Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 14, No. 4, pp. 567–580, 2018

# A PARAMETER FREE FILLED FUNCTION METHOD FOR GLOBAL OPTIMIZATION[*]

Haiyan Liu, Yuping Wang[†], Xiao-Zhi Gao, Chuangyin Dang and Zhiqing Meng

**Abstract:** The filled function method is an effective approach for global optimization. However, the existing filled functions have the disadvantages of discontinuity, non-differentiability and sensitivity to parameters, which limit their efficiency. In this paper, a parameter free filled function is proposed with two advantages: 1) continuity and differentiability, and 2) no need for adjusting parameters. Then a new filled function algorithm is designed based on the proposed filled function for global optimization problems. Furthermore, to make the filled function algorithm more effective, a new search method with restarting mechanism is designed. Finally, numerical experiments are conducted and comparisons are made with two representative filled function algorithms. The results show the proposed algorithm can find higher quality solutions and has faster convergence speed.

**Key words:** *filled function, parameter free, local search, global optimization, multi-modal function*

**Mathematics Subject Classification:** *65K05, 65K10*

## 1 Introduction

Global optimization is widely used in many fields such as economics and engineering [4, 3, 14]. Due to the existence of multiple local optimal solutions, the existing algorithms may often be trapped into a local optimal solution. So, a major problem is how to escape from the current local optimal solution and find better ones successively until a global optimal solution is found. The filled function method (briefly, FFM) proposed by Ge [11, 13, 12] is an effective way to tackle this problem and has been applied in many areas such as entomology and switched systems [29, 2, 1, 22]. It aims at finding the global optimal solution of a continuously differentiable function $F(x)$ on $R^n$. For convenience, we will use the following symbols in this paper:

. $n$: The dimension of the problem.

. $x^*$: The global optimal solution of the problem.

. $k$: The iteration counter.

. $x_k^*$: The local minimum of the problem in the $k$-th iteration. We assume the number of local minima is finite.

[*]This work was supported by NSFC (61872281) and Shaanxi Natural Science Foundation (BD11016030002).

[†]Corresponding author.

. $B_k^*$: The basin of the objective function at an isolated local minimum $x_k^*$.

. $F(x)$: The continuous differentiable objective function at $x \in \Omega \subset R^n$, where $\Omega$ is a closed bounded domain.

We first introduce the definition [11] of the filled function and the related concepts as follows:

**Definition 1.1** ([11])**.** The basin $B_k^*$ of $F(x)$ at an isolated minimum $x_k^*$ is a connected domain which contains $x_k^*$, and in this domain the steepest descent trajectory of $F(x)$ will converge to $x_k^*$ starting from any initial point, but outside the basin the steepest descent trajectory of $F(x)$ does not converge to $x_k^*$.

A basin $B_1^*$ at $x_1^*$ is lower (or higher) than basin $B_2^*$ at $x_2^*$ iff

$$F(x_1^*) \leq (or >) F(x_2^*)$$

**Definition 1.2** ([11])**.** A function $P(x, x_1^*)$ is called a filled function of $F(x)$ at a local minimum $x_1^*$ if it satisfies the following conditions:

(1) $x_1^*$ is a strictly local maximum of $P(x, x_1^*)$, and the whole basin $B_1^*$ of $F(x)$ becomes a part of a hill of $P(x, x_1^*)$.

(2) $P(x, x_1^*)$ has no minima or stable points in any basin of $F(x)$ higher than $B_1^*$.

(3) If $F(x)$ has a lower basin than $B_1^*$, then there is a point $\bar{x}$ in such a basin that minimizes $P(x, x_1^*)$ on the line through x and $x_1^*$.

It can be seen from condition (1) that it is easy to escape from the local minimum $x_1^*$ by minimizing the filled function$P(x, x_1^*)$ since $x_1^*$ is a local maximum of $P(x, x_1^*)$. Condition (2) ensures the FFM will not find a worse local minimum by minimizing the filled function $P(x, x_1^*)$, and condition (3) means that if a local minimum better than $x_1^*$ exists, FFM is able to enter a lower basin of $F(x)$. Thus it is possible to get a better local minimum by minimizing the filled function $P(x, x_1^*)$ along a line from $x$ to $x_1^*$.

The main idea of FFM is as follows: from an initial point $x_1$ we can get a local minimum $x_1^*$ of the original function $F(x)$ by minimizing $F(x)$, and then construct a filled function $P(x, x_1^*)$ at the local minimum $x_1^*$. Afterwards, we minimize the filled function $P(x, x_1^*)$ to get a local minimum $x'$ from an initial point $y_1$ near $x_1^*$. From condition (1), we can see that $y_1$ is not a peak of $P(x, x_1^*)$ and thus we can find a local minimum $x'$ of $P(x, x_1^*)$ by any descent direction method such as the steepest descent method. By condition (2), we can see that $x'$ will not be in a basin higher than $B_1^*$. Instead, this local minimum $x'$ will be in a lower basin than $B_1^*$ if such basin exists by condition (3). Furthermore, by minimizing $F(x)$ from the initial point $x'$ we can get a better local minimum $x_2^*$ of $F(x)$ by condition (3). Finally, we can get a global minimum by repeating this process until no better minimum of $F(x)$ can be found.

In fact, FFM gives the insight of gradually moving from one local minimum to better ones until a global minimum is found. However, condition 3 in definition 1.2 [11] is not easy to verify. To make the conditions of a filled function be easily verified, in this paper, we give a revised definition of the filled function and design a new filled function with the following advantages: 1) it is continuous differentiable, and 2) it has no parameter to tune.

