AIMS Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 10(10): 24270–24293.
DOI: 10.3934/math.20251076
Received: 23 August 2025
Revised: 13 October 2025
Accepted: 16 October 2025
Published: 23 October 2025

Research article

A simplified single-parameter filled function method for unconstrained global optimization

Zengfu Chao[*]

Faculty of Mathematical and Physical Sciences, Yibin University, Yibin 644007, China

[*] **Correspondence:** Email: chaozf@yibinu.edu.cn.

Abstract: To address the limitations of existing filled function methods—including complex multi-parameter tuning and ambiguous global-optimality verification—this paper proposes a simplified continuously differentiable filled function for unconstrained global optimization with only one interpretable parameter and no exponential or logarithmic terms. Under mild assumptions (continuous differentiability and coercivity of the objective function), we rigorously proved that the proposed function satisfies all essential axioms of a filled function, enabling explicit certification of global optimality. The resulting hybrid algorithm (SP-FFM) combines gradient-based local optimization with deterministic global search via grid sampling, requiring only a single grid-density parameter. This design eliminates the need for alternating between the original objective and auxiliary functions around local optima, significantly reducing computational effort and parameter-tuning complexity. Extensive numerical experiments on benchmark problems show that the algorithm converges to global minima within milliseconds, achieves a 100% success rate after grid refinement, and outperforms state-of-the-art methods in robustness and efficiency while maintaining insensitivity to initial points.

Keywords: global optimization; filled function; single-parameter; hybrid optimization; deterministic search; benchmark testing
Mathematics Subject Classification: 90C26, 90C30

1. Introduction

Global optimization problems have found extensive applications across various fields, including engineering [1], finance [2], transportation [3], and healthcare [4]. These problems can be formally expressed as the minimization of a function $f : \mathbb{R}^n \to \mathbb{R}$, subject to constraints in $\mathbb{R}^n$. However, many real-world optimization problems are non-convex and may contain numerous local minima, making it particularly challenging to escape from local optima and locate the global minimum [5]. Over the past decades, significant progress has been made in both theoretical and algorithmic aspects of global optimization. Existing methods can be broadly categorized into two classes: stochastic methods and deterministic approaches.

1.1. Stochastic methods

Stochastic methods, such as simulated annealing [6], genetic algorithms [7], ant colony optimization [8], grey wolf optimization [9], and the complex-valued artificial hummingbird algorithm [10], are probability-based techniques inspired by biological or physical phenomena. While these methods are relatively easy to implement and require minimal regularity conditions on the objective function, they often suffer from theoretical limitations, such as susceptibility to local optima and slow convergence rates [11]. Enhanced Social Learning Particle Swarm Optimization (ESLPSO) [12] recently offered a stability-guaranteed social learning Particle Swarm Optimization (PSO) with impressive accuracy in estimating photovoltaic (PV) parameters. Individual-Based Model Dynamic Multi-Swarm Snow Goose Algorithm (IBM-Dy-SGA) [13] has also been applied to optimize Proportional-Integral–Derivative (PID) parameters for high-speed rail platform doors in 10D–30D. However, like other stochastic swarms, both methods still depend on initial populations and random seeds, so the above-mentioned drawbacks remain.

1.2. Deterministic methods

In contrast to stochastic approaches, deterministic methods, including tunneling methods [5, 14, 15] and filled function techniques [16–20], typically exhibit faster convergence and demonstrate better capability to escape local minima. Among these, the filled function method has gained particular popularity due to its straightforward implementation. This approach systematically searches for improved local minima by constructing auxiliary functions at current local optima, then employing local optimization methods to minimize these auxiliary functions and obtain better initial points for the original problem.

The filled function method was first introduced by Ge [16] in 1983, and subsequent research has produced numerous variants [16, 17, 21–25]. Each variant presents unique advantages and limitations. For instance, the original filled function proposed by Ge [16] takes the form

$$P(x, x^*, r, p) = \frac{1}{r + f(x)} \exp\left(-\frac{\|x - x^*\|^2}{p^2}\right), \tag{1.1}$$

where r and p are parameters requiring careful selection. While this formulation successfully solved various standard test problems with $n > 2$ dimensions, both theoretical analysis and numerical experiments revealed that $P(x, x^*, r, p)$ and its gradient $\nabla P(x, x^*, r, p)$ exhibit slow variations, potentially leading to pseudo-minima or saddle points.

Several improvements have been proposed to address these limitations. El-Gindy et al. [26] developed a modified version eliminating the exponential term, though it still requires careful tuning of two parameters (r and α). Liu [27] introduced a single-parameter version that offers simpler implementation but suffers from discontinuity at certain points. Shang et al. [20] proposed a parameter-free continuously differentiable filled function without exponential or logarithmic terms, addressing the issue of parameter tuning and non-differentiability. Theoretical analysis and numerical verification show that their algorithm is feasible and effective, and it has been successfully applied to solve nonlinear equations and data fitting problems. More recently, Lin et al. [18] presented an improved continuous and differentiable single-parameter filled function, albeit with increased complexity.

In addition to these contributions, Yilmaz [28] introduced a new global optimization algorithm based on space-filling curves and an auxiliary function approach, which has shown promising results in various applications. This method combines the reducing dimension technique using space-filling curves with a continuously differentiable auxiliary function, providing a robust framework for solving unconstrained global minimization problems. The proposed algorithm is evaluated on various test problems and applied to economic load dispatch problems, yielding promising results.

Table 1 summarizes the advantages and disadvantages of the deterministic and meta-heuristic methods previously discussed.

Table 1. Advantages and disadvantages of deterministic and meta-heuristic unconstrained global optimization methods discussed in Sections 1.1 and 1.2.

Category	Representative work	Key advantages	Main drawbacks
Meta-heuristic (stochastic)	Simulated Annealing [6]	Gradient-free, simple implementation	Sensitive to cooling schedule, no iteration bound
	Genetic Algorithm [7]	Global exploration, easy parallelization	Premature convergence, parameter tuning load
	Ant Colony, GWO, Hummingbird [8–10]	Bio-inspired search, adaptive operators	Random-seed dependent, theoretical guarantee absent
	ESLPSO 2025 [12]	Stability guarantee, low RMSE on PV models	No worst-case bound, initial-population sensitive
	IBM-Dy-SGA 2025 [13]	Dynamic multi-swarm, 10D–30D ready	No theoretical certificate, population-based
Deterministic	Tunneling Methods [5, 14]	Finite-step certificate, fast local escape	Shape parameter tuning, gradient information needed
	Ge's Filled Function (1987) [16]	Straightforward concept, proven on $n > 2$	Two sensitive parameters (r, p), slow gradient variation
	Liu's Filled Function (2001) [27]	Only one coefficient	Discontinuous at some points
	Shang's Filled Function (2007) [29]	Single coefficient, no exponential	Requires global Lipschitz constant
	El-Gindy's Filled Function (2016) [26]	Removes exponential term	Still two parameters (r, α), manual calibration
	Lin's Filled Function (2019) [18]	Continuous and differentiable	Higher algebraic complexity
	Shang 2025 [20]	Parameter-free, effective	Limited complexity discussion
	Yilmaz 2025 [28]	Combines curves with functions	Limited convergence discussion

1.3. Our contribution

Although filled function methods have made notable progress, they still impose frustrating burdens: multiple coefficients that must be tuned by trial and error, exponential or logarithmic terms that overflow or flatten the landscape and create pseudo-minima, the absence of any computable stopping rule that forces premature termination by CPU or iteration limits, and single-parameter variants that sacrifice continuous differentiability or insist on global Lipschitz constants, thereby blocking efficient gradient-based solvers and leaving users without a verifiable global certificate.

To address these limitations, this paper proposes a simplified continuously differentiable filled function with only one interpretable parameter and no exponential or logarithmic terms. Under mild assumptions (continuous differentiability and coercivity of the objective function), we rigorously prove that the proposed function satisfies all essential axioms of a filled function, enabling explicit certification of global optimality. The resulting hybrid algorithm (SP-FFM) combines gradient-based local optimization (e.g., Broyden–Fletcher–Goldfarb–Shanno algorithm, BFGS) with deterministic global search via grid sampling, requiring only a single grid-density parameter. Extensive numerical experiments on classical benchmark functions demonstrate the method's effectiveness in achieving global convergence and robustness to initial points, while significantly reducing parameter-tuning effort compared to existing approaches.

