Random Orthogonal Search with Triangular and Quadratic Distributions (TROS and QROS)
Parameterless Algorithms for Global Optimization
Tong, Bruce Kwong-Bun; Sung, Chi Wan; Wong, Wing Shing

Published in:
Applied Sciences (Switzerland)

Published: 01/02/2023

Document Version:
Final Published version, also known as Publisher's PDF, Publisher's Final version or Version of Record

License:
CC BY

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.3390/app13031391

Publication details:

Citing this paper
Please note that where the full-text provided on CityU Scholars is the Post-print version (also known as Accepted Author Manuscript, Peer-reviewed or Author Final version), it may differ from the Final Published version. When citing, ensure that you check and use the publisher's definitive version for pagination and other details.

General rights
Copyright for the publications made accessible via the CityU Scholars portal is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights. Users may not further distribute the material or use it for any profit-making activity or commercial gain.

Publisher permission
Permission for previously published items are in accordance with publisher's copyright policies sourced from the SHERPA RoMEO database. Links to full text versions (either Published or Post-print) are only available if corresponding publishers allow open access.

Take down policy
Contact lbscholars@cityu.edu.hk if you believe that this document breaches copyright and provide us with details. We will remove access to the work immediately and investigate your claim.

Download date: 19/11/2023
Random Orthogonal Search with Triangular and Quadratic Distributions (TROS and QROS): Parameterless Algorithms for Global Optimization

Bruce Kwong-Bun Tong 1,2,*, Chi Wan Sung 3 and Wing Shing Wong 2

1 Department of Electronic Engineering and Computer Science, Hong Kong Metropolitan University, Hong Kong, China
2 Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, China
3 Department of Electrical Engineering, City University of Hong Kong, Hong Kong, China
* Correspondence: kbtong@hkmu.edu.hk; Tel.: +852-3120-2607

Abstract: In this paper, the behavior and performance of Pure Random Orthogonal Search (PROS), a parameter-free evolutionary algorithm (EA) that outperforms many existing EAs on the well-known benchmark functions with finite-time budget, are analyzed. The sufficient conditions to converge to the global optimum are also determined. In addition, we propose two modifications to PROS, namely Triangular-Distributed Random Orthogonal Search (TROS) and Quadratic-Distributed Random Orthogonal Search (QROS). With our local search mechanism, both modified algorithms improve the convergence rates and the errors of the obtained solutions significantly on the benchmark functions while preserving the advantages of PROS: parameterless, excellent computational efficiency, ease of applying to all kinds of applications, and high performance with finite-time search budget. The experimental results show that both TROS and QROS are competitive in comparison to several classic metaheuristic optimization algorithms.

Keywords: global optimization; metaheuristics; orthogonal search; parameterless; TROS; QROS

1. Introduction

Black-box optimization refers to optimizing an objective function in the absence of prior knowledge of the function. The only knowledge of the objective function is the observed outputs of the given inputs. Metaheuristics such as evolutionary computation (EC) techniques and evolutionary algorithms (EAs) are widely employed for black-box optimization of various fields across science and engineering [1–3] due to their certain primary advantages over the traditional techniques (such as Newton’s method and gradient descent) including high robustness in solving complex real-world problems, and domain knowledge and regularity (e.g., convexity, continuity, differentiability) of the functions to be optimized are generally not required [4].

However, these algorithms often require a large number of function evaluations to evolve or update the candidate solutions in order for errors reduced to a satisfactory level [5]. For many real-world problems, evaluating the objective function is often costly. These function evaluations may involve conducting computational intensive numerical simulations or expensive physical experiments [6–9]. When only a limited number of objective function evaluations is affordable, EC’s trial and error approach becomes unattractive.

Moreover, EC techniques are commonly recognized as heuristic search algorithms because their theoretical analysis often lags behind the development of the algorithms [4,10]. Their performance is often sensitive to the problem-dependent hyper-parameters [11]. Various approaches have been suggested to overcome these drawbacks. For example, building an inexpensive surrogate model for evaluation so as to reduce the number of
objective function evaluations \([5,12]\), and automatic or self-adaptive tuning of the hyper-parameters \([11,13,14]\).

In this paper, we propose two algorithms for expensive black-box optimization. The two algorithms are modified from Pure Random Orthogonal Search (PROS) \([15]\). PROS is a \((1 + 1)\) Evolutionary Strategy \((1 + 1)\) ES, a kind of EA, that has only one parent in the population and generates one child in each generation that competes with its parent. The finite-time performance of PROS is promising and outperforms several existing EC techniques on most benchmark functions \([15]\). Unlike those well known EAs, PROS is free of parameters and thus the tuning of parameters is not required. To the best of our knowledge, there were no mathematical analyses on the behaviour and performance of the algorithm reported in the literature. In this paper, we aim to fill this gap. The major contributions of this research work are summarized as follows:

1. The behavior and performance analysis of PROS, which are not available in \([15]\), are provided here.
2. Two effective novel \((1 + 1)\) ES based on PROS, namely TROS and QROS, are proposed and they outperform PROS on a set of benchmark problems.
3. The performance of TROS and QROS are found competitive with three well-known optimization algorithms (GA, PSO and DE) on a set of benchmark problems.

**Problem Formulation**

In this paper, we restrict attention to global optimization of expensive black-box functions. The goal of global optimization is to find \(x^*\) which is a global minimum of \(f\),

\[
x^* = \arg \min_{x \in \Omega} f(x)
\]

where \(f: \mathbb{R}^D \rightarrow \mathbb{R}\) is a scalar-valued objective function defined on the decision space \(\Omega \subseteq \mathbb{R}^D\) and \(x = (x_1, x_2, ..., x_D)\) represents a vector of decision variables with \(D\) dimensions.

We have made the following assumptions.

**Assumption 1.** \(f\) has a single global optimum \(f^* = \min_{x \in \Omega} f(x)\).

**Assumption 2.** \(f\) can be evaluated at all points of \(\Omega\) in an arbitrary order.

**Assumption 3.** The slope of \(f\) is bounded with a Lipschitz constant \(L\) such that

\[
|f(x_1) - f(x_2)| \leq L\|x_1 - x_2\|, \quad \forall x_1, x_2 \in \Omega,
\]

where \(L > 0\) and \(\|\cdot\|\) denotes the Euclidean norm.

It has to be noted that although \(f\) is assumed a Lipschitz function, the corresponding Lipschitz constant \(L\) is unknown to an optimization algorithm.

**2. Analysis of the Pure Random Orthogonal Search (PROS) Algorithm**

The PROS algorithm was originally proposed by Plevris et al. in \([15]\). We describe it with our revised notation as follows:

Initially, a candidate solution vector \(x^{(0)}\) is generated randomly from \(\Omega\) (lines 1 and 2). Then, for each iteration \(t\), \(j\) is chosen randomly from 1 to \(D\) (line 4) and a real number \(r\) is drawn randomly with uniform distribution within the search space of the \(j\)-th decision variable (line 5). A new candidate solution vector \(y\) is obtained by replacing the value of the \(j\)-th decision variable of the current best solution vector \(x^{(t)}\) with \(r\) (lines 6 and 7). The new candidate solution vector \(y\) is evaluated and the new best solution vector \(x^{(t+1)}\) is then updated based on the comparison result between \(f(y)\) and \(f(x^{(t)})\). If \(f(y)\) is smaller, \(y\) is accepted as the new best solution vector. Otherwise, \(y\) is rejected (lines 8 to 12). The iteration counter \(t\) is updated and the search process is repeated until a termination.
criterion is met (lines 13 and 14). Finally, the best solution vector found by the algorithm is returned as the final result (line 15).

In the following analysis, we assume the algorithm runs continuously until the global optimum is found (defined below of this section). It is because we are interested in analyzing its ultimate performance. In practice, it may take an infinitely long time to reach the global optimum. Therefore, other termination criteria are usually adopted. For example, the algorithm stops when no improvement is made after a certain number of iterations or when a pre-defined maximum number of iterations is reached.