The remainder of this paper is organized as follows: Section 2 gives a brief overview of filled functions. In Section 3, a revised definition of filled function is given and a novel filled function is designed. Then, the properties of the filled function are discussed. In Section 4, a new filled function algorithm is proposed. Numerical experiments are carried out and comparisons are made in Section 5. Section 6 makes the conclusions.

## $\boxed{2}$ A Brief Overview of Filled Functions

In this section we give a brief overview of filled functions.
The first filled function proposed by Ge [11] is

$$P(x, r, \rho) = \frac{1}{r + F(x)} exp(-\frac{\|x - x_1^*\|^2}{\rho^2}) \tag{2.1}$$

where $r$ and $\rho$ are two parameters which need to be carefully chosen to satisfy the aforementioned three conditions of a filled function. However, it is difficult to choose proper values of these parameters for this filled function. Also, $P(x, r, \rho)$ will approach to zero too fast when $\|x - x_1^*\|^2$ becomes larger and larger because of the exponential term, and its values change slowly (change near zero and are very close to zero). Thus, it is very hard to distinguish these changes and the algorithm may easily trap in a false minimum even a saddle point.

To avoid such effect of the term $exp(-\|x - x_1^*\|/\rho^2)$ of function P and overcome the difficulty of the mutual adjustment of two parameters, another filled function called function Q was proposed [13]. The function Q contains only one parameter and is defined by

$$Q(x, A) = -[F(x) - F(x_1^*)]exp[-A\|x - x_1^*\|^2] \tag{2.2}$$

$$\tilde{Q}(x, A) = -[F(x) - F(x_1^*)]exp[-A\|x - x_1^*\|] \tag{2.3}$$

This function also contains the exponential term and has the similar drawback as the filled function proposed by Ge [11]. Afterwards, many filled functions with one parameter were proposed (e.g., [21, 19, 18, 27, 15] ). A typical one called function H was proposed in [21] as follows:

$$H(x) = \frac{1}{ln(1 + F(x) - F(x_1^*))} - a\|x - x_1^*\|^2 \tag{2.4}$$

However, $H(x)$ has no definition at point $x$ where $F(x) = F(x_1^*)$ and the logistic term can cause the ill-conditioning problem [7]. To overcome this drawback, Liu [19] proposed function M without the exponential or logistic term:

$$M(x, a) = \frac{1}{(F(x) - F(x_1^*))^{1/m}} - a\|x - x_1^*\|^2 \tag{2.5}$$

but this function has two parameters and adjusting two parameters becomes more difficult than adjusting one parameter. To reduce the possibility of introducing additional local minima by non-differential term of filled functions, Liu first proposed a class of continuous differentiable filled functions without exponential and logistic terms [20]:

$$C(x, a) = -u[F(x) - F(x_1)]w^a(\|x - x_1^*\|^p) \tag{2.6}$$

where $u$ and $w$ are two real functions defined in $R^1$, and twice continuously differentiable in their domains satisfying the following conditions:

$$\begin{aligned} u(0) = 0 \quad w(0) = 1/a > 0 \\ u'(t) > 0 \quad w'(t) > 0, \quad \forall t \in [0, \infty) \\ \lim_{t \to 0} \frac{u(t)w'(t)}{u'(t)w(t)} = 0 \end{aligned}$$

To make the definition more clear and exact, Zhang [28] provided a revised definition of filled functions and gave some useful information to construct filled functions. Based on the revised definition, he proposed two filled functions with one parameter and two parameters, respectively. The two-parameter filled function is defined by:

$$P(x, x_1^*, r, a) = \varphi(r + F(x)) - a\|x - x_1^*\| \tag{2.7}$$

where $a$ and $r$ are the two parameters to tune with $a > 0$. Also, $\varphi(t)$ needs to satisfy the following conditions:
(1) $\varphi(t) > 0$ is continuously differentiable, $t \in (0, +\infty)$.
(2)$\varphi'(t) < 0, \varphi''(t) > 0, t \in (0, +\infty)$.
The one parameter filled function is as follows:

$$P(x, x_1^*, \mu) = -\|x - x_1^*\| + \mu\{max[0, F(x) - F(x_1^*)]\} + 1/\mu\{min[0, F(x) - F(x_1^*)]\}^3 \tag{2.8}$$

Using this definition, authors in [7] designed the following relatively simple filled function

$$F(x, x_1^*, P) = -[F(x) - F(x_1^*) + P]\|x - x_1^*\|^2 \tag{2.9}$$

with one parameter $P$. Authors in [5, 6] proposed a bounded filled function with two parameters which can not cause the computation overflow as follows:

$$P(x, x_1^*) = \frac{1}{1 + \|x - x_1^*\|^2} + g(F(x) - F(x_1^*)) \tag{2.10}$$

$$g(t) = \begin{cases} 0 & t \geq 0 \\ r.arctan(t^\rho) & t < 0 \end{cases}$$

Recently, authors in [23, 17] proposed two similar filled functions

$$F(x, x_1^*, r, q) = \frac{1}{1 + \|x - x_1^*\|} arctan(|\frac{F(x) - F(x_1^*) + r|}{|F(x) - F(x_1^*)| + q}) \tag{2.11}$$

and

$$P_{x_1^*, A, \epsilon}(x) = min\{-(\|x - x_1^*\|)^2 - \frac{F^2(x)}{A}, min\{A(F(x) - F(x_1^*)), -\epsilon\}(F(x) - F(x_1^*))^2\} \tag{2.12}$$

with two parameters, respectively, where $q$, $r$, $A$ and $\epsilon$ are parameters.

