J Glob Optim (2006) 36:609–626
DOI 10.1007/s10898-006-9030-3

ORIGINAL PAPER

A modification to the new version of the price's algorithm for continuous global optimization problems

Yong-Chang Jiao · Chuangyin Dang · Yee Leung · Yue Hao

Received: 12 December 2003 / Accepted: 1 March 2006 /
Published online: 14 June 2006
© Springer Science+Business Media B.V. 2006

Abstract This paper presents an algorithm for finding a global minimum of a multimodal, multivariate and nondifferentiable function. The algorithm is a modification to the new version of the Price's algorithm given in Brachetti et al. [J. Global Optim. **10**, 165–184 (1997)]. Its distinguishing features include: (1) The number-theoretic method is applied to generate the initial population so that the points in the initial population are uniformly scattered, and therefore the algorithm could explore uniformly the region of interest at the initial iteration; (2) The simplified quadratic approximation with the three best points is employed to improve the local search ability and the accuracy of the minimum function value, and to reduce greatly the computational overhead of the algorithm. Two sets of experiments are carried out to illustrate the efficiency of the number-theoretic method and the simplified quadratic model separately. The proposed algorithm has also been compared with the original one by solving a wide set of benchmark problems. Numerical results show that the proposed algorithm requires a smaller number of function evaluations and, in many cases, yields a smaller or more accurate minimum function value. The algorithm can also be used to deal with the medium size optimization problems.

Y. -C. Jiao
Institute of Antennas and EM Scattering, Xidian University,
Xi'an, Shaanxi 710071, China
e-mail: ychjiao@xidian.edu.cn

C. Dang (✉)
Department of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
e-mail: mecdang@cityu.edu.hk

Y. Leung
Department of Geography and Resource Management, The Chinese University of Hong Kong, New Territories, Shatin, Hong Kong

Y. Hao
Microelectronics Institute, Xidian University, Xi'an, Shaanxi 710071, China

Keywords Price's algorithm · Global optimization · Continuous spaces · Number-theoretic method

1 Introduction

Many practical engineering applications can be formulated as a global optimization problem, in which the objective function is not convex and possesses many local minima in the region of interest. In this paper, we consider the problem of finding a global minimum of the unconstrained optimization problem (P):

$$\begin{aligned} &\text{minimize } f(x), \\ &\text{subject to } x \in \mathcal{D}, \end{aligned}$$

where $f\colon \mathcal{R}^n \to \mathcal{R}$ and $\mathcal{D}$ is a compact set which contains in its interior a global minimum point x^* of $f(x)$. Usually, the search domain $\mathcal{D}$ is a hypercube. We will concern ourselves with a particularly difficult global optimization problem in which the evaluation of the objective function is very expensive, and the derivatives of the objective function are not available. This class of global optimization problems is very important in engineering applications.

In recent years, many algorithms have been proposed to solve the unconstrained global optimization problem (P) (see, e.g. [1]). Törn et al. [2] discussed the features of a global optimization problem and their contributions to the problem complexity. They also recognized different techniques that are applied in global optimization. Ali et al. [3] proposed a new controlled random search algorithm in which the simplex search originally proposed by Price [7] is replaced by the quadratic search and β-distribution sampling. In their algorithm, a three-point quadratic approximation is used to conduct global searches. Ali and Storey [4] proposed an aspiration based simulation algorithm. The advantage of their method is that it can memorize the best solution during a run. Storm and Price [5] proposed a new heuristic approach, called differential evolution, for global optimization over continuous spaces. Brachetti et al. [6] presented a new version of the well known Price's algorithm [7], which tries to make a better use of the values of the objective function already evaluated than in the basic Price's algorithm [7]. Three simple heuristic tools [6], the weighted centroid, the weighted reflection and the quadratic model of the objective function, are employed to improve the efficiency of Price's algorithm. Experimental results show that the algorithm performs better than the basic Price's algorithm and that the application of a weighted centroid and of a weighted reflection is effective in reducing the number of function evaluations. The use of a quadratic model of the objective function mainly improves the accuracy of the estimated minimum function value. However, to build quadratic approximations of the objective function, simultaneous linear equations with $2n + 1$ unknowns must be solved many times, and in many cases the objective function has no improvement. For relatively large n, this will increase the computational burden and hence is time-consuming.

In this paper, we are interested in the new version of the Price's algorithm proposed in [6] since it has been developed to tackle a class of particularly difficult global optimization problems. The number-theoretic method and the quadratic approximation with the three best points will be adopted in the new version of the Price's algorithm to improve its search ability and efficiency and to make it adapt to the solution of medium size problems.

Number-theoretic (NT) method or quasi-Monte Carlo method is a special method which represents a combination of number theory and numerical analysis. Like many mixed breeds, it has fascinations and attractions. The NT methods have been successfully applied in multidimensional numerical integration (quadrature), interpolation and numerical solutions of integral equations as well as differential equations [8, 10], and in various fields of statistics [9]. These applications illustrate that the NT method is a power tool. The essence of the NT method is to find a set of points that are uniformly scattered over an n-dimensional unit cube. Note that a "uniformly scattered" set of points as stated here means roughly that the set has a small discrepancy (defined in Sect. 2), not a set of points which are uniformly distributed in the usual statistical sense. Sometimes the NT set can be used instead of random numbers in the Monte Carlo method.

The population-based algorithm for the continuous optimization problem requires that the points in the initial population set should cover the entire region of interest. If the number of points in the initial population set is relatively small, the set produced by the Monte Carlo method is not distributed very uniformly, which will affect the efficiency of the algorithm. Instead of Monte Carlo method, the NT method will be used in the algorithm to generate its initial population.

In order to solve the problem (P) efficiently, we modify the new version of the Price's algorithm proposed in [6], which is a population-based algorithm with global and local search parts. The global search part consists of the weighted centroid and the weighted reflection. Our modification is twofold: (1) The NT method is applied to generate the initial population. We first produce by the NT method the uniformly scattered points in an n-dimensional unit cube, and then transform them into the points in the hypercube $\mathcal{D}$, which are chosen as the initial population. (2) A simplified quadratic approximation using the three best points is adopted, instead of the quadratic model of the objective function in [6]. When a point found by the global search part has the function value less than or equal to the third best within the current population, the quadratic approximation using the three best points is used. Two set of experiments are carried out to illustrate the efficiency of the NT method and the simplified quadratic model separately. The proposed algorithm has also been experimented on a large set of benchmark problems, taken from [6]. For all the benchmark problems, a comparison has been made with the original algorithm. Simulation results indicate that the proposed algorithm requires a smaller number of function evaluations and, in many cases, yields a smaller value of the objective function. The algorithm can also be used to solve medium size optimization problems. Therefore the proposed algorithm is efficient.