2. A new filled function and its properties

2.1. Preliminaries

The global optimization problem (P) can be generally formulated as finding the global minimum of a function $f(x)$, where $x \in \mathbb{R}^n$. Mathematically, it can be expressed as

$$(P)\ \min_{x\in\mathbb{R}^n}\ f(x)\ .$$

To ensure the existence of a global optimal solution for problem (P) and the feasibility of the proposed filled function method, the objective function should satisfy the following basic assumptions.

Assumption 2.1. *$f(x)$ is a continuously differentiable function on $\mathbb{R}^n$.*

Assumption 2.2. *$f(x)$ is coercive on $\mathbb{R}^n$, i.e., $\lim_{\|x\|\to\infty} f(x) = +\infty$.*

Assumptions 2.1 and 2.2 imply that the function $f(x)$ has a global minimum in the interior Ω° of a bounded closed region $\Omega \subset \mathbb{R}^n$, and the function values on the boundary $\partial\Omega$ are greater than any interior point, i.e., $\min_{x\in\partial\Omega} f(x) > \max_{x\in\Omega^\circ} f(x)$. Therefore, problem (P) can be transformed into

$$(P^*)\ \min_{x\in\Omega}\ f(x)\ .$$

Assumption 2.3. *Let X^* be the set of all local minimizers of $f(x)$, and then $f^* = \{f(x)|x \in X^*\}$ is a finite set.*

Assumption 2.3 implies that $f(x)$ may have infinitely many local minimizers in the bounded region Ω, but it has only finitely many distinct function values at these minimizers. By constructing filled functions for finite times, we can always find the global minimum $m = \min\{f^*\}$ of $f(x)$ on Ω.

Based on these assumptions, [17] proposed the original concept of filled functions.

Definition 2.1. *[17] A function $F(x, x^*)$ is called a filled function of $f(x)$ at x^* if it satisfies*

(1) x^* is a maximizer of $F(x, x^*)$, and the whole basin B^* of $f(x)$ at x^* becomes part of a hill of $F(x, x^*)$.
(2) $F(x, x^*)$ has no minimizers or saddle points in any basin higher than B^*.
(3) If $f(x)$ has a basin lower than B^*, then there exists a point x' in such basin such that $F(x, x^*)$ is minimized along the line passing through x and x^*.

(For definitions of a basin and hill, refer to [17].)

With the development of filled functions, various scholars have proposed different forms of filled function concepts [18, 19, 21, 24]. This paper adopts the following definition from [19].

Definition 2.2. *[19] A function $F(x, x_1)$ is called a filled function of $f(x)$ at a local minimizer x_1 if it satisfies*

(1) x_1 is a strict local maximizer of $F(x, x_1)$;
(2) $F(x, x_1)$ has no stationary points in $S_1 = \{x | f(x) \geq f(x_1), x \in \Omega \setminus \{x_1\}\}$;
(3) If x_1 is not a global minimizer of $f(x)$, then $F(x, x_1)$ must have a local minimizer in $S_2 = \{x | f(x) < f(x_1), x \in \Omega\}$.

This definition preserves the essential properties while presenting the concept in a more mathematically rigorous and concise form.

Theorem 2.1. *Based on the definition and assumptions, the following conclusions hold:*

(1) $S_1 \subseteq \Omega$;
(2) $S_1 \cup \{x_1\}$ is closed, and S_2 is open;
(3) $\min_{x\in\Omega} f(x) = \min_{x\in S_2} f(x)$;
(4) The global minimizer of $f(x)$ on Ω is among its local minimizers.

2.2. *A simplified single-parameter filled function and theoretical analysis*

Based on Definition 2.2, we propose the following single-parameter filled function:

$$F(x, x^*) = \frac{1}{1 + \|x - x^*\|^2} + A \min_{x\in\Omega}\{0, (f(x) - f(x^*))^3\}. \tag{2.1}$$

We now prove that function 2.1 is indeed a filled function of $f(x)$ at x^*.

Theorem 2.2. *$F(x, x^*)$ is continuously differentiable on $\mathbb{R}^n$.*

Proof. Let $F_1(t) = \frac{1}{1+t}$ and $F_2(t) = A \min\{0, t^3\}$, and then function 2.1 can be rewritten as

$$F(x, x^*) = F_1(\|x - x^*\|^2) + F_2(f(x) - f(x^*)).$$

We have

$$\begin{aligned}
&\nabla(\|x - x^*\|^2) = 2(x - x^*),\\
&\nabla(f(x) - f(x^*)) = \nabla f(x),\\
&\nabla F(x, x^*) = 2F_1'(\|x - x^*\|^2)(x - x^*) + F_2'(f(x) - f(x^*))\nabla f(x).
\end{aligned}$$

If $f(x)$, $F_1(t)$ (for $t \geq 0$), and $F_2(t)$ are all continuously differentiable, then $F(x, x^*)$ is continuously differentiable.

For $t \geq 0$, $F_1(t) = \frac{1}{1+t}$ is continuously differentiable. The derivative of F_2 is

$$F_2'(t) = \begin{cases} 0, & t \geq 0, \\ 3At^2, & t < 0, \end{cases}$$

which shows $F_2(t)$ is also continuously differentiable.

Combined with Assumption 2.1, $F(x, x^*)$ is continuously differentiable, with

$$\nabla F(x, x^*) = -\frac{2(x - x^*)}{(1 + \|x - x^*\|^2)^2} + F_2'(f(x) - f(x^*))\nabla f(x).$$

■

Theorem 2.3. *If x^* is a local minimizer of $f(x)$, then x^* is a strict local maximizer of $F(x, x^*)$.*

Proof. Since x^* is a local minimizer of $f(x)$, there exists $\delta > 0$ such that for $x \in B(x^*, \delta)$, $f(x) > f(x^*)$. In this case, $F_2(t) = 0$, so

$$F(x, x^*) = \frac{1}{1 + \|x - x^*\|^2} < 1 = F(x^*, x^*).$$

Thus, x^* is a strict local maximizer of $F(x, x^*)$. ■

Theorem 2.4. *$F(x, x^*)$ has no stationary points in $S_1 = \{x | f(x) \geq f(x^*), x \in \Omega \setminus \{x^*\}\}$.*

Proof. By the definition of $F(x, x^*)$, when $x \in S_1$, we have

$$F(x, x^*) = \frac{1}{1 + \|x - x^*\|^2},$$

with its gradient

$$\nabla F(x, x^*) = -\frac{2(x - x^*)}{(1 + \|x - x^*\|^2)^2}.$$

Since $x \neq x^*$, the gradient satisfies

$$\nabla F(x, x^*) \neq 0.$$

Therefore, $F(x, x^*)$ has no stationary points in $S_1 = \{x \mid f(x) \geq f(x^*), x \in \Omega \setminus \{x^*\}\}$. ■

We define an effective direction $d(x)$ at x as one that moves away from x^*, ensuring that the algorithm will consistently move away from x^* when designing the search direction. By definition, $d^T(x - x^*) > 0$.

Combining with Theorem 2.4, for all $x \in S_1$, we have $d^T \nabla F(x, x^*) < 0$, meaning $F(x, x^*)$ is monotonically decreasing along the effective direction $d(x)$. Together with Theorems 2.3 and 2.4, we conclude that $F(x, x^*)$ has no local minimizers in $S_1 \cup \{x^*\}$.

Theorem 2.5. *If x^* is not a global minimizer of $f(x)$, then $F(x, x^*)$ has a local minimizer in $S_2 = \{x | f(x) < f(x^*), x \in \Omega\}$.*

Proof. By Assumption 2.2, $f(x)$ has a global minimizer in Ω. If x^* is not the global minimizer, there must exist another local minimizer x_1 with $f(x_1) < f(x^*)$, so $S_2 \neq \emptyset$.

By the continuity of $f(x)$ (Assumption 2.1) and Theorem 2.1.4, there exists $\delta_1 > 0$ such that for $x \in B(x_1, \delta_1) \subset S_2$, $f(x) < f(x^*)$, and

$$F(x, x^*) = \frac{1}{1 + \|x - x^*\|^2} + A(f(x) - f(x^*))^3.$$

Moreover, S_2 is bounded, so there exists $M > 0$ with $\max\limits_{x \in S_2} \|x - x^*\| \leq M$.

Let $B(x^*, M) \subset \Omega$ be the closed ball centered at x^* with radius M. Since $F(x, x^*)$ is continuous on $\mathbb{R}^n$, by the Weierstrass extreme value theorem, $F(x, x^*)$ attains its minimum on $B(x^*, M)$.