All aforementioned filled functions have one or two parameters and these parameters are not easy to tune. To overcome this shortcoming, some filled functions without parameters are proposed [9, 16]:

$$P(x, x_1^*) = -sign(F(x) - F(x_1^*))\|x - x_1^*\|^2 \tag{2.13}$$

$$sign(t) = \begin{cases} 1 & t \geq 0 \\ -t & t < 0 \end{cases}$$

$$P(x, x_1^*) = -sign(F(x) - F(x_1^*))arctan(\|x - x_1^*\|^2) \tag{2.14}$$

$$sign(t) = \begin{cases} 1 & t \geq 0 \\ -1 & t < 0 \end{cases}$$

However, because function $sign(t)$ is not continuous, these two filled functions are not continuous and differentiable, and may result in extra local minima, which means extra search works needed in looking for a global minimum. In this paper, a new filled function without any parameter is proposed and it is continuous and differentiable. Based on this filled function, a new filled function algorithm is designed.

## 3 A New Filled Function and its Properties

The filled function method is an effective way to solve multi-modal unconstrained optimization problems. However, the definition of the filled function proposed by Ge is not clear and is confusing. To make the meaning of the definition clearer, some authors have given a few revised definitions of filled functions [10, 24, 28, 16]. In the definition given in literature [16], sub-gradients $\partial P(x, x_k^*)$ were used in condition (2) to make the definition clearer. However, according to this definition, the designed filled function may be not differentiable and may introduce extra local minima. This may result in more difficulty for an algorithm to look for a global minimum. To overcome this disadvantage, in this paper, we replace sub-gradients in condition (2) by gradients to ensure the designed filled function to be differentiable. The conditions in this new definition are stronger and more difficult to satisfy. Thus it is more difficult for one to construct a filled function based on the new definition. But as soon as such a filled function is designed, it is much easier for an algorithm to look for a global minimum. The new definition of the filled function is as follows:

**Definition 3.1.** A function $P(x, x_k^*)$ is called a filled function of $F(x)$ at a strictly local minimum $x_k^*$ if it satisfies the following conditions:
(1) $x_k^*$ is a strictly local maximum of $P(x, x_k^*)$.
(2) For any $x \in \Omega_1, \nabla P(x, x_k^*) \neq 0$, where $\Omega_1 = \{x \in \Omega | F(x) \geq F(x_k^*), x \neq x_k^*\}$.
(3) If $\Omega_2 = \{x \in \Omega | F(x) < F(x_k^*)\}$ is not empty, then there exists a stationary point $x' \in \Omega_2$ that fulfils $\nabla P(x', x_k^*) = 0$ and $x'$ is a local minimum of $P(x, x_k^*)$.

As mentioned above, the conditions (2) and (3) in definition 3.1 are stronger than those in definition [16], but they are convenient for readers to understand because many readers especially the engineers may not be familiar to sub-gradients.

According to the revised definition, we propose a new filled function without any parameter as follows:

$$P(x, x_k^*) = -\|x - x_k^*\|^2 g(F(x) - F(x_k^*))$$
$$g(t) = \begin{cases} 1 & t \geq 0 \\ ln(1 + t^2) + 1 & t < 0 \end{cases} \tag{3.1}$$

In the following, we suppose that $F(x)$ is a continuous differentiable function on $R^n$. We will first prove that the proposed filled function is continuous differentiable and then further prove that it satisfies the three conditions in definition 3.1.

**Theorem 3.2.** *The proposed filled function $P(x, x_k^*)$ in (3.1) is continuous differentiable.*

*Proof.* Note that the only point at which the filled function may be not continuous and differentiable is $t = 0$ for $g(t)$. So we only need to prove that $g(t)$ is continuous differentiable at $t = 0$.

Since $\lim\limits_{t\to 0^+} g(t) = \lim\limits_{t\to 0^-} g(t) = 1$, so $P(x, x_k^*) = -\|x - x_k^*\|^2 g(F(x) - F(x_k^*))$ is continuous. Now we further prove that $P(x, x_k^*)$ is differentiable. Since

$$g'_+(0) = \lim_{t\to 0^+} \frac{g(t) - g(0)}{t - 0} = \lim_{t\to 0^+} \frac{1 - 1}{t} = 0$$

and

$$g'_-(0) = \lim_{t\to 0^-} \frac{g(t) - g(0)}{t - 0} = \frac{\ln(1 + t^2)}{t} = 0.$$

Thus, $g'_+(0) = g'_-(0) = 0$, and $g'(0) = 0$. Note that $g'(t) = 0$ when $t > 0$ and $g'(t) = \frac{2t}{1+t^2}$ when $t < 0$, and $\lim_{t\to 0^-} g'(t) = \lim_{t\to 0^-} \frac{2t}{1+t^2} = 0$, thus $g'(t)$ is continuous at $t = 0$. □

**Theorem 3.3.** *Suppose $x_k^*$ is a local minimum of $F(x)$ and $P(x, x_k^*)$ is a filled function at $x_k^*$, then $x_k^*$ is a strictly local maximum of $P(x, x_k^*)$.*