The error of the algorithm after running for \( t \) iterations is given by

\[
e_t = f(x^{(t)}) - f(x^*).
\]

Here, \( x^{(t)} \) is the solution vector found by the algorithm in the \( t \)-th iteration and is also the best solution vector found by the algorithm after \( t \) iterations. It is because the values of the solution vectors found by PROS is monotonically non-increasing. That is,

\[
f(x^{(t)}) \leq f(x^{(t-1)}) \leq \cdots \leq f(x^{(0)}),
\]

Consider the pseudo code from line 8 to line 12 of Algorithm 1. In the \( t \)-th iteration, when a better solution \( y \) is found, it will be assigned to \( x^{(t+1)} \). In this case, \( f(x^{(t+1)}) < f(x^{(t)}) \). When no better solution is found in the \( t \)-th iteration, \( x^{(t)} \) will be assigned to \( x^{(t+1)} \). In this case, \( f(x^{(t+1)}) = f(x^{(t)}) \). Combining the two conditions, \( f(x^{(t+1)}) \leq f(x^{(t)}) \) and the inequalities then follow.

**Algorithm 1: Pure Random Orthogonal Search (PROS)**

```
input : nil  
output : the best solution vector \( x^{(t)} \) found by the algorithm 
\( t \leftarrow 0; \)
Initialize \( x^{(t)} = (x_1, x_2, \ldots, x_D) \) randomly from 
\( \Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_D, b_D]; \)
repeat 
  Sample a random integer \( j \in \{1, \ldots, D\}; \)
  Sample a random real number \( r \sim U(a_j, b_j); \)
  \( y \leftarrow x^{(t)}; \)
  \( y_j \leftarrow r; \)
  if \( f(y) < f(x^{(t)}) \) then 
    \( x^{(t+1)} \leftarrow y; \)
  else 
    \( x^{(t+1)} \leftarrow x^{(t)}; \)
  end 
  \( t \leftarrow t + 1; \)
until a termination criterion is met; 
return \( x^{(t)} \)
```

**Definition 1 (Region of Global Optimum).** It is said that a candidate solution vector \( x \) is in the region of global optimum \( R_\epsilon \) if the error of \( x \) is not larger than \( \epsilon \). That is,

\[
R_\epsilon = \{x : x \in \Omega \text{ and } f(x) - f^* \leq \epsilon\},
\]

where \( \epsilon \) is a small positive real number denoting the tolerance of error.

**Definition 2 (Convergence Time).** The convergence time \( T \) of an algorithm is defined to be the first time \( t \) when

\[
x^{(t)} \in R_\epsilon.
\]
2.1. One-Dimensional Functions

In this section, we study the performance of PROS on one-dimensional functions defined on the domain \( \Omega = [a, b] \). Since we are interested in the case where \( \epsilon \) is small, we assume that \( \frac{2\epsilon}{L} < b - a \).

**Lemma 1.** For any \( \epsilon > 0 \), the region of the global minimum contains an interval of minimum length \( \frac{\epsilon}{L} \) for all one-dimensional functions.

**Proof.** Let \( x^* = (x^*_1) \) be the global optimum, \( x^l = (x^l_1) \in R_{\epsilon} \) be a point on the left of \( x^* \) and \( x^r = (x^r_1) \in R_{\epsilon} \) be a point on the right of \( x^* \). We are going to find a sufficient condition to ensure that \([x^l_1, x^r_1]\) is in the region of global optimum. Since \( f \) is a Lipschitz function, the interval \([x^l_1, x^r_1]\) belongs to \( R_{\epsilon} \) if
\[
f(x^l) - f(x^*) \leq L\|x^l - x^*\| = L(x^*_1 - x^l_1) \leq \epsilon,
\]
or equivalently, \( x^r_1 \geq x^*_1 - \frac{\epsilon}{L} \). We want to make \( x^l_1 \) as small as possible, so we let
\[
x^l_1 = \max\{a, x^*_1 - \frac{\epsilon}{L}\}.
\]
Similarly, the interval \([x^*_1, x^r_1]\) belongs to \( R_{\epsilon} \) if
\[
x^r_1 = \min\{b, x^*_1 + \frac{\epsilon}{L}\}.
\]
Since we have assumed \( \frac{2\epsilon}{L} < b - a \), it is impossible that \( x^l_1 = a \) and \( x^r_1 = b \) hold simultaneously. Therefore, at least one of the intervals has length \( \frac{\epsilon}{L} \). Hence, the length of the region of global optimum is at least \( \frac{\epsilon}{L} \). \( \square \)

**Theorem 1.** The expected convergence time of PROS for one-dimensional functions is bounded above by \( \frac{L(b-a)}{\epsilon} \).

**Proof.** Let \( l \) be the length of the region of global optimum, \( R_{\epsilon} \). The probability that a point chosen uniformly at random falls in \( R_{\epsilon} \) is given by
\[
p = \frac{l}{b-a} \geq \frac{\epsilon}{L(b-a)}.
\]

The convergence time \( T \) is geometrically distributed with parameter \( p \). Its expected value is given by
\[
\mathbb{E}[T] = \frac{1}{p} \leq \frac{L(b-a)}{\epsilon}.
\]
\( \square \)

**Corollary 1.** PROS converges in probability to the region of global optimum for all one-dimensional functions, i.e.,
\[
\lim_{t \to \infty} P\{x^{(t)} \in R_{\epsilon}\} = 1, \quad \forall \epsilon > 0.
\]

**Proof.** For any realization \( \{x^{(t)} : t = 0, 1, 2, \ldots\} \), \( f(x^{(t)}) \) is monotonically non-increasing in \( t \). If \( x^{(t)} \in R_{\epsilon} \), then \( x^{(t+\tau)} \in R_{\epsilon} \) for all non-negative integer \( \tau \). Therefore, \( P\{x^{(t)} \in R_{\epsilon}\} \) is monotonically non-decreasing in \( t \). Since the sequence is bounded above by 1, it is convergent.

Given any \( \epsilon > 0 \), the expected convergence time is bounded according to Theorem 1. If \( \lim_{t \to \infty} P\{x^{(t)} \notin R_{\epsilon}\} \) was non-zero, then the expected convergence time would be unbounded, which leads to a contradiction. \( \square \)
2.2. Multi-Dimensional Functions

In this section, we study the performance of PROS on $D$-dimensional functions with domain $\Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_D, b_D]$, where $D > 1$. In particular, we focus on the class of totally separable functions as defined below. For notation simplicity, define $x_{-i} \triangleq (x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_D)$.

**Definition 3 (Partially Separable Function).** A function $f(x)$ is partially separable with coordinate $i$ if $\arg \min_{x_i} f(x_i, x_{-i})$ is independent of $x_{-i}$.

An example of partially separable functions with coordinate $i = 1$ is:

$$f(x) = x_1^2 + \sum_{i=2}^{D-1} (x_i - x_{i+1})^2.$$  

**Definition 4 (Totally Separable Function).** A function $f(x)$ is totally separable if it is partially separable with every of the $D$ coordinates.

An example of totally separable functions is:

$$f(x) = \prod_{i=1}^{D} (x_i^2 + 1).$$

In each iteration, PROS minimizes $f$ in one of the random coordinate $i$. As $f$ is totally separable, each coordinate $i$ can be minimized independently. The following result shows that PROS converges to the region of global optimum in probability.