In Sect. 2, we describe the NT method for generating the initial population. The proposed algorithm is then presented in Sect. 3, and the numerical experiment results and the comparison with the original algorithm are given in Sect. 4. We finally conclude our paper in Sect. 5.

2 Number-theoretic method for generating the initial population

This section introduces the so-called NT method that can be applied to generate the initial population for population-based algorithms. The essence of the NT method is to find a set of points which are uniformly scattered in an n-dimensional unit cube $C^n = [0, 1]^n$. What is the meaning of "uniformly scattered" in C^n? First, we describe the concept of discrepancy to measure the uniformity of a set of points in C^n.

Definition [9] Let $\mathcal{S} = \{x_k, k = 1, 2, \ldots, m\}$ be a set of points in C^n. For any $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_n)^\top \in C^n$, let $N(\Gamma, \mathcal{S})$ be the number of points in $\mathcal{S}$ satisfying $x_k \leq \Gamma$. Then

$$D(m, \mathcal{S}) = \sup_{\Gamma \in C^n} \left| \frac{N(\Gamma, \mathcal{S})}{m} - v([0, \gamma_1] \times \cdots \times [0, \gamma_n]) \right| \tag{1}$$

is called the discrepancy of $\mathcal{S}$, where $v([0, \gamma_1] \times \cdots \times [0, \gamma_n]) = \gamma_1 \cdots \gamma_n$ denotes the volume of the rectangle $[0, \gamma_1] \times \cdots \times [0, \gamma_n]$.

If the set $\mathcal{S}$ with m points in C^n has the lowest discrepancy $D(m, \mathcal{S})$ among all the sets of m points in C^n, then the points in the set $\mathcal{S}$ are uniformly scattered in C^n. That is why the discrepancy is used to measure the uniformity for a set of points.

Now the question is how to determine the set of points with the lowest discrepancy. The problem is solved for the case of $n = 1$ [9]. Let m be an integer such that $m \geq 1$, and let

$$\mathcal{Q} = \left\{ \frac{2i - 1}{2m}, \quad i = 1, 2, \ldots, m \right\}. \tag{2}$$

Then the set $\mathcal{Q}$ has the lowest discrepancy $1/(2m)$ among all the sets of m points in $C^1 = [0, 1]$, as shown in [9].

It is very difficult to find a set with the smallest discrepancy for the case $n \geq 2$, because the distributions of m points in C^n may be very complicated. In what follows, we will introduce several useful methods for generating a set of points in C^n with low discrepancies. This set is also called an NT-net on C^n. For the details of the related theoretical results, refer to [8–10].

(1). *A glp set* The set produced by a so-called good lattice point modulo m is called a *glp* set, which is often used in practice and is convenient for computation.
Let $(m; h_1, \ldots, h_n)$ be a vector with integral components satisfying $1 \leq h_i < m$, $h_i \neq h_j$ $(i \neq j)$, $n < m$ and the greatest common divisors $(m, h_i) = 1$, $i = 1, \ldots, n$. Let

$$\begin{aligned} q_{ki} &\equiv k h_i (\text{mod } m), \\ x_{ki} &= (2q_{ki} - 1)/(2m), \end{aligned} \qquad k = 1, \ldots, m, \quad i = 1, \ldots, n. \tag{3}$$

The set $\mathcal{S}_m = \{x_k = (x_{k1}, \ldots, x_{kn})^\top, k = 1, \ldots, m\}$ is called a lattice point set of the generating vector $(m; h_1, \ldots, h_n)$. If the set $\mathcal{S}_m$ has the smallest discrepancy among all possible generating vectors, then the set $\mathcal{S}_m$ is called a *glp* set. It can be seen that x_{ki} defined in (3) can be easily calculated by

$$x_{ki} = \left\{ \frac{2k h_i - 1}{2m} \right\}, \tag{4}$$

where $\{x\}$ stands for the fractional part of x. Some *glp* sets have already been derived in the references (see, e.g. [8]). When $2 \leq n \leq 18$, a few generating vectors $(m; h_1, \ldots, h_n)$ with $h_1 = 1$ for some m's can be found in Appendix A of [9].

(2). *A gp set* For any given $\Omega = (\mu_1, \ldots, \mu_n)^\top \in R^n$, let L be a set formed by the first m elements of the set

$$\{(\{\mu_1 k\}, \ldots, \{\mu_n k\})^\top, \ k = 1, 2, \ldots\}. \tag{5}$$

If L has a low discrepancy, then the set (5) is called a *gp* set and Ω a good point.

The *gp* set can be easily obtained if we have a good point. The following three good points are recommended in practice:

1. The square root sequence: We take
$$\Omega = \left(\sqrt{p_1}, \ldots, \sqrt{p_n}\right)^\top, \tag{6}$$
where p_j's are different prime numbers, e.g. the first n prime numbers.
2. Let p be a prime number and $q = p^{1/(n+1)}$. Take
$$\Omega = \left(q, q^2, \ldots, q^n\right)^\top. \tag{7}$$
3. The cyclotomic field method: We take
$$\Omega = \left(\left\{2\cos\frac{2\pi}{p}\right\}, \left\{2\cos\frac{4\pi}{p}\right\}, \ldots, \left\{2\cos\frac{2\pi n}{p}\right\}\right)^\top, \tag{8}$$
where p is a prime number such that $p \geq 2n + 3$.

(3). *An H-set* H-set is based on the p-adic representation of natural numbers. Let l be a natural number such that $l \geq 2$. Then any natural number k has a unique l-digits representation
$$k = b_0 + b_1 l + b_2 l^2 + \cdots + b_r l^r, \quad 0 \leq b_i \leq l - 1, \ \ i = 0, 1, \ldots, r, \tag{9}$$
where $l^r \leq k < l^{r+1}$. For any $c \in (0, 1)$, c has a unique l-digits representation
$$c = c_0 l^{-1} + c_1 l^{-2} + \cdots, \quad 0 \leq c_i \leq l - 1, \quad i = 0, 1, 2, \ldots$$
We write $k = b_r b_{r-1} \cdots b_1 b_0$ and $c = 0.c_0 c_1 \cdots$ A one-to-one correspondence between positive integers and rational numbers in $(0, 1)$ can be established as follows: For any integer $k \geq 1$ with the representation (9), let
$$y_l(k) = b_0 l^{-1} + b_1 l^{-2} + \cdots + b_r l^{-r-1}. \tag{10}$$
Then $y_l(k) \in (0, 1)$. Let p_i $(1 \leq i \leq n)$ be n distinct prime numbers. Then
$$x_k = \left(y_{p_1}(k), \ldots, y_{p_n}(k)\right)^\top, \quad k = 1, 2, \ldots \tag{11}$$
is called an H-set.