*Proof.* Suppose $B_k^*$ is the basin corresponding to $x_k^*$. Since $x_k^*$ is a local minimum of $F(x)$, then $\forall x \in B_k^*, x \neq x_k^*$, we have $F(x) > F(x_k^*)$.
So $P(x, x_k^*) = -\|x - x_k^*\|^2 < 0 = P(x_k^*, x_k^*)$. Thus, $x_k^*$ is a strictly local maximum of $P(x, x_k^*)$. □

**Theorem 3.4.** *For any $x \in \Omega_1, \nabla P(x, x_k^*) \neq 0$, where $\Omega_1 = \{x \in \Omega | F(x) \geq F(x_k^*), x \neq x_k^*\}$.*

*Proof.* For any $x \in \Omega_1$ with $x \neq x_k^*$, we have $F(x) \geq F(x_k^*)$, and
$\nabla P(x, x_k^*) = -2(x - x_k^*) \neq 0$. □

**Theorem 3.5.** *Suppose that $F(x)$ has a finite number of global minima. If $\Omega_2 = \{x \in \Omega | F(x) < F(x_k^*)\}$ is not empty, then there exists $x' \in \Omega_2$ that fulfils $\nabla P(x', x_k^*) = 0$ and $x'$ is a local minimum of $P(x, x_k^*)$.*

*Proof.* Let $\bar{\Omega}_2$ denote the closure of $\Omega_2$. Since $F(x)$ is continuous, then $\bar{\Omega}_2$ is a closed set. Since $F(x)$ has a finite number of global minima, thus the level set $\bar{\Omega}_2$ of $F(x)$ at $x_k^*$ is bounded (see theorem 4.3.1 in [8]). Therefore, $\bar{\Omega}_2$ is a non-empty closed bounded set. Thus, $F(x)$ has the minimum on $\bar{\Omega}_2$ . Since $\Omega_2$ is not empty, this minimum must be in $\Omega_2$.

Since $P(x, x_k^*)$ is continuous differentiable on $R^n$, then it is continuous differentiable on this closed bounded set $\bar{\Omega}_2$. Thus it has a minimum $x'$ at this closed bounded set. Because $P(x, x_k^*)$ is differentiable at $x'$, then this minimum $x'$ must be a stationary point, that is, $\nabla P(x', x_k^*) = 0$.

Since $\Omega_2$ is not empty, then there exists a point $z \in \Omega_2$ such that $P(z, x_k^*) < 0$. Thus $P(x', x_k^*) \leq P(z, x_k^*) < 0$ and $x' \neq x_k^*$. Also, it can be seen from Theorem 3.4 that $x' \notin \Omega_1$. Therefore, $x' \in \Omega_2$. □

## 4 A New Filled Function Algorithm

In this section, a new search method with restarting mechanism is proposed and a new filled function algorithm is designed.

### 4.1 A new search method with restarting mechanism

The new search method adopts the idea of evolutionary algorithm with restarting mechanism. Our motivation is that the parallel searching mechanism of evolutionary algorithm may accelerate the convergent speed. In the search method, a new crossover and mutation operators are designed to generate new population within a neighbourhood of the current best point with the radius $= range = (UBound - LBound)/n * 0.001$, where $UBound$ and $LBound$ refer to the upper and lower bounds respectively and $n$ is the dimension of the problem. Also, a restarting mechanism is designed to make the search method more robust. When the search method fails to find a better point, the range is expanded ($range = range * 5$) to give the search more chances to escape from the local minimum. The search method will stop when it fails for 3 successive times (for problems with no more than 10 dimensions) or 5 successive times (for problems with more than 10 dimensions). In this way, it is hoped that the global minimum can be found. The pseudo-code of the search method is summarized in algorithm 4.1.

**Algorithm 1** pseudo-code of the search method with restarting mechanism

```
$restart = 0$;
$range = (UBound - LBound)/n * 0.001$;
while $restart < count$ do
    for $i = 1$ to $n$ do
        for $j = 1$ to $NP$ do
            $new\_x(j) = best\_x$;
            $new\_x(j)(i) = new\_x(i) + range * rand$;
        end for
        evaluate all points of the population and get the best point $new\_best$;
        if $F(new\_best) < F(best\_x)$ then
            return $new\_best$;
        end if
        for $j = 1$ to $NP$ do
            $new\_x(j) = best\_x$;
            $new\_x(j)(i) = new\_x(i) - range + range * rand$;
        end for
        evaluate all points of the population and get the best point $new\_best$;
        if $F(new\_best) < F(best\_x)$ then
            return $new\_best$;
        end if
    end for
    $restart = restart + 1$;
    $range = range * 5$;
end while
```

## 4.2 A new filled function algorithm

Based on the preparation of the above mentioned filled function and search method, a new filled function algorithm is designed as follows:

Step 1. (initialization). Use uniform population initialization method to distribute NP (we use NP=10 in the algorithm) points in $\Omega$, set $\epsilon = 1.0e-10$ as the stopping criterion. Set k=0 as the initial value of the iteration counter.

Step 2. Choose the best point *best_x* from the population and its function value is recorded as *best_f*. Let $x_0 = best\_x$.

Step 3. By minimizing $F(x)$ from an initial point $x_0$, we get a local minimum $x_k^*$ and its function value $F(x_k^*)$. If $F(x_k^*) > best\_f$, go to step 4, otherwise, go to step 5.

Step 4. Use Algorithm 4.1 as the search method. If the best point obtained by the search method is better than *best_x*, update *best_x* and let $x_0 = best\_x$, and then go to step 5, otherwise, go to step 6.

