


Article

A New Parameterless Filled Function Method for Global Optimization

Haiyan Liu *, Siyan Xue, Yuan Cheng and Shouheng Tuo 

School of Computer Science and Technology, Xi'an University of Posts and Telecommunications,
Xi'an 710121, China

* Correspondence: hylu83@126.com

Abstract: The filled function method is an effective way to solve global optimization problems. However, its effectiveness is greatly affected by the selection of parameters, and the non-continuous or non-differentiable properties of the constructed filled function. To overcome the above-mentioned drawbacks, in this paper, a new parameterless filled function is proposed that is continuous and differentiable. Theoretical proofs have been made to show the properties of the proposed filled function. Based on the new filled function, a filled function algorithm is proposed to solve unconstrained global optimization problems. Experiments are carried out on widely used test problems and an application of supply chain problems with equality and inequality constraints. The numerical results show that the proposed filled function is effective.

Keywords: filled function method; global optimization; parameterless filled function; mathematical programming



Citation: Liu, H.; Xue, S.; Cheng, Y.; Tuo, S. A New Parameterless Filled Function Method for Global Optimization. *Axioms* **2022**, *11*, 746. <https://doi.org/10.3390/axioms11120746>

Academic Editor: Nodari Vakhania

Received: 6 November 2022

Accepted: 16 December 2022

Published: 19 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Global optimization is rich in content and widely used subject in mathematics. With the development of science and information technology, global optimization has been widely applied to economic models, finance, image processing, machine designing and so on. Therefore, the theories and methods for global optimization need to be studied deeply. With the efforts of scholars in this field, various methods have been developed for global optimization. However, finding the global optimal solution is usually not easy due to the two properties of the global optimization problems: (1) usually there exists a lot of local optimal solutions, and (2) optimization algorithms are very easy to be trapped in certain local optimal solutions and unable to escape. Therefore, one key problem is how to help the optimization method escapes from local optimal solutions. The filled function method is specifically designed to solve this problem. Now we will introduce some basic information about the filled function method.

The filled function method was first proposed by Ge [1] in which he constructed an auxiliary function named filled function to help the optimization algorithm escape from local optimal solutions. In the following, we will introduce the original definition of the filled function proposed by Ge [1] and its related concepts. In this paper, we consider the following optimization problem:

$$\begin{aligned} & \min F(x) \\ & s.t. x \in \Omega = [l, u] = \{x | l \leq x \leq u, l, u \in R^n\} \end{aligned}$$

where n is the dimension of the objective function $F(x)$, which is continuous and differentiable. $F(x)$ has a finite number of local optimal solutions $x_1^*, x_2^*, \dots, x_m^*$. Suppose x_k^* is the local optimal solution found by the optimization algorithm in the k th iteration; the definition of basin B_k^* is as follows.

Definition 1. The basin B_k^* of the objective function $F(x)$ at an isolated minimum (local optimal solution) x_k^* refers to the connected domain that 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

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

A basin is actually an area that contains one local optimal solution. Within this area, the gradient descent optimization algorithm will converge to the corresponding local optimal solution no matter what the initial point is. One basin is lower than the other basin if its corresponding local optimal solution is smaller (better for minimization problems).

Definition 2. 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 properties:

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 x' in such a basin that minimizes $P(x, x_1^*)$ on the line through x and x_1^* .

From the definition of the filled function, we can see that the three properties together ensure the optimization algorithm escapes from one local optimal solution to a better one. For example, the optimization algorithm is now trapped in a local optimal solution x_1^* and can not escape. Now we construct a filled function to help escape x_1^* . The first property of the filled function will make the local optimal solution x_1^* become a local worst solution of the filled function. In this case, when the filled function $P(x, x_1^*)$ is optimized, it will surely leave the local worst solution; that is, the local optimal solution will escape. Then the second property of the filled function will ensure that when optimizing $P(x, x_1^*)$, it will not end up at a worse solution than x_1^* because there are no minima or stable points in any basin higher than B_1^* . Instead, the optimization procedure of $P(x, x_1^*)$ will enter a basin that is better than B_1^* , if such a basin exists. The overall optimization procedure is as follows: First, it starts from an initial point to optimize the objective function $F(x)$ and find a locally optimal solution (e.g., x_1^*). Secondly, a filled function at this point is constructed (e.g., $P(x, x_1^*)$) and optimized starting from x_1^* . After the optimization of $P(x, x_1^*)$, the algorithm will enter a better region (basin) ensured by the properties of the filled function. Thirdly, starting from the new better basin, the algorithm continues to optimize $F(x)$ to find a better local optimal solution than x_1^* . Then, repeating the above steps, the algorithm will continuously move from one local optimal solution to better ones till the global optimal solution is found.

2. Related Work

With optimization algorithms extensively used in various fields, more and more efforts are devoted to optimization theory. As a deterministic algorithm for optimization, the filled function method has drawn a lot of attention. The main idea of the filled function method is to locate a current local optimal solution by any local search algorithm and then construct an auxiliary function called the filled function at that local optimal solution. The filled function should have three good properties in order to help it escape from the current local optimal solutions and enter regions that contain better solutions.

The first definition of a filled function was proposed by Ge in ref. [1], in which he constructed a filled function with two parameters:

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

Experiments show this filled function is effective. However, it has two disadvantages: first, this filled function has two adjusting parameters r and ρ , the values of the two parameters need to be adjusted in order to ensure the global optimal solution will not be missed in the optimization procedure; secondly, since there is an exponent term in denominator when it gets larger, the function value will become smaller, and thus, the filled function may locate a fake stable point.

In order to improve the efficiency of the filled function method, a lot of effort has been made, and new contributions have been achieved. In ref. [2], a filled function with only one parameter is proposed that also has no exponent term:

$$H(x) = 1/\ln[1 + f(x) - f(x_1)] - a\|x - x_1\|^2 \quad (2)$$

However, this filled function $H(x)$ is undefined at $f(x) - f(x_1) \leq -1$. To overcome the discontinuous and non-differential disadvantages of filled function, Liu proposed a class of filled functions that is continuous and differentiable [3]:

$$C(x, a) = -u[F(x) - F(x_1)]w^a(\|x - x_1^*\|^p) \quad (3)$$

where u and w are two real functions that are twice continuously differentiable in their domains, satisfying the following conditions:

$$\begin{aligned} u(0) &= 0 \quad w(0) = 1/a > 0 \\ u'(t) &> 0 \quad w'(t) > 0, \quad \forall t \in [0, \infty) \\ \lim_{t \rightarrow 0} \frac{u(t)w'(t)}{u'(t)w(t)} &= 0 \end{aligned} \quad (4)$$

