Hindawi
Discrete Dynamics in Nature and Society
Volume 2021, Article ID 1336929, 13 pages
https://doi.org/10.1155/2021/1336929

Research Article

Mutation Strategy Based on Step Size and Survival Rate for Evolutionary Programming

Libin Hong,[1] **Chenjian Liu**,[1] **Jiadong Cui**,[2] **and Fuchang Liu**[1]

[1]*School of Information Science and Technology, Hangzhou Normal University, Hangzhou 311121, China*
[2]*College of Electronics and Information, Hangzhou Dianzi University, Hangzhou 310018, China*

Correspondence should be addressed to Fuchang Liu; liufc@hznu.edu.cn

Received 29 July 2021; Accepted 14 September 2021; Published 15 October 2021

Academic Editor: Shi Cheng

Copyright © 2021 Libin Hong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Evolutionary programming (EP) uses a mutation as a unique operator. Gaussian, Cauchy, Lévy, and double exponential probability distributions and single-point mutation were nominated as mutation operators. Many mutation strategies have been proposed over the last two decades. The most recent EP variant was proposed using a step-size-based self-adaptive mutation operator. In SSEP, the mutation type with its parameters is selected based on the step size, which differs from generation to generation. Several principles for choosing proper parameters have been proposed; however, SSEP still has limitations and does not display outstanding performance on some benchmark functions. In this work, we proposed a novel mutation strategy based on both the "step size" and "survival rate" for EP (SSMSEP). SSMSEP-1 and SSMSEP-2 are two variants of SSMSEP, which use "survival rate" or "step size" separately. Our proposed method can select appropriate mutation operators and update parameters for mutation operators according to diverse landscapes during the evolutionary process. Compared with SSMSEP-1, SSMSEP-2, SSEP, and other EP variants, the SSMSEP demonstrates its robustness and stable performance on most benchmark functions tested.

1. Introduction

Evolutionary programming (EP) is a major evolutionary algorithm that attempts to find a global optimum for benchmark functions; mutation is the only available operator in EP. Cauchy [1], Gaussian [1], Lévy [2], and double exponential [3] distributions are used as mutation operators in EP, and the popular EP variants include fast evolutionary programming (FEP), classic evolutionary programming (CEP), and Lévy-based evolutionary programming (LEP).

The early version of EP employs a single-mutation operator to optimize functions. While researchers observed that EP employing a single-mutation operator has limitations, Kumar [4] proposed that different mutation operators can generate different step sizes on average; however, the same mutation operator with different parameter values can generate various step sizes. For example, the Cauchy distribution can generate larger random values, thereby allowing individuals to jump from local optima, unlike the Gaussian distribution. The Lévy distribution is compatible with two features by controlling the value of α. Combining mutation operators was proposed in [4]; the mutation operators are classified into very small, small, medium, large, and very large mutation types. Researchers have proposed several mixed-mutation strategies for EP because different mutation operators have different characteristics. Improved FEP (IFEP) [5] uses both Gaussian and Cauchy distributions as mutations. Hong et al. [6] proposed a mixed-mutation strategy called evolutionary programming (MSEP), wherein the appropriate mutation operator is selected during evolution based on the probabilities of the four mutation operators. MSEP with a local fitness landscape (LMSEP) [7] is an upgraded mutation strategy in which a local fitness landscape is proposed based on MSEP. Ensemble strategies with adaptive EP [8] combine a population with mutation operators, allowing every population to benefit from the function calls. Hong et al. [9] proposed an EP variant that employs a mutation strategy which is step-size-based and

has a self-adaptive mechanism (SSEP). In recent years, researchers have proposed the automatic design of mutation operators and adaptive mutation operators for EP using hyper-heuristic/genetic programming [10–12].

The previously proposed different mutation strategies attempt to use different step size mutation operators during the evolutionary process; however, one of the issues is that mutation strategies may cause loss of step size control [13, 14]. We herein propose a novel mutation strategy that uses both "step size" and "survival rate" to control the selection of mutation operator/type for evolutionary programming (SSMSEP). Meanwhile, the standard deviation of the Gaussian distribution is updated if a Gaussian mutation is selected. In [9], principles were proposed to guide the mutation strategy. SSMSEP-1 and SSMSEP-2 are two variants of SSMSEP, which use either the survival rate or step size, respectively. In this study, the principles are optimized, and the experimental data illustrate that SSMSEP is more robust and stable than existing mixed-mutation strategies and single-mutation operators in most cases. The proposed mutation strategy conquers the loss of step size control on our tested benchmark functions.

Section 2 describes function optimization and the basic EP algorithm. The details of the mutation strategy of SSMSEP are introduced in Section 3. Section 4 describes the implementation of the proposed algorithm. In Section 5, experimental results are presented, and SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, CEP, FEP, and LEP with different α values are tested. A comparison among SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, MSEP, and LMSEP is also presented in this section. We explain and discuss future work in Section 6. Section 7 summarizes and concludes the paper.

2. Function Optimization and Evolutionary Programming

The global minimization in $\mathbb{R}^n$ can be formalized as a pair (S, f), where $S \in \mathbb{R}^n$ is a bounded set on $\mathbb{R}^n$ and $f: S \longrightarrow \mathbb{R}$ is an n-dimensional real-valued function. The objective is to obtain a point $x_{\min} \in S$ such that $f(x_{\min})$ is a global optimum on S. More explicitly, it is necessary to obtain $x_{\min} \in S$ such that

$$\forall x \in S: \ f(x_{\min}) \leq f(x). \tag{1}$$

Here, f must be bounded and it does not need to be continuous or differentiable. The EP algorithm is described as follows [1, 15]:

(1) Generate p individuals for the initial population and set $k = 1$. Each individual is taken as a pair of real-valued vectors. (x_i, η_i), $\forall i \in \{1, \ldots, p\}$. The strategy parameter η as the initial value was set to 3.0.

