2016, 66, 3–20. [CrossRef]

31. Lepagnol, I.B.; Siarry, P. A survey on optimization metaheuristics. *Inf. Sci.* 2013, 237, 82–117.

32. Dokseroglu, T.; Sevinc, E.; Kucukyilmaz, T.; Cosar, A. A survey on new generation metaheuristic algorithms. *Comput. Ind. Eng.* 2019, 137, 106040. [CrossRef]

33. Hussain, K.; Salleh, M.N.; Cheng, S.; Shi, Y. Metaheuristic research: A comprehensive survey. *Artif. Intell. Rev.* 2019, 52, 2191–2233. [CrossRef]

34. Jain, N.K.; Nangia, U.; Jain, J. A Review of Particle Swarm Optimization. *J. Inst. Eng. India Ser. B* 2018, 99, 407–411. [CrossRef]

35. Khare, I.; Rangnekar, S. A review of particle swarm optimization and its applications in Solar Photovoltaic system. *Appl. Soft Comput.* 2013, 13, 2997–3006. [CrossRef]

36. Meneses, A.; Bornellos, M.; Schirru, M.R. Particle Swarm Optimization applied to the nuclear reload problem of a Pressurized Water Reactor. *Progr. Nucl. Energy* 2009, 51, 319–326. [CrossRef]

37. Shaw, R.; Srivastava, S. Particle swarm optimization: A new tool to invert geophysical data. *Geophysics* 2007, 72, F75–F83. [CrossRef]

38. Ouart, C.O.; Bicaia, E.C.; Pinto, J.C. The use of particle swarm optimization for dynamical analysis in chemical processes. *Comput. Chem. Eng.* 2002, 26, 1783–1793. [CrossRef]

39. Fang, H.; Zhou, J.; Wang, Z. Hybrid method integrating machine learning and particle swarm optimization for smart chemical process operations. *Front. Chem. Sci. Eng.* 2021, 16, 274–287. [CrossRef]

40. Wachowiak, M.P.; Smolikova, R.; Zheng, Y.J.M.; Zurada, A.S.E. An approach to multimodal biomedical image registration utilizing particle swarm optimization. *IEEE Trans. Evol. Comput.* 2004, 8, 289–301. [CrossRef]

41. Makris, Y.; Marinakis, M.; Dounias, G. Particle swarm optimization for pap-smear diagnosis. *Expert Syst. Appl.* 2008, 35, 1645–1656. [CrossRef]

42. Park, J.; Jeong, Y.; Shin, J.; Lee, K. An Improved Particle Swarm Optimization for Nonconvex Economic Dispatch Problems. *IEEE Trans. Power Syst.* 2010, 25, 156–166. [CrossRef]
44. Clerc, M. The swarm and the queen: Towserds a deterministic and adaptive particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999; Volume 3, pp. 1951–1957.

45. Juan, H.; Laihing, Y.; Kaira, Z. Enhanced Self-Adaptive Search Capability Particle Swarm Optimization. In Proceedings of the 2008 Eighth International Conference on Intelligent Systems Design and Applications, Kaohsiung, Taiwan, 28 November 2008; pp. 49–55.

46. Hou, Z.X. Wiener model identification based on adaptive particle swarm optimization. In Proceedings of the 2008 International Conference on Machine Learning and Cybernetics, Kunming, China, 10 January 2008; pp. 1041–1045.

47. Ratnaweera, A.; Halgamuge, S.K.; Watson, H.C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. 2004, 8, 240–255. [CrossRef]

48. Stacey, A.; Jancic, M.; Grundy, I. Particle swarm optimization with mutation. In Proceedings of the 2003 Congress on Evolutionary Computation, CEC ’03, Canberra, Australia, 12 December 2004; pp. 1425–1430.

49. Pant, M.; Thangaraj, R.; Abraham, A. Particle Swarm Optimization Using Adaptive Mutation. In Proceedings of the 2008 19th International Workshop on Database and Expert Systems Applications, Turin, Italy, 1–5 September 2008; pp. 519–523.

50. Higashi, N.; Iba, H. Particle swarm optimization with Gaussian mutation. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS’03 (Cat. No.03EX706), Indianapolis, IN, USA, 26–26 April 2003; pp. 72–79.

51. Engelbrecht, A. Particle swarm optimization: Velocity initialization. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, Australia, 10–15 June 2012; pp. 1–8.

52. Liu, B.; Wang, L.; Jin, Y.H.; Tang, F.; Huang, D.X. Improved particle swarm optimization combined with chaos. Chaos Solitons Fract. 2005, 25, 1261–1271. [CrossRef]

53. Shi, X.H.; Liang, Y.C.; Lee, H.P.; Lu, C.; Wang, L.M. An improved GA and a novel PSO-GA based hybrid algorithm. Inf. Proc. Lett. 2005, 93, 255–261. [CrossRef]

54. Garg, H. A hybrid PSO-GA algorithm for constrained optimization problems. Appl. Math. Comput. 2016, 274, 292–305. [CrossRef]

55. Schutte, J.F.; Reinbolt, J.A.; Fregly, B.J.; Haftka, R.T.; George, A.D. Parallel global optimization with the particle swarm algorithm. Int. J. Numer. Meth. Eng. 2004, 61, 2296–2315. [CrossRef]

56. Shi, X.H.; Liang, Y.C.; Lee, H.P.; Lu, C.; Wang, L.M. An improved GA and a novel PSO-GA based hybrid algorithm. Inf. Proc. Lett. 2005, 93, 255–261. [CrossRef]