For all $x \in S_1$, $F(x, x^*) = \frac{1}{1+\|x-x^*\|^2} > 0$.

Let $A = \frac{1}{(f(x^*) - f(x_1))^3}$, and then

$$A(f(x_1) - f(x^*))^3 = -1,$$

$$F(x_1, x^*) = \frac{1}{1 + \|x_1 - x^*\|^2} + A(f(x_1) - f(x^*))^3 < 0.$$

Thus, the minimizer x_k must lie in S_2. Since S_2 is open (Theorem 2.1.2), $x_k \in S_2$, making x_k a local minimizer of $F(x, x^*)$. Therefore

$$\nabla F(x_k, x^*) = -\frac{2(x_k - x^*)}{(1 + \|x_k - x^*\|^2)^2} + 3A(f(x_k) - f(x^*))^2 \nabla f(x_k) = 0,$$

$$\nabla f(x_k) = \frac{2}{3A(f(x_k) - f(x^*))^2 (1 + \|x_k - x^*\|^2)^2} (x_k - x^*).$$

■

Remarks 2.1.

- *The scaling parameter A serves as a quantitative measure of the error between the current best solution x^* and potentially better minima. See Section 3 for a concrete, accuracy-based calibration.*
- *The existence of any x_1 with $f(x_1) < f(x^*)$ guarantees $\exists \delta > 0$ such that $F(x, x^*) < 0$ for all $x \in B(x_1, \delta)$. The global optimality of x^* is certified when $F(x, x^*) > 0$ throughout Ω.*
- *The complete theoretical verification in Theorems 2.3–2.5 establishes that our proposed function 2.1 strictly satisfies all conditions in Definition 2.2 .*

Theorem 2.6. *Let x^* be a global minimum point of the function $f(x)$. The necessary and sufficient condition is that for any positive number A, the inequality $F(x, x^*) > 0$ holds.*

Proof. (Necessity) Suppose x^* is a global minimum point of the function $f(x)$. Then, for any $x \neq x^*$, we have $f(x) > f(x^*)$. Therefore, we obtain

$$F(x, x^*) = \frac{1}{1 + \|x - x^*\|^2} > 0.$$

(Sufficiency) If x^* is not a global minimum point of $f(x)$, then by Theorem 2.5, there must exist a point x_1 such that $F(x_1, x^*) < 0$, which contradicts the given condition.

■

3. The filled function algorithm

In the previous section, rigorous theoretical analysis showed that the constructed filled function $F(x, x^*)$ satisfies all the properties required by Definition 2.2. Building on this foundation, this section will further elaborate on the development of a new hybrid single-parameter filled function algorithm—the simplified-parameter filled function method (SP-FFM), which is derived from the formalized filled function represented by Eq (2.1).

Let x^* denote the current local minimizer, where the filled function $F(x, x^*)$ attains a strict local maximum at x^* (3.1). Conventional approaches [16, 19, 30, 31] employ fixed-direction initialization

$$x_i = x^* + \delta e_i. \tag{3.1}$$

Here, δ is a small step size, and e_i is the direction of the i-th unit vector. Although this method is simple, it is difficult to find a better solution when the minimum point of the filled function $F(x, x^*)$ is not near the coordinate axis direction [32].

Consider the three-hump back camel function

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

Initializing at $[-3, -3]$, the BFGS algorithm converges to $x_1^* = [-1.74755239, -0.8737762]$, yielding a local minimum value of 0.2986384422368723 (Figure 1).

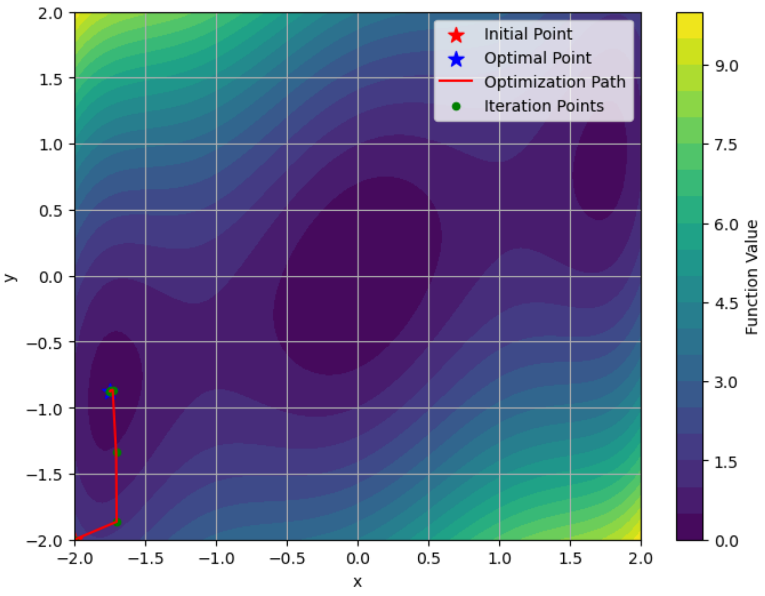

Figure 1. Optimization path of function f based on BFGS.

To escape from local minima, one can construct the corresponding filled function $F(x, x_1^*)$ according to Eq (2.1). Traditional algorithms require a slight shift from x_i^* according to Eq 3.1 to obtain a new initial point (the red pentagon stars x_1, x_2, x_3, x_4 in Figure 2), and then search for the local minimum of the filled function along the direction of the gradient descent. However, in such cases, if the optimal minimum point is far from the coordinate axis direction, it will cause the gradient descent process to fail to enter a more optimal region. Therefore, Wu [32] proposed a self-adaptive search strategy, generating uniformly distributed search directions around x_1^*, and added initial points (the green pentagon stars x_5, x_6, x_7, x_8 in Figure 2) on the basis of the classical algorithm, but did not specify how many to add.

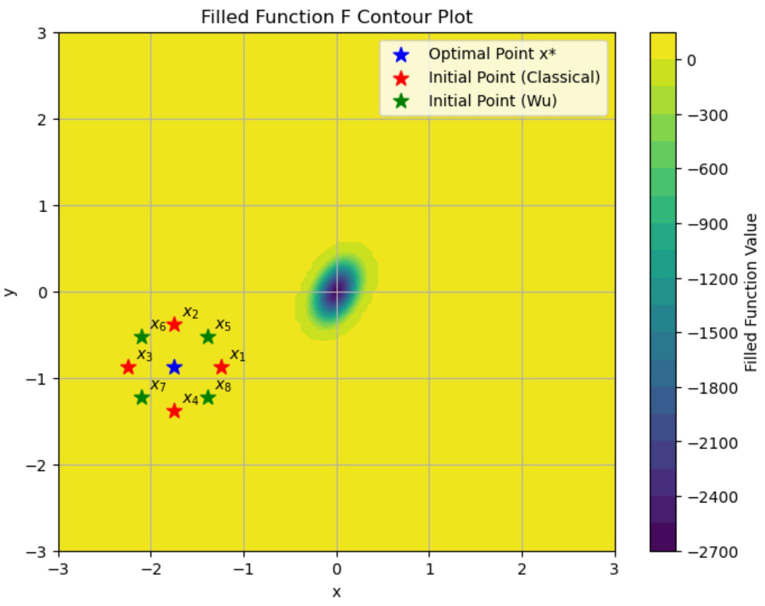

Figure 2. A minor deviation occurs at a local minimum point.

On the other hand, the region corresponding to the filled function S_2 is generally smaller, while the properties of the filled function in the S_1 region are mainly determined by $F_1(t)$, and the form of this function is relatively simple. It is not difficult to obtain from the gradient formula of the filled function that in the S_1 region, the direction of gradient descent is away from x^*, i.e., the direction of $(x - x^*)$. Therefore, in most cases, the significance of calculating the gradient is not great. At the same time, according to the theoretical proof process in this paper, it is known that in a certain domain where the local minimum value is smaller than the current local minimum value, the filled function $F(x, x_i^*) < 0$, while in S_1, $F(x, x_i^*) > 0$. This provides a way to escape from local optimality, i.e., as long as a point smaller than 0 can be found, one can escape from local optimality. If not, then the local optimal is the global optimal solution. Therefore, this paper uses a grid search [33], which is simple and intuitive, and convenient for parallel processing, and has a wide range of applications in machine learning and deep learning.