(2) Calculate the fitness value for each (x_i, η_i), $\forall i \in \{1, \ldots, p\}$.

(3) Each parent (x_i, η_i), $\forall i \in \{1, \ldots, p\}$, generates a single offspring $(x_i'(j), \eta'(j))$ (where $i = 1, \ldots, p$, $j = 1, \ldots, n$).

$$\begin{aligned} x_i'(j) &= x_i(j) + \eta_i(j) M_j, \\ \eta'(j) &= \eta_i(j) \exp\left(\gamma' N(0,1) + \gamma N_j(0,1)\right). \end{aligned} \tag{2}$$

The above two formulas are used to generate new offspring. The benchmark function is used to evaluate the fitness value. The factors γ and γ' are set as $(\sqrt{2\sqrt{n}})^{-1}$ and $(\sqrt{2n})^{-1}$.

(4) Calculate the fitness value for each offspring (x_i', η_i'), $\forall i \in \{1, \ldots, p\}$, according to $f(x')$.

(5) Conduct pairwise comparison over the set of parents (x_i, η_i) and offspring (x_i', η_i'), $\forall i \in \{1, \ldots, p\}$. For each individual, Y opponents were chosen randomly from the set of parents and offspring with equal probability. The individual gets a "reward" if its fitness value is smaller than that of the opponents in the comparison.

(6) Select the p individuals out of both parents and their offspring, $i \in \{1, \ldots, p\}$, which have the most rewards to be parents, for the next generation.

(7) Stop when the end condition is met; otherwise, k++ and go to Step 3.

It is CEP [1], FEP [5], and LEP [2] when M_j is a random number generated by Gaussian, Cauchy, and Lévy distributions, respectively. The above algorithm acts as a basic framework for SSMSEP, and the general parameter settings of EP are provided in Table 1.

3. Step Size and Survival Rate-Based Mutation Strategy

Here, we introduce the strategy used to design a step size and survival rate-based mutation strategy (SSMSEP) and explain how it works. The motivation of the SSMSEP is to solve the drawback of SSEP [9]. In SSEP, one of the issues is that the evolutionary process may have insufficient usage of long step-size-based mutation operators in the earlier generation of EP, which is also called loss of step size control [13, 14]. A typical case is the optimization of f_9; the mean best value over 50 runs is much worse than that of most single-mutation operators. From the experimental results, we observed that SSEP may fall into local optima in earlier generations because of insufficient mutation with a mutation operator which can generate a long step size. To solve this problem, we introduce the "survival rate" to control the mutation types in the proposed mutation strategy. The number of surviving offspring will be recorded to evaluate whether a long step size mutation is sufficient. In the proposed strategy, "step size" and "survival rate" are two keys to control mutation selection and parameter updating. Compared with SSEP, there are two significant changes.

(i) A "survival rate" is imported, and we work with "step size" to control mutation type selection.

(ii) Parameter calculation and updating strategy for mutation operators are proposed in each generation of EP.

Table 1: EP parameter settings.

Parameter	Settings
DIM	n in Table 2
POPNO	100
Tournament size	10
Lower bound	$1e-3$

Liang et al. [13, 14] proposed the lower bound control of the offspring to improve the EP performance and pointed out that self-adaptation may swiftly cause a search step size that is too narrow to further scan the search space, which is called the *loss of step size control*. In addition, Liang et al. [13, 14] analyzed how step size control was lost. Yao et al. [5] indicated that the Gaussian mutation is more likely to produce offspring closer to its parent than a Cauchy mutation. Thus, a Gaussian mutation is a better selection if the individual is near the global optimum; in contrast, the Cauchy mutation is a better selection for EP. Hong et al. [9] designed an SSEP that can explore space with a long step size at the outset and afterwards use a short step size mutation operator in the search, where "step size" denotes the distance between a "survived" offspring and its parent. In this paper, we propose using both "step size" and "survival rate" to control the mutation operator with parameter updating.

EP uses a static mutation operator, which leads to a few offspring surviving in the later generations of the run. A mutation strategy that switches mutation operators with related parameters is necessary to improve the performance of EP. The idea is to design an algorithm that can search a wider space to guarantee that more of the search space can be explored at the outset, and that search for a narrower space can be conducted later. $N(\mu, \sigma^2)$ is a Gaussian distribution, and $\mu = 0$ and $\sigma^2 = 1$ represent the standard normal distribution. Usually, $N(0, 1)$ represents a mutation operator that can generate a short step size, and the Cauchy distribution represents a long step size mutation operator. In SSMSEP, σ^2 is dynamically updated using the proposed equation. σ^2 is set to 0.1, 0.01, or 1 in SSEP, where the step size cannot effectively prevent the EP loss of step size control on some benchmark functions. In SSMSEP, "step size" and "survival rate" are combined and calculated to evaluate whether a mutation with a long step size is sufficient.

3.1. Symbols Used in the Novel Mutation Strategy. The symbols used by the mutation strategy are as follows:

(i) S_k is a single real value; it represents the step size at generation k. The new population comprises both "survived" parents and "survived" offspring after tournament selection at generation k; the parents and offspring which appear in the new population are called "survived" parents and offspring, respectively. S_k is evaluated as the mean **absolute** value of the jumped step size of all surviving offspring. This value is updated at generation k (in Algorithm 1 line 11), $\forall k \in \{1, \ldots, g\}$, where g is the maximum EP generation.

(ii) MD is taken as a single real-valued vector; it records the nonabsolute value of the step size MD_i of each individual i after tournament selection, $\forall i \in \{1, \ldots, p\}$, where p is the population number; MD_i can be either a positive or a negative value.

(iii) S_H_k is a single real value; it represents the mean step size from generation 1 to generation k. S_H_k is evaluated as $\mathrm{mean}(S_1, \ldots, S_k)$, where k is the number of current generations. This value is recalculated in every generation as well (in Algorithm 1 line 13).