However, this class of filled function is not easy to construct and still contains two parameters to adjust. Afterward, new continuous differentiable filled functions were proposed [4–6], yet these filled functions all have one or two parameters. In order to improve the parameter-adjusting problem of the filled function, the authors of [7] proposed a filled function with two parameters and gave a reasonable and effective way to choose the parameters. In ref. [8], the authors proposed a filled function without any parameter. This filled function contains no exponent term and is simple in form; however, it is not a continuous differentiable, which may produce extra local optimal solutions. To overcome this problem, the authors of [9] proposed a continuously differentiable filled function without any adjusting parameter:

$$\begin{aligned} P(x, x_k^*) &= -\|x - x_k^*\|^2 g(F(x) - F(x_k^*)) \\ g(t) &= \begin{cases} 1 & t \geq 0 \\ -\exp(t^2) + 2 & t < 0 \end{cases} \end{aligned} \quad (5)$$

Afterward, researchers have proposed more continuous differentiable filled functions without parameters, such as in ref. [10–14]. The authors of [12] proposed a new continuous differentiable filled function without any parameter or exponent term:

$$P(x, x^*, f(x)) = \cosh(1/(1 + \|x - x^*\|^2))\psi(f(x) - f(x_k^*)) \quad (6)$$

These parameterless and continuous differentiable filled functions have several advantages. First, more efficient local search algorithms can be applied. Secondly, it is not easy to produce extra fake local optimal solutions. Thirdly, no parameter adjusting is needed. Thus this kind of filled function can improve the efficiency of the performance of filled function methods.

To better enhance the efficiency of filled function methods, a two-stage method with a stretch function was proposed [15]. After a current local minimum is located in the stage of optimizing the objective function, a stretch function is used to make this local minimum

higher. Then a filled function is constructed and optimized in the second stage. However, this filled function is not continuous, which means classical efficient local search methods can not be applied to this method.

The authors of [16] proposed a new algorithm based on filled function. First, a multi-dimensional objective function is transformed into one-dimensional functions, and then, for each direction, a filled function is constructed to optimize the one-dimensional function. To overcome the potential failure that only local minimum is found, the authors of [17] proposed a new filled function method. By combining an adaptive strategy of determining the initial points and a narrow valley widening strategy, the ability to escape the local minimum and locate the global minimum is further enhanced. In ref. [18], the authors proposed a new filled function using a smoothing method to eliminate local optimal solutions. Further, an adaptive method is used to determine the step length and shallow valleys.

Now the filled function method is not only used in unconstrained optimization problems but is extended to constrained optimization problems with inequalities, bi-level programming, nonlinear integer programming and non-smooth constrained problems. The authors of [19] proposed a continuous differentiable filled function with one parameter to solve constrained optimization problems. The authors of [20] proposed a single-parameter filled function and applied it to a supply chain problem, which is a nonlinear programming problem with equality and inequality constraints. For bi-level programming with inequality and equality constraints, the authors of [21] first transformed the bi-level programming problem into a single-layer constrained optimization problem and then constructed the filled function combining penalty functions.

The authors of [22] first transformed the original problem into an equivalent constrained optimization problem and then constructed a filled function to solve it.

For the following type of constrained global optimization P :

$$g(t) = \begin{cases} \min f(x) \\ s.t. g_i(x) \leq 0, i = 1, 2, \dots, m \\ x \in Z^n \end{cases} \quad (7)$$

where Z^n is an integer set of R^n and $S = \{x \in Z^n | g_i(x) \leq 0, i = 1, 2, \dots, m\}$ is bounded. The authors of [23] proposed a method to transform this constrained problem into a box-constrained integer programming problem and then constructed a filled function to solve it. In ref. [24], the authors proposed a parameterless filled function to solve nonlinear equations with box constraints.

The filled function method is also extended to non-smoothing optimization problems. The authors of [25,26] proposed a one-parameter filled function based on a new definition of filled function for a non-smoothing-constrained programming problem.

Based on the idea of filled function, in this paper, a new parameterless filled function is proposed that is continuous and differentiable. The properties of the new filled function is proven in section 3. Based on it, a filled function algorithm is also proposed to handle unconstrained optimization problems. Numerical experiments are carried out, and comparisons are made in Section 4.

3. A New Parameterless Filled Function and a Filled Function Algorithm

In this section, a new filled function is proposed with the advantages of being parameterless, continuous and differentiable. The three properties of the proposed filled function are described and proven. Based on it, a new filled function method is designed to solve unconstrained optimization problems.

3.1. A New Parameterless Filled Function and Its Properties

The first definition of the filled function is defined in [1]. However, the third property of the definition is not so clear, e.g., it is not clear where the point x' is and where the line is through x and x_1^* . To make the definition more clear and more strict, some scholars gave several revised definitions of the filled function [27,28]. In this paper, we will use the

revised definition from ref. [9] since it is more clear and strict by using the gradient. The revised definition is as follows.

Definition 3. A function $P(x, x_k^*)$ is called a filled function of $F(x)$ at a local minimum x_k^* if it satisfies the following properties:

1. x_k^* is a strictly local maximum of $P(x, x_k^*)$, and the whole basin B_k^* of $F(x)$ becomes a part of a hill of $P(x, x_k^*)$.
2. For any $x \in \Omega_1$, we have $\nabla P(x, x_k^*) \neq 0$, 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 $x'_k \in \Omega_2$, such that x'_k is a local minimum of $P(x, x_k^*)$.

Now we give a brief explanation of the revised definition of the filled function. Property 1 is the same as the original definition, which turns the local minimum x_k^* of the objective function into a local maximum of the filled function. In this case, when optimizing the filled function, it is easy to escape x_k^* since it is a local maximum (worst solution for the minimization problem). Property 2 makes sure the optimization procedure will not end up with solutions worse than the current local minimum x_k^* because there are no stationary points there. Property 3 means that it is easy for the optimization procedure to end in a region that contains a better solution than the current local minimum because in that region, there exists a local minimum. Therefore, the three properties together will drive the optimization procedure to escape from the current local minimum and enter a better region that contains a better solution.

Based on Definition 3, we design a new parameterless filled function that is also continuous and differentiable:

$$P(x, x_k^*) = \frac{1}{1 + \|x - x_k^*\|} \cdot g(F(x) - F(x_k^*))$$

$$g(t) = \begin{cases} 1 & t \geq 0 \\ t^3 + 1 & t < 0 \end{cases} \quad (8)$$

The new filled function mainly has two advantages. First, it has no parameter to adjust, which makes it easier to apply to different optimization problems. Secondly, the new filled function is continuous and differentiable. Note that continuity and differentiability are two excellent properties of filled function. Compared to filled functions that are not continuous or differentiable, it is easier to optimize since more choices of algorithms, especially more efficient algorithms designed for continuous differentiable functions can be used, and it is also not easy to generate extra local optimal solutions during the optimization. Now, we will first prove that the new filled function is continuously differentiable and then prove that it fulfills the three properties of the definition of the filled function.