Taking the three-hump back camel function as an example, we perform the following steps:

- Set the initial point to $[-3, -3]$, and employ the BFGS algorithm to obtain the local minimum point $x_1^* = [-1.74755239, -0.8737762]$;
- Construct the filled function $F(x, x_1^*)$ at the local minimum point $x_1^* = [-1.74755239, -0.8737762]$, set $A = 1000$ (corresponding to target accuracy $\varepsilon \approx 0.1$; choose $A = \varepsilon^{-3}$ for any required gap ε, ensuring the obtained solution differs from the global optimum by at most ε);
- Set $k = 10$, obtain k^2 grid points on Ω, and it is easy to find point $x_1 = [0.30424534, 0.30154536]$ where $F(x, x_1^*) < 0$ (the red pentagon star in Figure 3) on these grid points;

- Take x_1 as the initial point, utilize the BFGS algorithm to find the minimum of $f(x, y)$, obtaining $x_2^* = [6.89984918 \times 10^{-8}, 2.72200279 \times 10^{-8}]$, and the corresponding local minimum value is $8.384372780392737 \times 10^{-15}$;
- Construct the filled function $F(x, x_2^*)$ at x_2^*;
- Set $n = 10$, obtain n^2 grid points on Ω, and fail to find points where $F(x, x_2^*) < 0$, thus x_2^* is identified as the global minimum point of the function $f(x, y)$, with the global minimum value being $8.384372780392737 \times 10^{-15}$. In fact, the global minimum point of the three-hump back camel function is known to be at $[0, 0]$, with the global minimum value being 0.

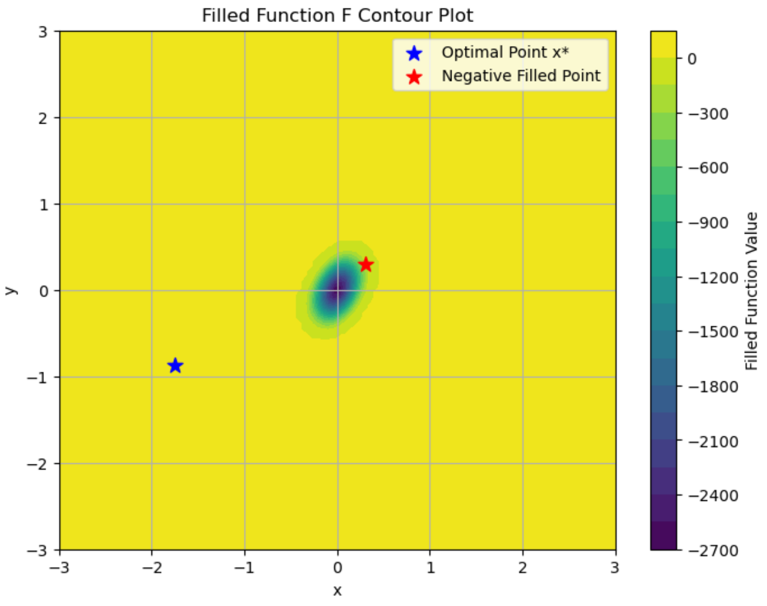

Figure 3. Escaping local minimum basins via negative filling point.

Below, we present the complete SP-FFM algorithm:

Algorithm 1 summarizes the complete SP-FFM procedure. The algorithm first invokes the BFGS solver to obtain a local minimizer x^* of the objective function f; then it constructs the filled function $F(x, x^*, f(x^*))$ and performs a grid-based search to detect any point with $F < 0$, i.e., a candidate lying in a deeper basin. If such a point is found, it becomes the starting point of the next local optimization; otherwise the iterative process stops and the best solution recorded so far is returned.

Remarks 3.1. *(Algorithm analysis and complexity)*

- *The filled parameter A reflects the advantage of the current minimum value. Theoretically, the difference between the current minimum value and the global minimum will not exceed* $\frac{1}{A}^{\frac{1}{3}}$. *Larger A values promote exploration of deeper basins by increasing sensitivity to better solutions. Since the target improvement is unknown in practice, users conservatively set* $A = \varepsilon^{-3}$ *for their required accuracy* ε*, ensuring any improvement* $\geq \varepsilon$ *is detected.*
- *The grid parameter n is the number of grids in each dimension, reflecting the smallest basin that the algorithm can find. Theoretically, the algorithm can find all basins with a diameter greater than* $\sqrt{\sum_{i=1}^{n} \frac{l_i}{k}}$, *where* l_i *is the length of the i-th dimension in the region* Ω.

Algorithm 1 Simplified Single-Parameter Filled-Function Method (SP-FFM)

```
Require: $f(x)$, $x_0$, bounds $\underline{x}, \overline{x}$, $A > 0$, grid density $n$, $max\_iter$
Ensure: best point $x_{best}$ and value $f_{best}$
 1: $t \leftarrow 0, \quad f_{best} \leftarrow +\infty$
 2: while $t < max\_iter$ do
 3:     $x^* \leftarrow \text{BFGS}(f, x_t)$                                    ▷ local minimizer
 4:     if $f(x^*) < f_{best}$ then
 5:         $(x_{best}, f_{best}) \leftarrow (x^*, f(x^*))$
 6:     end if
 7:     Build $F(x, x^*, f(x^*)) = \frac{1}{1+\|x-x^*\|^2} + A\min(0, f(x) - f(x^*))^3$
 8:     $P \leftarrow \{x \mid F(x, x^*, f(x^*)) < 0$ and $x$ on $n$-grid inside bounds$\}$
 9:     if $P = \emptyset$ then
10:         break                                                      ▷ no better basin detected
11:     else
12:         pick any $x_t \in P$                                       ▷ start next local search
13:     end if
14:     $t \leftarrow t + 1$
15: end while
16: return $x_{best}, f_{best}$
```

- *The time complexity of the grid search algorithm is $O(d^n)$, making it suitable for low-dimensional non-convex function global optimization problems. For high-dimensional scenarios, it can be modified to an adaptive random search algorithm.*
- *Compared to traditional algorithms, this hybrid approach only requires setting the grid density n and the filled parameter A, both of which are clearly defined and can be flexibly adjusted according to actual needs, significantly reducing the cost of tuning. The combination of gradient-based local search and gradient-free global exploration ensures efficient convergence.*

4. Numerical experiments

To evaluate the effectiveness of the proposed algorithm, we conducted numerical experiments on several typical test functions. These experiments were implemented using Python 3.12.3 and run on a desktop with an AMD Ryzen 5 2400G processor with Radeon Vega Graphics (3.60 GHz) and 16.0 GB of RAM.

Problem 1. (Two-dimensional Shubert function)

$$\begin{aligned} \min f(x) &= \{\sum_{i=1}^{5} i\cos[(i+1)x_1 + i]\}\{\sum_{i=1}^{5} i\cos[(i+1)x_2 + i]\} \\ \text{s.t.} &- 10 \le x_1, x_2 \le 10. \end{aligned} \tag{4.1}$$

The 2-dimensional Shubert function, with its 760 local minima, serves as a rigorous benchmark for evaluating global optimization algorithms. The global minimum value is -186.7309 [24, 32].

Two initial points selected for the SP-FFM are $(1,1)$ and $(4,4)$, where $(1,1)$ is the same as that in literatures [24] and [32]. The SP-FFM successfully finds the global minimum with $f(x^*) = -186.73090883102353$. The results of the SP-FFM and the algorithms proposed in literature [32] are displayed in Table 2.

Since the filled function method based on adaptive search direction and valley widening strategy (FFAW) [32] and the parameter-free filled function (PFFF) [24] are deterministic, their reported FEs depend solely on the starting point and parameter set; platform differences affect CPU time but not the iteration count. Thus the literature FE values in Table 2 are comparable to those of the SP-FFM without re-execution under identical hardware.

From the results, it can be found that the SP-FFM and the algorithms in [32] can successfully find the global minimum on this problem. The result of the SP-FFM using the initial point $(4,4)$ is shown in Table 3. From Tables 2 and 3, the SP-FFM can jump out of the local optimal solutions successfully from two different points.