The aforementioned three main kinds of sets have wide applications in practice [8–10]. The *glp* set is a finite set while the *gp* set and the H-set have infinite number of elements. Each set has its advantages and also shortcomings [9]. For the *glp* set, a different generating vector $(h_2, \ldots, h_n)$ will be chosen for each value of m, only a few generating vectors for some values of m are provided in Appendix A of [9], and there are no generating vectors for $n > 18$. The *gp* set is convenient to use. Suppose one first generates an NT-net of m_1 points and then finds that in fact m_2 $(> m_1)$ points are needed. We can use the *gp* set or H-set method to generate only an additional set of $m_2 - m_1$ points. The H-set method has the heaviest computational burden, and it is suitable only for small n.

Niederreiter and Peart [11] proposed the quasi-random search method for global optimization problems. Here the NT method is used to generate the initial population for population-based algorithms. First, we produce the NT-net points in C^n, and then transform them to the hypercube $\mathcal{D}$. In view of its powerful way of determining a set of points that are uniformly scattered in an n-dimensional unit cube, the NT method can

be regarded as an efficient method for producing the initial population in a population-based algorithm. As a matter of fact, the NT method makes the algorithm explore the search space uniformly, enhance the diversity of the population, and reduce the chance of being trapped in local minima at the initial iteration.

3 A modified algorithm

In this section, we put forward a modification to the new version of the Price's algorithm [6] for solving the problem (P). As shown in [6], the original algorithm has global and local search parts. The global search part consists of two heuristic tools: the weighted centroid and the weighted reflection, and the quadratic model of the objective function is used in the local search part. The use of a weighted centroid and of a weighted reflection is effective in reducing the number of function evaluations, while the use of a quadratic model of the objective function is effective mainly in improving the accuracy of the estimated minimum objective function value. However, the solution of simultaneous linear equations with $2n+1$ unknowns for building the quadratic model of the objective function greatly increases the computational burden and hence is time-consuming. In many cases the objective function has no improvement. The three-point quadratic approximation is used in [3] to replace the simplex search originally proposed by Price [7], and to conduct global searches. Here, the simplified quadratic approximation with the three best points in the current population is used to reduce the computational burden and to improve the local search ability as well as the solution accuracy of the algorithm.

Our modification to the new version of the Price's algorithm is twofold: (1) The NT method is applied to generate the initial population. We first produce the uniformly scattered points in an n-dimensional unit cube by using the NT method such as the square root sequence in the gp set, and then transform them into the points in the hypercube $\mathcal{D}$, which are chosen as the initial population. (2) The simplified quadratic approximation using the three best points in the current population is adopted, instead of the quadratic model of the objective function in [6]. When a point found by the global search part has the function value less than or equal to the third best within the current population, the quadratic approximation using the three best points is used.

We now present the proposed algorithm for solving the problem (P) as follows.

Data. Choose a positive integer m such that $m \geq \max(n+1, 3)$, a suitable small positive value ϵ, and a positive constant ω.

Step 0 Let $k = 0$, apply the NT method to generate the initial population set

$$\mathcal{S}^k = \{x_1^k, \ldots, x_m^k\},$$

where $x_i^k \in \mathcal{D}$, $i = 1, \ldots, m$, and compute $f(x_i^k)$, $i = 1, \ldots, m$.

Step 1 Determine two points $x_{\max}^k$ and $x_{\min}^k$ and their objective function values $f_{\max}^k$ and $f_{\min}^k$ such that

$$f_{\max}^k = f(x_{\max}^k) = \max_{x \in \mathcal{S}^k} f(x)$$

and

$$f_{\min}^k = f(x_{\min}^k) = \min_{x \in \mathcal{S}^k} f(x).$$

If the stopping criterion $f^k_{\max} - f^k_{\min} < \epsilon$ is satisfied, the algorithm terminates; otherwise determine the third best point $x^k_{\min 3}$ in $\mathcal{S}^k$ and its objective function value $f^k_{\min 3} = f(x^k_{\min 3})$.

Step 2 Choose at random $n+1$ points $x^k_{i_0}, x^k_{i_1}, \ldots, x^k_{i_n}$ in $\mathcal{S}^k$. Determine the weighted centroid c^k_w of the n points $x^k_{i_1}, \ldots, x^k_{i_n}$:

$$c^k_w = \sum_{j=1}^{n} w^k_j x^k_{i_j},$$

where

$$w^k_j = \frac{\eta^k_j}{\sum_{j=1}^{n} \eta^k_j}$$

with

$$\eta^k_j = \frac{1}{f(x^k_{i_j}) - f^k_{\min} + \phi^k}$$

and

$$\phi^k = \omega \frac{(f^k_{\max} - f^k_{\min})^2}{f^0_{\max} - f^0_{\min}}.$$

Step 3 Determine the trial point $\tilde{x}^k$ by performing a weighted reflection: let

$$f^k_w = \sum_{j=1}^{n} w^k_j f(x^k_{i_j})$$

and

$$\tilde{x}^k = \begin{cases} c^k_w - \alpha^k (x^k_{i_0} - c^k_w), & \text{if } f^k_w \le f(x^k_{i_0}), \\ x^k_{i_0} - \alpha^k (c^k_w - x^k_{i_0}), & \text{if } f^k_w > f(x^k_{i_0}), \end{cases}$$

where

$$\alpha^k = \begin{cases} 1 - \frac{f(x^k_{i_0}) - f^k_w}{f^k_{\max} - f^k_{\min} + \psi^k}, & \text{if } f^k_w \le f(x^k_{i_0}), \\ 1 - \frac{f^k_w - f(x^k_{i_0})}{f^k_{\max} - f^k_{\min} + \psi^k}, & \text{if } f^k_w > f(x^k_{i_0}) \end{cases}$$

with

$$\psi^k = \omega \frac{(f^k_{\max} - f^k_{\min})^2}{f^0_{\max} - f^0_{\min}}.$$

If $\tilde{x}^k \notin \mathcal{D}$, go to step 2; otherwise compute $f(\tilde{x}^k)$.

Step 4 If $f(\tilde{x}^k) \ge f^k_{\max}$, let

$$\mathcal{S}^{k+1} = \mathcal{S}^k$$

and $k = k+1$, and go to step 2.