Since the only point that may cause the filled function $P(x, x_k^*)$ to not be continuously differentiable is $t = 0$ in $g(t)$, so if $g(t)$ is continuously differentiable at $t = 0$, the filled function $P(x, x_k^*)$ is continuously differentiable.

Since $\lim_{t \rightarrow 0^+} g(t) = \lim_{t \rightarrow 0^-} g(t) = 1$, the new filled function $P(x, x_k^*)$ is continuous.

Since

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

and

$$g'_-(0) = \lim_{t \rightarrow 0^-} \frac{g(t) - g(0)}{t - 0} = \lim_{t \rightarrow 0^-} \frac{t^3 + 1 - 1}{t} = \lim_{t \rightarrow 0^-} t^2 = 0.$$

Thus, $g'_+(0) = g'_-(0) = 0$, so the new filled function $P(x, x_k^*)$ is differentiable. Now we will prove that $P(x, x_k^*)$ satisfies the three properties of filled function.

Theorem 1. Suppose x_k^* is a local minimum of the objective function $F(x)$ and $P(x, x_k^*)$ is the filled function constructed at x_k^* , then x_k^* is a strictly local maximum of $P(x, x_k^*)$.

Proof. Suppose B_k^* is the basin containing x_k^* (please refer to Definition 1 about basin), since x_k^* is a local minimum of $F(x)$, so $\forall x \in B_k^*, x \neq x_k^*$, we have $F(x) > F(x_k^*)$. Thus, $F(x) - F(x_k^*) > 0$, in this case $g(F(x) - F(x_k^*)) = 1$. According to the construction of the filled function $P(x, x_k^*)$, we get

$$P(x, x_k^*) = \frac{1}{1 + \|x - x_k^*\|} \cdot g(F(x) - F(x_k^*)) = \frac{1}{1 + \|x - x_k^*\|} < 1$$

$$P(x_k^*, x_k^*) = \frac{1}{1 + \|x_k^* - x_k^*\|} \cdot g(F(x_k^*) - F(x_k^*)) = g(0) = 1$$

Thus, $P(x, x_k^*) < P(x_k^*, x_k^*)$, which means that x_k^* is the strict local maximum of $P(x, x_k^*)$.

Theorem 2. For any $x \in \Omega_1$, we have $\nabla P(x, x_k^*) \neq 0$, where $\Omega_1 = \{x \in \Omega | F(x) \geq F(x_k^*), x \neq x_k^*\}$.

Proof. Since $\Omega_1 = \{x \in \Omega | F(x) \geq F(x_k^*), x \neq x_k^*\}$, for any $x \in \Omega_1$, we have $F(x) \geq F(x_k^*)$, thus

$$P(x, x_k^*) = \frac{1}{1 + \|x - x_k^*\|} \cdot g(F(x) - F(x_k^*)) = \frac{1}{1 + \|x - x_k^*\|}$$

and

$$\nabla P(x, x_k^*) = -\frac{1}{1 + \|x - x_k^*\|^2} \neq 0$$

This proves Theorem 2.

Theorem 3. If $\Omega_2 = \{x \in \Omega | F(x) < F(x_k^*)\}$ is not empty, then there exists $x'_k \in \Omega_2$, such that x'_k is a local minimum of $P(x, x_k^*)$.

Proof. Since Ω_2 is not empty, then $F(x)$ must have a minimum in Ω_2 .

Since $P(x, x_k^*)$ is continuous and differentiable on R^n , it must have a minimum, say x'_k at Ω_2 . Because $P(x, x_k^*)$ is differentiable at x'_k , then this minimum x'_k must be a stationary point, that is, $\nabla P(x'_k, x_k^*) = 0$.

Since Ω_2 is not empty, then there exists a point $z \in \Omega_2$ such that $P(z, x_k^*) < 0$. Thus $P(x'_k, x_k^*) \leq P(z, x_k^*) < 0$ and $x'_k \neq x_k^*$. Therefore, we know that $x'_k \notin \Omega_1$ (according to the definition of Ω_1 from Theorem 2); therefore, $x'_k \in \Omega_2$.

3.2. A Filled Function Algorithm to Solve Unconstrained Optimization Problems

Based on the proposed filled function, we design a filled function algorithm to solve unconstrained optimization problems. The steps of the algorithm are as follows.

1. Initialization. Randomly generate 10 points in the feasible region and choose the point with the best function value as the initial point x_0 . Then, set $bestX = x_0$, $bestVal = F(x_0)$ to record the best solution and its corresponding function value, we set $\epsilon = e - 10$ as the stopping criteria and $k = 1$ as the iteration counter.
2. Optimize the objective function $F(x)$. Starting from the initial point x_0 , we use the BFGS Quasi-Newton Method as the local search method to optimize the objective function to locate a local optimal point x_k^* . The main steps of the BFGS method are shown in Algorithm 1.

- Construct the filled function at x_k^* :

$$P(x, x_k^*) = \frac{1}{1 + \|x - x_k^*\|} \cdot g(F(x) - F(x_k^*))$$

$$g(t) = \begin{cases} 1 & t \geq 0 \\ t^3 + 1 & t < 0 \end{cases} \quad (9)$$

- Optimize the filled function $P(x, x_k^*)$. Set $x_k^* + 0.1$ as the initial point, and use the BFGS Quasi-Newton Method local search method to optimize the filled function $P(x, x_k^*)$ to obtain a local minimum point x'_k of $P(x, x_k^*)$. It is known from property 3 of the filled function that point x'_k must lie in a lower basin than x_k^* .
- Set the point $x'_k + 0.1$ as the initial point, and continue to optimize the objective function $F(x)$ to obtain a new local minimum point x_{k+1}^* of $F(x)$.
- Determine whether $F(x_{k+1}^*) - bestVal$ is less than $-\epsilon$. If satisfied, update $bestX$ by x_{k+1}^* and $bestVal$ by $F(x_{k+1}^*)$, let $k = k + 1$. Go to step 2, otherwise, $bestX$ is the global optimum and the algorithm terminates.

Algorithm 1 Main steps of the BFGS Quasi-Newton Method

- Given an initial value x_0 and an accuracy threshold ϵ , set $D_0 = I, k := 0$.
 - Determine the direction of the search: $d_k = -D_k \cdot g_k$.
 - set $S_k = \lambda_k d_k, X_{k+1} := X_k + S_k$, and $\lambda_k = \operatorname{argmin} f(X_k + \lambda d_k), \lambda \in R$.
 - if $\|g_{k+1}\| < \epsilon$, then, the algorithm ends.
 - Calculate $y_k = g_{k+1} - g_k$.
 - Calculate $D_{k+1} = (I - \frac{S_k y_k^T}{y_k^T S_k}) D_k (I - \frac{y_k S_k^T}{y_k^T S_k}) + \frac{S_k S_k^T}{y_k^T S_k}$.
 - let $k := k + 1$, go to Step 2
-

In the following, we use an example to demonstrate the optimization procedure of the filled function algorithm. Figure 1 shows the objective function $f(x) = x + 10 \sin(5x) + 7 \cos(4x)$ with the search region $[-2, 2]$. From Figure 1, we can see that $f(x)$ has three basins B_1^*, B_2^* and B_3^* in the search region, where B_3^* is the lowest basin that contains the global optimal solution. Suppose the optimization procedure starts from x_0 ; using the BFGS local search method we can obtain a local minimal solution x_1^* of the objective function $f(x)$.