**Theorem 2.** PROS converges in probability to the region of global optimum for all totally separable functions, i.e.,

$$\lim_{t \to \infty} P\{x^{(t)} \in R_\epsilon\} = 1, \ \forall \epsilon > 0.$$  

**Proof.** Since $f$ is Lipschitz, given any $x_{-i}$, we have

$$|f(x_i, x_{-i}) - f(\tilde{x}_i, x_{-i})| \leq L|x_i - \tilde{x}_i|.$$  

Therefore, $f$ can be regarded as a one-dimensional Lipschitz function in $x_i$ with the same constant $L$. As shown in the previous subsection, there is an interval $V_{\epsilon_i}$ of minimum length $\frac{\epsilon_i}{L}$ such that for all $x_i \in V_{\epsilon_i}$,

$$|f(x_i, x_{-i}) - f(x_i', x_{-i})| \leq \epsilon_i.$$  

Note that $V_{\epsilon_i}$ is independent of $x_{-i}$. Under PROS, it is clear that

$$\lim_{t \to \infty} P\{x_i^{(t)} \in V_{\epsilon_i}\} = 1, \ \forall \epsilon_i > 0.$$  

Since $f$ is Lipschitz, it is a continuous function. Given any $\epsilon > 0$, there exists sufficiently small positive $\epsilon_i$’s such that when $x_i^{(t)} \in V_{\epsilon_i}$ for all $i$, we must have $x^{(t)} \in R_\epsilon$. The statement then follows from

$$P\{x^{(t)} \in R_\epsilon\} \geq P\{x_i^{(t)} \in V_{\epsilon_i}, \forall i\} = \prod_{i=1}^{D} P\{x_i^{(t)} \in V_{\epsilon_i}\}.$$  

The expected convergence time can be obtained for the following subclass of totally separable functions.
Definition 5 (Additively Separable Function). A function $f$ is additively separable if $f(x)$ can be written in the form of $f_1(x_1) + f_2(x_2) + \cdots + f_D(x_D)$, where $x = (x_1, x_2, \ldots, x_D)$ and $f_1, f_2, \ldots, f_D$ are one-dimensional functions.

An example of additively separable functions is the sum-of-spheres function:

$$f_{sphere}(x) = \sum_{i=1}^{D} x_i^2.$$  

For the optimization of $D$-dimensional additively separable functions, it is equivalent to optimize $D$ one-dimensional functions independently, i.e.,

$$\min_{x \in \Omega} f(x) = \sum_{i=1}^{D} \min_{x_i \in [a_i, b_i]} f_i(x_i).$$

Theorem 3. The expected convergence time of PROS for $D$-dimensional additively separable functions is bounded above by

$$\mathbb{E}[T] \leq \frac{D^2 L}{\epsilon} \sum_{i=1}^{D} (b_i - a_i).$$

Proof. A point $x = (x_1, x_2, \ldots, x_D)$ belongs to $R_\epsilon$ if

$$\sum_{i=1}^{D} (f_i(x_i) - f_i^*) \leq \epsilon,$$

where $f_i^*$ is the global minimum of the $i$-th one-dimensional function. Then,

$$f_i(x_i) - f_i^* \leq \frac{\epsilon}{D}, \text{ for } i \in \{1, \ldots, D\}$$

is a sufficient condition for $x \in R_\epsilon$.

Let $S_i$ be the iteration time PROS enters the region for $x_i$ as stated in (1). Under PROS, at each iteration $t$, the coordinate to be optimized is chosen uniformly at random. As in the proof of Theorem 1, $S_i$ is a geometric random variable with parameter

$$p_i \geq \frac{1}{D} \left( \frac{\epsilon / D}{L(b_i - a_i)} \right) = \frac{\epsilon}{D^2 L(b_i - a_i)}.$$  

Note that $S_i$’s are not independent. We bound the convergence time $T$ as follows:

$$T \leq \max\{S_1, \ldots, S_D\} \leq S_1 + \cdots + S_D.$$  

Hence,

$$\mathbb{E}[T] \leq \sum_{i=1}^{D} \mathbb{E}[S_i] = \sum_{i=1}^{D} \frac{1}{p_i} \leq \frac{D^2 L}{\epsilon} \sum_{i=1}^{D} (b_i - a_i).$$

3. Modified PROS with Local Search Mechanism

Although the PROS algorithm converges to the global optimum, provided that the sufficient conditions are satisfied, it converges slowly when compared with other well-known EC algorithms [15]. The major reason is PROS simply performs uniform orthogonal search in every iteration with no local search mechanism. The probability of finding an improved solution (defined in the below subsection) diminishes as it moves closer to the global optimum. Consider the situation that in the $t$-th iteration, $x^{(t)}$ is already very close
to \( x^* \) but has not yet fallen into the region of global optimum. There is a high chance that \( x^{(t+1)} \) would reach the region of the global optimum if a narrow-range local search is performed in the \((t+1)\)-th iteration. However, with uniform orthogonal search, the chance to reach the global optimum is relatively low, thus making it converge slowly. One may consider using a sampling policy other than uniform to perform local search.

3.1. Triangular-Distributed Random Orthogonal Search (TROS)

In this section, we present our first proposed algorithm called Triangular-Distributed Random Orthogonal Search (TROS). The TROS algorithm (Algorithm 2) is presented as follows:

**Algorithm 2:** Triangular-Distributed Random Orthogonal Search (TROS)

```plaintext
input : nil
output : the best solution vector \( x^{(t)} \) found by the algorithm

\( t \leftarrow 0; \)
Initialize \( x^{(t)} = (x_1, x_2, \ldots, x_D) \) randomly from \( \Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_D, b_D]; \)
repeat
  Sample a random integer \( j \in \{1, \ldots, D\}; \)
  Sample a random real number \( r \sim T(a_j, b_j, x_j^{(t)}) \);
  \( y \leftarrow x^{(t)}; \)
  \( y_j \leftarrow r; \)
  if \( f(y) < f(x^{(t)}) \) then
    \( x^{(t+1)} \leftarrow y; \)
  else
    \( x^{(t+1)} \leftarrow x^{(t)}; \)
  end
  \( t \leftarrow t + 1; \)
until a termination criterion is met;
return \( x^{(t)} \)
```

Compared with PROS, TROS has one change in line 5 of Algorithm 1. Instead of sampling the next point of the \( j \)-th decision variable using uniform distribution, the triangular distribution \( T \) is used in the TROS algorithm. The probability density function of the triangular distribution \( T(a, b, c) \) is

\[
f_T(x) = \begin{cases} \frac{2}{b-a} \left( \frac{x-a}{b-a} \right), & a < x \leq c \\ \frac{2}{b-a} \left( \frac{b-x}{b-a} \right), & c < x < b \\ 0, & x \leq a \text{ or } x \geq b \end{cases} \quad (2)
\]

where \( a, b \) and \( c \) are the parameters of the distribution that represents the lower limit of \( x \), upper limit of \( x \) and mode of \( x \), respectively. The triangular distribution is illustrated in Figure 1.

In each iteration of TROS, the next point is sampled using the triangular distribution with the settings \( a = a_j, b = b_j, c = x_j^{(t)} \) where \( j \in \{1, 2, \ldots, D\} \) is the randomly chosen decision variable for the current iteration, \( a_j \) and \( b_j \) are the lower bound and upper bound of the \( j \)-th decision variable, and \( x_j^{(t)} \) is the value of the \( j \)-th decision variable of the current best solution vector. With this distribution, there is a higher chance to draw a sample that is near to the current position \( x_j^{(t)} \) than is far from the current position. As a result, the algorithm performs exploitation (that is, encourages local search) on the \( j \)-th decision variable.
Figure 1. The probability density function of the triangular distribution $T(a, b, c)$, the quadratic distribution $Q(a, b, c)$, and uniform distribution $U(a, b)$.

3.2. Quadratic-Distributed Random Orthogonal Search (QROS)

In this section, we present our second proposed algorithm called Quadratic-Distributed Random Orthogonal Search (QROS). The QROS algorithm (Algorithm 3) is presented as follows:

**Algorithm 3: Quadratic-Distributed Random Orthogonal Search (QROS)**

- **input**: nil
- **output**: the best solution vector $x^{(t)}$ found by the algorithm

$t \leftarrow 0$;

Initialize $x^{(t)} = (x_1, x_2, \ldots, x_D)$ randomly from

$\Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_D, b_D]$;

repeat

Sample a random integer $j \in \{1, \ldots, D\}$;

Sample a random real number $r \sim Q(a_j, b_j, x_j^{(t)});$  

$y \leftarrow x^{(t)}$;  

$y_j \leftarrow r$;

if $f(y) < f(x^{(t)})$ then

$x^{(t+1)} \leftarrow y$;

else

$x^{(t+1)} \leftarrow x^{(t)}$;

end

$t \leftarrow t + 1$;

until a termination criterion is met;

return $x^{(t)}$
Compared with TROS, QROS has one change in line 5 of Algorithm 2. The value of the $j$-th decision variable is sampled using the quadratic distribution $Q$. The probability density function of the quadratic distribution $Q(a, b, c)$ is

$$
f_Q(x) = \begin{cases} 
\alpha_Q(x - a)^2, & a < x \leq c \\
\beta_Q(x - b)^2, & c < x < b \\
0, & x \leq a \text{ or } x \geq b 
\end{cases}
$$

where $\alpha_Q = \frac{3}{(b-a)(c-a)}$, $\beta_Q = \frac{3}{(b-a)(c-b)}$, $a$, $b$, and $c$ are the lower limit of $x$, upper limit of $x$ and mode of $x$, respectively. The quadratic distribution is illustrated in Figure 1.