Table 2. Results for Problem 1 with initial point $(1,1)$.

k	SP-FFM		FFAW [32]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix}1.08652152\\1.08652152\end{pmatrix}$	$2.8125603455779463 \times 10^{-21}$	$\begin{pmatrix}2.04669717\\2.04669717\end{pmatrix}$	$1.6324937065 \times 10^{-15}$
1	$\begin{pmatrix}-7.08350642\\2.78593446\end{pmatrix}$	-38.29597288161607	$\begin{pmatrix}0.00000000\\5.48286417\end{pmatrix}$	-64.6800711725
2	$\begin{pmatrix}-7.70831374\\3.28002671\end{pmatrix}$	-46.51131427388805	$\begin{pmatrix}5.48286414\\6.08779953\end{pmatrix}$	-123.5767708592
3	$\begin{pmatrix}-7.08350641\\4.85805687\end{pmatrix}$	-186.73090883102353	$\begin{pmatrix}5.48286418\\4.85805688\end{pmatrix}$	-186.730908831024

Table 3. Results for Problem 1 with initial point $(4,4)$.

k	x_k	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix}4\\4\end{pmatrix}$	$\begin{pmatrix}3.99024797\\3.99024797\end{pmatrix}$	$5.391438155433615 \times 10^{-14}$
1	$\begin{pmatrix}-10\\-1.11111111\end{pmatrix}$	$\begin{pmatrix}-7.08350642\\2.78593446\end{pmatrix}$	-38.29597288161607
2	$\begin{pmatrix}-7.77777778\\3.33333333\end{pmatrix}$	$\begin{pmatrix}-7.70831374\\3.28002671\end{pmatrix}$	-46.51131427388805
3	$\begin{pmatrix}-7.77777778\\5.55555556\end{pmatrix}$	$\begin{pmatrix}-7.08350641\\4.85805687\end{pmatrix}$	-186.73090883102353

Furthermore, we randomly selected 20 initial points (uniformly distributed within the search space), with the error tolerance set to 10^{-4}, and successfully identified the global optimum using the SP-FFM algorithm in all trials. The average computation time was 0.0111913800239563 seconds.

Problem 2. (Six-hump back camel function)

$$\begin{aligned} &\min f(x) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 - x_1x_2 - 4(1 - x_2^2)x_2^2 \\ &\text{s.t. } -3 \le x_1, x_2 \le 3. \end{aligned} \tag{4.2}$$

The global minimum is located at $x^* = (-0.0898, -0.7127)^T$ or $x^* = (0.0898, 0.7127)^T$, with a function value of $f(x^*) = -1.0316$ [31].

In the experiments, take $(-2, -1)$ and $(-2, 1)$ as the two different initial points which are the same as those in literature [32]. The experimental results are shown in Tables 4 and 5. Experimental results confirm that the SP-FFM reliably converges to the global minimum across both initialization points.

Table 4. Results for Problem 2 with initial point $(-2, -1)$.

k	SP-FFM		FFAW [32]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix} -0.08984202 \\ -0.71265641 \end{pmatrix}$	-1.0316284534898765	$\begin{pmatrix} 0.08984199 \\ 0.71265637 \end{pmatrix}$	-1.031628453489865
1			$\begin{pmatrix} 0.08984201 \\ 0.71265640 \end{pmatrix}$	-1.031628453489878

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

k	SP-FFM		FFAW [32]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix} 0.08984184 \\ 0.71265627 \end{pmatrix}$	-1.0316284534896283	$\begin{pmatrix} -0.08984199 \\ -0.71265637 \end{pmatrix}$	-1.031628453489870
1			$\begin{pmatrix} -0.08984201 \\ -0.71265640 \end{pmatrix}$	-1.031628453489878

Across 20 independent runs with uniformly random initializations, the SP-FFM located the true global optimum in 12 instances (average CPU time 4.06 ms, tolerance 1×10^{-4}). In the remaining eight runs, the algorithm converged to a spurious minimum at -0.21546. Figure 4 reveals that once the iterate reaches $(1.70360669, 0.79608359)$, the basin of attraction surrounding the global minimizer $(-0.0898, -0.7127)$ becomes extremely narrow, preventing the exploratory samples from entering it. Increasing the grid density parameter from $n = 10$ to $n = 20$ successfully eliminates this failure mode: all 20 runs now converge to the global optimum (average CPU time 6.08 ms). The finer discretization raises the computational cost by roughly 50% relative to the coarser setting.

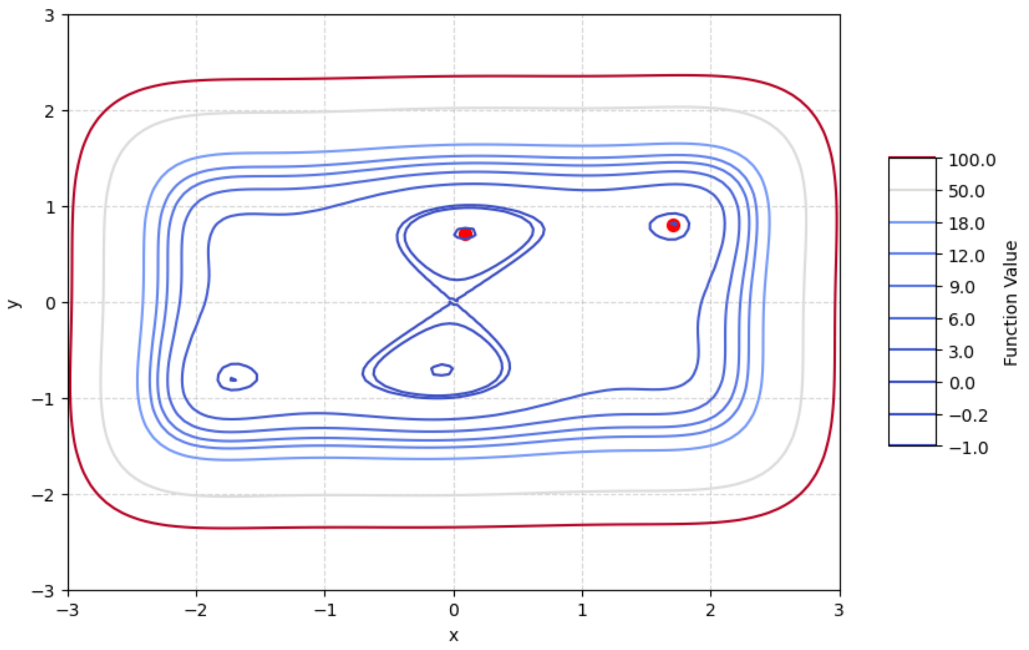

Figure 4. Contour plot of Problem 3.

Problem 3. (Treccani function)

$$
\begin{aligned}
&\min f(x) = x_1^4 + 4x_1^3 + 4x_1^2 + x_2^2 \\
&\text{s.t.} \; -3 \le x_1, x_2 \le 3.
\end{aligned}
\tag{4.3}
$$

The global minimum is located at $x^* = (0,0)^T$ or $x^* = (-2,0)^T$, with a function value of 0.

In this problem, we adopt the initial point $(-1,-2)$ as employed in the literature [32]. The experimental results are shown in Table 6. Experimental results confirm that the SP-FFM reliably converges to the global minimum from the given initialization point.

Table 6. Results for Problem 3 with initial point $(-1,-2)$.

k	SP-FFM		FFAW [32]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix} -0.999999941 \\ -2.53762337 \times 10^{-8} \end{pmatrix}$	1.0000	$\begin{pmatrix} 6.10441121 \times 10^{-8} \\ -1.56869273 \times 10^{-7} \end{pmatrix}$	3.9514×10^{-14}
1	$\begin{pmatrix} -1.99999891 \\ -1.84948901 \times 10^{-8} \end{pmatrix}$	4.7628×10^{-12}	$\begin{pmatrix} -7.91711881 \times 10^{-27} \\ 3.22874133 \times 10^{-28} \end{pmatrix}$	2.5072×10^{-52}

Furthermore, we randomly sampled and the proposed SP-FFM algorithm successfully converged to the global optimum in all trials (error tolerance 10^{-4}), with an average runtime of 2.33 milliseconds.

Problem 4. (Goldstein-Price function)

$$
\begin{aligned}
\min\ & f(x) = t_1(x) \times t_2(x) \\
& t_1(x) = 1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \\
& t_2(x) = 30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \\
\text{s.t.}\ & -3 \le x_1, x_2 \le 3.
\end{aligned}
\tag{4.4}
$$

The global minimum is located at $x^* = (0, -1)^T$, with a function value of 3 [21].

We first note that the function was not tested in [32]; therefore, we adopt the initial point $(-1, -1)$ used in [24] for a direct comparison. With a grid density of $n = 20$, the SP-FFM converged to the global minimizer $(-1.56836534 \times 10^{-8}, -1.00000002)$ in only two iterations (Table 7) with an objective value of 3.000000000000131 in 6.92 ms.