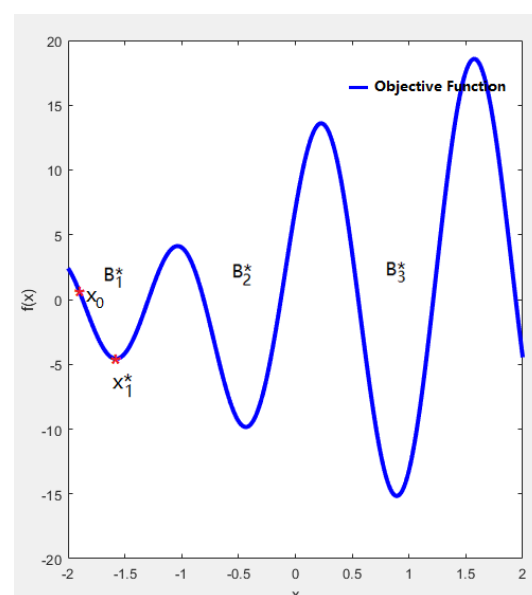


Figure 1. Illustration of steps 1 and 2 of the filled function algorithm.

To escape from this local minimum x_1^* , we construct the filled function $P(x, x_1^*)$ at x_1^* , as shown in Figure 2.

From Figure 2, we can see that x_1^* is a strictly local maximum (maximal point) of $P(x, x_1^*)$, which is guaranteed by the definition of the filled function. Therefore, a local search of $P(x, x_1^*)$, starting from point $x_1^* + 0.1$, can easily escape from this point and yield a local minima x_1' of $P(x, x_1^*)$. Next, using x_1 ($x_1 = x_1' + 0.1$) as the initial point to optimize the objective function $f(x)$, we can obtain another local minimal solution x_2^* that is better than x_1^* . At this time, the first iteration is completed.

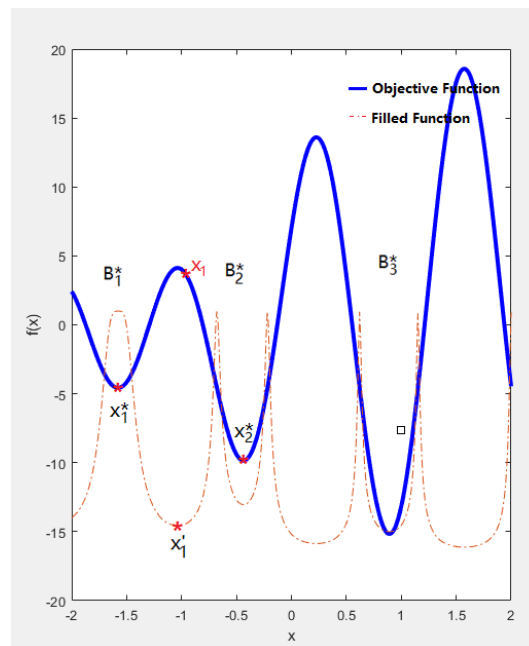


Figure 2. Illustration of steps 3 to step 5 of the filled function algorithm.

To escape from the local minimum x_2^* , we repeat the above steps to construct the filled function $P(x, x_2^*)$ at x_2^* , as shown in Figure 3.

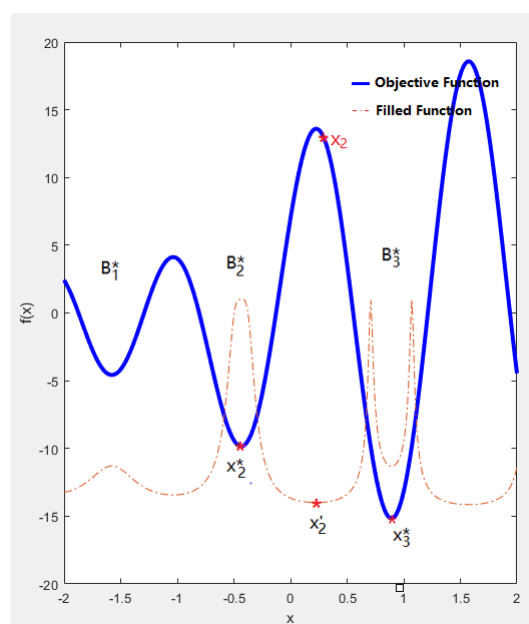


Figure 3. Illustration of the filled function algorithm in the second iteration.

Similarly, $P(x, x_2^*)$ peaks at point x_2^* , which makes it easy to escape from this point. We continue to optimize $P(x, x_2^*)$ to obtain a local minimal point x_2' . Then, using $x_2 = x_2' + 0.1$ as the initial point to optimize the objective function $f(x)$, a new better local optimal solution x_3^* is obtained. Now, the second iteration is completed. We continue the above procedure to optimize the objective function and filled function alternately to escape from the current local optimal solution to a better one till the global optimal solution is located.

From the above optimization procedure, we can clearly see that the proposed method can easily and continuously escape from a current local optimal solution to obtain a better one till the global optimal solution is located. This is a good way to overcome the disadvantage of premature convergence of optimization algorithms. Moreover, the proposed method also has three other advantages. First, since the proposed filled function is parameterless; the algorithm has no adjustable parameters to tune for different problems. Secondly, since the new filled function is continuous and differentiable, the proposed algorithm is less apt to produce an extra local minimum while more choices of local search methods, especially the efficient gradient-based ones, can be applied to make the optimization more efficient and effective. Thirdly, once the filled function is designed and constructed, it is easy to implement and apply to different optimization problems.

There are mainly two disadvantages of the filled function method. First, it is not easy to design a good filled function and each time when a local optimal solution is found, the filled function has to be constructed. Secondly, the filled function method becomes less effective when the dimensionality of the problem is large. More research is needed to extend the scope of the filled function method.

4. Numerical Experiments

The proposed filled function algorithm is implemented in Matlab 2021 and tested on wildly used test problems. Comparisons are made with a state-of-the-art filled function algorithm [18], another continuous differentiable filled function algorithm [5] and Ge's filled function algorithm [1]. The test problems used in this paper are listed as follows.

Test case 1. (The rastrigin function)

$$\begin{aligned} \min F(x) &= x_1^2 + x_2^2 - \cos 18x_1 - \cos 18x_2 \\ \text{s.t. } &-3 < x_1 < 3, -3 < x_2 < 3 \end{aligned}$$

The global minimum solution is $x^* = (0, 0)^T$, and the corresponding function value is $F(x^*) = -2$.