Step 5 If $f^k_{\min 3} < f(\tilde{x}^k) < f^k_{\max}$, let

$$\mathcal{S}^{k+1} = \mathcal{S}^k \cup \{\tilde{x}^k\} - \{x^k_{\max}\}$$

and $k = k + 1$, and go to step 1.

Step 6 If $f(\tilde{x}^k) \le f^k_{\min 3}$, let

$$\tilde{\mathcal{S}} = \mathcal{S}^k \cup \{\tilde{x}^k\} - \{x^k_{\max}\}$$

and select three best points $\hat{x}_{l_1}$, $\hat{x}_{l_2}$ and $\hat{x}_{l_3}$ in $\tilde{\mathcal{S}}$ corresponding to the smallest objective function values of f. Let

$$\hat{x}_{l_1} = (\hat{x}_{l_1 1}, \hat{x}_{l_1 2}, \ldots, \hat{x}_{l_1 n})^\top, \quad \hat{x}_{l_2} = (\hat{x}_{l_2 1}, \hat{x}_{l_2 2}, \ldots, \hat{x}_{l_2 n})^\top$$

and

$$\hat{x}_{l_3} = (\hat{x}_{l_3 1}, \hat{x}_{l_3 2}, \ldots, \hat{x}_{l_3 n})^\top$$

and determine

$$\hat{f}^k_{\max} = f(\hat{x}_{\max}) = \max_{\hat{x} \in \tilde{\mathcal{S}}} f(\hat{x}).$$

Step 7 Compute the trial vector $\hat{x}^k_q = (\hat{x}^k_{q1}, \hat{x}^k_{q2}, \ldots, \hat{x}^k_{qn})^\top$ by

$$\hat{x}^k_{qi} = \frac{1}{2}\left[\frac{(\hat{x}^2_{l_2 i} - \hat{x}^2_{l_3 i}) f(\hat{x}_{l_1}) + (\hat{x}^2_{l_3 i} - \hat{x}^2_{l_1 i}) f(\hat{x}_{l_2}) + (\hat{x}^2_{l_1 i} - \hat{x}^2_{l_2 i}) f(\hat{x}_{l_3})}{(\hat{x}_{l_2 i} - \hat{x}_{l_3 i}) f(\hat{x}_{l_1}) + (\hat{x}_{l_3 i} - \hat{x}_{l_1 i}) f(\hat{x}_{l_2}) + (\hat{x}_{l_1 i} - \hat{x}_{l_2 i}) f(\hat{x}_{l_3})}\right], \quad i = 1, 2, \ldots, n.$$

If $\hat{x}^k_q \notin \mathcal{D}$, let

$$\mathcal{S}^{k+1} = \tilde{\mathcal{S}}$$

and $k = k + 1$, and go to step 1; otherwise compute $f(\hat{x}^k_q)$. If $f(\hat{x}^k_q) \ge \hat{f}^k_{\max}$, let

$$\mathcal{S}^{k+1} = \tilde{\mathcal{S}};$$

otherwise, let

$$\mathcal{S}^{k+1} = \tilde{\mathcal{S}} \cup \{\hat{x}^k_q\} - \{\hat{x}_{\max}\}.$$

Let $k = k + 1$ and go to step 1.

A disturbing fact concerning the proposed algorithm is its totally heuristic nature with no theoretical convergence properties. Here, we give some remarks on the convergence of the proposed algorithm. As shown in [6], the algorithm could be easily modified in order to prove theoretically that the algorithm produces a sequence of solutions globally convergent in probability towards a global minimum point. In this case, we need to choose at random a vector in $\mathcal{D}$ and to compare the vector to the worst point $x^k_{\max}$ by using the greedy criterion, as discussed in [6].

Since the NT method is applied to generate the initial population, the value m is smaller than $25n$, which is used by the original algorithm [6]. We suggest to choose m as follows: $m = 12$, when $n = 1$; $m = 20n$, when $2 \le n \le 7$; and $m = 19n$, when $n \ge 8$. However, this is a heuristic choice for m. For relatively larger n, smaller m may be chosen.

4 Numerical results and comparisons

The proposed algorithm is experimented on a large set of benchmark problems given in [6], on which the new version of the Price's algorithm [6] has been tested. In all the cases, $\mathcal{D}$ is a hypercube. All the test functions and the corresponding hypercubes are described in the Appendix.

In the proposed algorithm, we choose $\epsilon = 10^{-6}$. In some cases, smaller ϵ will be chosen in order to yield more accurate minimum function values. Based on extensive simulation results, we suggest to choose $\omega = 1.1$, which is smaller than $\omega = 10^3$ chosen in the original algorithm [6]. However, this is a heuristic choice for ω. Simulation results show that this choice makes the proposed algorithm perform better.

Since the simplified quadratic approximation with the three best points is adopted in the proposed algorithm instead of the quadratic model of the objective function used in the original algorithm, the proposed algorithm avoids the solution of simultaneous linear equations with $2n + 1$ unknowns, which can reduce greatly the computational overhead of the proposed algorithm. Although the CPU time taken by the proposed algorithm may be much less than that by the original algorithm, we use the number of function evaluations as our measure of efficiency, because it is machine-independent.

Although the NT method is a deterministic method for generating the initial population, globally the proposed algorithm includes random selections in Step 2. For each of the benchmark problems, 15 independent runs are performed, and the numerical results obtained from the average of these 15 executions are presented in following tables. In these tables, n_t is the average total number of function evaluations; n_{f1} is the average number of function evaluation at the trial $\tilde{x}^k$ that have not given a value $f(\tilde{x}^k)$ smaller than $f^k_{\max}$; n_q is the average number of function evaluations totally computed at the trial $\hat{x}^k_q$; n_{f2} is the average number of function evaluations at the trial $\hat{x}^k_q$ that have not given a value $f(\hat{x}^k_q)$ smaller than $\hat{f}^k_{\max}$; and $f_{\min}$ is the average value of $f^k_{\min}$ when the stop occurs.

As is shown in Sect. 3, the proposed algorithm has been derived from two modifications to the new version of the Price's algorithm [6]: (1) Generating the initial population of trial points by the NT method; (2) Using a simplified quadratic model of the objective function based on the three best points in the current population instead of the quadratic model of the objective function in [6]. Here, two sets of experiments are carried out to illustrate the efficiency of these two modifications separately. For convenience, `Algorithm A` stands for the new version of the Price's algorithm given in [6], `Algorithm B` stands for the `Algorithm A` where the quadratic model of the objective function in [6] is replaced by the simplified quadratic model, and `Algorithm C` stands for our modification to `Algorithm A`.