(iv) SR_k is a single real value; it represents the survival rate at each generation k. Each parent has an offspring after mutation, and after tournament selection, both parent and offspring have the opportunity to be selected as parents for the next generation. In SSMSEP, the number of selected offspring (which is summed up) divided by the population size is called the survival rate.

(v) $N(\mu, \sigma^2)$ represents the Gaussian distribution; in SSMSEP, $\mu = 0$, and σ^2 is updated dynamically.

(vi) C indicates the Cauchy distribution.

(vii) T is the distance coefficient; it helps EP to control the usage of the long step size mutation operator, when S_k is very small, to prevent EP from being trapped in a local optimum prematurely. Table 3 provides the value of T which is empirically determined for each benchmark function. The values of T follow [9].

(viii) S_RATE represents survival rate; it is an empirically determined constant; it is set to 0.12.

3.2. Mutation Strategy for SSMSEP. The step size and survival rate-based mutation strategy for EP is as follows:

(i) If $S_H_k > 1$, $S_H_k < S_k \times T$, or survival rate $\mathrm{SR}_k \geq 0.12$ and the mutation type is Cauchy distribution in current generation, then Cauchy distribution is selected for the next generation.

(ii) If $S_k \in (0, 1e-2]$ and $S_H_k \geq S_k \times T$ and survival rate $\mathrm{SR}_k \geq 0.12$ at the current generation, Gaussian distribution is selected and set to σ^2 by the following equation, where p is the population number, for the next generation:

$$\sigma^2 = \sqrt{\left(\sqrt{\frac{\sum_{i=1}^{p}\left(\left|\mathrm{MD}_i\right| - \left(\sum_{i=1}^{p}\left(\left|\mathrm{MD}_i\right|\right)/p\right)\right)^2}{p}} * 100.\right)} \tag{3}$$

(iii) For all other cases, Gaussian distribution is selected and set $\mu = 0$ and $\sigma^2 = 1$ for $N(\mu, \sigma^2)$, as the mutation operator for EP.

In this study, for the earlier generation of EP, the Cauchy distribution was used as the mutation operator. We used both the step size and survival rate to evaluate whether the

Table 2: The 23 benchmark functions used in this study, where n is the benchmark function's dimensional size, f_{min} is the benchmark function's minimum value, and $S \subseteq \mathbb{R}^n$.

Function	n	S	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^n$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]^n$	0
$f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]^n$	0
$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	30	$[-100, 100]^n$	0
$f_5(x) = \sum_{i=1}^{n} [(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^n$	0
$f_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]^n$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]^n$	0
$f_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]^n$	−12569.5
$f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]^n$	0
$f_{10}(x) = -20\exp(-0.2\sqrt{(1/n)\sum_{i=1}^{n} x_i^2}) - \exp((1/n)\sum_{i=1}^{n} \cos 2\pi x_i) + 20 + e$	30	$[-32, 32]^n$	0
$f_{11}(x) = (1/4000)\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	30	$[-600, 600]^n$	0
$f_{12}(x) = (\pi/n)\{10\sin^2(\pi y_i) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1}) + (y_n - 1)^2]\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + (1/4)(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]^n$	0
$f_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]^n$	0
$f_{14}(x) = [(1/500) + \sum_{i=1}^{25} (1/(j + \sum_{i=1}^{2} (x_i - w_{ij})^6))]^{-1}$	2	$[-65.536, 65.536]^n$	0.998
$f_{15}(x) = \sum_{i=1}^{11} [w_i - (x_1(y_i^2 + y_i x_2)/(y_i^2 + y_i x_3 + x_4))]^2$	4	$[-5, 5]^n$	0.0003075
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + (1/3)x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^n$	−1.0316285
$f_{17}(x) = (x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6)^2 + 10(1 - (1/8\pi))\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$	2	$[-2, 2]^n$	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{4} a_{ij}(x_j - p_{ij})^2]$	3	$[0, 1]^n$	−3.86
$f_{20}(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2]$	6	$[0, 1]^n$	−3.32
$f_{21}(x) = -\sum_{i=1}^{5} [\sum_{j=1}^{4} (x_j - a_{ij}) + c_i]^{-1}$	4	$[0, 10]^n$	−10.15
$f_{22}(x) = -\sum_{i=1}^{7} [\sum_{j=1}^{4} (x_j - a_{ij}) + c_i]^{-1}$	4	$[0, 10]^n$	−10.34
$f_{23}(x) = -\sum_{i=1}^{10} [\sum_{j=1}^{4} (x_j - a_{ij}) + c_i]^{-1}$	4	$[0, 10]^n$	−10.54

Table 3: Values of distance coefficient T for $f_1 - f_{23}$ in SSMSEP, SSMSEP-1, SSMSEP-2, and SSEP.

f_n	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}	f_{23}
T	150	100	150	100	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150

```
(1)  T/*Set constant T*/
(2)  S_RATE/*Set constant S_RATE*/
(3)  Update both popStepSize[1...POPNO] and popStepSizeRaw[1...POPNO] after tournament selection;
(4)  MD[1, ..., POPNO] = popStepSizeRaw[1...POPNO];
(5)  for i: POPNO do
(6)    if pop[i] is offspring then
(7)      totalPopStepSize + = popStepSize[i];
(8)      survivedOffspringNo ++;
(9)    end if
(10) end for
(11) S_k = totalPopStepSize/survivedOffspringNo;/*Renew S_k*/
(12) SR = survivedOffspringNo/POPNO;/*Renew survival rate*/
(13) S_H_k = mean(S_1, ..., S_k);/*Renew S_H_k*/
(14) if S_H_k > 1 or S_H_k < S_k × T or (muttype == 1 and SR ≥ S_RATE) then
(15)   muttype←1;/*Set mutation type C*/
(16) else if S_k ≤ 1e − 2 and S_k > 0 and S_H_k ≥ S_k × T and SR ≥ S_RATE then
(17)   muttype←2;/*Set mutation type N(μ, σ^2)*/
(18)   σ^2 ← sqrt((sqrt((Σ_{i=1}^{POPNO} (|MD_i| − (Σ_{i=1}^{POPNO} (|MD_i|)/POPNO))^2/POPNO)) * 100;/*Renew σ^2 for N(μ, σ^2)*/
(19) else
(20)   muttype←2;/*Set mutation type N(μ, σ^2)*/
(21)   σ^2←1;/*Set 1 for σ^2*/
(22) end if
```

Algorithm 1: Algorithm of SSMSEP.