Test case 2. (Two-dimensional function)

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

where $c = 0.05, 0.2, 0.5$. The global minimum solution is $x^* = (1, 0)^T$, and the corresponding function value is $F(x^*) = 0$ for all values of c .

Test case 3. (Three-hump back camel function)

$$\begin{aligned} \min F(x) &= 2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 - x_1x_2 + x_2^2 \\ \text{s.t. } &-3 < x_1 < 3, -3 < x_2 < 3 \end{aligned}$$

The global minimum solution is $x^* = (0, 0)^T$, and the corresponding function value is $F(x^*) = 0$.

Test case 4. (Six-hump back camel function)

$$\begin{aligned} \min F(x) &= 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 - x_1x_2 - 4x_2^2 + 4x_2^4 \\ \text{s.t. } &-3 < x_1 < 3, -3 < x_2 < 3 \end{aligned}$$

The global minimum solution is $x^* = (\pm 0.0898, \pm 0.7127)^T$, and the corresponding function value is $F(x^*) = -1.0316$.

Test case 5. (Treccani function)

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

The global minimum solution is $x^* = (-2, 0)^T$ and $x^* = (0, 0)^T$, and the corresponding function value is $F(x^*) = 0$.

Test case 6. (Two-dimensional Shubert function)

$$\begin{aligned} \min F(x) &= \left\{ \sum_{i=1}^5 i \cos[(i+1)x_1 + i] \right\} \left\{ \sum_{i=1}^5 i \cos[(i+1)x_2 + i] \right\} \\ \text{s.t. } &0 < x_1 < 10, 0 < x_2 < 10 \end{aligned}$$

There are multiple local minimum solutions in the feasible region, and the global minimum function value is $F(x^*) = -186.7309$.

Test case 7. (n -dimensional function)

$$\begin{aligned} \min F(x) &= \frac{\pi}{n} [10 \sin^2(\pi x_1) + g(x) + (x_n - 1)^2] \\ \text{s.t. } &-10 < x_i < 10, i = 1, 2, \dots, n \end{aligned}$$

where

$$g(x) = \sum_{i=1}^{n-1} [(x_i - 1)^2 (1 + 10 \sin^2(\pi x_{i+1}))]$$

The global minimum solution is $x^* = (1, 1, \dots, 1)^T$, and the corresponding function value is $F(x^*) = 0$ for all values of n .

First, all results obtained by the new filled function algorithm are listed in Tables 1–14. Further, the comparisons are made with another continuous differentiable filled function algorithm, CDFA in [5]. In these tables, we use the following notations:

x_k^* : the local minimum of the objective function in the k th iteration.

f_k^* : the function value of the objective function at x_k^* .

k : the iteration counter.

F_f : the total function evaluations of the objective function and the filled function.

CDFA: the filled function algorithm proposed in [5].

FFFA: the filled function algorithm proposed in [18].

Table 1. Results of Problem 1.

k	Ours	f_k^*	CDFA	
	x_k^*		x_k^*	f_k^*
1	$(-0.7168, -0.0344)^T$	−1.2439	/	/
2	$(-0.6938, 0.0000)^T$	−1.5156	/	/
3	$(-0.7815 \times 10^{-7}, -0.0179 \times 10^{-7})^T$	−2.0000	/	/

Table 2. Results of Problem 2 ($c = 0.2$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(8.8344, -3.3354)^T$	9.4093	$(5.7221, -1.8806)^T$	2.5070
2	$(1.8784, -0.3458)^T$	1.9805×10^{-15}	$(3.7387, -1.2649)^T$	0.6165
3			$(1.5909, -0.2703)^T$	2.8126×10^{-9}

Table 3. Results of Problem 2 ($c = 0.5$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(3.9733, -1.7709)^T$	3.3347	$(0.0420, -0.0948)^T$	0.5175
2	$(1.4513, 0)^T$	0.2264	$(1.0000, 0)^T$	5.7949×10^{-16}
3	$(1.0000, 0)^T$	0		

Table 4. Results of Problem 2 ($c = 0.05$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(4.0604, -2.1833)^T$	7.2212	$(8.7299, -3.2965)^T$	9.0733
2	$(1.8513, -0.4021)^T$	1.1414×10^{-12}	$(7.7280, -0.4022)^T$	6.5031
3			$(1.8513, -0.4021)^T$	4.3885×10^{-11}

Table 5. Results of Problem 3.

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(-0.2565, -0.4722)^T$	0.2289	$(-1.7476, -0.8738)^T$	0.2986
2	$(-0.4064 \times 10^{-7}, -0.3367 \times 10^{-7})^T$	3.0680×10^{-15}	$(-0.0000, -0.0000)^T$	4.0157×10^{-10}

Table 6. Results of Problem 4.

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(0.1021, 0.5426)^T$	-0.8448	$(-1.6071, 0.5687)^T$	2.1043
2	$(0.0898, 0.7127)^T$	-1.0316	$(0.0898, 0.7127)^T$	-1.0316

Table 7. Another results of Problem 4.

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(0.2434, -0.7892)^T$	-0.5178	$(1.6071, -0.5687)^T$	2.1043
2	$(-0.0898, -0.7127)^T$	-1.0316	$(-0.0898, -0.7127)^T$	-1.0316

Table 8. Results of Problem 5.

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(-0.9200, -1.0954)^T$	2.1871	$(-1.0000, 0)^T$	1.0000
2	$(-0.7800 \times 10^{-8}, -0.3252 \times 10^{-8})^T$	2.5397×10^{-16}	$(-0.0000, -0.0000)^T$	2.4048×10^{-17}

Table 9. Results of Problem 6.

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(1.2424, 7.2516)^T$	−6.0455	$(2.0467, 2.0467)^T$	0
2	$(5.4829, 3.7723)^T$	−52.0504	$(3.2800, 4.8581)^T$	−46.511
3	$(5.4892, 4.8581)^T$	−186.7309	$(4.2760, 4.8581)^T$	−79.411
4			$(5.4892, 4.8581)^T$	−186.7309

Table 10. Results of Problem 7 ($n = 2$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(-0.9500, -0.0717)^T$	11.1395	/	/
2	$(1.0000, 1.0000)^T$	7.7954×10^{-14}	/	/

Table 11. Results of Problem 7 ($n = 3$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(-0.5665, -4.3568, -2.8414)^T$	147.6024	/	/
2	$(1.9900, 1.0000, 1.0000)^T$	1.0367	/	/
3	$(1.0000, 1.0000, 1.0000)^T$	1.2257×10^{-13}	/	/

Table 12. Results of Problem 7 ($n = 5$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(4.5092, 4.9506, -0.8846, -3.0831, 5.4416)^T$	165.8637	/	/
2	$(1.9900, 1.0000, 1.0000, 1.0000, 1.0000)^T$	0.6220	/	/
3	$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	8.6588×10^{-12}	/	/

Table 13. Results of Problem 7 ($n = 7$).

k	Ours		CDFA	
	x_k^*	f_k^*	x_k^*	f_k^*
1	$(2.1849, 9.6164, -1.8296, 0.5062, 0.1504, 4.9519, -5.0729)^T$	359.2339	$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	2.3538×10^{-13}
2	$(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T$	8.2241×10^{-13}		

Tables 1–14 show the numerical results of the test problems in different test criteria (different parameters and different dimensions) obtained by the proposed filled function

algorithm. In these tables, k means the iterations for the filled function algorithm to locate the global minimum solution x^* , $F(x^*)$ is the corresponding function value and n is the dimension of the test problem.

