Optim Lett (2019) 13:761–776
https://doi.org/10.1007/s11590-018-1275-5

ORIGINAL PAPER

A filled function which has the same local minimizer of the objective function

Hongwei Lin[1] · Yuelin Gao[2] · Xiaoli Wang[3] · Shoubao Su[4]

Received: 5 October 2016 / Accepted: 10 May 2018 / Published online: 22 May 2018

Abstract Auxiliary function methods have been considered to be practical approaches for finding the global minimizer of multi-model functions.Filled function methods, as a typical representative of auxiliary function methods, obtain a global minimizer by minimizing the objective function and the filled function cyclically. In order to improve the efficiency of the filled function, this paper presents a new filled function which has the same local minimizers of the objective function, and these minimizers are all better than the current minimizer of the objective function. Therefore, it does not need to minimize the objective function except for the first iteration in the filled function method. Additionally, the proposed filled function excludes some disadvantages of conventional filled functions and a classical local optimization method can be applied directly to the new filled function to obtain a better minimizer of the original problem. Finally, numerical experiments are made and the results show the effectiveness of the proposed method.

Keywords Global optimization · Filled function method · Global minimizer · Local minimizer

✉ Hongwei Lin
lhw@jit.edu.cn

[1] Faculty of science, Jinling Institute of Technology, Nanjing 211169, People's Republic of China

[2] Information and System Science, Beifang University for Nationalities, Yinchuan 750021, People's Republic of China

[3] School of Computer Science and Technology, Xidian University, Xi'an 710071, People's Republic of China

[4] School of Computer Engineering, Jinling Institute of Technology, Nanjing 211169, People's Republic of China

1 Introduction

Global optimization has extensive applications in many social practices, such as engineering, finance, management, decision science and so on. In recent years, many theoretical and computational contributions have been reported for solving global optimization problems. In general, existing approaches can be classified into two categories: deterministic methods and probabilistic methods. The former includes the auxiliary function method [1–4], trajectory method [5,6], and covering method [7,8] and so on. The typical representatives of the latter are genetic algorithm [10], simulated annealing method [11], particle swarm optimization [12] and differential evolution algorithm [13] and so on. Although the latter methods have been effectively applied to solve many complex problems, there still some issues to solve, such as convergence analysis, the decrease of the computational time, the selection of the parameters, etc. Compared with the latter, the former methods turn out to be more mature in theorems, but many of them have shown to be dependent on the conditions of the specific problem. Additionally, in practical problems, most of the objective functions have several local minimizers, which leads a great challenge for global optimization. The key of global optimization is finding a way to escape from the current local minimizer. Among the existing methods, the auxiliary function method appears to have several advantages over others mainly due to its relatively easy actualization with a process that aims at successively finding a smaller local minimizer. This paper mainly focuses on one representative method of auxiliary function methods: filled function method (FFM), whose main idea can be described as follows:

1. An arbitrary point is taken as an initial point to minimize the objective function by using a local optimization method, and a minimizer of the objective function is obtained.
2. Based on the minimizer of the objective function, construct a filled function and take some points around this minimizer as initial points to minimize the constructed filled function. A minimizer of the filled function falls into a better region which contains a better minimizer with smaller objective function value and it will be found.
3. Take the minimizer of the filled function obtained in the second step as an initial point to minimize the objective function and obtain a better minimizer of the objective function.

By repeating process 2 and 3, a global minimizer will be found at last.

The filled function method has been applied to solve various global optimization problems, e.g., bound-constrained and nonlinearly constrained optimization problems [2], integer programming [14], max-cut problems [15] and so on. As one of the main methods to solve unconstrained global optimization, since the filled function method was proposed, considerable attention has been paid on the constructing of the new filled functions [9,16,17]. Both theoretical and algorithmic studies demonstrate that the filled function method has potential. The filled function method is an efficient global optimization method and different filled functions have been proposed. Conventional filled functions either are non-differentiable (even discontinuous) [5,6], need more than one adjustable parameter [1,3,4] or contain ill-conditioned term (exponential terms or logarithmic terms) [1,2]. Additionally, from the general framework of the FFM, we can see that the minimizer of the filled function is generally not a minimizer of the objective function, and a better local minimizer obtained need to solve the original problem. Consequently, if the filled function which has the same local minimizer of the objective function, and these local minimizers are better than the current one of the original problem, then a minimizer of the filled function will be a better minimizer of the original problem. Therefore the efficiency of the FFM will be higher.

To overcome the shortcomings of the conventional filled functions, a new filled function with only one easy to adjust parameter is proposed. It is a continuously differentiable function which excludes exponential terms or logarithmic terms. What is more satisfactory is that any one local minimizer of the new proposed function is a better local minimizer of the original problem.

The rest of this paper is organized as follows. Some knowledge that is basic for the subsequent sections are given in Sect. 2. In Sect. 3, a new continuously differentiable filled function with only one parameter is proposed and its properties are analyzed. An algorithm with some practical considerations is given in Sect. 4, some numerical experiment results on some testing problems are also shown in this section. Finally, some concluding remarks are given in Sect. 5.

2 Preliminaries

In this paper, we consider the following unconstrained global optimization problem :

$$(P)\begin{cases}\min\ f(x),\\ s.t.\ \ x\in R^n.\end{cases}$$

where $f : R^n \to R$ is a continuously differentiable function.