Cauchy distribution was sufficiently applied. 0.12 is an empirically determined value based on a large number of experiments. When the step size is sufficiently small, a Gaussian distribution is applied with the updated σ^2 for each generation. Instead of fixing the value of σ^2, which is set to 0.1 and 0.01, we control the shape of the Gaussian distribution in SSEP. The proposed strategy can also effectively avoid getting trapped in local optima during the evolutionary process.

4. SSMSEP Implementation

In this section, we present the pseudocode of the SSMSEP algorithm designed in accordance with the description of the mutation strategy in Section 3.2. The absolute value and non-absolute value of the step size for the population at each generation of EP are calculated using Algorithm 2. The calculation of S_k and S_H_k in Algorithm 1 uses the values prepared in Algorithm 2. Algorithm 1 implements the strategy proposed in Section 3.2 and describes how to switch the mutation operators with related parameters. Algorithms 1 and 2 are inserted into the EP algorithm described in Section 2.

To better observe the influence of "step size" and "survival rate" on the algorithm, we designed two variants of SSMSEP: SSMSEP-1 only uses "survival rate" (muttype == 1 and SR ≥ S_RATE in line 14, SR ≥ S_RATE in line 16 in Algorithm 1) to select the mutation operator; SSMSEP-2 only uses "step size" ($S_H_k > 1$ or $S_H_k < S_k \times T$ in line 14, $S_k \leq 1e-2$ and $S_k > 0$ and $S_H_k \geq S_k \times T$ in line 16 in Algorithm 1) to select the mutation operator.

5. Experimental Results

Table 2 lists the 23 benchmark functions, which are also commonly used by other researchers [1, 5, 7, 9] in experiments. The benchmark functions include unimodal benchmark functions, multimodal benchmark functions with many local optima, and multimodal benchmark functions with a few local optima, specified as $f_1 - f_7$, $f_8 - f_{13}$, and $f_{14} - f_{23}$, respectively [5]. The results of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, EP using Cauchy distribution (FEP), EP using Gaussian distribution (CEP), and EP using Lévy distributions with different α values (1.2,1.4,1.6, and 1.8) are provided for each function in Table 4. We retain 4 digits after the decimal point on most benchmark functions in this table.

The results of the Wilcoxon signed-rank test within a 95% confidence interval among SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, and other single-mutation operators are given in Table 5. "≥" and "≤" indicate that SSMSEP can perform better or worse on average, respectively, compared

```
(1) DIM/*Set dimensional size*/
(2) POPNO/*Set population size*/
(3) pop[POPNO × 2][DIM]/*Parents and offspring in present generation*/
(4) for i: POPNO do
(5)    indStepSize = 0;
(6)    for j: DIM do
(7)       temp(i, j) = pop[i][j] - pop[POPNO + i][j];/*Compute single step size for an individual*/
(8)       indStepSize + = abs(temp(i, j));/*Compute value of total absolute step size for overall individuals*/
(9)       indStepSizeRaw + = temp(i, j);/*Compute value of total step size for overall individuals*/
(10)   end for
(11)   popStepSize(POPNO + i) = indStepSize/DIM;/*Compute mean absolute step size for pop[i]*/
(12)   popStepSizeRaw(POPNO + i) = indStepSizeRaw/DIM;/*Compute mean non-absolute step size for pop[i]*/
(13) end for
```

Algorithm 2: Algorithm to calculate the absolute value and the non-absolute value of the step size for a population.

to a specific existing mutation strategy or mutation operator in this table; ">" and "<" denote statistical significance for better or worse on average, respectively.

5.1. Comparison and Analysis. The experimental data in Table 4 show that SSMSEP achieves the best performance on average compared with SSMSEP-1, SSMSEP-2, SSEP, and EP using a single-mutation operator on f_1, f_4, f_6, f_{10}, f_{13}, and f_{20}, the second-best performance on average of f_8, and the third-best performance on average of f_2, f_9, and f_{11}. The SSMSEP provides the best fitness value on f_6, f_7, and f_8 and the third-best fitness value on f_1, f_2, f_4, f_{10}, f_{12}, and f_{13}. f_{16}, f_{17}, f_{18}, and f_{19} have equivalent values for both the averaged values and best fitness values over 50 independent runs. f_{20} and f_{21} have equivalent best fitness values for all EP variants except for FEP, and f_{22} and f_{23} have equivalent best fitness values for all EP variants.

Table 5 lists the results of the 2-tailed Wilcoxon signed-rank tests, showing a statistically significant difference among SSMSEP and SSMSEP-1, SSMSEP-2, SSEP, CEP, FEP, and LEP variations. SSMSEP is significantly better than SSMSEP-1 on f_1, f_2, f_3, f_4, f_6, f_7, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, and f_{15}; it shows better performance on f_{20}, f_{21}, f_{22}, and f_{23}. SSMSEP is significantly better than SSMSEP-2 on f_8, f_9, and f_{20}; it shows better performance than SSMSEP-2 on f_1, f_4, f_5, f_{10}, f_{11}, f_{13}, and f_{23}. SSMSEP is significantly better than SSEP on f_4 and f_9; it shows better performance than SSEP on f_1, f_8, f_{10}, f_{13}, f_{14}, f_{15}, f_{20}, and f_{23}.