Step 5. If $F(x_k^*) - best\_f < -\epsilon$, go to step 6, otherwise, set $best\_f = F(x_k^*)$, and construct the filled function:

$$P(x, x_k^*) = -\|x - x_k^*\|^2 g(F(x) - F(x_k^*))$$
$$g(t) = \begin{cases} 1 & t \geq 0 \\ ln(1+t^2) + 1 & t < 0 \end{cases}$$

Make a tiny disturbance on $x_k^*$ and then minimize $P(x, x_k^*)$ using the disturbed point as the initial point. We get a local minimum $x_k^{'}$ of $P(x, x_k^*)$. Set $x_0 = x_k^{'}$, $k = k+1$, and go to step 3.

Step 6. (termination). Set $x^* = best\_x$ as the global minimum and the algorithm stops.

Note that we always record the best point and its function value we have found so far in each cycle and will update them if a better point appears. Also, the reason we use uniformly distributed points other than randomly given points as in some literatures is that, for an unknown problem, uniformly distributed points have more opportunities to get closer to the local or even global minimum and have a better diversity which can help the algorithm to obtain good results. For detailed information about uniform design, the readers can refer to [25, 26].

Figure 1 shows the optimization process of filled function methods. When a local minimum $x_1^*$ of the objective function $F(x)$ is found, the filled function $P(x, x_1^*)$ will be constructed on $x_1^*$. Then we will minimize $P(x, x_1^*)$ from the point $x_1^{'} = x_1^* + 0.01$ (disturbance of $x_1^*$), which is very easy because $x_1^*$ is a strictly maximum of $P(x, x_1^*)$. The minimization process of $P(x, x_1^*)$ will lead the search to a lower basin of $F(x)$ ensured by conditions 2 and 3 of the definition of the filled function. For example, the local minimum $x_2^{'}$ of $P(x, x_1^*)$ is in a lower basin of $F(x)$ than that of $F(x)$ near $x_1^*$. We then minimize $F(x)$ from the initial point $x_2 = x_2^{'}$ and can get a better minimum $x_2^*$ of $F(x)$. Repeat this process until a global optimum is found or the termination criterion of the algorithm is met.

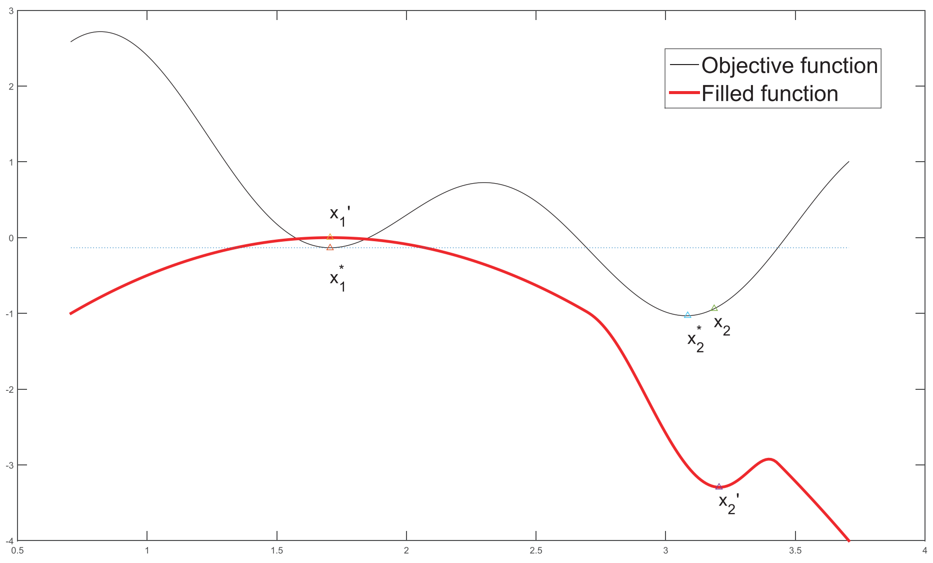

Figure 1: Illustration of optimization process of filled function algorithms

## 5 Numerical Experiments

In this section, to demonstrate the performance of the proposed algorithm, some experiments are conducted using Matlab R2014b on the widely used benchmark problems 1-7 which are taken from problems 1-7 in [17]. For details of these problems, please refer to paper [17]. The proposed algorithm is compared with a state-of-the-art algorithm [17] as well as Ge's method on these benchmark problems. Table 1 compares the overall results on all test problems, and Tables 2 - 8 present the results in the optimization process on one random run of the proposed algorithm. The meanings of the symbols are as follows:
No.: The problem number.
n: The dimension of the problem.

$K$ (iter): The iteration counter.
$F_f$: The total number of function evaluations of $F(x)$ and $P(x, x_k^*)$.
$F_g$: The total number of gradient evaluations of $F(x)$ and $P(x, x_k^*)$.
SR: The successful rate of ten repeated runs. A successful run means the algorithm finds a solution with an accuracy of $\epsilon = 1e - 8$ before termination.
- : The results not available.

It can be seen from Table 1 that for problems with dimensions no more than 10, all the three algorithms can successfully find the global minima (or with high successful rates). As the dimension of the problem increases, more function evaluations are needed. This is because when the dimension increases, the search space increases exponentially. Thus the better exploring ability is required for an algorithm to explore a larger search space effectively.