Assumption 1 $f(x)$ is a coercive function on R^n, namely, $f(x)\to +\infty$, as $\|x\|\to +\infty$.

Assumption 1 implies that there exists a box $\Omega=\prod_{i=1}^{n}[l_i,u_i]\subset R^n$ whose interior contains all global minimizers of $f(x)$. Thus the problem (P) is equivalent to the following problem:

$$\min_{x\in\Omega} f(x) \tag{1}$$

Then, we only consider problem (1) in the following. In order to analyze the properties of the new filled function, the following assumptions are all necessary.

Assumption 2 $f(x)$ has only a finite number of minimizers in Ω and the set of the minimizers is denoted as

$$M^l=\left\{x_i^*|i=1,2,\ldots,N\right\}$$

where N is the number of minimizer of $f(x)$

Assumption 3 All of the local minimizers of $f(x)$ fall into the interior of Ω, and each point y on the boundary of Ω satisfies $f(y) > c$, where c satisfies $c = \max\{f(x)|x \in M^l\}$.

Additionally, some useful symbols will be adopted in the following.

k: the iteration number;
x_k: the initial point in the k-th iteration;
x_k^*: the local minimizer of $f(x)$ in the k-th iteration;
f_k^*: the function value at x_k^*;
S_1: the set defined by $S_1 = \{x|f(x) \geq f(x_k^*), x \in \Omega\backslash\{x_k^*\}\}$;
S_2: the set defined by $S_2 = \{x|f(x) < f(x_k^*), x \in \Omega\}$;
m: a constant defined by $m = \min_{i,j\in\{1,2,\cdots N\}, f(x_i^*)\neq f(x_j^*)} |f(x_i^*) - f(x_j^*)|$.

Based on the above symbols and assumptions, the FFM was first proposed by Ge [1,2]and has been updated as the following generations. The examples of the filled function in the first generation are P-function [1] and G-function [2] which are listed as follows:

$$P(x, r, \rho) = \exp\left(-\|x - x_k^*\|/\rho^2\right)/(r + f(x)) \tag{2}$$

$$G(x, r, \rho) = -\left[\rho^2 \ln(r + f(x)) + \left\|x - x_k^*\right\|^p\right] \tag{3}$$

where p is an integer, for example, p can be taken as 2.

The P-function and G-function have a common feature: there are two adjustable parameters, r and ρ, which need to be appropriately iterated and coordinated. Because of this limitation, the second-generation filled functions were proposed. Among them, the best known is the Q-function [2] given by

$$Q(x, A) = -\left(f(x) - f\left(x_k^*\right)\right)\exp\left(A\left\|x - x_k^*\right\|^2\right). \tag{4}$$

The Q-function has only one adjustable parameter A, so the algorithm is significantly efficient than those in the first generation. However, the Q-function is susceptible to exponential terms when applied to global optimizations since its magnitude increases exponentially against parameter A. The larger A, which is required by the property of the filled function, the larger exponential may result in an overflow in the computation. To overcome this shortcoming, the H-function was proposed in [18] as follows:

$$H(x, A) = 1/\ln\left(1 + f(x) - f\left(x_k^*\right)\right) - A\left\|x - x_k^*\right\|^2. \tag{5}$$

The H-function keeps the advantage of the Q-function that it has only one adjustable parameter and that it does not include the exponential term. The H-function can be regarded as an example of the third-generation filled functions. Nevertheless, it discontinues at the points whose function value is equal to the one at x_k^*. Therefore, most local minimization algorithms used in the filled function may lose efficiency, and the FFM will be failure to find a global minimizer of the problem (1). This leads to the fourth-generation (C-function) of the FFM [19]. Based on the thinking of the C-function, a new filled function which has the same properties as the C-function is constructed in this paper. The new filled function not only excluded from some disadvantages of the conventional filled function, but also has the same local minimizer with the original problem.

3 A new filled function and its properties

The definition of the filled function is first introduced by Ge in [1]. Since the definition of the filled function was introduced, many variations of the definition of the filled function are given. In this paper, we adopt the revised definition of the filled function as follows:

Definition 1 Suppose x_k^* is a current local minimizer of $f(x)$, A continuously differentiable function $\mathcal{F}(x)$ is said to be a filled function of $f(x)$ at x_k^*, if it satisfies the following properties:

(1) x_k^* is a strict local maximizer of $\mathcal{F}(x)$;
(2) $\mathcal{F}(x)$ has no stationary point in the set S_1;
(3) If x_k^* is not a global minimizer of $f(x)$, namely S_2 is not empty, then there exists a point $x_k^{'}$ such that it is a local minimizer of $\mathcal{F}(x)$ on S_2 which is a better local minimizer of $f(x)$.