From the numerical results, we can see that the proposed filled function algorithm can locate all the global minima solutions successfully (some with a precision error of less than 10^{-10}) and within small iterations.

Table 14. Results of Problem 7 ($n = 10$).

k	Ours	f_k^*	CDFA	f_k^*
	x_k^*		x_k^*	
1	(6.2612, 7.8479, 2.3710, 0.8488, 8.7385, 0.0580, 0.4644, 7.5710, 7.2606, 2.2505) ^T	359.2339	(0.0101, 0.0103, 0.0103, 0.0104, 0.0103, 0.0102, 1.0000, 6.0000, 6.0000, 6.0000) ^T	2.6653
2	(1.0013, 0.5326, −0.9803, 1.0066, 1.0152, 0.9937, 0.5966, 1.0802, 0.9954, −0.1773) ^T	1.8292	(1.1615, 1.1651, 0.4418, 0.9258, 0.9638, −0.4809, 0.9926, 6.0000, 6.0000, 6.0000) ^T	2.4443
3	(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000) ^T	1.5113×10^{-13}	(1.9900, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 6.0000, 6.0000, 6.0000) ^T	0.4443
4			(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000) ^T	0

For these test problems, we also listed the results of another continuous differentiable filled function algorithm, CDFA [5]. Since CDFA just carried out parts of the test problems, we use slashes (/) to indicate the missed values. From the comparison, we can see that for problem 2 ($c = 0.2$), we use one less iteration to locate a minimum solution with six orders of magnitude higher accuracy than CDFA. For problem 2 ($c = 0.5$), although we use one more iteration than CDFA, we successfully located the global minimum 0. For problem 2 ($c = 0.05$), we use one less iteration and obtain a better result than CDFA. For problem 3 we locate a better result (five orders of magnitude higher accuracy) than CDFA with the same iterations. For problems 5 and 7 ($n = 7$), our algorithm use one more iteration than CDFA. For problem 6 we use one less iteration to locate the global minimum than CDFA. For problem 7 ($n = 10$), although we use one less iteration, CDFA located the global minimum 0 while we get 1.51×10^{-13} . From the above analysis, we can achieve the comparison results that our algorithm has four wins (problem 2 with $c = 0.2$, $c = 0.05$, and problems 3 and 6), two losses (problems 5 and 7 with $n = 7$), and three ties (problem 2 with $c = 0.5$, and problems 4 and 7 with $n = 10$) out of the overall ten test problems. Therefore, we come to the conclusion that the proposed filled function algorithm is more effective than CDFA.

Comparisons are also made with a state-of-the-art filled function algorithm, FFFA [18] and Ge's filled function [1]. The comparison results are listed in Table 15, where $No.$ is the number of test problems and n is the dimension, F_f refers to the total number of function evaluations consumed to obtain the optimal solutions (minimum solutions for minimization problems).

Since all three filled function algorithms can find the global minimum solutions, we compare their efficiency by the total number of iterations and function evaluations consumed by each algorithm. From Table 15, we can see that for all test problems, our algorithm is much better than Ge's algorithm. As for the comparison with FFFA, we can see that for test problems 1, 3 and 4, although our algorithm takes one more iteration to get the optimal solution, we use much fewer function evaluations. For problems 2, 5 and 6, our algorithm uses fewer iterations and fewer function evaluations than that of FFFA. For problem 7, we can see that for dimension $n = 2$, our algorithm uses fewer iterations but more function evaluations than FFFA, for $n = 3$ and $n = 7$, our algorithm uses more function evaluations than that of FFFA, but for $n = 5$ and $n = 10$, our algorithm

performs much better than that of FFFA. We can see that for $n = 5$ our algorithm uses three fewer iterations and only uses 2287 function evaluations, while FFFA uses 12,681 function evaluations; for $n = 10$, our algorithm only uses 12,795 function evaluations (which is nearly half of that of FFFA's 20,044) to get the global optimal solution. Overall, we come to the conclusion that the filled function algorithm proposed in this paper is more efficient than FFFA. From the numerical results and comparison with the other three filled function algorithms, we come to the conclusion that the new filled function algorithm is effective and efficient for solving unconstrained global optimization problems.

Table 15. The overall comparisons.

No.	n	Ours		FFFA		Ge's Filled Function	
		k	F_f	k	F_f	k	F_f
1	2	3	553	2	1255	4	21,276
2	2($c = 0.2$)	2	392	3	1997	15	27,500
	2($c = 0.5$)	3	470	4	1400	34	54,505
	2($c = 0.05$)	2	493	4	3888	35	38,424
3	2	2	378	1	1789	32	92,498
4	2	2	277	1	1792	44	35,171
5	2	2	259	4	2089	14	15,759
6	2	2	484	4	3576	20	103,988
7	2	2	463	3	294	22	107,899
	3	3	962	3	510	6	248,407
	5	2	2287	5	12,681	16	1,229,860
	7	2	2590	2	811	21	1,443,686
	10	5	12,795	3	20,044	16	1,829,898

5. An Application of the Filled Function Algorithm

In this section, the proposed filled function algorithm is applied to the supply chain problem. Supply chain problems can be divided into three types, namely manufacturer's core supply chain, supplier core supply chain and seller core supply chain. In this paper, we mainly consider the manufacturer's core supply chain. For the manufacturer's core supply chain, there are multiple suppliers, multiple shippers, multiple generalized transportation methods, multiple sellers and one manufacturer. In this supply chain, the manufacturer uses different raw materials to produce various products that are sold by multiple sellers. The optimization objective of the supply chain is to minimize the total transportation cost.

We suppose there is a supply chain with a manufacturer as the core, one supplier and one kind of raw material required for production. The unit raw material cost of this kind of raw material supplied by the supplier is 2000 USD/t, the maximum supply is 5000 t, and all shippers can deliver it. The manufacturer produces only one product and requires 1.2 t of raw material per ton of product. There are two shippers, both of which can provide services of two generalized modes of transport. There are three sellers, and the order quantity of each seller must be strictly satisfied. The manufacturer initially has no inventory products, the production cost per unit product is 1000 USD, and the maximum production capacity is 4500 t. The relevant unit costs are shown in Tables 16–18.