In each iteration of QROS, the next point is sampled using the quadratic distribution with the settings $a = a_j$, $b = b_j$, $c = x_j(t)$, where $j \in \{1, 2, \ldots, D\}$ is the randomly chosen decision variable for the current iteration, $a_j$ and $b_j$ are the lower bound and upper bound of the $j$-th decision variable, and $x_j(t)$ is the value of the $j$-th decision variable of the current best solution vector. With this distribution, there is a higher chance to draw a sample that is near to the current position $x_j(t)$ than is far from the current position. Compared with TROS, QROS encourages exploitation even more than TROS.

It has to be noted that both TROS and QROS are still “parameter-free” algorithms, as $a$ and $b$ are fixed boundaries, and $c$ is determined based on the current best solution vector. With a simple modification, both TROS and QROS are able to improve the convergence speed of PROS and keep the major properties of PROS: parameterless, excellent computational efficiency, easy to apply to all kinds of applications, and high performance with finite-time search budget.

3.3. Analysis of the Modified Algorithms

In this subsection, we are going to explain the motivation and analyze the performance of the two algorithms. Sampling using uniform distribution can be considered as performing global search on a decision variable, while sampling using the triangular distribution and the quadratic distribution can be considered as performing local search on a decision variable.

**Definition 6 (Improved Solution).** The candidate solution vector $x^{(t+1)}$ found by the algorithm is said to be improved if its error is less than the error of the current best solution. That is,

$$f(x^{(t+1)}) - f(x^*) < f(x^{(t)}) - f(x^*)$$

or simply

$$f(x^{(t+1)}) < f(x^{(t)})$$

where $t$ denotes the current iteration.

For continuous (or Lipschitz) unimodal functions defined on a bounded domain, finding an improved solution vector always lead to the convergence to the global optimum. However, it may not be true for multimodal functions. Therefore, we are not interested in simply finding an improved solution vector. Here, we define a restrictive subset of improved solution vectors.

**Definition 7 (Tame Solution).** The candidate solution vector $x^{(t+1)}$ found by the algorithm is said to be a tame solution if the errors of all points in between the line containing the candidate solution and the optimum solution are all less than the error of the current best solution. That is,

$$f(ax^{(t+1)} + (1-a)x^*) < f(x^{(t)}), \forall a \in [0, 1]$$

where $t$ denotes the current iteration.
A local optimum in a multimodal function is not considered as a tame solution if there exists a high hill in between the local optimum and the global optimum. The purpose of defining tame solution is when a tame solution is found, it does not only mean the solution is improved but also implies one could apply some simple techniques to converge to the global optimum. Therefore, we are interested in knowing the probability of finding a tame solution by each of the algorithms.

We define following terms for the analysis of the performance of the proposed algorithms. Let $x^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_D^{(t)})$ be the current best solution vector, and $j \in \{1, \ldots, D\}$ be the chosen decision variable of the current iteration. Given $x_{-j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t)}, \ldots, x_D^{(t)})$, assume $f(x_{-j}^{(t)}, x_j)$ has a unique global minimum and is denoted as $x_j^*$. That is,

$$f(x_{-j}^{(t)}, x_j^*) \leq f(x_{-j}^{(t)}, x_j), \forall x_j \in [a_j, b_j].$$

Let $R_j$ be the set of values of the $j$-th decision variable that belong to improved solution vectors. That is,

$$R_j = \{x_j | f(x_{-j}^{(t)}, x_j) < f(x_{-j}^{(t)}, x_j^*), x_j \in [a_j, b_j]\}.$$

Let $S_j$ be the set of values of the $j$-th decision variable that belong to tame solution vectors. That is,

$$S_j = \{x_j | (ax_j + (1-a)x_j^*) \in R_j, \forall a \in [0, 1], x_j \in [a_j, b_j]\}.$$

Let $x_j^l$ be the smallest $x_j$ value and $x_j^r$ be the largest $x_j$ value of the tame solutions, respectively. That is,

$$x_j^l = \inf S_j$$

and

$$x_j^r = \sup S_j.$$

Let $l_j$ be the length of the interval $S_j$. That is,

$$l_j = x_j^r - x_j^l.$$

By Lemma 1, $l_j$ is non-negative and is bounded below by

$$l_j \geq \frac{e}{L},$$

where $e$ is the error of $f(x_{-j}^{(t)}, x_j^*)$ with respect to $f(x_{-j}^{(t)}, x_j^r)$. That is,

$$e = f(x_{-j}^{(t)}, x_j^*) - f(x_{-j}^{(t)}, x_j^r).$$

Figure 2 shows two examples of multimodal functions with the corresponding set of tame solutions denoted by $S_j$.

**Definition 8** (Probability of Tame Convergence). The probability of tame convergence is defined as the probability of an algorithm with sampling policy $\pi$ to find a tame solution in its next iteration, given the current best solution vector. That is,

$$P\{x_j^{(t+1)} \in S_j | x_j^{(t)}, x_j^{(t+1)} \sim \pi_x^{(t)}\}.$$

**Lemma 2.** The probability of tame convergence of PROS is $\frac{l_j}{b_j-a_j}$. 
Figure 2. Left: an example function where $x^{(t)} < x^*_j$. Right: another example function where $x^{(t)} > x^*_j$. The set of tame solutions is denoted as $S_j$.

**Proof.** Let $P_U^j$ be the probability that a tame solution is found by sampling using uniform distribution on the $j$-th decision variable in the $(t + 1)$-th iteration, given the best solution vector in the $t$-th iteration. Then

$$P_U^j = P\{x_j^{(t+1)} \in S_j | x_j^{(t)}, x_{j}^{(t+1)} \sim U(a_j, b_j)\} = \frac{l_j}{b_j - a_j}.$$ 

□

**Theorem 4.** The probability of tame convergence of TROS is $l_j b_j - a_j (l_j + 2q_j)$ when $x_j^{(t)} < x^*_j$ and is $l_j b_j - a_j (l_j + 2q_j)$ when $x_j^{(t)} > x^*_j$ where $u_j = x_j^* - x_j^{(t)}$, $v_j = b_j - x^*_j$, $q_j = x_j^* - a_j$ and $r_j = x_j^{(t)} - x^*_j$.

**Proof.** Let $P_T^j$ be the probability that a tame solution is found by sampling using the triangular distribution on the $j$-th decision variable in the $(t + 1)$-th iteration, given the best solution vector in the $t$-th iteration. That is,

$$P_T^j = P\{x_j^{(t+1)} \in S_j | x_j^{(t)}, x_{j}^{(t+1)} \sim T(a_j, b_j, x_j^{(t)})\}.$$ 

For case 1, $x_j^{(t)} < x^*_j$:

$$P_T^j = \frac{l_j}{2} \left( \frac{2}{b_j - a_j} \frac{v_j}{u_j + l_j + v_j} + \frac{2}{b_j - a_j} \frac{l_j + v_j}{u_j + l_j + v_j} \right) = \frac{l_j}{b_j - a_j} \left( \frac{l_j + 2v_j}{u_j + l_j + v_j} \right).$$

Similarly, for case 2, $x_j^{(t)} > x^*_j$:

$$P_T^j = \frac{l_j}{2} \left( \frac{2}{b_j - a_j} \frac{q_j}{q_j + l_j + r_j} + \frac{2}{b_j - a_j} \frac{l_j + q_j}{q_j + l_j + r_j} \right) = \frac{l_j}{b_j - a_j} \left( \frac{l_j + 2q_j}{q_j + l_j + r_j} \right).$$

□

**Corollary 2.** The conditions that TROS has a higher probability of tame convergence than that of PROS, the same probability as that of PROS, and a lower probability than that of PROS are as follows, respectively.
For case 1, \( x_j^{(t)} < x_j^* \):
\[
P_j^T = \begin{cases} 
 p_j^{U}, v_j > u_j \\
 p_j^{U}, v_j = u_j \\
 p_j^{U}, v_j < u_j 
\end{cases}
\]

For case 2, \( x_j^{(t)} > x_j^* \):
\[
P_j^T = \begin{cases} 
 p_j^{U}, q_j > r_j \\
 p_j^{U}, q_j = r_j \\
 p_j^{U}, q_j < r_j 
\end{cases}
\]

**Corollary 3.** The probability of tame convergence of TROS is higher than or equal to that of PROS for all convex functions.