In order to construct a new filled function, two functions with one variable are introduced firstly:

$$h(t) = \begin{cases} 1, & t \geq 0, \\ 1 - 2t^3 - 3t^2, & -1 \leq t < 0, \\ 0, & t \leq -1. \end{cases}$$

$$g(t) = \begin{cases} 0, & t \geq 0, \\ t^3, & t < 0. \end{cases}$$

it can be seen that $h(t)$ and $g(t)$ are all continuously differentiable functions over R. Based on functions $h(t)$ and $g(t)$, we construct a filled function for solving the problem (1) at a local minimizer x_k^* as follows:

$$\mathcal{F}\left(x, x_k^*, P\right) = \frac{1}{1 + \left\|x - x_k^*\right\|^2} \times h\left(P \times \left(f(x) - f\left(x_k^*\right)\right)\right) + g\left(f(x) - f\left(x_k^*\right)\right) \tag{6}$$

where $P > 0$ is a parameter. We can see that the formula (6) is a continuously differentiable function. The following theorems indicate that the function (6) is a filled function which satisfies Definition 1.

Definition 2 A set $N(x, \delta) = \{y | \|y - x\| < \delta\}$ is called a neighborhood of x.

Definition 3 A point y is said to be a local minimizer (maximizer) of $f(x)$ on a set A, if there is δ, for $\forall z \in N(y, \delta) \bigcap A$, one has $f(z) \geqslant (\leqslant) f(y)$. If the equality does not hold, the minimizer (maximizer) is called strict. For $\forall z \in A$, $f(z) \geqslant (\leqslant) f(y)$ holds, y is called a global minimizer (maximizer) of $f(x)$ on A

Theorem 1 *Suppose x_k^* is a local minimizer of $f(x)$ and $\mathcal{F}(x, x_k^*, P)$ is defined by (6), then x_k^* is a strict local maximizer of $\mathcal{F}(x, x_k^*, P)$ on Ω for all $P > 0$.*

Proof Since x_k^* is a local minimizer of the problem $f(x)$, there exists $\delta > 0$, such that $f(x) \geqslant f(x_k^*)$ for any $x \in \Omega \bigcap N(x_k^*, \delta)$. Then, for any $x \in \Omega \bigcap N(x_k^*, \delta)$, $x \neq x_k^*$, one has

$$\mathcal{F}\left(x, x_k^*, P\right) = \frac{1}{1 + \|x - x_k^*\|^2} < 1 = \mathcal{F}\left(x_k^*, x_k^*, P\right). \tag{7}$$

Thus, x_k^* is a strict local maximizer of $\mathcal{F}(x, x_k^*, P)$. □

Theorem 2 *Suppose x_k^* is a local minimizer of $f(x)$, if x is a point such that in set S_1, then x is not a stationary point of $\mathcal{F}(x, x_k^*, P)$ for all $P > 0$.*

Proof By $x \in S_1$, one has $x \neq x_k^*$ and

$$\mathcal{F}\left(x, x_k^*, P\right) = \frac{1}{1 + \left\|x - x_k^*\right\|^2}, \quad \nabla \mathcal{F}\left(x, x_k^*, P\right) = \frac{-2\left(x - x_k^*\right)}{\left(1 + \left\|x - x_k^*\right\|^2\right)^2} \neq \mathbf{0}$$

Namely x is not a stationary point of $\mathcal{F}(x, x_k^*, P)$ (where$\nabla\mathcal{F}$ is the gradient of $\mathcal{F}$). □

By the continuously differentiability of $\mathcal{F}(x, x_k^*, P)$ and definition of S_1, we know that $\forall y \in S_1$ is not a local minimizer of $\mathcal{F}(x, x_k^*, P)$.

Theorem 3 *Suppose x_k^* is a local minimizer but not a global minimizer of $f(x)$ on Ω, it means S_2 is not empty, then there exists a point $x^{'} \in S_2$ such that $x^{'}$ is a local minimizer of $\mathcal{F}(x, x_k^*, P)$ when $P > \frac{1}{m}$.*

Proof Since x_k^* is a local but not a global minimizer of $f(x)$, there exists another local minimizer x^* of $f(x)$ such that $f(x^*) < f(x_k^*)$

By definition of m and the continuity of $f(x)$, if $P > \frac{1}{m}$, then $P \times (f(x^*) - f(x_k^*)) \leq -mP < -1$. Consequently

$$\mathcal{F}\left(x^*, x_k^*, P\right) = \left(f\left(x^*\right) - f\left(x_k^*\right)\right)^3 < 0 \tag{8}$$

By the Assumption 3, $\forall y \in \partial\Omega$, one has $f(y) > f(x_k^*)$, there by $\mathcal{F}(y, x_k^*, P) = \frac{1}{1+\|y-x_k^*\|^2} > 0$. Thus, by the intermediate value theorem of continuous functions, there exists a point on the segment between x^* and y denoted by $[x^*, y]$ and the function value at this point is 0. Assume that z is the closest point to x^* on this segment with $\mathcal{F}(z, x_k^*, P) = 0$, then we can obtain a segment $[x^*, z]$.