Table 7. Results for Problem 4 with initial point $(-1, -1)$.

k	SP-FFM		PFFF [24]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix}-0.60000001\\-0.4\end{pmatrix}$	30.000000000000036	$\begin{pmatrix}-0.6000\\-0.4000\end{pmatrix}$	30.0000
1	$\begin{pmatrix}-1.56836534 \times 10^{-8}\\-1.00000002\end{pmatrix}$	3.000000000000131	$\begin{pmatrix}0.0000\\-1.0000\end{pmatrix}$	3.0000

Next, starting from the initial point $(-2, 2)$, the SP-FFM reached the global minimizer $(-1.03 \times 10^{-8}, -1.00)$ in one iteration, achieving the same objective value of 2.999999999999997 in 7.13 ms. To examine robustness to initialization, 20 additional points were uniformly sampled from the region shown in Figure 5. Under the default setting ($n = 10$), the algorithm converged to the global optimum in 19 of 20 runs (tolerance 10^{-4}; average CPU time 7.30 ms). The sole failure started from $(-0.26358009, 1.71105577)$ and terminated at the local minimizer $(-0.6, -0.4)$ with an objective value of 30.000000000000075. Increasing the grid density to $n = 20$ removed this outlier, yielding global convergence in all 20 trials with an average runtime of 10.5 ms.

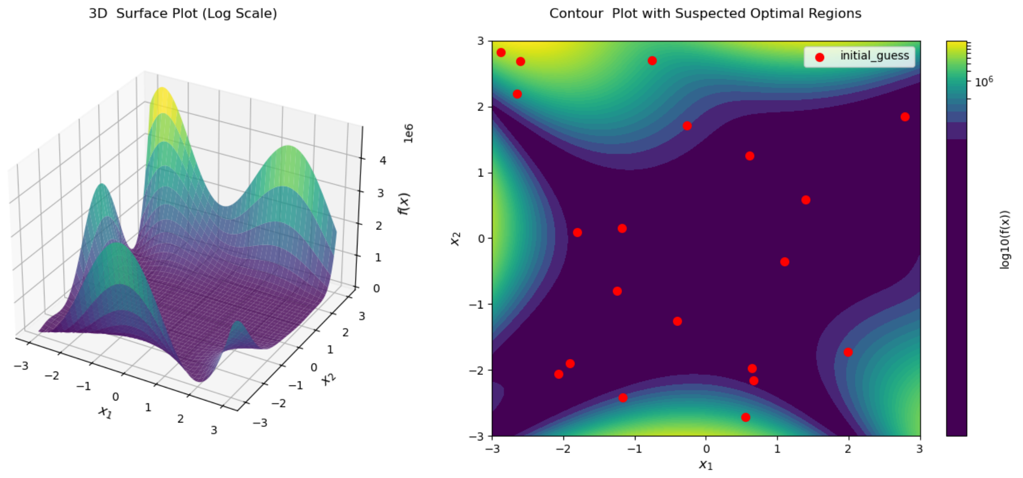

Figure 5. 3D surface and contour plot of the Goldstein-Price function with random initial points.

Problem 5. (n-dimensional function)

To further validate the effectiveness of the proposed algorithm in high-dimensional scenarios, we tested the algorithm on n-dimensional functions. The specific function is defined as follows:

$$\begin{aligned} \min\ f(x) = {} & \frac{\pi}{n}\{10\sin^2(\pi x_1) \\ & + \sum_{i=1}^{n-1}\left[(x_i-1)^2(1+10\sin^2(\pi x_{i+1})\right] \\ & + (x_n-1)^2\} \\ \text{s.t.}\ & -10 \le x_i \le 10, \quad i = 1, 2, \ldots, n. \end{aligned} \tag{4.5}$$

This function is frequently mentioned in high-order global optimization problems, with its global minimum located at the point $(1, 1, \ldots, 1)$ and the global minimum value being 0 [34].

We initially examined the $n = 2$ case across four representative starting points: $(-4, -4)$, $(2, 2)$, $(0, 0)$, and $(10, -2)$. Among these, the choice of $(-4, -4)$ is aligned with the starting point used in [32], facilitating a direct comparison with their results. The iteration trajectory for this specific starting point is detailed in Table 8, highlighting the effectiveness of our SP-FFM under the same conditions as those reported in their study. The detailed iteration trajectories for the remaining three points are reported in Table 9.

Table 8. Results for Problem 5 (n = 2) with initial point (−4, −4).

k	SP-FFM		FFAW [32]	
	x_k^*	$f(x_k^*)$	x_k^*	$f(x_k^*)$
0	$\begin{pmatrix}-3.94896524\\-3.99793237\end{pmatrix}$	78.12635489981538	$\begin{pmatrix}2.97983349\\2.99484261\end{pmatrix}$	12.4870637154
1	$\begin{pmatrix}-1.96972423\\0.00114755\end{pmatrix}$	15.56393532344705	$\begin{pmatrix}0.99999380\\0.99937733\end{pmatrix}$	$6.1505677343 \times 10^{-7}$
2	$\begin{pmatrix}0.99999998\\1.00000001\end{pmatrix}$	$3.8484089334481006 \times 10^{-14}$	$\begin{pmatrix}5.48286414\\6.08779953\end{pmatrix}$	$5.5318868676 \times 10^{-21}$

Table 9. Performance on Problem 5 (n = 2) across different initial points.

k	Initial	Iter	Process	Best solution	Time (ms)
1	(2, 2)	2	(1.98985997, 1.98975808) (0.99999999, 0.99999997)	$7.79824244679955 \times 10^{-15}$	2.04
2	(0, 0)	2	(0.0101401, 0.0102419) (0.99999999, 0.99999997)	$7.79824244679955 \times 10^{-15}$	8.44
3	(10, −2)	4	(9.90407911, −1.99961666) (−2.95942979, 3.9980629) (−0.97987294, 0.00257823) (0.99999999, 0.99999997)	$7.79824244679955 \times 10^{-15}$	11.68

To further systematically assess the sensitivity of our method to initialization, we randomly selected 20 additional starting points from the domain depicted in Figure 6. In each of these trials, the SP-FFM successfully converged to the global optimum within the specified tolerance of 10^{-4}, with an average CPU time of 10.03 ms. This outcome further underscores the robustness and efficiency of our method across a variety of starting conditions.

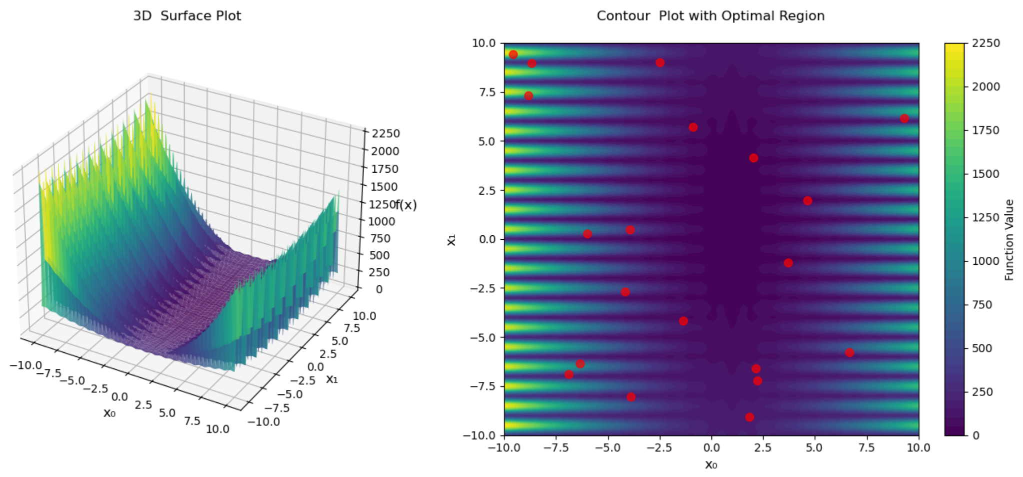

Figure 6. Surface and contour plot of Problem 5 for $n = 2$ with random initial points.

Next, we examine the $n = 5$ case. Three representative starting points were selected: (2, 2, 2, 2, 2),

$(0,0,0,0,0)$, and $(10,-8,2,-1,5)$. The proposed SP-FFM algorithm was applied to locate the global minimizer within the domain $(-10,10)^5$; the corresponding iteration histories are detailed in Table 10. To further probe the sensitivity to initialization, 20 additional points were sampled uniformly at random from the same domain. In every trial, the SP-FFM converged to the global optimum (error tolerance 10^{-4}), with an average CPU time of 8.3793 s.