**Proof.** For convex functions, \( u_j = 0 \) (for case 1) and \( r_j = 0 \) (for case 2). An example convex function is shown in Figure 3.

For case 1, \( x_j^{(t)} < x_j^* \):
\[
P_j^T = \frac{l_j}{b_j - a_j} \left( \frac{l_j + 2v_j}{u_j + l_j + v_j} \right) 
= p_j^{U} \left( 1 + \frac{v_j}{l_j + v_j} \right) 
\therefore p_j^T \begin{cases} 
 > p_j^{U}, v_j > 0 \\
 = p_j^{U}, v_j = 0 
\end{cases}
\]

Similarly, for case 2, \( x_j^{(t)} > x_j^* \):
\[
P_j^T = \frac{l_j}{b_j - a_j} \left( \frac{l_j + 2r_j}{q_j + l_j + r_j} \right) 
= p_j^{U} \left( 1 + \frac{r_j}{q_j + r_j} \right) 
\therefore p_j^T \begin{cases} 
 > p_j^{U}, q_j > 0 \\
 = p_j^{U}, q_j = 0 
\end{cases}
\]

Therefore, for convex functions, \( p_j^T \) can never be less than \( p_j^{U} \). □

**Figure 3.** An example convex function. Left: for case 1, \( x_j^{(t)} < x_j^* \), \( u_j = 0 \). Right: case 2, \( x_j^{(t)} > x_j^* \), \( r_j = 0 \).
Theorem 5. The probability of tame convergence of QROS is
\[ P_j^Q \overset{\text{def}}{=} \frac{I_j}{b_j - a_j} \frac{(l_j + v_j)(l_j + 2v_j) + v_j^2}{(u_j + l_j + v_j)^2} \]
when \( x_j^{(t)} < x_j^* \) and is
\[ P_j^Q \overset{\text{def}}{=} \frac{I_j}{b_j - a_j} \frac{(q_j + l_j)(2q_j + l_j) + q_j^2}{(q_j + l_j + r_j)^2} \]
when \( x_j^{(t)} > x_j^* \) where \( u_j = x_j^t - x_j^{(t)} \), \( v_j = b_j - x_j^* \), \( q_j = x_j^t - a_j \) and \( r_j = x_j^{(t)} - x_j^t \).

Proof. Let \( P_j^Q \) be the probability that a tame solution is found by sampling using the quadratic distribution on the \( j \)-th decision variable in the \( (t+1) \)-th iteration, given the best solution vector in the \( t \)-th iteration. That is,
\[ P_j^Q = P\{x_j^{(t+1)} \in S_j | x_j^{(t)} \sim Q(a_j, b_j, x_j^{(t)}) \} \]  

For case 1, \( x_j^{(t)} < x_j^* \):
\[ P_j^Q = \int_{x_j^*}^{x_j^t} f_Q(x) \, dx \]
\[ = \int_{x_j^*}^{x_j^t} \frac{3(x - b_j)^2}{(b_j - a_j)(x_j^t - b_j)^2} \, dx \]
\[ = \frac{I_j}{b_j - a_j} \frac{(l_j + v_j)(l_j + 2v_j) + v_j^2}{(u_j + l_j + v_j)^2} \]  

Similarly, for case 2, \( x_j^{(t)} > x_j^* \):
\[ P_j^Q = \int_{x_j^*}^{x_j^t} f_Q(x) \, dx \]
\[ = \int_{x_j^*}^{x_j^t} \frac{3(x - a_j)^2}{(b_j - a_j)(x_j^t - a_j)^2} \, dx \]
\[ = \frac{I_j}{b_j - a_j} \frac{(q_j + l_j)(2q_j + l_j) + q_j^2}{(q_j + l_j + r_j)^2} \]  

\[ \square \]

Corollary 4. The probability of tame convergence of QROS is higher than or equal to that of TROS and PROS for all convex functions.

Proof. For convex functions, \( u_j = 0 \) (for case 1) and \( r_j = 0 \) (for case 2).

For case 1, \( x_j^{(t)} < x_j^* \):
\[ P_j^Q = \frac{I_j}{b_j - a_j} \frac{(l_j + v_j)(l_j + 2v_j) + v_j^2}{(u_j + l_j + v_j)^2} \]
\[ = P_j^U \left( 1 + \frac{v_j}{l_j + v_j} + \frac{v_j^2}{(l_j + v_j)^2} \right) \]
\[ = P_j^U + P_j^U \frac{v_j^2}{(l_j + v_j)^2} \]
\[ = P_j^U + P_j^U \frac{v_j^2}{(l_j + v_j)^2} \]
\[ \geq P_j^T > P_j^U, \quad v_j > 0 \]
\[ = P_j^T = P_j^U, \quad v_j = 0 \]
For case 2, \( x_j^{(t)} > x^* \):

\[
p_j^Q = \frac{1}{b_j - a_j} \frac{(q_j + I_j)(2q_j + l_j) + q_j^2}{(q_j + l_j + r_j)^2}
\]

\[
= p_j^U \left( 1 + \frac{q_j}{q_j + l_j} + \frac{q_j^2}{(q_j + l_j)^2} \right)
\]

\[
\therefore \begin{cases} 
  p_j^T > p_j^U, & q_j > 0 \\
  p_j^T = p_j^U, & q_j = 0
\end{cases}
\]

Therefore, for convex functions, \( p_j^Q \) can never be less than \( p_j^T \) and \( p_j^U \). □

The probability density function of distributions \( \mathcal{U} \), \( \mathcal{T} \) and \( \mathcal{Q} \) are characterized by a degree-0, degree-1 and degree-2 polynomials, respectively. From Corollary 4, the increment in the probability of \textit{tame} convergence diminishes as the degree of the polynomial increases. We expect sampling using an even higher degree distribution (e.g., cubic distribution with the mode narrower than \( \mathcal{Q} \) and \( \mathcal{T} \)) would give higher probability of \textit{tame} convergence, but the gain diminishes as the degree grows.

4. Experiments

In order to evaluate the merits of the two modified algorithms (TROS and QROS), a set of benchmark test problems is selected from the literature [15–17] and shown in Table 1. The benchmark problems include unimodal and highly multimodal functions, convex and non-convex functions, and separable and non-separable functions. TROS and QROS were compared with the original PROS algorithm and three well-known and widely used EC algorithms for optimization: genetic algorithm (GA) [18], particle swarm optimization (PSO) [19] and differential evolution (DE) [20]. PROS, TROS and QROS were implemented in Java by the authors. GA, PSO and DE were run using pymoo, an open source framework for multi-objective optimization in Python, version 0.6.0 [21]. The code for the benchmark problems running on pymoo were revised by the authors (for experiments of shifted search space). The hyper-parameters of GA, PSO and DE are selected using the default values from the pymoo package (they are the typical values suggested in the literature).