Let $S(x^*)$ be the set of all the above line segments $[x^*, z]$ when $y \in \partial\Omega$, then it is a closed region. By continuity of $\mathcal{F}(x, x_k^*, P)$, there exists a $x^{'} \in S(x^*)$ such that it is a minimizer $\mathcal{F}(x, x_k^*, P)$ and $\mathcal{F}(x^{'}, x_k^*, P) < 0$. Since $\mathcal{F}(x, x_k^*, P)$ is continuously differentiable, thus

$$\nabla \mathcal{F}\left(x^{'}, x_k^*, P\right) = \mathbf{0}$$

□

From Theorems 1, 2 and 3, we know that if x_k^* is not a global minimizer of $f(x)$ on Ω, there exists a local minimizer $x^{'}$ of $\mathcal{F}(x, x_k^*, P)$ on Ω which satisfies $x^{'} \in S_2$ when parameter P is taken as large as possible. Meanwhile, if x_k^* is not a global minimizer of $f(x)$, there exists another local minimizer x^* of $f(x)$ such that $f(x^*) < f(x_k^*)$, then $\exists N(x^*, \delta)$ such that $P \times (f(x) - f(x_k^*)) < -1$ for $\forall x \in N(x^*, \delta)$, there by for $\forall x \in N(x^*, \delta)$, $\mathcal{F}(x, x_k^*, P) = (f(x) - f(x_k^*))^3$. It is obvious that x^* is a local minimizer of $\mathcal{F}(x, x_k^*, P)$. Thus, we can see that a local minimizer of $\mathcal{F}(x, x_k^*, P)$ is also a minimizer of $f(x)$ whose function value is less than $f(x_k^*)$. Therefore, the filled function method only need to minimize the filled function for finding a global minimizer of the problem (1).

4 Filled function algorithm and numerical experiments

4.1 filled function algorithm and some explanations

Based on the theorems and discussions in the above section, a new filled function algorithm for finding a global minimizer of $f(x)$ and some explanations of the algorithm are given. The algorithm is described firstly.

The filled function algorithm

Step0: Choose an upper bound Ubp of P (e.g., take it as 10^6) and a constant $\rho > 1$ (e.g., take it as $\rho = 10$); give the initial P (e.g., take it as 1)respectively; Some directions $e_i, i = 1, 2, \ldots, K, K \geq 2n$ are given in advance, where n is the dimension of the optimization problems. Set $k := 1$, and choose a point $x_k \in \Omega$.

Step1: Minimize $f(x)$ starting from an initial point $x_k \in \Omega$ and obtain a minimizer x_k^* of $f(x)$.

Step2: Construct

$$\mathcal{F}\left(x, x_k^*, P\right) = \frac{1}{1+\|x-x_k^*\|^2} \times h\left(P \times \left(f(x) - f\left(x_k^*\right)\right)\right) + g\left(f(x) - f\left(x_k^*\right)\right).$$

Set $i = 1$.

Step3: If $i \leqslant K$, then set $x = x_k^* + \delta e_i$ and go to step 4; otherwise, go to Step5.

Step4: Use x as an initial point for minimization of $\mathcal{F}(x, x_k^*, P)$. If the minimization sequences of $\mathcal{F}(x, x_k^*, P)$ go out of Ω, set $i = i+1$ and go to Step3; otherwise, a minimizer $x^{'}$ will be found by minimizing $\mathcal{F}(x, x_k^*, P)$, set $x_{k+1}^* = x^{'}$,$k = k+1$ and go to Step2.

Step5: If $P < Ubp$, then increase P by setting $P := \rho \times P$ and go to Step 2; Otherwise, the algorithm stops and x_k^* is taken as a global minimizer of problem (P).

Some explanations about the above filled function algorithm are necessary.

1. In Step0, K directions need to be chosen, in general, take $K = 2n$, $e_i = (0, \dots, \underset{i}{1}, \dots, 0)^T$, $i = 1, 2, \dots, n$ and $e_i = -e_{i-n}$, $i = n+1, \dots, 2n$.
2. In minimization of $f(x)$ and $FF(x, x_k^*, P)$, a local optimization method needs to be selected firstly. In our algorithm, the SQP method is chosen.
3. In Step3, the value of δ needs to be selected accurately. In our algorithm, δ is selected to guarantee $\|\nabla \mathcal{F}(x, x_k^*, P)\|$ is greater than a threshold (e.g.,take the threshold as 10^{-2}).
4. Step 4 means that the local minimizer $x^{'}$ of $\mathcal{F}(x, x_k^*, P)$ satisfies $x^{'} \in S_2$ which is a better local minimizer of $f(x)$.
5. We notice that the Assumption 3 is necessary for analyzing the properties of $\mathcal{F}(x, x_k^*, P)$. In implementation of the FFM, this assumption is not necessary.

4.2 Numerical experiments

In this section, the proposed algorithm is tested on the following ten benchmark problems which are usually used as test functions.

Problem 1 (Two-dimensional function)

$$\begin{aligned}&\min\ f(x) = [1 - 2x_2 + c\sin(4\pi x_2) - x_1]^2 + [x_2 - 0.5\sin(2\pi x_1)]^2,\\ &s.t.\ \ 0 \le x_1 \le 10, \quad -10 \le x_2 \le 0,\end{aligned}$$

where $c = 0.2, 0.5, 0.05$. The global minimum function value $f(x^*) = 0$ for all c.

Problem 2 (Three-hump camel-back function)

$$\begin{aligned}&\min\ f(x) = 2x_1^2 - 1.05x_1^4 + \tfrac{1}{6}x_1^6 - x_1x_2 + x_2^2,\\ &s.t.\ \ -3 \le x_1 \le 3, \quad -3 \le x_2 \le 3.\end{aligned}$$

The global minimizer is $x^* = (0, 0)^T$.