Table 10. Performance on Problem 5 (n = 5) across different initial points.

k	Initial	Iter	Process	Best solution	Time (s)
1	(2,2,2,2,2)	2	$(1.98985779, 1.98965067, 1.98964642, 1.98964851, 1.98975373)$ $(0.99999999, 0.99999979, 1.00000013, 1.00000009, 1.00000012)$	6.5378×10^{-14}	2.1926
2	(0,0,0,0,0)	2	$(0.0101422, 0.01034931, 0.01035356, 0.01035148, 0.01024625)$ $(0.99999999, 0.99999979, 1.00000013, 1.00000009, 1.00000012)$	6.5378×10^{-14}	2.1943
3	$(10, -8, 2, -1, 5)$	6	$(9.90406656, -7.99884996, 1.99986979, -0.97967038, 4.98967815)$ $(-7.90407913, 3.99961665, 0.99999999, 1.00000069, 1.00000043)$ $(-2.9594382, -0.99870495, -0.99492644, -0.99490731, -0.99492012)$ $(-0.97983326, -0.99482954, -0.99491972, 0.99999999, 1.00000001)$ $(-0.97988607, 0.99999999, 0.99999999, 0.99999999, 1.00000002)$ $(0.99999999, 0.99999979, 1.00000013, 1.00000009, 1.00000012)$	6.5378×10^{-14}	4.2388

For the scenario where $n = 7$, we initiated our analysis with three distinct starting points: $(2, 2, 2, 2, 2, 2, 2)$, $(0, 0, 0, 0, 0, 0, 0)$, and $(10, -8, 2, -1, 5, -5, 1)$. The global optimum was reached after 2, 2, and 6 iterations, respectively, at the point $(0.99999999, 0.99999979, 1.00000013, 1.00000009, 1.00000012, 1.00000012, 1.00000012)$ with a function value of 0 taking times of 284.26 s, 288.62 s, and 545.78 s, respectively. Table 11 details the iterative progression of these initial points. Subsequently, to systematically evaluate the influence of initial values on optimization outcomes, we randomly sampled 20 initial points within the specified domain. Under the preset error tolerance of 10^{-4}, all 20 trials successfully converged to the global optimum. The average computational time across these experiments was 958.96 s, highlighting both robustness and computational expense of the proposed method. As shown in Figure 7, CPU time grows exponentially with n, explicitly demonstrating the performance and limitation of the current grid search for moderately higher dimensions.

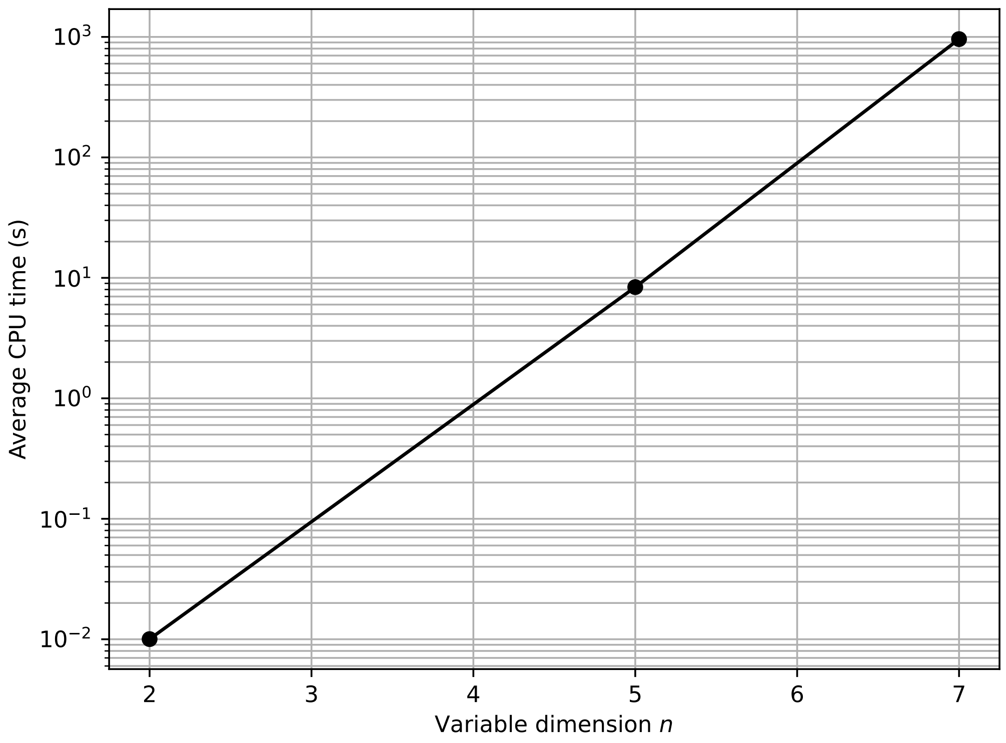

Figure 7. CPU time versus variable dimension n obtained by SP-FFM on Problem 5.

Table 11. Performance on Problem 5 (n = 7) across different initial points.

k	Initial	Ite	Process	Best solution	Time (s)
1	$(2,2,2,2,2,2,2)$	2	$(1.98985779, 1.98965067, 1.98964638, 1.98964629, 1.98964633, 1.9896485, 1.98975373)$ $(0.99999992, 0.99999886, 0.99999884, 1.00000287, 0.99999448, 1.0000024, 0.99999564)$	2.9999×10^{-11}	284.26
2	$(0,0,0,0,0,0,0)$	2	$(0.0101422, 0.01034931, 0.01035361, 0.01035369, 0.01035365, 0.01035148, 0.01024626)$ $(0.99999992, 0.99999886, 0.99999884, 1.00000287, 0.99999448, 1.0000024, 0.99999564)$	2.9999×10^{-11}	288.62
3	$(10,-8,2,-1,5,-5,1)$	6	$(9.90406656, -7.99884996, 1.99986979, -0.97966978, 4.98966333, -4.99618281, 0.99999999)$ $(-7.90408052, 1.32897109\times10^{-4}, -0.979827935, -0.994829513, -0.994919724, 0.999999993, 0.999999731)$ $(-2.9594382, -0.99870495, -0.99492644, -0.99490731, -0.99492012, 0.99999999, 0.99999986)$ $(-0.97983326, -0.99482954, -0.99491973, 0.99999999, 0.99999984, 0.99999963, 0.99999983)$ $(-0.97988608, 0.99999999, 1.0000004, 0.99999987, 0.99999979, 0.99999866, 1.00000016)$ $(0.99999994, 0.9999991, 1.00000168, 1.00000178, 1.00000177, 1.00000172, 1.00000107)$	6.4619×10^{-12}	545.78

Through numerical simulations, the single-parameter filled function algorithm demonstrates good performance on several typical test functions. On one hand, its performance is stable; setting the "fineness parameter" $n = 10$, in most cases (except for 8/20 random tests in the 6-hump back camel function and 1/20 random tests in the Goldstein Price function), the global optimum can be found. Setting $n = 20$, the global optimum is always obtained. On the other hand, the algorithm exhibits good robustness, almost ignoring the influence of the initial point selection on the algorithm's effect. However, when the dimension is high ($n > 5$), the "curse of dimensionality" gradually manifests, leading to significant time consumption during the search for the optimal value in the 7-dimensional function.

5. Conclusions

In this paper, we construct a single-parameter filled function for solving global optimization problems. This parameter has a clear practical significance, and the geometric interpretation of the filled function itself is intuitive, facilitating both understanding and algorithmic implementation. Notably, the proposed filled function does not contain exponential or logarithmic terms, ensuring numerical stability. Theoretically, this function satisfies the fundamental properties of filled functions, effectively enabling the original problem to escape local minima. Compared to traditional filled functions, our formulation additionally provides a criterion to determine whether the current minimum is the global minimum—a significant theoretical contribution.

Based on this single-parameter filled function, we design a novel hybrid filled function algorithm (the SP-FFM) that combines gradient-based local optimization with deterministic global search. In practical applications, the balance between computational efficiency and accuracy can be flexibly adjusted via a "fineness" parameter n. Numerical experiments on several classical global optimization problems demonstrate that the algorithm can locate the global optimum with very few iterations, confirming its feasibility. Moreover, when tested with different initial points (including randomly selected starting positions), the algorithm consistently converges to the global optimum, highlighting its strong robustness.

However, the current method is not yet efficient for large-scale global optimization; future work will continue to refine the filled function itself and integrate it with dimension-reduction techniques or meta-heuristics to extend its applicability in high-dimensional spaces, while also extending the present single-parameter filled function to constrained global optimization via penalty or augmented-Lagrangian methods to further broaden its scope.