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
<th>Formulation and Global Optimum</th>
<th>Search Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>Sphere</td>
<td>( \sum_{i=1}^{D} x_i^2 ), ( x^* = (0,0,\ldots,0) ), ( f(x^*) = 0 )</td>
<td>([-10,10]^D)</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>Ellipsoid</td>
<td>( \sum_{i=1}^{D} i x_i^2 ), ( x^* = (0,0,\ldots,0) ), ( f(x^*) = 0 )</td>
<td>([-10,10]^D)</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>Schwefel 1.2</td>
<td>( \sum_{i=1}^{D} (\sum_{j=1}^{D} x_j)^2 ), ( x^* = (0,0,\ldots,0) ), ( f(x^*) = 0 )</td>
<td>([-5.12,5.12]^D)</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>Rosenbrock</td>
<td>( \sum_{i=1}^{D} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2] ), ( x^* = (1,1,\ldots,1) ), ( f(x^*) = 0 )</td>
<td>([-2.048,2.048]^D)</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>Zakharov</td>
<td>( \sum_{i=1}^{D} x_i^2 + (\frac{1}{4} \sum_{i=1}^{D} i x_i)^2 + (\frac{1}{4} \sum_{i=1}^{D} i x_i)^4 ), ( x^* = (0,0,\ldots,0) ), ( f(x^*) = 0 )</td>
<td>([-10,10]^D)</td>
</tr>
</tbody>
</table>
Windows 10 platform. Multiple runs were conducted for each problem by each algorithm with a different seed for the generation of random numbers for each run. For fair comparison, all algorithms end with the same maximum number of objective function evaluations. In order to increase the reliability of the statistical results, the number of runs was increased to 100, 100 and 30 for 5D, 10D, and 50D problems, respectively (instead of 10 for all dimensions in [15]). The settings are summarized in Table 2.

### Table 2. Settings of the numerical experiments.

<table>
<thead>
<tr>
<th>Settings</th>
<th>D = 5</th>
<th>D = 10</th>
<th>D = 50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>10D</td>
<td>100</td>
<td>500</td>
</tr>
<tr>
<td>Max. No. of Generations</td>
<td>(20D − 50)</td>
<td>50</td>
<td>150</td>
</tr>
<tr>
<td>Max. No. of Objective</td>
<td>10D(20D − 50)</td>
<td>2500</td>
<td>15,000</td>
</tr>
<tr>
<td>Function Evaluations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Runs</td>
<td>100</td>
<td>100</td>
<td>30</td>
</tr>
</tbody>
</table>

### 4.1. Experiment I: Basic Benchmark Problems

Figure 4 shows the experimental results of the mean error for 100 independent runs of each of the algorithms on each benchmark problem for dimension D = 5. The x-axes are the number of objective function evaluations (in log scale) and the y-axes are the average error of the algorithm after certain objective function evaluations. As shown in the figure, QROS converged faster than the other five algorithms except on f_3 and f_5 where PSO converged the fastest. TROS and PROS had similar trends as QROS on the benchmark functions: they converged faster than GA, PSO and DE on most benchmark functions.
except on \(f_3\) and \(f_5\). Figures 5 and 6 show the experimental results for dimension \(D = 10\) and \(D = 50\), respectively. Similar to the results for \(D = 5\), QROS converged faster than the other five algorithms except on \(f_3\) (for \(10D\)) and \(f_5\) (for \(10D\) and \(50D\)) where PSO converged the fastest.

![Figure 4. The convergence curve of 100 runs on 5D benchmark functions.](image)

![Figure 5. The convergence curve of 100 runs on 10D benchmark functions.](image)
Figure 6. The convergence curve of 30 runs on 50D benchmark functions.

Table 3 shows the mean of the errors of the final solution vectors returned by PROS, TROS and QROS for 100 independent runs of on each benchmark problem for dimension \( D = 5 \). The mean errors represent the medium to long term performance of the algorithms. The best results (the smallest mean final errors) for each benchmark problem are highlighted in bold font. The corresponding standard deviations are placed next with parenthesis. Owing to limited space, the results of GA, PSO and DE are not shown here, as they have been reported in [15] already. As seen from the table, QROS had the smallest mean final errors on most of the benchmark problems. Similar results can be seen from Tables 4 and 5 which show the means of the final errors for dimension \( D = 10 \) and \( D = 50 \), respectively.

In general, QROS and TROS are more efficient in reducing the mean final errors on most benchmark problems when compared with PROS.

<table>
<thead>
<tr>
<th>( f )</th>
<th>PROS ((1.06e-03 \pm 8.88e-04))</th>
<th>TROS ((1.21e-03 \pm 1.33e-03))</th>
<th>QROS ((4.32e-02 \pm 2.99e-02))</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>4.65e-03 (4.85e-03)</td>
<td>1.06e-03 (8.88e-04)</td>
<td>4.13e-04 (4.54e-04)</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>3.38e-02 (3.68e-03)</td>
<td>3.13e-02 (2.65e-03)</td>
<td>1.21e-03 (1.33e-03)</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>2.17e-01 (1.80e-01)</td>
<td>7.61e-02 (5.05e-02)</td>
<td>4.32e-02 (2.99e-02)</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>1.34e+00 (1.17e+00)</td>
<td>1.48e+00 (1.66e+00)</td>
<td>1.55e+00 (1.17e+00)</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>5.13e+00 (6.46e+00)</td>
<td>5.33e-01 (8.45e-01)</td>
<td>1.28e-01 (1.81e-01)</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>4.81e-03 (3.68e-03)</td>
<td>2.38e-03 (9.69e-04)</td>
<td>1.62e-03 (7.19e-04)</td>
</tr>
<tr>
<td>( f_7 )</td>
<td>2.41e-01 (2.49e-01)</td>
<td>5.30e-02 (4.49e-02)</td>
<td>2.91e-02 (3.01e-02)</td>
</tr>
<tr>
<td>( f_8 )</td>
<td>7.48e-01 (4.80e-01)</td>
<td>2.81e-01 (1.65e-01)</td>
<td>1.53e-01 (8.95e-02)</td>
</tr>
<tr>
<td>( f_9 )</td>
<td>2.69e-01 (1.41e-01)</td>
<td>1.33e-01 (6.76e-02)</td>
<td>8.98e-02 (3.69e-02)</td>
</tr>
<tr>
<td>( f_{10} )</td>
<td>4.01e-01 (1.40e-01)</td>
<td>3.34e-01 (1.24e-01)</td>
<td>2.83e-01 (1.26e-01)</td>
</tr>
<tr>
<td>( f_{11} )</td>
<td>4.67e-01 (1.34e-01)</td>
<td>4.05e-01 (1.20e-01)</td>
<td>3.72e-01 (9.48e-02)</td>
</tr>
<tr>
<td>( f_{12} )</td>
<td>3.49e-01 (1.04e-01)</td>
<td>2.22e-01 (6.44e-02)</td>
<td>1.77e-01 (4.67e-02)</td>
</tr>
</tbody>
</table>
Table 4. Statistical results of 100 runs on 10D benchmark functions.

<table>
<thead>
<tr>
<th>f</th>
<th>PROS</th>
<th>TROS</th>
<th>QROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>8.62e-04 (7.26e-04)</td>
<td>2.38e-04 (1.85e-04)</td>
<td>1.03e-04 (7.51e-05)</td>
</tr>
<tr>
<td>f2</td>
<td>4.45e-03 (3.33e-03)</td>
<td>1.34e-03 (1.23e-03)</td>
<td>5.68e-04 (4.63e-04)</td>
</tr>
<tr>
<td>f3</td>
<td>6.84e-01 (3.62e-01)</td>
<td>2.52e-01 (1.35e-01)</td>
<td>1.35e-01 (7.28e-02)</td>
</tr>
<tr>
<td>f4</td>
<td>3.40e+00 (3.28e+00)</td>
<td>3.72e+00 (2.99e+00)</td>
<td>4.53e+00 (2.91e+00)</td>
</tr>
<tr>
<td>f5</td>
<td>4.69e+01 (2.47e+01)</td>
<td>1.40e+01 (9.29e+00)</td>
<td>6.26e+00 (4.77e+00)</td>
</tr>
<tr>
<td>f6</td>
<td>3.10e-03 (9.52e-04)</td>
<td>1.70e-03 (5.30e-04)</td>
<td>1.10e-03 (3.52e-04)</td>
</tr>
<tr>
<td>f7</td>
<td>4.48e-02 (3.77e-02)</td>
<td>1.24e-02 (9.46e-03)</td>
<td>5.24e-02 (3.56e-03)</td>
</tr>
<tr>
<td>f8</td>
<td>1.61e-01 (8.23e-02)</td>
<td>7.13e-02 (3.00e-02)</td>
<td>4.32e-02 (1.55e-02)</td>
</tr>
<tr>
<td>f9</td>
<td>1.78e-01 (6.53e-02)</td>
<td>9.15e-02 (3.68e-02)</td>
<td>6.77e-02 (2.74e-02)</td>
</tr>
<tr>
<td>f10</td>
<td>4.73e-01 (2.46e-01)</td>
<td>4.35e-01 (1.97e-01)</td>
<td>3.76e-01 (1.17e-01)</td>
</tr>
<tr>
<td>f11</td>
<td>4.90e-01 (1.65e-01)</td>
<td>4.30e-01 (1.49e-01)</td>
<td>1.68e-01 (3.85e-02)</td>
</tr>
<tr>
<td>f12</td>
<td>3.40e-01 (8.16e-02)</td>
<td>2.20e-01 (4.97e-02)</td>
<td>1.68e-01 (3.85e-02)</td>
</tr>
</tbody>
</table>

Table 5. Statistical results of 30 runs on 50D benchmark functions.