Problem 3 (Six-hump camel-back function)

$$\begin{aligned}&\min\ f(x) = 4x_1^2 - 2.1x_1^4 + \tfrac{1}{3}x_1^6 - x_1x_2 - 4x_2^2 + 4x_2^4,\\ &s.t.\ \ -3 \le x_1 \le 3, \quad -3 \le x_2 \le 3.\end{aligned}$$

The global minimizer is $x^* = (-0.0898, -0.7127)^T$ and $x^* = (0.0898, 0.7127)^T$.

Problem 4 (Treccani function)

$$\begin{aligned}&\min\ f(x) = x_1^4 + 4x_1^3 + 4x_1^2 + x_2^2,\\ &s.t.\ \ -3 \le x_1 \le 3, \quad -3 \le x_2 \le 3.\end{aligned}$$

The global minimizers are $x^* = (0, 0)^T$ and $x^* = (-2, 0)^T$.

Problem 5 (Goldstein and Price function)

$$\begin{aligned}&\min\ f(x)=g(x)h(x),\\ &s.t.\ -3\le x_1\le 3,\quad -3\le x_2\le 3,\end{aligned}$$

where $g(x)=1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)$, and $h(x)=30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)$. The global minimizers is $x^*=(0,-1)^T$.

Problem 6 (Rosenbrock function)

$$\begin{aligned}&\min\ f(x)=100(x_1^2-x_2)^2+(x_1-1)^2,\\ &s.t.\ -10\le x_1\le 10,\quad -10\le x_2\le 10,\end{aligned}$$

The global minimizer is $(1,1)^T$.

Problem 7 (Branin function)

$$\begin{aligned}&\min\ f(x)=\left(x_2-1.275\tfrac{x_1^2}{\pi^2}+5\tfrac{x_1}{\pi}-6\right)^2+10\left(1-\tfrac{0.125}{\pi}\right)\cos x_1+10,\\ &s.t.\ -5\le x_1\le 10,\quad 0\le x_2\le 15,\end{aligned}$$

The global minimizer are $(-3.142, 12.275)^T$, $(3.142, 2.275)^T$ and $(9.425, 2.475)^T$.

Problem 8 (Two-dimensional Shubert function)

$$\begin{aligned}&\min\ f(x)=\left\{\sum_{i=1}^{5} i\cos[(i+1)x_1]+i\right\}\left\{\sum_{i=1}^{5} i\cos[(i+1)x_2]+i\right\},\\ &s.t.\ -10\le x_1\le 10,\quad -10\le x_2\le 10.\end{aligned}$$

This problem has 760 minimizers in total. The global minimum value is $f(x^*)=-186.7309$.

Problem 9 (Shekel's function)

$$\begin{aligned}&\min\ f(x)=-\sum_{i=1}^{5}\left[\sum_{j=1}^{4}(x_j-a_{i,j})^2+c_i\right]^{-1},\\ &s.t.\ 0\le x_j\le 10,\quad j=1,2,3,4,\end{aligned}$$

where the coefficients $a_{i,j}$,c_i,$i=1,2,3,4,5$, $j=1,2,3,4$ are given in Table 1.

All local minimizers are approximately equal to $(a_{i,1},a_{i,2},a_{i,3},a_{i,4})^T$ with function value $-1/c_i$,$i=1,2,3,4,5$.

Problem 10 (n-dimensional Sine-square II function)

$$\begin{aligned}&\min\ f(x)=\tfrac{\pi}{n}\left\{10\sin^2\pi x_1+g(x)+(x_n-1)^2\right\},\\ &s.t.\ -10\le x_i\le 10,\quad i=1,2,\ldots,n,\end{aligned}$$

Table 1 The coefficient for Problem 9

i	$a_{i,1}$	$a_{i,2}$	$a_{i,3}$	$a_{i,4}$	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	0.3
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.5

where $g(x) = \sum_{i=1}^{n-1}[(x_i - 1)^2(1 + 10\sin^2 \pi x_{i+1})]$. The global minimizer of this problem is $x^* = (1, \dots, 1)$ for all n.

The proposed algorithm is executed on the above 10 test problems and the performance is compared with that of the algorithms in [18–20]. In the related Tables, the following symbols are adopted.

Prob: the problem number 1–10;
ObjEval: the number of the total function evaluations of the original objective;
FilledEval: the number of the total function evaluations of the filled function;
x_0: an initial point;

The initial value of P is taken as 1 and Ubp is taken as 10^6 for all problems;

NFFM: the algorithm proposed in this paper;
HFA: the algorithm proposed in [18].
CFA: the algorithm proposed in [19].
NFA: the algorithm proposed in [20].
Time: the CPU time for the algorithm to stop. In general, CPU-time is dependent on the tuning of parameter P and the performance of the computer.

The choice of the tuning parameter P is given by the filled function algorithm. The proposed algorithm is programmed in Matlab 2014a for working on the Windows 10 system with Intel(R) Core(TM)i5-3340M CPU and 4G RAM.

The proposed algorithm is executed on the above 10 test problems, for Problems 1–5, 8–10, the performance is compared with that of the algorithm in [20]. The minimizers obtained by the above two algorithms are listed in Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, and 16.

From Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, and 17, We can see that for all test problems, both the proposed algorithm (NFFM) and the algorithm proposed in [20] (NFA) can find the global optimal solutions for 8 test problems. However, the NFFM used no more iterations to find the global optimal solution. In particular, for test Problem 1 in all three cases, and Problems 8 and 10, the NFFM used fewer iterations to find the optimal solutions than the NFA. For example, for Problem 1 in the case that $c = 0.05$ and $x_0 = (10, -10)$ shown in Table 4, the NFFM only needs 7 iterations to find the global optimal solution, but the NFA needs 8 iterations. For Problem 6, there are 760 local minimizers, and the NFFM only needs 4 iterations to find the global optimal solutions, but the NFA needs 6 iterations. Thus, the proposed algorithm is more efficient than the algorithm in [20].

Table 2 Results for Problem 1 with $c = 0.2$ and $x_0 = (6, -2)^T$

NFFM			NFA		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(5.7221, -1.8806)^T$	2.5070	1	$(5.7221, -1.8806)^T$	2.5070
2	$(3.7387, -1.2649)^T$	0.6165	1	$(4.7387, -1.7417)^T$	1.6212
3	$(1.0000, -0.0000)^T$	2.8229e−010	10	$(4.7096, -1.3985)^T$	1.3566
4				$(3.7387, -1.2649)^T$	0.61647
5				$(2.7380, -0.78836)^T$	0.088673
6				$(1.8784, -0.34585)^T$	0

Table 3 Results for Problem 1 with $c = 0.5$ and $x_0 = (0, 0)^T$

NFFM			NFA		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(0.0420, -0.0948)^T$	0.5175	1	$(0.042023, -0.094772)^T$	0.51745
2	$(1.0000, 0)^T$	1.5257e−009	10	$(0.99991, -1.2524e{-4})^T$	2.2389e−7
3				$(1.0000, -2.2205e{-14})^T$	0

Table 4 Results for Problem 1 with $c = 0.05$ and $x_0 = (10, -10)^T$

NFFM			NFA		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(8.7299, -3.2965)^T$	9.0733	1	$(8.7299, -3.2965)^T$	9.0733
2	$(7.7280, -2.8347)^T$	6.5031	1	$(7.7280, -2.8347)^T$	6.5031
3	$(6.7248, -2.3724)^T$	4.3943	1	$(6.7248, -2.3724)^T$	4.3943
4	$(5.7198, -1.9162)^T$	2.7434	1	$1(5.7198, -1.9162)^T$	2.7434
5	$(4.7129, -1.4890)^T$	1.5351	1	$(4.7129, -1.4891)^T$	1.5351
6	$(2.7300, -0.7934)^T$	0.1022	1	$(3.7305, -1.2306)^T$	0.61844
7	$(1.8513, -0.4021)^T$	4.3885e−011	10	$(2.7300, -0.79341)^T$	0.10216
8				$(1.8513, -0.40209)^T$	0

Table 5 Results for Problem 2 with initial point $(-2, -1)^T$

NFFM			NFA		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(-1.7476, -0.8738)^T$	0.2986	1	$(-1.7476, -0.87378)^T$	0.29864
2	$1.0e{-004} \times (0.8343, 0.9603)^T$	1.5130e−008	10	$(0, 0)^T$	0

Table 6 Results for Problem 2 with initial point $(2, 1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(1.7476, 0.8738)^T$	0.2986	1	$(1.7476, 0.87378)^T$	0.29864
2	$1.0e{-}004 \times (0.8343, 0.9603)^T$	1.5130e−008	10	$(0, 0)^T$	0

Table 7 Results for Problem 3 with initial point $(-2, 1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(-1.6071, 0.5687)^T$	2.1043	1	$(-1.6071, 0.56865)^T$	2.1043
2	$(0.0898, 0.7127)^T$	−1.0316	1	$(0.089842, 0.71266)^T$	−1.0316

Table 8 Results for Problem 3 with initial point $(2, -1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(1.6071, -0.5687)^T$	2.1043	1	$(1.6071, -0.56865)^T$	2.1043
2	$(-0.0898, -0.7127)^T$	−1.0316	1	$(-0.089842, -0.71266)^T$	−1.0316

Table 9 Results for Problem 3 with initial point $(-2, -1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(-1.7036, -0.79608)^T$	−0.2155	1	$(1.7036, -0.79608)^T$	−0.21546
2	$(-0.0899, -0.7127)^T$	−1.0316	10	$(-0.089842, -0.71266)^T$	−1.0316

Table 10 Results for Problem 4 with initial point $(-1, 0)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(-1.0000, 0)^T$	1.0000	1	$(-1.0000, 0)^T$	1.0000
2	$(0, 0)^T$	0	10	$(0, 0)^T$	0

Table 11 Results for Problem 5 with initial point $(-1, -1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(-0.6000, -0.4000)^T$	30.0000	1	$(-0.60000, -0.40000)^T$	30.000
2	$(0, -1.0000)^T$	3.0000	1	$(0, -1.0000)^T$	3.0000