Table 1: The overall comparisons

| | Problem 1 | | Problem 3 | | Problem 4 | |
|---|---|---|---|---|---|---|
| k | $x_k^*$ | $f_k^*$ | $x_k^*$ | $f_k^*$ | | |
| 1 | $(0.7125,0.0002)^T$ | -1.4593 | $(-0.1899,-0.4819)^T$ | 0.2115 | $(-0.5961,-0.2927)^T$ | 0.6833 |
| 2 | $(0.3469,0.0000)^T$ | -1.8789 | $(3.1700e-6, 5.2420e-6)^T$ | 3.0963e-15 | $(-0.0898, -0.7127)^T$ | -1.0316 |
| 3 | $(1.0910e-6,8.6900e-5)^T$ | -2.0000 | | | | |

From Table 1 we can also see that the proposed algorithm uses much fewer function and gradient evaluations than Ge's. From the aspect of using fewer function and gradient evaluations, the proposed algorithm outperforms the algorithm in [17] on 14 problems out of all 17 problems while for problems 1, 2 (with c=0.5) and 7 (with n=2), the proposed algorithm uses more function evaluations. So we can get the conclusion that the proposed

Table 2: Results of Problem 1, 3 and 4

| | Problem 2( c=0.2) | | Problem 2( c=0.5) | | Problem 2( c=0.05) | |
|---|---|---|---|---|---|---|
| k | $x_k^*$ | $f_k^*$ | $x_k^*$ | $f_k^*$ | | |
| 1 | $(1.6785,-0.3910)^T$ | 0.0932 | $(0.2342,-0.3173)^T$ | 3.8141 | $(5.0556,-0.2373)^T$ | 13.0468 |
| 2 | $(1.8784,-0.3458)^T$ | 8.8459e-14 | $(1.0000, 0)^T$ | 6.0202e-16 | $(0.5487, 0)^T$ | 0.2264 |
| 3 | $(1.0000,0)^T$ | 5.7949e-16 | | | $(1.0000, 0)^T$ | 6.0335e-16 |

Table 3: Results of Problem 2

| | Problem 2( c=0.2) | | Problem 2( c=0.5) | | Problem 2( c=0.05) | |
|---|---|---|---|---|---|---|
| k | $x_k^*$ | $f_k^*$ | $x_k^*$ | $f_k^*$ | | |
| 1 | $(1.6785,-0.3910)^T$ | 0.0932 | $(0.2342,-0.3173)^T$ | 3.8141 | $(5.0556,-0.2373)^T$ | 13.0468 |
| 2 | $(1.8784,-0.3458)^T$ | 8.8459e-14 | $(1.0000, 0)^T$ | 6.0202e-16 | $(0.5487, 0)^T$ | 0.2264 |
| 3 | $(1.0000,0)^T$ | 5.7949e-16 | | | $(1.0000, 0)^T$ | 6.0335e-16 |

Table 4: results of Problem5, 6 and 7(with n=2)

| | Problem 5 | | Problem 6 | | Problem 7( n=2) | |
|---|---|---|---|---|---|---|
| k | $x_k^*$ | $f_k^*$ | $x_k^*$ | $f_k^*$ | | |
| 1 | $(0.0948,-0.0697)^T$ | 0.0443 | $(6.2547,8.7374)^T$ | -9.0081 | $(1.0871,0.2019)^T$ | 2.2003 |
| 2 | $(-2.5690e-6, 2.6000e-6)^T$ | 3.3168e-15 | $(5.4829, 4.8581)^T$ | -186.7309 | $(1.0000, 1.0000)^T$ | 5.9169e-14 |

Table 5: results of Problem 7( with n = 3 and n=5).

| | Problem 7(n=3) | | Problem 7(n=5) | |
|---|---|---|---|---|
| k | $x_k^*$ | $f_k^*$ | | |
| 1 | $(2.8043,5.3125,-0.9250)^T$ | 64.4347 | $(-2.7759,-0.0265,-5.7860, 7.0329,5.2683)^T$ | 209.2577 |
| 2 | $(1.0000, 1.0000,1.0000)^T$ | 8.9037e-12 | $(-4.9383, -3.9986,-1.9988,-2.9955,0.0006)^T$ | 54.4155 |
| 3 | | | $(1.0000, 1.0000,1.0000,1.0000,1.0000)^T$ | 1.6161e-11 |

Table 6: results of Problem 7 with n = 7.

| k | $x_k^*$ | $f_k^*$ |
|---|---|---|
| 1 | $(-3.9366, 2.9254, -1.7531, 4.0212,-0.3777,-5.7159,-4.7161)^T$ | 233.2210 |
| 2 | $(0.0127, 5.9476,3.9988,4.9955,1.0000, 1.0000,1.0000)^T$ | 22.7584 |
| 3 | $(1.0000, 1.0000,1.0000,1.0000,1.0000, 1.0000,1.0000)^T$ | 2.3169e-12 |

Table 7: results of Problem 7 with n = 10.

| k | $x_k^*$ | $f_k^*$ |
|---|---|---|
| 1 | $(2.0482,4.4854, 7.8205,0.8499,-1.9279,-0.2314,5.2956,-0.0948,0.0377,5.0596)^T$ | 98.2744 |
| 2 | $(1.9890,-1.9688,8.9908,3.9995,1.0000,1.0000, 1.0000,1.0000,1.0000, 1.0000)^T$ | 26.0195 |
| 3 | $(1.0000, 1.0000,1.0000,1.0000,1.0000,1.0000, 1.0000,1.0000,1.0000,1.0000)^T$ | 7.1366e-13 |