<table>
<thead>
<tr>
<th>f</th>
<th>PROS</th>
<th>TROS</th>
<th>QROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>1.12e-04 (3.30e-05)</td>
<td>2.82e-05 (7.60e-06)</td>
<td>1.24e-05 (4.68e-06)</td>
</tr>
<tr>
<td>f2</td>
<td>2.86e-03 (9.68e-04)</td>
<td>7.25e-04 (2.22e-04)</td>
<td>3.29e-04 (1.32e-04)</td>
</tr>
<tr>
<td>f3</td>
<td>1.34e+01 (3.13e+00)</td>
<td>6.45e+00 (1.57e+00)</td>
<td>3.75e+00 (8.21e-01)</td>
</tr>
<tr>
<td>f4</td>
<td>8.06e+01 (3.80e+01)</td>
<td>6.05e+01 (3.41e+01)</td>
<td>5.48e+01 (2.46e+01)</td>
</tr>
<tr>
<td>f5</td>
<td>9.66e+02 (1.63e+02)</td>
<td>7.58e+02 (1.51e+02)</td>
<td>6.44e+02 (1.36e+02)</td>
</tr>
<tr>
<td>f6</td>
<td>2.46e-03 (3.48e-04)</td>
<td>1.32e-03 (1.93e-04)</td>
<td>8.85e-04 (1.26e-04)</td>
</tr>
<tr>
<td>f7</td>
<td>5.81e-03 (1.72e-03)</td>
<td>1.32e-03 (2.79e-04)</td>
<td>6.25e-04 (1.60e-04)</td>
</tr>
<tr>
<td>f8</td>
<td>2.06e-02 (3.28e-03)</td>
<td>9.90e-03 (1.13e-03)</td>
<td>6.49e-03 (9.52e-04)</td>
</tr>
<tr>
<td>f9</td>
<td>2.63e-02 (1.51e-02)</td>
<td>1.17e-02 (1.01e-02)</td>
<td>1.39e-02 (1.77e-02)</td>
</tr>
<tr>
<td>f10</td>
<td>5.99e-01 (2.48e-01)</td>
<td>5.80e-01 (2.25e-01)</td>
<td>5.66e-01 (2.27e-01)</td>
</tr>
<tr>
<td>f11</td>
<td>6.51e-01 (1.28e-01)</td>
<td>6.17e-01 (1.11e-01)</td>
<td>6.21e-01 (1.08e-01)</td>
</tr>
<tr>
<td>f12</td>
<td>5.34e-01 (4.80e-02)</td>
<td>3.43e-01 (2.93e-02)</td>
<td>2.61e-01 (2.84e-02)</td>
</tr>
</tbody>
</table>

4.2. Experiment II: Random Shifted Benchmark Problems

One may notice that both TROS and QROS are in favor of objective functions with the global optimum located at the center of the search space. For example, the global optimum of $f_1$, $f_2$, $f_3$, $f_5$, $f_6$, $f_7$, $f_9$, $f_{10}$, $f_{12}$ is $(0, 0, \ldots, 0)$. In order to test the effectiveness of TROS and QROS in a general case, another experiment was conducted. The same set of benchmark problems were used in this experiment, but the search space was shifted so that the global optimum may be located anywhere within the bounded search space. A random point $o = (o_1, o_2, \ldots, o_D)$ is drawn uniformly from $\Omega' = [a_1 - x^*_1, b_1 - x^*_1] \times [a_2 - x^*_2, b_2 - x^*_2] \times \cdots \times [a_D - x^*_D, b_D - x^*_D]$ for each benchmark function for each run where $a_i, b_i, x^*_i$ are the lower limit, upper limit and the optimum value of the $i$-th decision variable in the original search space of the benchmark function being optimized and $i \in \{1, 2, \ldots, D\}$. The shifted search space becomes $\Omega'' = [a_1 - o_1, b_1 - o_1] \times [a_2 - o_2, b_2 - o_2] \times \cdots \times [a_D - o_D, b_D - o_D]$. It has to be noted that the benchmark problems are carefully chosen in this experiment to
ensure for each benchmark problem, $x^*$ is still the global optimum in the shifted search space but $x^*$ is not necessarily be located at the center of the new search space.

Figures 7–9 show the experimental results of the mean error of each of the algorithms on each benchmark problem for dimension $D = 5$, $D = 10$ and $D = 50$, respectively. Similar to the non-shifted experiments, PROS, TROS and QROS converged faster than GA, PSO and DE on most benchmark problems except on $f_5$ when $D = 50$. When only considering the three random orthogonal search algorithms, PROS usually converged quickly initially, then TROS surpassed PROS, followed by QROS surpassed TROS on most of the benchmark problems.

Tables 6–8 show the statistical results of the final error of each of the algorithms on each benchmark problem for dimension $D = 5$, $D = 10$ and $D = 50$, respectively. Similar to the non-shifted experiments, both QROS and TROS often improve the mean final errors. The performance of QROS and TROS are quite promising on the shifted benchmark functions.

### Table 6. Statistical results of 100 runs on 5D benchmark functions with shifted global optimum.

<table>
<thead>
<tr>
<th>$f$</th>
<th>PROS</th>
<th>TROS</th>
<th>QROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>4.31e-03 (4.38e-03)</td>
<td>1.67e-03 (3.92e-03)</td>
<td>1.15e-03 (5.90e-03)</td>
</tr>
<tr>
<td>$f_2$</td>
<td>1.24e-02 (1.20e-02)</td>
<td>4.41e-03 (7.05e-03)</td>
<td>2.60e-03 (9.36e-03)</td>
</tr>
<tr>
<td>$f_3$</td>
<td>2.64e-01 (2.23e-01)</td>
<td>9.43e-02 (8.36e-02)</td>
<td>5.54e-02 (4.39e-02)</td>
</tr>
<tr>
<td>$f_4$</td>
<td>1.38e+00 (3.51e+00)</td>
<td>9.47e-01 (3.44e+00)</td>
<td>1.82e+00 (1.12e+01)</td>
</tr>
<tr>
<td>$f_5$</td>
<td>7.76e+00 (1.24e+01)</td>
<td>8.39e-01 (1.16e+00)</td>
<td>2.60e-01 (4.88e-01)</td>
</tr>
<tr>
<td>$f_6$</td>
<td>4.62e-03 (2.28e-03)</td>
<td>6.57e-03 (9.53e-03)</td>
<td>1.21e-02 (9.64e-02)</td>
</tr>
<tr>
<td>$f_7$</td>
<td>2.23e-01 (2.23e-01)</td>
<td>1.39e-01 (3.07e-01)</td>
<td>2.34e-01 (3.01e-01)</td>
</tr>
<tr>
<td>$f_8$</td>
<td>7.34e-01 (4.52e-01)</td>
<td>3.27e-01 (2.86e-01)</td>
<td>2.34e-01 (3.01e-01)</td>
</tr>
<tr>
<td>$f_9$</td>
<td>2.94e-01 (1.97e-01)</td>
<td>1.54e-01 (7.26e-02)</td>
<td>1.18e-01 (1.06e-01)</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>3.90e-01 (1.68e-01)</td>
<td>3.18e-01 (1.22e-01)</td>
<td>3.13e-01 (1.38e-01)</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>4.82e-01 (1.21e-01)</td>
<td>4.28e-01 (1.43e-01)</td>
<td>4.15e-01 (1.93e-01)</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>3.47e-01 (1.03e-01)</td>
<td>2.64e-01 (1.33e-01)</td>
<td>2.79e-01 (2.32e-01)</td>
</tr>
</tbody>
</table>