Table 12 Results for Problem 8 with initial point $(1, 1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(1.08651.0865)T$	0	1	$(1.0865, 1.0865)^T$	2.8841e−17
2	$(3.2800, 4.8581)^T$	− 46.511	1	$(1.3200, 1.8703, \text{e}{-12})^T$	− 13.052
3	$(4.2760, 4.8581)^T$	− 79.411	1	$(1.3200, 4.8581)^T$	− 37.681
4	$(5.4892, 4.8581)^T$	− 186.7309	1	$(3.2800, 4.8581)^T$	− 46.511
5				$(4.2760, 4.8581)^T$	− 79.411
6				$(5.4892, 4.8581)^T$	− 186.73

Table 13 Results for Problem 9 with initial point $(1, 1, 1, 1)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(1.0001, 1.0002, 1.0001, 1.0002)^T$	− 5.0552	1	$(1.0001, 1.0002, 1.0001, 1.0002)^T$	− 5.0552
2	$(4.0000, 4.0001, 4.0000, 4.0001)^T$	− 10.1532	1	$(4.0000, 4.0001, 4.0000, 4.0001)^T$	− 10.153

Table 14 Results for Problem 9 with initial point $(6, 6, 6, 6)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(5.9987, 6.0003, 5.9987, 6.0003)^T$	− 2.6829	1	$(5.9987, 6.0003, 5.9987, 6.0003)^T$	− 2.6829
2	$(7.9996, 7.9996, 7.9995, 7.9996)^T$	− 5.1008	1	$(7.9996, 7.9996, 7.9996, 7.9996)^T$	− 5.1008
3	$(4.0000, 4.0001, 4.0000, 4.0001)^T$	− 10.1532	1	$(4.0000, 4.0001, 4.0000, 4.0001)^T$	− 10.153

Table 15 Results for Problem 10 with $n = 7$ and initial point $(2, 2, 2, 2, 2, 2, 2)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(1.9899, 1.9897, 1.9896, 1.9896 1.9896, 1.9896, 1.9898)^T$	2.1767	1	$(1.9899, 1.9897, 1.9896, 1.9896 1.9896, 1.9896, 1.9898)^T$	2.1767
2	$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	7.4665e-011	1	$(1.0000, 1.0000, 1.0000, 1.0000 1.0000, 1.0000, 1.0000)^T$	0

Table 16 Results for Problem 10 with $n = 10$ and initial point $(6, 6, 6, 6, 6, 6, 6, 6, 6, 6)^T$

NFFM			*NFA*		
k	x_k^*	f_k^*	P	x_k^*	f_k^*
1	$(5.9490, 5.9979, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	78.4316	1	$(5.9490, 5.9979, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	78.432
2	$(-0.97956, 5.9871, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	71.8841	1	$(-1.9696, 5.9943, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	73.450
3	$(1.9900, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 6.0000, 6.0000, 6.0000)^T$	26.7135	1	$(-0.97956, 5.9871, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	71.884
4	$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	0	1	$(0.012709, 5.9476, 5.9979, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980, 5.9980)^T$	70.890
5				$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	0

Table 17 The results obtained by *NFFM* and *NFA* on Problems 1–5, 8–10

Prob	x_0	*NFFM*		*NFA*	
		Time	*ObjEval* + *FilledEval*	*Time*	*ObjEval* + *FilledEval*
1 ($c = 0.2$)	$(6, -2)^T$	2.653418	60 + 71	5.837520	222 + 486
1 ($c = 0.5$)	$(0, 0)^T$	1.825776	30 + 58	3.103819	63 + 283
1 ($c = 0.05$)	$(10, -10)^T$	2.840004	42 + 194	4.057149	217 + 469
2	$(-2, -1)^T$	2.182257	27 + 22	4.400483	57 + 265
2	$(2, 1)^T$	2.762833	27 + 22	4.983860	57 + 265
3	$(-2, 1)^T$	1.884487	27 + 33	3.963400	51 + 237
3	$(2, -1)^T$	1.873671	27 + 33	3.842301	51 + 237
3	$(-2, -1)^T$	1.979559	39 + 53	3.950833	53 + 252
4	$(-1, 0)^T$	1.268946	3 + 31	3.998480	36 + 243
5	$(-1, -1)^T$	2.929603	42 + 79	4.796678	87 + 295
8	$(1, 1)^T$	2.706970	24 + 60	4.449131	144 + 240
9	See Table 13	6.416832	20 + 140	9.545932	85 + 380
9	See Table 14	7.051653	20 + 231	9.820829	135 + 571
10 $n = 7$	See Table 15	12.404419	56 + 634	22.592902	232 + 1090
10 $n = 10$	See Table 16	17.074316	88 + 795	25.662697	392 + 862

Table 18 The average total number of the evaluations of the objective function and the filled function

Prob	The performance of *NFFA*	The performance of *HFA*	The performance of *CFA*
2	57×10	218×4	216×4
3	119.6×10	103.5×4	99.25×4
4	41.1×10	74×4	74×4
5	142×10	207×4	195×4
6	164.2×10	392×4	392×4
7	27.3×10	80.75×4	80.75×4
8	53.3×10	168×3	1563×3
9	272.4×10	9805.25×4	3070.25×4

To evaluate its effectiveness, the new filled function algorithm was used to find the global minimizers of the Problems 2–8 and 9. In all tests, ten initial points were randomly generated. The experimental results are presented in Table 18, in which the data are the average total number of evaluations of function $f(x)$ and $FF(x, x_k^*, P)$ when the global minimizer was found. (The number of directions is taken as $2n$, in Problem 6, and Ω is taken as $[-10, 10] \times [-10, 10]$.)