Table 6 gives function evaluations (FEs) of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, CEP, FEP, and LEP with $\alpha = (1.2, 1.4, 1.6, 1.8)$.

The mean best fitness values of $f_1 - f_{10}$ over 50 runs for each generation are plotted in Figures 1 and 2. SSMSEP not only demonstrates a consistent convergence rate for f_1, f_2, f_4, f_7, f_8, f_9, and f_{10} throughout the evolutionary process but also shows its robustness when compared with SSMSEP-1, SSMSEP-2, and SSEP. SSMSEP shows the stability of the convergence rate on f_1, f_4, f_9, and f_{10} in both the earlier and later generations of EP.

5.2. Comparison of Existing EP Mutation Strategies. SSEP, MSEP, and LMSEP are recent EPs that use mutation strategies with different mutation operators. The experimental results of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, LMSEP, and MSEP are displayed in Table 7, among which SSMSEP demonstrates the best performance on f_1, f_4, f_9, and f_{10}. Compared with SSEP, SSMSEP significantly improves f_1, f_4, f_9, and f_{10}. SSMSEP demonstrates the second-best performance on f_7 and third-best performance on f_2, f_{11}, f_{12}, and f_{15}. SSMSEP, SSMSEP-2, SSEP, and LMSEP exhibit equivalent performance on f_6. The EP variants demonstrate equivalent performance on f_{16}, f_{18}, and f_{19}. SSMSEP demonstrates outstanding performance on f_9 among all compared mixed-mutation strategies (including SSMSEP-1 and SSMSEP-2) in this study. Overall, the mutation strategy of SSMSEP provides very competitive performance on unimodal benchmark functions and multimodal functions with many local optima. We believe that the proposed SSMSEP can better fit the optimized two types of benchmark functions in various stages of EP. LMSEP performs better on average on multimodal functions with a few local optima (especially on f_{15}, f_{21}, f_{22}, and f_{23}). Reference [7] does not provide results for f_3, f_5, f_8, f_{13}, f_{14}, and f_{17}; thus here we only list the benchmark functions with existing results.

From the observation of the experimental results, the SSMSEP overcomes "loss of step size control" on most of the benchmark functions and benefits from the "survival rate," compared with SSEP, SSMSEP-1, and SSMSEP-2. The experimental results of f_5 are an exception in Table 4; SSMSEP-1 demonstrates the best performance on f_5, but its performance is worse than or equivalent to SSMSEP on the rest of the benchmark functions; LEP with $\alpha = 1.4$ also demonstrates a much better performance on f_5. We think

Table 4: The results of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, FEP, CEP, and LEP with $\alpha = (1.2, 1.4, 1.6, 1.8)$ on $f_1 - f_{23}$, where "Mean" denotes the mean best fitness values calculated in the final generation over 50 runs, "Min" denotes the minimum fitness value in final generation in 50 runs, and "Std Dev" denotes standard deviations.