In order to illustrate the efficiency of the simplified quadratic model, we execute `Algorithm B` (case 1), in which $m = 25n$ is chosen and the Monte Carlo method is used to generate its initial population, for every benchmark problem, and compare the results with those obtained by `Algorithm A`. The comparison results are shown in Table 1. The results of `Algorithm A` presented in the table are cited from [6]. From Table 1, one can see that `Algorithm B` (case 1) requires a smaller number of function evaluations and it performs favorably. In particular, in 13 cases `Algorithm B` (case 1) yields a smaller minimum objective function value, and in six cases $f_{\min}$ found by `Algorithm B` (case 1) is equal to that obtained by `Algorithm A`. For Problem 15, we obtain more accurate minimum function values, as shown in [12].

Table 1 Comparison results for illustrating the efficiency of the simplified quadratic model

Problems		Algorithm A			Algorithm B (case 1)				
	n	n_t	n_{f1}	$f_{\min}$	n_t	n_{f1}	n_q	n_{f2}	$f_{\min}$
1	2	3837	1428	6.5×10^{-9}	3004	683	330	70	2.6×10^{-9}
2	3	1648	634	1.9×10^{-3}	1321	379	39	10	1.9×10^{-3}
3	3	3150	1632	2.2×10^{-8}	1662	297	45	2	3.0×10^{-9}
4	4	3500	2318	4.7×10^{-10}	1903	762	31	10	2.3×10^{-10}
5	4	5089	2417	3.2×10^{-8}	3109	587	155	23	1.8×10^{-8}
6	2	722	229	-1.0	592	124	20	7	-1.031628
7	2	903	261	5.8×10^{-12}	876	157	20	0	1.2×10^{-10}
”	4	2374	688	7.7×10^{-11}	2266	444	44	1	3.9×10^{-9}
”	6	3921	1092	3.8×10^{-10}	3850	744	60	2	6.3×10^{-9}
”	8	5427	1440	4.2×10^{-10}	5368	1007	84	2	1.9×10^{-8}
”	10	7081	1815	1.8×10^{-10}	6926	1246	94	4	4.8×10^{-8}
8	2	800	263	2.2×10^{-9}	716	129	20	0	8.9×10^{-10}
”	4	2195	642	2.2×10^{-9}	2143	414	39	1	1.7×10^{-9}
”	6	3790	1082	2.2×10^{-10}	3746	747	58	1	2.3×10^{-9}
”	8	5191	1331	1.7×10^{-9}	5105	1000	72	1	1.8×10^{-8}
”	10	7037	1826	9.7×10^{-11}	6931	1250	94	3	1.8×10^{-8}
9($l = 5$)	4	5403	2841	-10.05	5349	3226	43	18	-10.1532
9($l = 7$)	4	5386	2837	-10.06	5045	2646	40	16	-10.40294
9($l = 10$)	4	5862	3235	-10.07	4959	2571	38	14	-10.53641
10	3	1014	250	-3.86	968	157	19	7	-3.862782
”	6	4154	1432	-3.32	3945	1114	39	14	-3.322368
11	2	936	279	3.00	712	127	22	9	3.00
12	2	586	162	-1.00	412	59	11	0	-1.00
”	4	1655	754	-1.00	1428	271	25	0	-1.00
13	2	723	225	-0.20	537	106	15	0	-0.20
”	4	2327	826	-0.40	1952	484	32	0	-0.40
14	2	710	164	-95.28	645	114	13	4	-95.28289
15($l = 4$)	1	236	89	15.28	205	45	5	1	15.28187
15($l = 10$)	1	203	61	44.95	195	32	4	1	44.95739
15($l = 25$)	1	332	113	261.78	295	74	14	4	261.7863

However, in eight cases $f_{\min}$ found by Algorithm B (case 1) is a little bit larger than that obtained by Algorithm A. Therefore, Algorithm B (case 1) requires a smaller number of function evaluations and yields relatively accurate solutions. These results indicate that in many cases the simplified quadratic model is favorable for achieving better accuracy of the estimated minimum value $f_{\min}$. It is clear that the computational burden of the simplified quadratic model is much less than that of the quadratic model of the objective function in [6].

In order to illustrate the efficiency of the NT method, we execute Algorithm B (case 2), in which $m = 12$, when $n = 1$; $m = 20n$, when $2 \le n \le 7$; $m = 19n$, when $n \ge 8$, and the Monte Carlo method is used to generate its initial population, for every benchmark problem, and compare the results with those obtained by Algorithm C, in which the identical m is chosen, and the square root sequence in the *gp* set given in Sect. 2 is used to generate its initial population. As an exception, in Algorithm B (case 2), we choose $m = 17$ for Problem 15 with $l = 25$, in order to make the algorithm converge to the optimal solution in 15 independent runs. The comparison results are shown in Table 2. From Table 2, one can see that Algorithm C requires a smaller

Table 2 Comparison results for illustrating the efficiency of the number-theoretic method

Problems		Algorithm B (case 2)					Algorithm C				
	n	n_t	n_{f1}	n_q	n_{f2}	$f_{\min}$	n_t	n_{f1}	n_q	n_{f2}	$f_{\min}$
1	2	2928	429	495	139	4.0×10^{-8}	2744	444	418	121	9.3×10^{-10}
2	3	1252	268	64	29	1.9×10^{-3}	984	221	39	9	1.9×10^{-3}
3	3	1443	246	73	7	5.2×10^{-9}	1327	231	56	5	2.1×10^{-9}
4	4	1627	673	37	11	1.1×10^{-10}	1326	376	31	6	1.9×10^{-11}
5	4	2787	415	213	30	2.7×10^{-8}	2478	389	173	23	2.2×10^{-8}
6	2	498	118	21	9	−1.031628	451	94	17	5	−1.031628
7	2	858	159	22	0	1.9×10^{-12}	802	140	18	0	1.5×10^{-12}
”	4	2264	424	44	1	1.3×10^{-11}	2096	365	44	1	2.6×10^{-11}
”	6	3708	660	69	5	7.7×10^{-11}	3541	863	59	3	2.7×10^{-10}
”	8	5108	852	88	3	1.1×10^{-10}	4875	1320	77	6	3.7×10^{-10}
”	10	6884	1070	108	5	2.4×10^{-10}	6593	1800	88	6	1.6×10^{-10}
8	2	591	108	18	0	1.6×10^{-9}	533	83	18	0	1.1×10^{-9}
”	4	1800	356	43	1	1.2×10^{-9}	1708	330	41	0	1.2×10^{-9}
”	6	3642	701	65	3	4.0×10^{-11}	3524	852	55	2	6.9×10^{-11}
”	8	4520	781	92	4	9.5×10^{-10}	4349	1142	73	4	8.0×10^{-10}
”	10	6729	1093	109	5	1.2×10^{-9}	6571	1791	86	6	4.9×10^{-11}
9($l = 5$)	4	4921	2924	44	19	−10.1532	4348	2504	42	16	−10.1532
9($l = 7$)	4	4014	2108	43	16	−10.40294	3724	1825	43	16	−10.40294
9($l = 10$)	4	3929	2036	42	15	−10.53641	3768	1878	42	16	−10.53641
10	3	824	139	19	6	−3.862782	755	109	17	6	−3.862782
”	6	3132	847	36	13	−3.322368	2953	1161	34	13	−3.322368
11	2	611	108	23	9	3.00	534	85	21	7	3.00
12	2	348	50	12	0	−1.00	309	41	13	0	−1.00
”	4	1160	204	25	0	−1.00	970	166	25	0	−1.00
13	2	491	102	16	0	−0.20	412	78	15	0	−0.20
”	4	1520	361	35	0	−0.40	1290	295	33	0	−0.40
14	2	601	146	12	4	−95.28289	545	115	12	5	−95.28289
15($l = 4$)	1	112	24	7	2	15.28187	86	14	6	2	15.28187
15($l = 10$)	1	109	21	6	1	44.95739	82	11	5	1	44.95739
15($l = 25$)	1	235	54	16	6	261.7863	118	16	12	4	261.7863