The performance of the HFA and CFA can be seen in reference [19]. In [19], four initial points were taken randomly, then the average number of evaluations of the objective function can be computed excluding unsuccessful tests. The results are listed in Table 18.

The means of the dates in Table 18 are as follows: When used HFA to solve problem 2, the number of evaluations of the objective function are 345, 109, 340, 78 respectively, then the date in table is denoted as $(345+109+340+78)/4 \times 4$; When used HFA to solve Problem 8, the number of evaluations of the objective function are –, 259, 131, 114 (– denote the test is unsuccessful) respectively, then the date in table is denoted as $(259 + 131 + 114)/3 \times 3$

From Table 18, the total number of evaluations of function $f(x)$ and $FF(x, x_k^*, P)$ of the new FFM is less than the ones HFA and CFA except for Problem 3. Therefore, the new algorithm is better than HFA and CFA in performance.

5 Concluding remarks

This paper presented a new filled function. It excluded some limitations of conventional filled functions and had the same local minimizer which is better than the current local minimizer with the original problem. Therefore, the computational cost would be highly reduced since we avoid minimizing the objective function and the filled function cyclically. We compared our proposed method with existing ones on several testing optimization problems. The experimental results showed the effectiveness of our method on practical problems.

Acknowledgements This work was supported by the National Nature Science Foundation of China (Nos. 11161001, 61402350, 61375121), the Research Foundation of Jinling Institute of Technology (Nos. jit-b-

201314, jit-n-201309, jit-rcyj-201505) and the Funds for Nanjing Creative Team of Swarm Computing & Smart Software Led by Prof. S. Su.

References

1. Ge, R.: A filled function method for finding a global minimizer of a function of several variables. Math. Program. **46**, 191–204 (1990)
2. Ge, R.: The theory of filled function method for finding global minimizers of nonlinearly constrained minimization problems. J. Comput. Math. **5**, 1–9 (1987)
3. Levy, A.V., Montalvo, A.: The tunneling algorithm for the global minimization of functions. SIAM J. Sci. Stat. Comput. **6**, 15–29 (1985)
4. Wang, Y., Fang, W., Wu, T.: A cut-peak function method for global optimization. J. Comput. Appl. Math. **230**, 135–142 (2009)
5. Branin, F.H.: Widely convergent method for finding multiple solutions of simultaneous nonlinear equations. IBM J. Res. Dev. **16**, 504–522 (1972)
6. Snyman, J.A., Fatti, L.P.: A multi-start global minimization algorithm with dynamic search trajectories. J. Optim. Theory Appl. **54**, 121–141 (1987)
7. Basso, P.: Iterative methods for the localization of the global maximum. SIAM J. Numer. Anal. **19**, 781–792 (1982)
8. Mladineo, R.H.: An algorithm for finding the global maximum of a multimodal, multivariate function. Math. Program. **34**, 188–200 (1986)
9. Lin, H., Gao, Y., Wang, Y.: A continuously differentiable filled function method for global optimization. Numer. Algorithms **66**, 511–523 (2014)
10. Leung, Y.W., Wang, Y.: Multiobjective programming using uniform design and genetic algorithm. IEEE Trans. Syst. Man Cybern. Part C **30**, 293–304 (2000)
11. Dang, C., Ma, W., Liang, J.: A deterministic annealing algorithm for approximating a solution of the min-bisection problem. Neural Netw. **22**, 58–66 (2009)
12. Li, X., Yao, X.: Cooperatively coevolving particle swarms for large scale optimization. IEEE Trans. Evolut. Comput. **16**, 210–224 (2012)
13. Mininno, E., Neri, F., Cupertino, F., Naso, D.: Compact differential evolution. IEEE Trans. Evolut. Comput. **15**, 32–54 (2011)
14. Woon, S.F., Rehbock, V.: A critical review of discrete filled function methods in solving nonlinear discrete optimization problems. Appl. Math. Comput. **217**, 25–41 (2010)
15. Ling, A.F., Xu, C.X., Xu, F.M.: A discrete filled function algorithm for approximate global solutions of max-cut problems. J. Comput. Appl. Math. **220**, 643–660 (2008)
16. Gao, Y., Yang, Y., You, M.: A new filled function method for global optimization. Appl. Math. Comput. **268**, 685–695 (2015)
17. El-Gindy, T.M., Salim, M.S., Ahmed, A.I.: A new filled function method applied to unconstrained global optimization. Appl. Math. Comput. **273**, 1246–1256 (2016)
18. Liu, X.: Finding global minima with a computable filled function. J. Glob. Optim. **19**, 151–161 (2001)
19. Liu, X.: A class of continuously differentiable filled functions for global optimization. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **38**, 38–47 (2008)
20. Zhang, L.S., Ng, C.K., Li, D., Tian, W.W.: A new filled function method for global optimization. J. Glob. Optim. **28**, 17–43 (2004)