Table 8: results of Problem 7 with n = 10.

| k | $x_k^*$ | $f_k^*$ |
|---|---|---|
| 1 | $(1.9290,-3.3797,-4.1400,0.0816,6.8956,8.9133,6.9662,0.8804,0.4168,0.5244)^T$ | 102.1356 |
| 2 | $(1.9895,-0.9791,-3.9871,0.0005,4.9593,8.9948,6.9990,1.0000,1.0000,1.0000)^T$ | 46.0795 |
| 3 | $(1.9894,-0.9799,0.9998, 0.9916,0.9856,0.9600,1.0045,1.0061,1.0000,1.0000)^T$ | 1.5554 |
| 4 | $(1.9895,-0.9795,1.0000,1.0000,1.0000,1.0000, 1.0000,1.0000,1.0000,1.0000)^T$ | 1.5548 |
| 5 | $(1.9899, 0.0102,1.0000,1.0000,1.0000,1.0000, 1.0000,1.0000,1.0000,1.0000)^T$ | 0.6220 |
| 6 | $(1.0000, 1.0000,1.0000,1.0000,1.0000,1.0000, 1.0000,1.0000,1.0000,1.0000)^T$ | 5.9254e-14 |

algorithm generally uses fewer function evaluations than that in [17]. Moreover, as the dimension increases, the difference becomes more significant. It can be seen from Table 1 that when the dimensions are more than 5, the proposed algorithm uses much fewer function evaluations than that in [17]. When the dimension reaches 50, algorithm in [17] needs at least $10^7$ function evaluations. By contrast, the proposed algorithm uses much fewer function evaluations (only 228478 for dimension 50 and 258125 for dimension 100 respectively) with the successful rates of 10/10 and 6/10 respectively. So we can get the conclusion that the proposed algorithm performs better than the algorithm in [17] on higher dimensional problems.

It can be seen from Tables 2-8 that the solution obtained for each problem is improved obviously at each iteration. This demonstrates the proposed algorithm converges fast and is effective.

It should be noted that Tables 7 and 8 show the results of two typical runs of the proposed algorithm on problem 7 with dimension 10. Table 7 shows the best results and Table 8 shows the results using the restarting mechanism of the algorithm. Note that although the properties of filled function ensure that the filled function method can find a better solution at each iteration in theory, a filled function method often can not realize this goal in practice. For example, when a better basin is so narrow that the search method can not easily focus the search on that narrow region, the search method may make a great effort (use much more function evaluations), but it finally terminates outside that better basin. This will results in the inefficiency of the filled function methods. However, the restarting mechanism can help the algorithm to avoid this situation and save the function evaluations. By restarting, the algorithm has more chances to find a global minimum within reasonable cost. In fact, although the algorithm in [17] uses very few iterations, e.g., it only uses 3 iterations on problem 7 with dimension 30, it uses as much as 376885 function evaluations while the proposed algorithm with the restarting mechanism uses much fewer function evaluations (only 41214 function evaluations). Also, when the dimension increases, although the proposed algorithm uses more iterations than the algorithm in [17], it uses much fewer function evaluations than that in [17]. This indicates that the restarting mechanism can help an algorithm to focus its search on better basin and save the computation cost.

Overall, we can see that the proposed filled function algorithm is more efficient and has better exploring ability than the compared ones. Also, from Table 1 we can see that, the filled function methods are efficient for solving small scale problems (with dimension smaller than 100). However, for large scale problems (with dimension more than 1000), to the best of our knowledge, there is not any experimental result reported for the filled function algorithms. In fact, the existing filled function methods become inefficient for large scale problems. It is very necessary to develop new methods for large scale problems. One possible and reasonable way is to decompose the whole problem into some small scale (low dimensional) sub-problems (e.g., use random grouping, differential grouping to make the decomposition), and then get the optimal solution by solving these sub-problems and integrating the solutions of these sub-problems.

## 6 Conclusions and Future Work

In this paper, a continuous differentiable filled function without any parameter is proposed. The new filled function mainly has three advantages: Firstly, since it is continuous differentiable, a wide range of efficient optimization algorithms such as quasi-Newton method can be chosen as the local search method. Secondly, optimizing a continuous differentiable function is much easier than optimizing a non-continuous differentiable function. Thirdly, the new filled function has no parameter to tune. Based on the new filled function, a filled function algorithm is proposed, and a new search method with a restarting mechanism is designed to make the algorithm more effective. Numerical experiments are carried out on some widely used benchmark functions. Comparisons with a state-of-the-art algorithm and the first filled function method are made and the results show that the proposed algorithm is more effective and efficient. The numerical results also indicate that filled function methods are not able to handle very high dimensional problems effectively and the optimization of high dimensional problems remains an open problem. For the very high dimensional problems, one possible way is to use divide and conquer strategy to decompose the high dimensional problem into several low dimensional problems, or use the hybridization of different algorithms. So how to design decomposition strategy and effectively integrate the filled function method with other techniques for very high dimensional problems are our future work.

## References

[1] A. Sahiner, N. Yilmaz, and O. Demirozer, Mathematical modeling and an application of the filled function method in entomology, *Int. J. Pest Manage* 60 (2014) 232–237.

[2] B. Ling, C. Wu, K. Teo, and V. Rehbock, Global optimal design of iir filters via constraint transcription and filled function methods, *Circuits Systems Signal Process* 32 (2013) 1313–1334.