	f_n	SSMSEP	SSMSEP-1	SSMSEP-2	SSEP	FEP	CEP	$\alpha = 1.2$	$\alpha = 1.4$	$\alpha = 1.6$	$\alpha = 1.8$
f_1	Mean	**3.7814E − 08**	1.0901$E-04$	1.4066$E-07$	5.5197$E-08$	4.5635$E-04$	6.0315$E-05$	2.5421$E-04$	1.6934$E-04$	1.3440$E-04$	1.0516$E-04$
	Min	1.4472$E-08$	2.5524$E-08$	1.3595$E-08$	**7.0480E − 09**	3.0863$E-04$	3.2239$E-05$	1.5908$E-04$	1.2202$E-04$	8.8775$E-05$	6.3256$E-05$
	Std Dev	(3.7792$E-08$)	(2.3378$E-04$)	(6.1386$E-07$)	(6.5450$E-08$)	(7.0235$E-05$)	(2.5166$E-05$)	(3.4971$E-05$)	(2.6149$E-05$)	(2.7323$E-05$)	(1.8110$E-05$)
f_2	Mean	3.6408$E-04$	4.0167$E-04$	3.2274$E-04$	**2.4660E − 04**	7.0726$E-02$	2.2304$E-02$	5.1594$E-02$	4.2239$E-02$	3.6880$E-02$	3.3471$E-02$
	Min	2.9214$E-04$	3.3439$E-04$	2.6128$E-04$	**2.0121E − 04**	5.1482$E-02$	1.8824$E-02$	4.3426$E-02$	3.3697$E-02$	3.2571$E-02$	2.6264$E-02$
	Std Dev	(3.0086$E-05$)	(3.0845$E-05$)	(2.8645$E-05$)	(1.5612$E-05$)	(5.5881$E-03$)	(1.5150$E-03$)	(3.2306$E-03$)	(3.0114$E-03$)	(2.1175$E-03$)	(2.5190$E-03$)
f_3	Mean	1.0407$E-02$	4.8819$E-02$	9.0411$E-03$	9.3736$E-03$	8.7540$E-03$	1.0524$E-02$	9.3985$E-03$	7.1744$E-03$	4.2293$E-03$	**3.0734E − 03**
	Min	2.4395$E-03$	8.1053$E-04$	2.4140$E-03$	2.3030$E-03$	1.9117$E-03$	**8.5077E − 05**	9.4563$E-04$	7.5277$E-04$	4.5609$E-04$	2.2686$E-04$
	Std Dev	(6.7484$E-03$)	(5.7250$E-02$)	(5.5701$E-03$)	(6.4085$E-03$)	(6.4514$E-03$)	(1.9002$E-02$)	(1.2368$E-02$)	(1.1998$E-02$)	(6.5552$E-03$)	(5.2651$E-03$)
f_4	Mean	**6.4065E − 04**	9.7131$E-01$	8.8194$E-04$	1.4405$E-03$	6.7516$E-03$	1.2210$E+00$	6.5288$E-03$	5.4754$E-02$	2.8222$E-01$	5.9637$E-01$
	Min	3.1376$E-05$	2.8928$E-03$	2.8940$E-05$	**2.1072E − 05**	5.4115$E-03$	1.4146$E-01$	4.2954$E-03$	3.2552$E-03$	3.6687$E-03$	3.1680$E-03$
	Std Dev	(1.8023$E-03$)	(8.1875$E-01$)	(1.4856$E-03$)	(4.0274$E-03$)	(7.8715$E-04$)	(9.0747$E-01$)	(5.6855$E-03$)	(1.4088$E-01$)	(4.8753$E-01$)	(6.0409$E-01$)
f_5	Mean	2.9275$E+01$	**3.1508**	2.9490$E+01$	2.1359$E+01$	2.9350$E+01$	7.5479	1.9830$E+01$	6.3888	8.0059	1.1073$E+01$
	Min	4.6981$E-02$	**9.9139E − 04**	1.5319$E-03$	5.5730$E-02$	4.7199$E-01$	1.8391$E-02$	3.3474$E-01$	1.0103$E-01$	8.0644$E-02$	4.2083$E-02$
	Std Dev	(3.2710$E+01$)	(1.0064$E+01$)	(3.1230$E+01$)	(2.6718$E+01$)	(3.2291$E+01$)	(1.4348$E+01$)	(2.3933$E+01$)	(6.0507$E+00$)	(1.3649$E+01$)	(1.9445$E+01$)
f_6	Mean	0	64.98	0	0	0	136.14	0	0	0.62	29.88
	Min	0	0	0	0	0	0	0	0	0	0
	Std Dev	(0)	(128.02)	(0)	(0)	(0)	(367.07)	(0)	(0)	(2.62)	(183.78)
f_7	Mean	8.4609$E-03$	2.8136$E-02$	8.1618$E-03$	**7.9013E − 03**	8.3416$E-03$	2.2382$E-02$	8.4703$E-03$	1.1287$E-02$	1.2451$E-02$	1.5274$E-02$
	Min	**2.7008E − 03**	1.4340$E-02$	4.0161$E-03$	3.7373$E-03$	3.7395$E-03$	8.4446$E-03$	4.7781$E-03$	4.8277$E-03$	5.7806$E-03$	7.6249$E-03$
	Std Dev	(2.6809$E-03$)	(9.8926$E-03$)	(2.1752$E-03$)	(2.1289$E-03$)	(2.8422$E-03$)	(7.6244$E-03$)	(2.4166$E-03$)	(3.6673$E-03$)	(3.9032$E-03$)	(5.3375$E-03$)
f_8	Mean	−1.1243$E+04$	−9.0460$E+03$	−1.1053$E+04$	−1.1113$E+04$	**−1.1300E + 04**	−7.9096$E+03$	−1.0582$E+04$	−1.0070$E+04$	−9.4480$E+03$	−8.7650$E+03$
	Min	**−1.1977E + 04**	−1.1145$E+04$	−1.1740$E+04$	−1.1976$E+04$	**−1.1977E + 04**	−9.0739$E+03$	−1.1661$E+04$	−1.0892$E+04$	−1.0870$E+04$	−1.0418$E+04$
	Std Dev	(2.7196$E+02$)	(6.4104$E+02$)	(3.5576$E+02$)	(4.0896$E+02$)	(2.7570$E+02$)	(5.9723$E+02$)	(4.8018$E+02$)	(4.3464$E+02$)	(5.3691$E+02$)	(6.0867$E+02$)
f_9	Mean	3.3160$E-01$	8.4770$E+01$	3.8155$E+01$	4.3687$E+01$	3.7750$E-02$	9.0465$E+01$	**2.9352E − 02**	2.8450$E+00$	1.9402$E+01$	6.6639$E+01$
	Min	2.9129$E-02$	3.3829$E+01$	3.1528$E-02$	2.2605$E-02$	2.3464$E-02$	4.1792$E+01$	**1.8232E − 02**	2.1729$E-02$	5.9871$E+00$	3.2844$E+01$
	Std Dev	(1.0996$E+00$)	(2.4674$E+01$)	(2.3952$E+01$)	(2.9723$E+01$)	(8.6261$E-03$)	(2.3879$E+01$)	(1.3161$E-02$)	(1.7927$E+00$)	(6.9163$E+00$)	(2.1773$E+01$)
f_{10}	Mean	**1.6879E − 03**	1.9694$E+01$	2.4643$E-03$	8.7447$E-02$	1.6157$E-02$	8.1049$E+00$	1.1862$E-02$	1.1219$E-02$	8.8736$E-01$	4.4735$E+00$
	Min	6.8753$E-05$	6.7988$E+00$	6.6043$E-05$	**4.7010E − 05**	1.4046$E-02$	3.4041$E+00$	9.2344$E-03$	7.6683$E-03$	8.4415$E-03$	1.1552$E+00$
	Std Dev	(4.8030$E-03$)	(1.8609$E+00$)	(5.9462$E-03$)	(4.1917$E-01$)	(1.1968$E-03$)	(2.5300$E+00$)	(1.3762$E-03$)	(4.7517$E-03$)	(1.4152$E+00$)	(2.1722$E+00$)
f_{11}	Mean	2.2300$E-02$	7.4742$E-02$	2.2230$E-02$	**1.7921E − 02**	2.4342$E-02$	1.2093$E-01$	4.1413$E-02$	4.3552$E-02$	6.7181$E-02$	1.1395$E-01$
	Min	1.4659$E-08$	2.1056$E-09$	9.3328$E-10$	**4.5294E − 10**	1.5764$E-05$	2.5166$E-06$	9.3085$E-06$	4.6435$E-06$	5.5552$E-06$	3.7370$E-06$
	Std Dev	(2.7383$E-02$)	(5.9743$E-02$)	(3.4994$E-02$)	(1.9481$E-02$)	(2.8117$E-02$)	(1.6475$E-01$)	(4.3791$E-02$)	(7.0052$E-02$)	(7.6744$E-02$)	(2.2874$E-01$)
f_{12}	Mean	4.1633$E-03$	1.3546$E+00$	4.1507$E-03$	4.1515$E-03$	**2.0812E − 03**	1.1365$E+00$	2.0741$E-02$	1.5691$E-01$	3.5859$E-01$	4.4462$E-01$
	Min	6.5400$E-10$	2.0109$E-06$	4.0296$E-10$	**1.5497E − 10**	4.0330$E-06$	8.8123$E-07$	2.0292$E-06$	1.8880$E-06$	1.2414$E-06$	9.6019$E-07$
	Std Dev	(2.0519$E-02$)	(2.3403$E+00$)	(2.0521$E-02$)	(2.0520$E-02$)	(1.4661$E-02$)	(1.6131$E+00$)	(7.5513$E-02$)	(3.6762$E-01$)	(5.7222$E-01$)	(6.2836$E-01$)
f_{13}	Mean	**3.9555E − 05**	6.4014$E+00$	6.8845$E-04$	2.5325$E-04$	8.5921$E-05$	4.5179$E+00$	7.1911$E-05$	4.9155$E-03$	5.8544$E-01$	2.0819$E+00$
	Min	6.3527$E-09$	1.2170$E-06$	2.2008$E-09$	**1.7797E − 09**	5.6319$E-05$	9.0896$E-06$	3.2237$E-05$	2.3006$E-05$	1.9697$E-05$	3.5239$E-05$
	Std Dev	(5.0506$E-05$)	(6.9850$E+00$)	(2.6287$E-03$)	(1.5500$E-03$)	(1.8465$E-05$)	(5.9895$E+00$)	(4.6632$E-05$)	(1.4952$E-02$)	(1.7692$E+00$)	(3.2842$E+00$)