number of function evaluations, and it performs better than Algorithm B (case 2). In particular, in nine cases Algorithm C yields a smaller minimum objective function value, and in 17 cases $f_{\min}$ found by Algorithm C is equal to that obtained by Algorithm B (case 2). However, in four cases $f_{\min}$ found by Algorithm C is a little bit larger than that obtained by Algorithm B (case 2). These results indicate that in most cases the NT method is effective in reducing the number of function evaluations and in improving the accuracy of the minimum objective function value, when the number of points in the initial population set is relatively small. In fact, the NT method is a powerful way for generating the initial population of a population-based algorithm.

For all the benchmark problems presented in the Appendix, a comparison between Algorithm A in Table 1 and Algorithm C in Table 2 has been made. From Tables 1 and 2, one can see that Algorithm C performs better on all the benchmark problems. In particular, both n_t and n_{f1} are smaller; in 21 cases $f_{\min}$ is smaller and in other cases $f_{\min}$ is equal to the minimum function value obtained by Algorithm A, except for problem 15, we obtain more accurate minimum function values. Therefore,

Table 3 Results of `Algorithm C` with smaller m

Problems	n	m	n_t	n_{f1}	n_q	n_{f2}	$f_{\min}$
1	2	36	2326	390	373	99	1.5×10^{-9}
2	3	54	814	180	32	8	1.9×10^{-3}
3	3	45	911	147	48	4	3.8×10^{-9}
4	4	60	979	242	37	6	4.0×10^{-11}
5	4	76	2420	358	200	24	6.4×10^{-8}
6	2	20	222	42	19	7	−1.031628
7	2	26	512	82	26	1	1.7×10^{-12}
”	4	56	1533	262	54	2	1.2×10^{-11}
”	6	96	2798	465	65	2	7.6×10^{-11}
”	8	120	4217	980	75	4	1.2×10^{-10}
”	10	150	5715	1499	89	7	1.7×10^{-10}
8	2	28	424	72	22	0	1.4×10^{-10}
”	4	56	1193	197	44	1	5.2×10^{-10}
”	6	96	2788	494	62	3	9.2×10^{-11}
”	8	112	3265	676	73	3	8.0×10^{-10}
”	10	140	5172	1253	87	7	5.8×10^{-11}
9($l = 5$)	4	56	2771	1533	42	17	−10.1532
9($l = 7$)	4	56	2438	1155	42	15	−10.40294
9($l = 10$)	4	56	2447	1160	45	17	−10.53641
10	3	36	453	60	21	6	−3.862782
”	6	84	2075	502	39	15	−3.322368
11	2	24	319	49	24	9	3.00
12	2	24	169	17	13	0	−1.00
”	4	48	628	84	29	0	−1.00
13	2	24	262	44	16	0	−0.20
”	4	56	975	191	37	0	−0.40
14	2	24	314	60	10	3	−95.28289

`Algorithm C` requires a much smaller number of function evaluations and yields a smaller or more accurate minimum objective function value in many cases.

From the analysis of the behavior of `Algorithm C`, we remark that the NT method makes the algorithm explore uniformly the region of interest at the initial iteration, and that the applications of the simplified quadratic model and the NT method is effective in reducing the number of function evaluations and in improving the accuracy of the estimated minimum objective function value.

Thanks to the advantage of the NT method for producing the uniformly scattered initial population over the search space, we may choose smaller m in `Algorithm C` and solve many benchmark problems. Table 3 shows the results of `Algorithm C` with smaller m for 14 benchmarks. Compared with the corresponding results given in Table 2, one can see that for all the cases `Algorithm C` with smaller m needs a smaller number of function evaluations. Although in eight cases $f_{\min}$ given in Table 3 is a little bit larger than that in Table 2, `Algorithm C` still yields relatively accurate solutions.

Since the three-point quadratic approximation in `Algorithm C` can reduce greatly its computational overhead, the proposed algorithm can be used to solve medium size problems. We have tested `Algorithm C` on 5 benchmark problems of medium size. For relative simple problems, e.g. problems 12 and 13, we choose smaller m. However, for relatively difficult problems, e.g. problems 1 and 8, in order to make the algorithm

Table 4 Results of `Algorithm C` for 5 benchmark problems with medium sizes

Problems	n	m	n_t	n_{f1}	n_q	n_{f2}	$f_{\min}$
1	10	250	24715	2076	1400	418	1.2×10^{-6}
”	15	840	97386	9165	2299	1091	1.1×10^{-5}
7	20	280	11467	1179	325	50	2.1×10^{-5}
”	30	420	17634	1941	484	109	1.6×10^{-4}
”	40	760	33250	4129	455	108	2.3×10^{-4}
”	50	950	49089	5353	579	170	6.8×10^{-4}
8	20	380	14827	1874	272	35	7.1×10^{-6}
”	30	600	24279	3144	424	95	5.2×10^{-5}
”	40	1200	50578	7375	370	73	8.6×10^{-5}
”	50	1500	74873	9586	502	161	4.6×10^{-4}
12	20	120	5681	1684	158	10	−1.00
”	30	240	23570	11495	129	10	−1.00
”	40	240	38330	19867	208	28	−1.00
”	50	250	47652	22457	260	47	−1.00
13	20	160	6696	992	240	13	−2.00
”	30	240	12374	2018	335	25	−3.00
”	40	320	19479	3416	459	37	−4.00
”	50	400	24811	3849	538	46	−5.00

converge to the optimal solution in 15 independent runs, larger m may be chosen. The results of `Algorithm C` for the medium size problems are shown in Table 4. From the table, one can see that `Algorithm C` performs favorably for the medium size problems.