[3] C. Dang, W. Ma, and J. Liang, A deterministic annealing algorithm for approximating a solution of the min-bisection problem, *Neural Networks* 22 (2009) 58 – 66.

[4] E. Campana, G. Liuzzi, S. Lucidi, D. Peri, V. Piccialli, and A. Pinto, New global optimization methods for ship design problems, *Optim. Eng.* 10 (2009) 533–555.

[5] F. Wei and Y. Wang, A new filled function method with one parameter for global optimization, *Math. Probl. Eng.* 2013 (2013) p. 12.

[6] F. Wei, Y. Wang and H. Lin, A new filled function method with two parameters for global optimization, *J. Optim. Theory Appl.* 163 (2014) 510–527.

[7] H. Lin, Y. Wang and L. Fan, A filled function method with one parameter for unconstrained global optimization, *Appl. Math. Comput.* 218 (2011) 3776 – 3785.

[8] J.M. Ortega and W.C. Rheinboldt, Iterative solution of nonlinear equations in several variables, *Acedemic Press* 3 (1970) 108–109.

[9] L. An, L. Zhang and M. Chen, A parameter-free filled function for unconstrained global optimization, *J. Shanghai Univ. Nat. Sci.* 8 (2004) 117–123.

[10] L. Zhang, C. K. Ng, D. Li and W. W. Tian, A new filled function method for global optimization, *J. Global Optim.* 28 (2004), 17–43.

[11] R. Ge, A filled function method for finding a global minimizer of a function of several variables, *Math. Program.* 46 (1990) 191–204.

[12] R. Ge, The theory of filled function method for finding global minimizers of nonlinearly constrained minimization problems, *J. Comput. Math.* 5 (1987) 1–9.

[13] R. Ge and Y. Qin, A class of filled functions for finding global minimizers of a function of several variables, *J Optim Theory Appl,* 54 (1987) 241–252.

[14] R.Q. Daoerji and Y. Wang, A new hybrid genetic algorithm for job shop scheduling problem, *Comput. Oper. Res.* 39 (2012) 2291– 2299.

[15] S. He, W. Chen and H. Wang, A new filled function algorithm for constrained global optimization problems, *Appl. Math. Comput.* 217 (2011) 5853–5859.

[16] S. Ma, Y. Yang and H. Liu, A parameter free filled function for unconstrained global optimization, *Appl. Math. Comput.* 215 (2010) 3610– 3619.

[17] T. El-Gindy, M. Salim and A. Ahmed, A new filled function method applied to unconstrained global optimization, *Appl. Math. Comput.* 273 (2016) 1246 –1256.

[18] W. Wang, Y. Shang and L. Zhang, A filled function method with one parameter for box constrained global optimization, *Appl. Math. Comput.* 194 (2007) 54–66.

[19] X. Liu and W. Xu, A new filled function applied to global optimization, *Comput. Oper. Res.* 31 (2004) 61–80.

[20] X. Liu, A class of continuously differentiable filled functions for global optimization, *IEEE Trans. Syst. Man Cyber. C* 38 (2008) 38–47.

[21] X. Liu, Finding global minima with a computable filled function, *J. Global Optim.* 19 (2001) 151–161.

[22] X. Wu, K. Zhang and C. Sun, Constrained optimal control of switched systems based on modified bfgs algorithm and filled function method, *Int. J. Comput. Math.* 91 (2014) 1713–1729.

[23] Y. Gao, Y. Yang and M. You, A new filled function method for global optimization, *Appl. Math. Comput.* 268 (2015) 685–695.

[24] Y. Shang, D. Pu and A. Jiang, Finding global minimizer with one-parameter filled function on unconstrained global optimization, *Appl. Math. Comput.* 191 (2007) 176–182.

[25] Y. Wang and K. Fang, A note on uniform distribution and experimental design, *Chin. Sci. Bull* 26 (1981) 485–489.

[26] Y.W. Leung and Y. Wang, Multiobjective programming using uniform design and genetic algorithm, *IEEE Trans. Syst. Man Cyber. C* 30 (2000) 293–304.

[27] Y. Zhang and Y.T. Xu, A one-parameter filled function method applied to nonsmooth constrained global optimization, *Comput. Math. Appl.* 58 (2009) 1230–1238.

[28] Y. Zhang, L. Zhang and Y. Xu, New filled functions for nonsmooth global optimization, *Appl. Math. Modelll* 33 (2009) 3114–3129.

[29] Z. Wan, L. Yuan and J. Chen, A filled function method for nonlinear systems of equalities and inequalities, *J. Comput. Appl. Math.* 31 (2012) 391–405.

*Manuscript received 21 November 2016*
*revised 27 March 2017*
*accepted for publication 8 May 2017*

Haiyan Liu
School of Computer Science and Technology
Xidian University, Xi'an 710071, China
E-mail address: hyliu83@126.com

Yuping Wang
School of Computer Science and Technology
Xidian University, Xi'an 710071, China
E-mail address: ywang@xidian.edu.cn

Xiao-Zhi Gao
School of Computing, University of Eastern Finland
Kuopio, Finland
E-mail address: yxiao.z.gao@gmail.com

Chuangyin Dang
ring Management
Department of Systems Engineering and Engineering Management
City University of Hong Kong, Hong Kong 999077, China
E-mail address: mecdang@cityu.edu.hk

Zhiqing Meng
College of Economics and Management
Zhejiang University of Technology, Hangzhou 310014, China
E-mail address: mengzhiqing@zjut.edu.cn