Table 4: Continued.

	f_n	SSMSEP	SSMSEP-1	SSMSEP-2	SSEP	FEP	CEP	$\alpha = 1.2$	$\alpha = 1.4$	$\alpha = 1.6$	$\alpha = 1.8$
f_{14}	Mean	1.3946	1.7563	**1.2205**	1.3946	1.2805	1.6321	1.4542	1.5371	1.3177	1.7693
	Min	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**
	Std Dev	$(8.7265E-01)$	$(1.1928E+00)$	$(5.0212E-01)$	$(8.7181E-01)$	$(5.7523E-01)$	$(1.1595E+00)$	$(8.9826E-01)$	$(9.8139E-01)$	$(6.7645E-01)$	$(1.4785E+00)$
f_{15}	Mean	$4.3569E-04$	$4.5400E-04$	$4.3569E-04$	$5.0894E-04$	$4.5400E-04$	$4.0332E-04$	$3.9906E-04$	$5.0894E-04$	**3.8074E – 04**	$4.9063E-04$
	Min	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**	**3.0749E – 04**
	Std Dev	$(3.2096E-04)$	$(3.3910E-04)$	$(3.2096E-04)$	$(3.8317E-04)$	$(3.3910E-04)$	$(2.9089E-04)$	$(2.7749E-04)$	$(3.8317E-04)$	$(2.5094E-04)$	$(3.6999E-04)$
f_{16}	Mean	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Min	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Std Dev	$(1.0087E-09)$	$(3.1347E-10)$	$(3.9007E-11)$	$(1.7266E-12)$	$(3.0332E-08)$	$7.9887E-09)$	$(3.3852E-08)$	$(2.0453E-08)$	$(2.2805E-08)$	$(1.9422E-08)$
f_{17}	Mean	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
	Min	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
	Std Dev	$(1.2995E-10)$	$(3.0825E-11)$	$(2.2603E-11)$	$(7.8869E-13)$	$(7.0398E-09)$	$5.1135E-09)$	$(1.4545E-08)$	$(9.8416E-09)$	$(8.3266E-09)$	$(8.0297E-09)$
f_{18}	Mean	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**
	Min	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**
	Std Dev	$(3.5756E-08)$	$(4.3186E-09)$	$(2.8254E-09)$	$(1.0494E-10)$	$(1.1868E-06)$	$8.8630E-07)$	$(1.1530E-06)$	$(1.3493E-06)$	$(1.4221E-06)$	$(7.6097E-07)$
f_{19}	Mean	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**
	Min	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**
	Std Dev	$(2.6613E-06)$	$(2.0399E-08)$	$(2.0050E-06)$	$(1.3995E-06)$	$(1.9028E-06)$	$6.1612E-07)$	$(1.5250E-06)$	$(9.6434E-07)$	$(1.0641E-06)$	$(1.2872E-06)$
f_{20}	Mean	**−3.2842**	−3.2675	−3.2563	−3.2675	−3.2437	−3.2794	−3.2675	−3.2675	−3.2651	**−3.2842**
	Min	**−3.3224**	**−3.3224**	**−3.3224**	**−3.3224**	−3.3223	**−3.3224**	**−3.3224**	**−3.3224**	**−3.3224**	**−3.3224**
	Std Dev	$(5.9431E-02)$	$(6.0015E-02)$	$(5.9333E-02)$	$(6.0014E-02)$	$(5.7039E-02)$	$(5.6169E-02)$	$(6.0012E-02)$	$(6.0014E-02)$	$(6.0159E-02)$	$(5.6174E-02)$
f_{21}	Mean	−6.4811	−5.4657	−6.5003	−6.6111	−6.4439	**−8.4928**	−6.3396	−7.0739	−7.3097	−7.8652
	Min	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	−10.1531	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**
	Std Dev	$(2.4491E+00)$	$(1.3963E+00)$	$(2.3030E+00)$	$(2.7105E+00)$	$(2.5162E+00)$	$(2.4704E+00)$	$(2.3080E+00)$	$(2.5335E+00)$	$(2.6909E+00)$	$(2.6294E+00)$
f_{22}	Mean	−8.2317	−5.8052	−8.3806	−8.3522	−7.5690	**−9.6849**	−7.7189	−8.1391	−8.3426	−8.5489
	Min	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**
	Std Dev	$(2.7034E+00)$	$(2.2835E+00)$	$(2.8997E+00)$	$(2.8100E+00)$	$(2.9246E+00)$	$(1.9791E+00)$	$(3.0032E+00)$	$(2.8505E+00)$	$(2.8157E+00)$	$(2.7805E+00)$
f_{23}	Mean	−8.7721	−8.5261	−8.3264	−8.1230	−8.5698	**−9.6378**	−8.7256	−8.9864	−8.5304	−8.9635
	Min	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**
	Std Dev	$(3.0564E+00)$	$(2.8798E+00)$	$(3.3848E+00)$	$(3.3768E+00)$	$(2.8178E+00)$	$(2.1817E+00)$	$(3.1400E+00)$	$(2.6928E+00)$	$(3.1457E+00)$	$(2.8776E+00)$