Apparently, the values of n_q and n_{f2} given in Tables 1–5 indicate that the three-point quadratic approximation is an important operator. In most cases, the operator is successful for improving the estimated minimum objective function value and makes the algorithm converge faster.

5 Conclusions

We have presented a modification to the new version of the Price's algorithm given in [6] for the unconstrained optimization problem (P). Its dominant factor is that the application of the NT method for generating the initial population, the weighted centroid, the weighted reflection, and the simplified quadratic approximation with the three best points. The three-point quadratic approximation in the algorithm avoids the solution of simultaneous linear equations, which can reduce greatly the computational overhead of the algorithm. The simplified quadratic model can also improve the accuracy of the minimum function value and enhance the local search ability of the algorithm. The close integration of the global search part of the algorithm with the simplified quadratic approximation makes the proposed algorithm converge fast and adapt itself to the problem being solved. Two sets of experiments have been carried out to illustrate the efficiency of the simplified quadratic model and the number-theoretic method separately. The comparison results with the original algorithm show that the proposed algorithm requires a smaller number of function evaluations and, in many cases, yields a smaller or more accurate minimum function value. Thanks to the

advantage of the three-point quadratic approximation, the proposed algorithm can also be used to deal with medium size problems. The performance of the proposed algorithm is efficient.

For the first time, the NT method has been applied to generate the initial population of a population-based algorithm. Owing to its powerful way of determining a set of uniformly scattered vectors in an n-dimensional unit cube, the NT method can be regarded as a general efficient method for producing the initial population of a population-based algorithm. As a result, the algorithm could explore the search space uniformly, enhance the diversity of the population, and reduce the chance of being trapped in local minima at the initial iteration. With the NT method, the initial population of the algorithm is generated by a deterministic way, which is different from the Monte Carlo method in the usual statistical sense. We believe that the deterministic method for producing the uniformly scattered initial population in the search space will be well accepted by the users for engineering applications.

The choice of the initial population by the NT method may reduce the value of m and improve the efficiency of the algorithm. How to select a suitable value of m and to make the algorithm converge faster remains to be studied further.

A Appendix: Benchmark problems [6]

1. Extended Rosenbrock

$$f(x) = \sum_{i=1}^{n-1} \left[(x_i - 1)^2 + 100(x_i^2 - x_{i+1})^2\right],$$
$$x^* = (1, 1, \ldots, 1)^\top, \quad f(x^*) = 0.$$

   The region of interest is $-1000 \le x_i \le 1000$, $i = 1, 2, \ldots, n$.

2. Meyer and Roth

$$f(x_1, x_2, x_3) = \sum_{i=1}^{l} \left[Y_i(t, v, x) - y_i\right]^2,$$

   in which

$$Y_i(t, v, x) = \frac{x_1 x_3 t_i}{1 + x_1 t_i + x_2 v_i}$$

   and t_i, v_i and y_i are given the following table.

i	t_i	v_i	y_i
1	1.0	1.0	0.126
2	2.0	1.0	0.219
3	1.0	2.0	0.076
4	2.0	2.0	0.126
5	0.1	0.0	0.186

   For this problem $l = 5$ and $n = 3$,

$$x^* = (3.13, 15.16, 0.78)^\top, \quad f(x^*) = 4.36 \times 10^{-5}.$$

The region of interest is $-10 \le x_i \le 10$, $i = 1, 2, 3$. In this region,

$$x^* = (3.52, 10.0, 0.571)^\top, \quad f(x^*) = 1.9 \times 10^{-3}.$$

3. Flether and Powell

$$f(x_1, x_2, x_3) = 100\big[(x_3 - 10\theta)^2 + (r-1)^2\big] + x_3^2,$$

where

$$r = \left(x_1^2 + x_2^2\right)^{1/2}$$

and

$$\theta = \begin{cases} \frac{1}{2\pi}\tan^{-1}\frac{x_2}{x_1}, & (x_1 > 0), \\ \frac{1}{2\pi}\tan^{-1}\frac{x_2}{x_1} + \frac{1}{2}, & (x_1 < 0). \end{cases}$$

$$x^* = (1, 0, 0)^\top, \quad f(x^*) = 0.$$

The region of interest is $-10 \le x_i \le 10$, $i = 1, 2, 3$.

4. Miele and Cantrell

$$f(x_1, x_2, x_3, x_4) = \big[\exp(x_1) - x_2\big]^4 + 100(x_2 - x_3)^6 + \big[\tan(x_3 - x_4)\big]^4 + x_1^8,$$
$$x^* = (0, 1, 1, 1)^\top, \quad \text{or} \quad (0, 1, 1, 1 + j\pi)^\top, \quad j = -3, -2, -1, 1, 2, \ f(x^*) = 0.$$

The region of interest is $-10 \le x_i \le 10$, $i = 1, 2, 3, 4$.

5. Wood's function quoted by Colville

$$\begin{aligned} f(x_1, x_2, x_3, x_4) &= 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 \\ &\quad + 10.1\big[(x_2 - 1)^2 + (x_4 - 1)^2\big] + 19.8(x_2 - 1)(x_4 - 1), \\ x^* &= (1, 1, 1, 1)^\top, \quad f(x^*) = 0. \end{aligned}$$

The region of interest is $-10 \le x_i \le 10$, $i = 1, 2, 3, 4$.

6. Six-hump camel back function

$$f(x_1, x_2) = \big(4 - 2.1x_1^2 + x_1^4/3\big)x_1^2 + x_1x_2 + \big(-4 + 4x_2^2\big)x_2^2.$$

The region of interest is $-2.5 \le x_1 \le 2.5$, $-1.5 \le x_2 \le 1.5$. The approximate optimal value $f^* = -1.031628$. This function exhibits six local minimizers, two of which are also global.

7. 10^n local minima

$$f(x) = (\pi/n)\left\{10\sin^2(\pi x_1) + \sum_{i=1}^{n-1}\Big[(x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1}))\Big] + (x_n - 1)^2\right\},$$
$$f(x^*) = 0.$$

The region of interest is $-10 \le x_i \le 10$, $i = 1, 2, \ldots, n$. This function has roughly 10^n local minimizers and a unique global minimizer located at $x_i^* = 1$, $i = 1, 2, \ldots, n$.