57. Venter, G.; Sobieszczanski-Sobieski, J. Parallel Particle Swarm Optimization Algorithm Accelerated by Asynchronous Evaluations. J. Aerosp. Comput. Inf. Commun. 2006, 3, 123–137. [CrossRef]

58. Gaing, Z.L. Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Trans. Power Syst. 2003, 18, 1187–1195. [CrossRef]

59. Yang, X.; Yuan, J.; Yuan, J.; Mao, H. A modified particle swarm optimizer with dynamic adaptation. Appl. Math. Comput. 2007, 189, 1205–1213. [CrossRef]

60. Jiang, Y.; Hu, T.; Huang, C.; Wu, X. An improved particle swarm optimization algorithm. Appl. Math. Comput. 2007, 193, 231–239. [CrossRef]

61. Bogdanova, A.; Junior, J.P.; Aranha, C. Franken-Swarm: Grammatical Evolution for the Automatic Generation of Swarm-like Meta-Heuristics. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, New York, NY, USA, 15 July 2018; pp. 411–412.

62. O’Neill, M.; Ryan, C. Grammatical evolution. IEEE Trans. Evol. Comput. 2001, 5, 349–358. [CrossRef]

63. Pan, X.; Xue, L.; Lu, Y. Hybrid particle swarm optimization with simulated annealing. Multimed Tools Appl. 2019, 78, 29921–29936. [CrossRef]

64. Mughal, M.A.; Ma, Q.; Xiao, C. Photovoltaic cell parameter estimation using hybrid particle swarm optimization and simulated annealing. Energies 2017, 10, 1213. [CrossRef]

65. Lin, G.H.; Zhang, J.; Liu, Z.H. Hybrid particle swarm optimization with differential evolution for numerical and engineering optimization. Int. J. Autom. Comput. 2018, 15, 103–114. [CrossRef]

66. Epitropakis, M.G.; Plagianakos, V.P.; Vrahatis, M.N. Evolving cognitive and social experience in particle swarm optimization through Differential Evolution: A hybrid approach. Inf. Sci. 2012, 216, 50–92. [CrossRef]

67. Wang, W.; Wu, J.M.; Liu, J.H. A Particle Swarm Optimization Based on Chaotic Neighborhood Search to Avoid Premature Convergence. In Proceedings of the 2009 Third International Conference on Genetic and Evolutionary Computing, Washington, DC, USA, 14 October 2009; pp. 633–636.

68. Eberhart, R.C.; Shi, Y.H. Tracking and optimizing dynamic systems with particle swarms. In Proceedings of the Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001.

69. Shi, Y.H.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the Congress on Evolutionary Computation, Washington, DC, USA, 6–9 July 1999.

70. Shi, Y.H.; Eberhart, R.C. Experimental study of particle swarm optimization. In Proceedings of the SCI2000 Conference, Orlando, FL, USA, 23–26 July 2000.

71. Zheng, Y.; Ma, L.; Zhang, L.; Qian, J. Empirical study of particle swarm optimizer with an increasing inertia weight. IEEE Congr. Evol. Comput. 2003, 1, 221–226.
72. Zheng, Y.; Ma, L.; Zhang, L.; Qian, J. On the convergence analysis and parameter selection in particle swarm optimization. In Proceedings of the Second International Conference on Machine Learning and Cybernetics, Xi’an, China, 5 November 2003.

73. Tsoulos, I.G.; Lagaris, I.E. MinFinder: Locating all the local minima of a function. Comput. Phys. Commun. 2006, 174, 166–179. [CrossRef]

74. Ali, M.M.; Kaelo, P. Improved particle swarm algorithms for global optimization. Appl. Math. Comput. 2008, 196, 578–593. [CrossRef]

75. Tsoulos, I.G. Modifications of real code genetic algorithm for global optimization. Appl. Math. Comput. 2008, 203, 598–607. [CrossRef]

76. Ali, M.M.; Khompatraporn, C.; Zabinsky, Z.B. A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems. J. Glob. Optim. 2005, 31, 635–672. [CrossRef]

77. Floudas, C.A.; Pardalos, P.M.; Adjiman, C.; Esposito, W.; Gümüş, Z.; Harding, S.; Klepeis, J.; Meyer, C.; Schweiger, C. Handbook of Test Problems in Local and Global Optimization; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1999.

78. Koyuncu, H.; Ceylan, R. A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems. J. Comput. Des. Eng. 2019, 6, 129–142. [CrossRef]

79. Siarry, P.; Berthiau, G.; François, D.; Haussy, J., Enhanced simulated annealing for globally minimizing functions of many-continuous variables. ACM Trans. Math. Softw. 1997, 23, 209–228. [CrossRef]

80. Tsoulos, I.G.; Lagaris, I.E. GenMin: An enhanced genetic algorithm for global optimization. Comput. Phys. Commun. 2008, 178, 843–851. [CrossRef]

81. Gaviano, M.; Ksasov, D.E.; Lera, D.; Sergeyev, Y.D. Software for generation of classes of test functions with known local and global minima for global optimization. ACM Trans. Math. Softw. 2003, 29, 469–480. [CrossRef]

82. Lennard-Jones, J.E. On the Determination of Molecular Fields. Proc. R. Soc. Lond. A 1924, 106, 463–477.

83. Powell, M.J.D. A Tolerant Algorithm for Linearly Constrained Optimization Calculations. Math. Program. 1989, 45, 547–566. [CrossRef]