Use of Generative-AI tools declaration

The author declares he has not used Artificial Intelligence (AI) tool in the creation of this article.

Acknowledgments

This work was supported by the Scientific Research Fund of Yibin University (2018YY04).

Conflict of interest

The author declares that there is no conflict of interest.

References

1. D. N. Truong, J. S. Chou, Metaheuristic algorithm inspired by enterprise development for global optimization and structural engineering problems with frequency constraints, *Eng. Struct.*, **318** (2024), 118679. https://doi.org/10.1016/j.engstruct.2024.118679

2. G. J. Wang, H. Huai, Y. Zhu, C. Xie, G. S. Uddin, Portfolio optimization based on network centralities: Which centrality is better for asset selection during global crises?, *J. Manage. Sci. Eng.*, **9** (2024), 348–375. https://doi.org/10.1016/j.jmse.2024.04.001

3. M. Elassy, M. Al-Hattab, M. Takruri, S. Badawi, Intelligent transportation systems for sustainable smart cities, *Transp. Eng.*, **16** (2024), 100252. https://doi.org/10.1016/j.treng.2024.100252

4. T. Qin, Z. Chen, J. D. Jakeman, D. Xiu, Data-driven learning of nonautonomous systems, *SIAM J. Sci. Comput.*, **43** (2021), A1607–A1624. https://doi.org/10.1137/20M1342859

5. A. V. Levy, A. Montalvo, The tunneling algorithm for the global minimization of functions, *SIAM J. Sci. Stat. Comput*, **6** (1985), 15–29. https://doi.org/10.1137/0906002

6. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680. https://doi.org/10.1126/science.220.4598.671

7. W. C. Yeh, M. C. Chuang, Using multi-objective genetic algorithm for partner selection in green supply chain problems, *Expert Syst. Appl.*, **38** (2011), 4244–4253. https://doi.org/10.1016/j.eswa.2010.09.091

8. N. K. Sreelaja, Ant colony optimization based light weight binary search for efficient signature matching to filter ransomware, *Appl. Soft Comput.*, **111** (2021), 107635. https://doi.org/10.1016/j.asoc.2021.107635

9. X. Liu, Y. Wang, M. Zhou, Dimensional learning strategy-based grey wolf optimizer for solving the global optimization problem, *Comput. Intell. Neurosci.*, **2022** (2022), 3603607. https://doi.org/10.1155/2022/3603607

10. L. Feng, Y. Zhou, Q. Luo, Y. Wei, Complex-valued artificial hummingbird algorithm for global optimization and short-term wind speed prediction, *Expert Syst. Appl.*, **246** (2024), 123160. https://doi.org/10.1016/j.eswa.2024.123160

11. Y. Zhang, L. Zhang, Y. Xu, New filled functions for nonsmooth global optimization, *Appl. Math. Model.*, **33** (2009), 3114–3129. https://doi.org/10.1016/j.apm.2008.10.015

12. L. Deng, S. Liu, Advancing photovoltaic system design: An enhanced social learning swarm optimizer with guaranteed stability, *Comput. Ind.*, **164** (2025), 104209. https://doi.org/10.1016/j.compind.2024.104209

13. D. Zhan, A. Q. Tian, S. Q. Ni, Optimizing PID control for multi-model adaptive high-speed rail platform door systems with an improved metaheuristic approach, *Int. J. Electr. Power Energy Syst.*, **169** (2025), 110738. https://doi.org/10.1016/j.ijepes.2025.110738

14. Y. Yao, Dynamic tunneling algorithm for global optimization, *IEEE Trans. Syst., Man, Cybern.*, **19** (1989), 1222–1230. https://doi.org/10.1109/21.44040

15. Y. T. Xu, Y. Zhang, S. G. Wang, A modified tunneling function method for non-smooth global optimization and its application in artificial neural network, *Appl. Math. Model.*, **39** (2015), 6438–6450. https://doi.org/10.1016/j.apm.2015.01.059

16. R. P. Ge, Y. F. Qin, A class of filled functions for finding global minimizers of a function of several variables, *J. Optim. Theory Appl.*, **54** (1987), 241–252. https://doi.org/10.1007/BF00939433

17. R. Ge, A filled function method for finding a global minimizer of a function of several variables, *Math. Program.*, **46** (1990), 191–204. https://doi.org/10.1007/BF01585737

18. H. Lin, Y.Gao, X. Wang, S. Su, A filled function which has the same local minimizer of the objective function, *Optim. Lett.*, **13** (2019), 761–776. https://doi.org/10.1007/s11590-018-1275-5

19. Q. Chen, X. M. Yang, Q. Yan, A new class of filled functions with two parameters for solving unconstrained global optimization problems, *J. Oper. Res. Soc. China*, **12** (2024), 921–936. https://doi.org/10.1007/s40305-024-00548-x

20. Y. Shang, D. Qu, J. Li, R. Zhang, A new parameter-free continuously differentiable filled function algorithm for solving nonlinear equations and data fitting problems, *J. Comput. Appl. Math.*, **454** (2025), 116198. https://doi.org/10.1016/j.cam.2024.116198

21. C. Wang, Y. Yang, J. Li, A new filled function method for unconstrained global optimization, *J. Comput. Appl. Math.*, **225** (2009), 68–79. https://doi.org/10.1016/j.cam.2008.07.001

22. S. Li, Y. L. Shang, D. Qu, A novel parameter-free filled function and its application in least square method, *Chinese Quart. J. Math.*, **36** (2021), 263–274. https://doi.org/10.13371/j.cnki.chin.q.j.m.2021.03.005

23. X. Liu, A class of generalized filled functions with improved computability, *J. Comput. Appl. Math.*, **137** (2001), 61–69. https://doi.org/10.1016/S0377-0427(00)00697-X

24. R. Pandiya, W. Widodo, Salmah, I. Endrayanto, Non parameter-filled function for global optimization, *Appl. Math. Comput.*, **391** (2021), 125642. https://doi.org/10.1016/j.amc.2020.125642

25. G. Sun, Y. Shang, X. Wang, R. Zhang, D. Qu, An efficient algorithm via a novel one-parameter filled function based on general univariate functions for unconstrained global optimization, *J. Comput. Appl. Math.*, **468** (2025), 116632. https://doi.org/10.1016/j.cam.2025.116632

26. T. El-Gindy, M. Salim, A. Ahmed, A new filled function method applied to unconstrained global optimization, *Appl. Math. Comput.*, **273** (2016), 1246–1256. https://doi.org/10.1016/j.amc.2015.08.091

27. X. Liu, Finding global minima with a computable filled function, *J. Global Optim.*, **19** (2001), 151–161. https://doi.org/10.1023/A:1008330632677

28. N. Yilmaz, A new global optimization algorithm based on space-filling curve and auxiliary function approach and its applications, *Commun. Nonlinear Sci. Numer. Simul.*, **149** (2025), 108920. https://doi.org/10.1016/j.cnsns.2025.108920

29. Y. L. Shang, D. G. Pu, A. P. Jiang, Finding global minimizer with one-parameter filled function on unconstrained global optimization, *Appl. Math. Comput.*, **191** (2007), 176–182. https://doi.org/10.1016/j.amc.2007.02.074

30. Q. Yan, W. Chen, X. Yang, A novel one-parameter filled function method with an application to pathological analysis, *Optim. Lett.*, **18** (2024), 803–824. https://doi.org/10.1007/s11590-023-02010-y
31. J. Li, Y. Gao, T. Chen, X. Ma, A new filled function method based on global search for solving unconstrained optimization problems, *AIMS Math.*, **9** (2024), 18475–18505. https://doi.org/10.3934/math.2024900
32. X. Wu, Y. Wang, N. Fan, A new filled function method based on adaptive search direction and valley widening for global optimization, *Appl. Intell.*, **51** (2021), 6234–6254. https://doi.org/10.1007/s10489-020-02179-0
33. Y. Zhao, W. Zhang, X. Liu, Grid search with a weighted error function: Hyper-parameter optimization for financial time series forecasting, *Appl. Soft Comput.*, **154** (2024), 111362. https://doi.org/10.1016/j.asoc.2024.111362
34. S. Ma, Y. Yang, H. Liu, A parameter free filled function for unconstrained global optimization, *Appl. Math. Comput.*, **215** (2010), 3610–3619. https://doi.org/10.1016/j.amc.2009.10.057

AIMS Press

©2025 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0)