8. 15^n local minima

$$f(x) = (1/10)\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}\left[(x_i-1)^2(1+10\sin^2(3\pi x_{i+1}))\right]\right\}$$
$$+(1/10)(x_n-1)^2\left[1+\sin^2(2\pi x_n)\right],$$
$$f(x^*) = 0.$$

The region of interest is $-10 \le x_i \le 10$, $i = 1,2,\ldots,n$. This function has roughly 15^n local minimizers and a unique global minimizer located at $x_i^* = 1$, $i = 1,2,\ldots,n$.

9. Shekel's family

$$f(x) = -\sum_{i=1}^{l} \frac{1}{(x-a_i)^\top(x-a_i)+c_i}.$$

We studied this function with $l = 5$, $l = 7$, $l = 10$ and $n = 4$. The values of $a_i = (a_{i1},\ldots,a_{in})^\top$ and $c_i > 0$ $(i = 1,\ldots,l)$ are given in the following table.

i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

The region of interest is $0 \le x_i \le 10$, $i = 1,2,\ldots,n$. This function has l local minima in positions a_i with levels c_i.

10. Hartman's family

$$f(x) = -\sum_{i=1}^{l} c_i \exp\Big[-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\Big].$$

We studied this function with $l = 4$, $n = 3$ and $n = 6$. The corresponding values of $a_i = (a_{i1},\ldots,a_{in})^\top$, $p_i = (p_{i1},\ldots,p_{in})^\top$ and $c_i > 0$ $(i = 1,\ldots,l)$ are given in following tables.

i	a_{i1}	a_{i2}	a_{i3}	c_i	p_{i1}	p_{i2}	p_{i3}
1	3.0	10.0	30.0	1.0	0.3689	0.1170	0.2673
2	0.1	10.0	35.0	1.2	0.4699	0.4387	0.7470
3	3.0	10.0	30.0	3.0	0.1091	0.8732	0.5547
4	0.1	10.0	35.0	3.2	0.03815	0.5743	0.8828

i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	c_i
1	10.0	3.0	17.0	3.5	1.7	8.0	1.0
2	0.05	10.0	17.0	0.1	8.0	14.0	1.2
3	3.0	3.5	1.7	10.0	17.0	8.0	3.0
4	17.0	8.0	0.05	10.0	0.1	14.0	3.2

i	p_{i1}	p_{i2}	p_{i3}	p_{i4}	p_{i5}	p_{i6}
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

The region of interest is $0 \le x_i \le 1, i = 1, 2, \ldots, n$. This function has l local minima in positions p_i with levels c_i.

11. Goldstein and Price

$$f(x) = \big[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\big] \big[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\big],$$

$$x^* = (0, -1)^\top, \quad f(x^*) = 3.$$

The region of interest is $-2 \le x_i \le 2, i = 1, 2$.

12. Exponential

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

$$f(x^*) = -1.$$

The region of interest is $-1 \le x_i \le 1, i = 1, 2, \ldots, n$.

13. Cosine mixture

$$f(x) = -0.1\sum_{i=1}^{n} \cos(5\pi x_i) + \sum_{i=1}^{n} x_i^2,$$

$$f(x^*) = -0.1n.$$

The region of interest is $-1 \le x_i \le 1, i = 1, 2, \ldots, n$.

14. Poissonian pulse-train likelihood

$$f(x) = \sum_{i=1}^{p} \Big\{\lambda_i(x) - \hat{n}_i \log[\lambda_i(x)]\Big\},$$

where

$$\lambda_i(x) = 2\Big\{1 + 2.5\exp\Big[-0.5\left(\frac{i - x_1}{x_2}\right)^2\Big]\Big\} + 3$$

and $p = 21$. The values of $\hat{n}_i$, $i = 1, \ldots, 21$ are: 5, 2, 4, 2, 7, 2, 4, 5, 4, 4, 15, 10, 8, 15, 5, 6, 3, 4, 5, 2, 6. The approximate optimal value $f(x^*) = -95.28289$. The region of interest is $1 \le x_1 \le 21, 1 \le x_2 \le 8$.

15. Cauchy likelihood

$$f(x) = \sum_{i=1}^{l} \Big\{\log(\pi) + \log[1 + (y_i - x)^2]\Big\}.$$

We studied this function with $l = 4, l = 10$, and $l = 25$. The values of y_i are given in the following table:

$l = 4$	3	7	12	17						
$l = 10$	2	5	7	8	11	15	17	21	23	26
$l = 25$	4.1	7.7	17.5	31.4	32.7	92.4	115.3	118.3	119.0	129.6
	198.6	200.7	242.5	255.0	274.7	274.7	303.8	334.1	430.0	489.1
	703.4	978.0	1656.0	1697.8	2745.6					

The region of interest is $y_1 \le x \le y_l$.

Acknowledgements The authors would like to thank two anonymous referees for their invaluable comments and remarks. This work was supported by SRG: 7001364 of City University of Hong Kong.

References

1. Horst, R.H., Pardalos, P. M.: Handbook of Global Optimization. Kluwer Academic Publishers, Dordrecht (1995)
2. Törn, A., Ali, M.M., Vitanen, S.: Stochastic global optimization: problem classes and solution techniques. J. Global Optimi. **14**, 437–447 (1999)
3. Ali, M.M., Törn, A., Vitanen, S.: A numerical comparison of some modified controlled random search algorithms. J. Global Optim. **11**, 377–385 (1997)
4. Ali, M.M., Storey, C.: Aspiration based simulated annealing algorithm. J. Global Optim. **11**, 181–191 (1997)
5. Storn, B., Price, K.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**, 341–359 (1997)
6. Brachetti, P., De Felice Ciccoli, M., Di Pillo, G., Lucidi, S.: A new version of the price's algorithm for global optimization. J. Global Optim. **10**, 165–184 (1997)
7. Price, W.L.: A controlled random search procedure for global optimization. In: Dixon, L. C. W., Szegö, G. P. (eds.) Towards Global Optimization 2. North Holland, Amsterdam (1978)
8. Hua, L.K., Wang, Y.: Application of Number Theory to Numerical Analysis. Springer-Verlag and Science Press, Berlin and Beijing (1981)
9. Fang, K.T., Wang, Y.: Number-Theoretic Methods in Statistics. Chapman & Hall, London (1994)
10. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia (1992)
11. Niederreiter, H., Peart, P.: Localization of search in quasi-Monte Carlo methods for global optimization. SIAM J. Sci. Stat. Comput. **7**(2), 660–664 (1986)
12. Breiman, L., Cutler, A.: A deterministic algorithm for global optimization. Math. Programming **58**, 179–199 (1993)