Table 16. Unit transportation cost and maximum transportation capacity of shippers.

Mode of Transportation	Shippers	Shipper 1		Shipper 2	
		Unit Cost (USD/ton)	Max Capacity (ton)	Unit Cost (USD/ton)	Max Capacity (ton)
Generalized transportation method 1	Seller 1	220		200	
	Seller 2	250	1000	220	1500
	Seller 3	210		210	
Generalized transportation method 2	Seller 1	180		200	
	Seller 2	200	1200	210	1000
	Seller 3	210		220	

Table 17. Sales capacity and unit sales cost.

	Seller 1	Seller 2	Seller 3
unit product cost of sales (USD/ton)	80	90	85
Product Demand (ton)	1000	1200	800

Table 18. Unit cost and maximum transportation capacity of shippers.

Mode of Transportation	Shippers	Shipper 1		Shipper 2	
		Unit Cost (USD/ton)	Max Capacity (ton)	Unit Cost (USD/ton)	Max Capacity (ton)
Generalized transportation method 1	supplier 1	180	2000	190	2500
Generalized transportation method 2	supplier 2	210	2200	220	2000

The optimization model used here is:

$$\begin{aligned} \min ZC = & (216\beta_{1111} + 252\beta_{1121} + 228\beta_{1211} + 264\beta_{1221}) \times Q + \\ & 3700x_{1111} + 3740x_{1112} + 3695x_{1113} + 3660x_{1121} + 3690x_{1122} + 3695x_{1123} + \\ & 3680x_{1211} + 3710x_{1212} + 3695x_{1213} + 3680x_{1221} + 3700x_{1222} + 3705x_{1223}. \end{aligned}$$

The constraints are:

$$\begin{aligned} Q & \leq 4500 \\ x_{1111} + x_{1112} + x_{1113} & \leq 1000 \\ x_{1121} + x_{1122} + x_{1123} & \leq 1200 \\ x_{1211} + x_{1212} + x_{1213} & \leq 1500 \\ x_{1221} + x_{1222} + x_{1223} & \leq 1000 \\ x_{1111} + x_{1121} + x_{1211} + x_{1221} & = 1000 \\ x_{1112} + x_{1122} + x_{1212} + x_{1222} & = 1200 \\ x_{1113} + x_{1123} + x_{1213} + x_{1223} & = 800 \\ 1.2 \times Q & \leq 5000 \\ 1.2 \times \beta_{1111} \times Q & \leq 2000 \\ 1.2 \times \beta_{1121} \times Q & \leq 2200 \\ 1.2 \times \beta_{1211} \times Q & \leq 2500 \\ 1.2 \times \beta_{1221} \times Q & \leq 2000 \\ \beta_{1111} + \beta_{1121} + \beta_{1211} + \beta_{1221} & = 1 \end{aligned}$$

where:

$$Q = x_{1111} + x_{1112} + x_{1113} + x_{1121} + x_{1122} + x_{1123} + x_{1211} + x_{1212} + x_{1213} + x_{1221} + x_{1222} + x_{1223}.$$

where x_{ijnk} are non-negative integers, $k = 1, 2, 3; j = 1, 2; n = 1, 2; \beta_{1jnl}$ are non-negative numbers. The symbols used in the model are explained as follows:

ZC: Total supply chain cost;

x_{ijnk} : The number of i -th product delivered to the k -th seller use the n -th generalized transportation method by the j -th transporter;

β_{ijnk} : The ratio of j -th transporter using the n -th generalized transportation method from the l -th supplier to the r -th raw material to the manufacturer's demand for the kind of raw material.

It can be seen from the model that the objective function is nonlinear, and the constraints of suppliers and the transportation of raw materials are also nonlinear, so the model is a nonlinear mixed integer programming model.

We applied the proposed filled function algorithm to this supply chain model to optimize the total transportation cost. We used MATLAB2021b programming on a 64-bit Windows 10, Intel(R) Core(TM) i5-9400F CPU@2.90 GHz memory personal computer to calculate it, we executed 20 independent runs, and the results are listed in Table 19.

Table 19. The optimization results.

x_{1113}	x_{1121}	x_{1122}	x_{1213}	x_{1222}	β_{1111}	β_{1211}	ZC(million)
800	1000	200	0	1000	0.556	0.444	1171.8
423	1000	200	377	1000	0.556	0.444	1171.8
300	1000	200	500	1000	0.556	0.444	1171.8
17	1000	200	783	1000	0.556	0.444	1171.8
2	1000	200	798	1000	0.556	0.444	1171.8

Table 19 shows the optimization results of nonzero variables (the values of other variables are all 0). We can see that this supply chain has multiple optimal solutions. By careful observation, we can see that when x_{1121} , x_{1122} , x_{1222} , β_{1111} and β_{1211} are fixed to the values shown in Table 19, and x_{1113} and x_{1213} satisfy the following conditions:

$$x_{1113} + x_{1213} = 800.$$

Therefore, the minimum transportation cost of this example is USD 1171.8 million. In this case, $x_{1122} = 200$ means that the manufacturer in this supply chain should arrange shipper 1 to deliver 200 t of the product to the second seller using the second generalized mode of transportation. $\beta_{1211} = 0.444$ means that the manufacturer should let shipper 2 complete 44.4 percent of the transportation task using generalized transportation method 2.

We compare our results with the results from ref. [20], the proposed filled function algorithm in this paper finds multiple optimal solutions and takes less computational time. From Table 19, we can see that our algorithm successfully finds five optimal solutions while the algorithm in [20] only finds single optimal solution $x^* = (0, 0, 800, 0, 1000, 200, 0, 0, 0, 0, 1000, 0)$, $\beta^* = (0.556, 0, 0.444, 0)$. Moreover, the average running time of our algorithm is 1106 s, while the running time of [20] is 5128 seconds. Therefore, we can come to the conclusion that the filled function algorithm in this paper is more effective and efficient.

6. Conclusions

In this paper, we design a new filled function method to solve unconstrained global optimization problems. The new filled function has two advantages. First, it has no adjustable parameters to tune for different optimization problems; Secondly, the new filled function is continuous and differentiable, which are very good properties. These good

properties mean the filled function is less apt to produce extra fake local minimum during the optimization; further, more choices of local search methods, especially some efficient gradient-based ones, can be applied to make the optimization more efficient and effective. The proposed filled function algorithm is tested on widely used benchmark problems and is also applied to a supply chain problem. Numerical experiments show the algorithm is effective and efficient. However, we also notice that the filled function algorithm become less effective with high dimensional optimization problems. The reason may lie in two aspects: First, the search space grows exponentially with the increase in the distention, and secondly, the local search method is not efficient enough for high dimensional problems. The time complexity of the proposed filled function algorithm is dependent on the local search method adopted. Since the filled function designed in this paper has the advantage of continuous and differentiable, the efficient gradient-based algorithm BFGS Quasi-Newton Method can be used in the proposed model. BFGS is well known for its fast super-linear convergence speed, although its time complexity is $O(n^2)$. This may be one of the reasons that when the optimization problem grows, its performance degrades. The proposed algorithm mainly helps the optimization process to repeatedly escape from local optimal solutions to better ones to locate the global optimal solution. In this process, a different local search method can be used. In our future work, we will continue to work on this issue and design new and better local search methods to make the filled function algorithm more efficient and perform better on higher dimensional problems.