### Table 7. Statistical results of 100 runs on 10D benchmark functions with shifted global optimum.

<table>
<thead>
<tr>
<th>$f$</th>
<th>PROS</th>
<th>TROS</th>
<th>QROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>8.71e-04 (5.89e-04)</td>
<td>2.55e-04 (1.99e-04)</td>
<td>1.64e-04 (2.97e-04)</td>
</tr>
<tr>
<td>$f_2$</td>
<td>4.64e-03 (3.58e-03)</td>
<td>1.33e-03 (1.18e-03)</td>
<td>9.91e-04 (2.60e-03)</td>
</tr>
<tr>
<td>$f_3$</td>
<td>7.99e-01 (4.64e-01)</td>
<td>2.78e-01 (1.76e-01)</td>
<td>1.62e-01 (9.67e-02)</td>
</tr>
<tr>
<td>$f_4$</td>
<td>1.35e+00 (6.26e+00)</td>
<td>8.60e-01 (2.81e+00)</td>
<td>8.20e-01 (2.68e+00)</td>
</tr>
<tr>
<td>$f_5$</td>
<td>6.23e+01 (4.86e+01)</td>
<td>1.80e+01 (1.70e+01)</td>
<td>7.34e+00 (7.04e+00)</td>
</tr>
<tr>
<td>$f_6$</td>
<td>3.12e-03 (9.80e-04)</td>
<td>2.38e-03 (2.74e-03)</td>
<td>6.05e-03 (9.72e-03)</td>
</tr>
<tr>
<td>$f_7$</td>
<td>4.53e-02 (3.06e-02)</td>
<td>6.38e-02 (2.58e-01)</td>
<td>2.65e-01 (5.19e-01)</td>
</tr>
<tr>
<td>$f_8$</td>
<td>1.64e-01 (7.02e-02)</td>
<td>8.10e-02 (3.39e-02)</td>
<td>6.33e-02 (1.06e-01)</td>
</tr>
<tr>
<td>$f_9$</td>
<td>1.85e-01 (6.35e-02)</td>
<td>9.56e-02 (3.76e-02)</td>
<td>7.97e-02 (3.49e-02)</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>3.84e-01 (2.06e-01)</td>
<td>3.61e-01 (1.80e-01)</td>
<td>3.37e-01 (1.96e-01)</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>4.50e-01 (1.46e-01)</td>
<td>4.51e-01 (2.05e-01)</td>
<td>4.45e-01 (1.92e-01)</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>3.31e-01 (7.39e-02)</td>
<td>2.52e-01 (9.44e-02)</td>
<td>3.03e-01 (2.26e-01)</td>
</tr>
</tbody>
</table>
Table 8. Statistical results of 30 runs on 50D benchmark functions with shifted global optimum.

<table>
<thead>
<tr>
<th>f</th>
<th>PROS</th>
<th>TROS</th>
<th>QROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>1.13e-04 (2.63e-05)</td>
<td>2.88e-05 (1.00e-05)</td>
<td>1.30e-05 (4.43e-06)</td>
</tr>
<tr>
<td>$f_2$</td>
<td>2.90e-03 (7.79e-04)</td>
<td>7.43e-04 (3.28e-04)</td>
<td>3.31e-04 (1.23e-04)</td>
</tr>
<tr>
<td>$f_3$</td>
<td>1.78e+01 (4.20e+00)</td>
<td>8.55e+00 (1.75e+00)</td>
<td>5.46e+00 (1.20e+00)</td>
</tr>
<tr>
<td>$f_4$</td>
<td>2.02e+01 (3.98e+01)</td>
<td>2.59e+00 (8.90e+00)</td>
<td>3.84e+00 (1.35e+01)</td>
</tr>
<tr>
<td>$f_5$</td>
<td>1.93e+03 (4.23e+02)</td>
<td>1.55e+03 (3.24e+02)</td>
<td>1.30e+03 (2.85e+02)</td>
</tr>
<tr>
<td>$f_6$</td>
<td>2.62e-03 (3.48e-04)</td>
<td>1.69e-03 (5.51e-04)</td>
<td>3.19e-03 (2.83e-03)</td>
</tr>
<tr>
<td>$f_7$</td>
<td>5.89e-03 (1.37e-03)</td>
<td>1.58e-03 (4.07e-04)</td>
<td>2.99e-01 (5.23e-01)</td>
</tr>
<tr>
<td>$f_8$</td>
<td>2.09e-02 (2.66e-03)</td>
<td>9.85e-03 (1.31e-03)</td>
<td>6.68e-03 (1.06e-03)</td>
</tr>
<tr>
<td>$f_9$</td>
<td>2.77e-02 (1.27e-02)</td>
<td>1.92e-02 (1.70e-02)</td>
<td>5.13e-02 (4.70e-02)</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>3.68e-01 (1.11e-01)</td>
<td>3.70e-01 (1.55e-01)</td>
<td>3.32e-01 (9.46e-02)</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>5.82e-01 (1.11e-01)</td>
<td>5.74e-01 (9.44e-02)</td>
<td>6.06e-01 (1.34e-01)</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>5.51e-01 (4.15e-02)</td>
<td>3.93e-01 (1.21e-01)</td>
<td>5.14e-01 (2.79e-01)</td>
</tr>
</tbody>
</table>

Figure 7. The convergence curve of 100 runs on 5D benchmark functions with shifted global optimum.
Figure 8. The convergence curve of 100 runs on 10D benchmark functions with shifted global optimum.

Figure 9. The convergence curve of 30 runs on 50D benchmark functions with shifted global optimum.

5. Future Work

In our future work, we plan to extend the algorithms in two directions. The first one is to allow switching between various sampling policies (e.g., uniform, triangular, quadratic)
in an adaptive way. This is particularly useful for black-box optimization where the form of the function is unknown to the optimization algorithm. One could learn the best sampling policy for that particular function and adapt gradually from the results of previous function evaluations. The second direction is to gradually rotate the objective function based on the current best solution set [22,23]. By doing so, a complex function would be transformed to a simpler convex function in the small area of concern. Any simple techniques such as TROS and QROS would quickly converge to the global optimum if it is in the area of concern. We are interested in investigating the convergence of TROS and QROS with the addition of transformation techniques.

6. Conclusions

In this paper, we perform an analysis of PROS. PROS is a parameterless EA that outperforms several well-known optimization algorithms on benchmark functions when the search budget is limited. It is attractive for expensive optimizing problems. However, to the best of our knowledge, there is no mathematical analysis of the performance and behavior of PROS reported in the literature. In this paper, we fill this gap.

Moreover, we propose TROS and QROS, which preserve the advantages of PROS and outperform it. We perform an analysis of tame convergence for PROS, TROS and QROS. We conduct two sets of experiments to evaluate the performance of TROS and QROS. We increase the number of runs compared to those conducted in [15] in order to increase the reliability of the statistical results. Experimental results illustrate that TROS and QROS outperform PROS to a large extent. Both TROS and QROS converge quickly on the benchmark problems. Although GA, PSO and DE usually have better performance in terms of smaller error in the final solution when a considerable amount of objective function evaluation is given, the performances of QROS and TROS are not much inferior to that of GA, PSO and DE. In fact, QROS and TROS are even sensible choices when we consider the computational complexity.

Author Contributions: Conceptualization, B.K.-B.T., C.W.S. and W.S.W.; methodology, B.K.-B.T. and W.S.W.; software, B.K.-B.T.; validation, B.K.-B.T.; formal analysis, B.K.-B.T., C.W.S. and W.S.W.; investigation, B.K.-B.T.; resources, B.K.-B.T.; data curation, B.K.-B.T.; writing—original draft preparation, B.K.-B.T.; writing—review and editing, B.K.-B.T., C.W.S. and W.S.W.; visualization, B.K.-B.T.; supervision, W.S.W.; project administration, B.K.-B.T.; funding acquisition, B.K.-B.T. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was partially funded by Hong Kong Metropolitan University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available from the authors upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References


**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.