Author Contributions: Conceptualization, H.L. and S.X.; methodology, H.L. and S.X.; software, S.X.; validation, Y.C.; writing—original draft, H.L. and S.X.; writing—review and editing, Y.C. and S.T.; supervision, H.L.; funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (No.62002289).

Data Availability Statement: The authors confirm that the data supporting the findings of this study are available within the article.

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

References

1. Ge, R. A filled function method for finding a global minimizer of a function of several variables. *Math. Program.* **1990**, *46*, 191–204. [\[CrossRef\]](#)
2. Liu, X. Finding Global Minima with a Computable Filled Function. *J. Glob. Optim.* **2001**, *19*, 151–161. [\[CrossRef\]](#)
3. Liu, X. A Class of Continuously Differentiable Filled Functions for Global Optimization. *Syst. Man Cybern. Part A Syst. Hum. IEEE Trans.* **2008**, *38*, 38–47. [\[CrossRef\]](#)
4. Lin, H.; Wang, Y.; Fan, L.; Gao, Y. A new discrete filled function method for finding global minimizer of the integer programming. *Appl. Math. Comput.* **2013**, *219*, 4371–4378. [\[CrossRef\]](#)
5. Lin, H.; Gao, Y.; Wang, Y. A continuously differentiable filled function method for global optimization. *Numer. Algorithms* **2014**, *66*, 511–523. [\[CrossRef\]](#)
6. Gao, Y.; Yang, Y.; You, M. A new filled function method for global optimization. *Appl. Math. Comput.* **2015**, *268*, 685–695. [\[CrossRef\]](#)
7. El-Gindy, T.; Salim, M.; Ahmed, A. A new filled function method applied to unconstrained global optimization. *Appl. Math. Comput.* **2016**, *273*, 1246–1256. [\[CrossRef\]](#)
8. Ma, S.; Yang, Y.; Liu, H. A parameter free filled function for unconstrained global optimization. *Appl. Math. Comput.* **2010**, *215*, 3610–3619. [\[CrossRef\]](#)
9. Liu, H.; Wang, Y.; Guan, S.; Liu, X. A new filled function method for unconstrained global optimization. *Int. J. Comput. Math.* **2017**, *94*, 2283–2296. [\[CrossRef\]](#)
10. Pandiya, R.; Widodo, W.; Endrayanto, I. Non parameter-filled function for global optimization. *Appl. Math. Comput.* **2021**, *391*, 125642. [\[CrossRef\]](#)
11. Ahmed, A. A new parameter free filled function for solving unconstrained global optimization problems. *Int. J. Comput. Math.* **2021**, *98*, 106–119. [\[CrossRef\]](#)
12. Ahmed, A. A new filled function for global minimization and system of nonlinear equations. *Optimization* **2021**, *71*, 1–24. [\[CrossRef\]](#)
13. Pandiya, R.; Salmah; Widodo; Endrayanto, I. A Class of Parameter-Free Filled Functions for Unconstrained Global Optimization. *Int. J. Comput. Methods* **2022**, *19*, 2250003. [\[CrossRef\]](#)

14. Qu, D.; Shang, Y.; Zhan, Y.; Wu, D. A new parameter-free filled function for the global optimization problem. *Oper. Res. Trans.* **2021**, *25*, 89–95.
15. Wang, Y.J.; Zhang, J.S. A new constructing auxiliary function method for global optimization. *Math. Comput. Model.* **2008**, *47*, 1396–1410. [[CrossRef](#)]
16. Sahiner, A.; Abdulhamid, İ.A.M.; Ibrahem, S.A. A new filled function method with two parameters in a directional search. *J. Multidiscip. Model. Optim.* **2019**, *2*, 34–42.
17. Wu, X.; Wang, Y.; Fan, N. A new filled function method based on adaptive search direction and valley widening for global optimization. *Appl. Intell.* **2021**, *51*, 6234–6254. [[CrossRef](#)]
18. Liu, J.; Wang, Y.; Wei, S.; Sui, X.; Tong, W. A Filled Flatten Function Method Based on Basin Deepening and Adaptive Initial Point for Global Optimization. *Int. J. Pattern Recognit. Artif. Intell.* **2020**, *34*, 2059011. [[CrossRef](#)]
19. Lin, H.; Wang, Y.; Gao, Y.; Wang, X. A filled function method for global optimization with inequality constraints. *Comput. Appl. Math.* **2018**, *37*, 1524–1536. [[CrossRef](#)]
20. Qu, D.; Shang, Y.; Wu, D.; Sun, G. Filled function method to optimize supply chain transportation costs. *J. Ind. Manag. Optim.* **2022**, *18*, 3339. [[CrossRef](#)]
21. Yuan, L.; Li, Q. Filling function method for solving two-level programming problems. *J. Math.* **2022**, *42*, 153–161.
22. Wan, Z.; Yuan, L.; Chen, J. A filled function method for nonlinear systems of equalities and inequalities. *Comput. Appl. Math.* **2012**, *31*, 391–405. [[CrossRef](#)]
23. Ma, S.; Gao, Y.; Zhang, B.; Zuo, W. A New Nonparametric Filled Function Method for Integer Programming Problems with Constraints. *Mathematics* **2022**, *10*, 734. [[CrossRef](#)]
24. Yuan, L.y.; Wan, Z.p.; Tang, Q.h.; Zheng, Y. A class of parameter-free filled functions for box-constrained system of nonlinear equations. *Acta Math. Appl. Sin. Engl. Ser.* **2016**, *32*, 355–364. [[CrossRef](#)]
25. Zhang, Y.; Xu, Y.T. A one-parameter filled function method applied to nonsmooth constrained global optimization. *Comput. Math. Appl.* **2009**, *58*, 1230–1238. [[CrossRef](#)]
26. Zhang, Y.; Xu, Y.; Zhang, L. A filled function method applied to nonsmooth constrained global optimization. *J. Comput. Appl. Math.* **2009**, *232*, 415–426. [[CrossRef](#)]
27. Zhang, L.S.; Ng, C.K.; Li, D.; Tian, W.W. A New Filled Function Method for Global Optimization. *J. Glob. Optim.* **2004**, *28*, 17–43. [[CrossRef](#)]
28. Yang, Y.; Shang, Y. A new filled function method for unconstrained global optimization. *Appl. Math. Comput.* **2006**, *173*, 501–512. [[CrossRef](#)]