Table 5: 2-tailed Wilcoxon signed-rank test for SSMSEP vs SSMSEP-1, SSMSEP-2, SSEP, FEP, CEP, and LEP with $\alpha = (1.2, 1.4, 1.6, 1.8)$ on $f_1 - f_{10}$ (NGs: number of generations).

f_n	NGs	vs SSMSEP-1	vs SSMSEP-2	vs SSEP	vs FEP	vs CEP	vs $\alpha = 1.2$	vs $\alpha = 1.4$	vs $\alpha = 1.6$	vs $\alpha = 1.8$
f_1	1500	>	≥	≥	>	>	>	>	>	>
f_2	2000	>	<	<	>	>	>	>	>	>
f_3	5000	>	≤	<	<	>	<	<	<	<
f_4	5000	>	≥	>	>	>	>	>	>	>
f_5	20000	<	≥	≤	≥	<	≤	<	<	<
f_6	1500	>	=	=	=	>	=	=	>	>
f_7	3000	>	<	≤	≤	>	≥	>	>	>
f_8	9000	>	>	≥	≤	>	>	>	>	>
f_9	5000	>	>	>	<	>	<	>	>	>
f_{10}	1500	>	≥	≥	>	>	>	>	>	>
f_{11}	2000	>	≥	≤	≥	>	>	≥	>	>
f_{12}	1500	>	≤	≤	<	>	≥	>	>	>
f_{13}	1500	>	≥	≥	>	>	>	>	>	>
f_{14}	100	>	≤	≥	≥	≥	≥	≥	≤	≥
f_{15}	4000	>	≤	≥	≥	<	≤	≥	≤	≥
f_{16}	100	=	=	=	=	=	=	=	=	=
f_{17}	100	=	=	=	=	=	=	=	=	=
f_{18}	100	=	=	=	=	=	=	=	=	=
f_{19}	100	=	=	=	=	=	=	=	=	=
f_{20}	200	≥	>	≥	>	>	≥	≥	≥	=
f_{21}	100	≥	≤	≤	≥	<	≥	<	<	<
f_{22}	100	>	≤	≤	≥	<	≥	≥	≤	≤
f_{23}	100	≥	≥	≥	≥	<	≥	≤	≥	≤

Table 6: Function evaluations of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, CEP, FEP, and LEP with $\alpha = (1.2, 1.4, 1.6, 1.8)$ on $f_1 - f_{23}$.

f_n	FEs
f_1	$4.5E+6$
f_2	$6.0E+6$
f_3	$1.5E+7$
f_4	$1.5E+7$
f_5	$6.0E+7$
f_6	$4.5E+6$
f_7	$9.0E+6$
f_8	$2.7E+7$
f_9	$1.5E+7$
f_{10}	$4.5E+6$
f_{11}	$6.0E+6$
f_{12}	$4.5E+6$
f_{13}	$4.5E+6$
f_{14}	$2.0E+4$
f_{15}	$1.6E+6$
f_{16}	$2.0E+4$
f_{17}	$2.0E+4$
f_{18}	$2.0E+4$
f_{19}	$3.0E+4$
f_{20}	$1.2E+5$
f_{21}	$4.0E+4$
f_{22}	$4.0E+4$
f_{23}	$4.0E+4$

that SSMSEP still has limitations; this also inspires the use of Cauchy distribution as a fixed long step size mutation while there is room for improvement. Currently, there is no mechanism for selecting and updating long step size mutations in the proposed mutation strategy.

6. Future Work and Discussion

The objective of this study is to design a strategy that can select a proper mutation operator for EP at each generation and update the parameters to select the probability distribution.

The mutation strategy proposed by SSEP involves updating the step size S_k and the historical step size S_H_k at each generation and using the distance coefficient T to dominate SSEP to avoid getting trapped into a local optimum in earlier generations of EP. However, SSEP cannot solve all evolving situations and easily falls into a local optimum, such as f_9, which also exists in other mixed-mutation strategies (LMSEP and MSEP). To solve this common issue, the SSMSEP importing both step size and survival rate to control and avoid EP will fall into a local optimum in the earlier generations. It is similar to the analogy of two keyholes being present in the door, and one having to use two keys to open this door. The "step size" and "survival rate" are two keys used simultaneously for the door. Through both keys, the mutation strategies can select the proper mutation operator by automatically updating the parameters.

The mutation strategy proposed in Section 3.2 was implemented in SSMSEP. We also notice that the algorithm does not demonstrate outstanding performance on a few functions. We believe that the mutation strategy still has room for improvement. Below are possible directions for future research to promote the robustness and accuracy of the mutation strategy.

(i) Automatically tune α of Lévy distribution during the evolution. The Cauchy mutation can be replaced

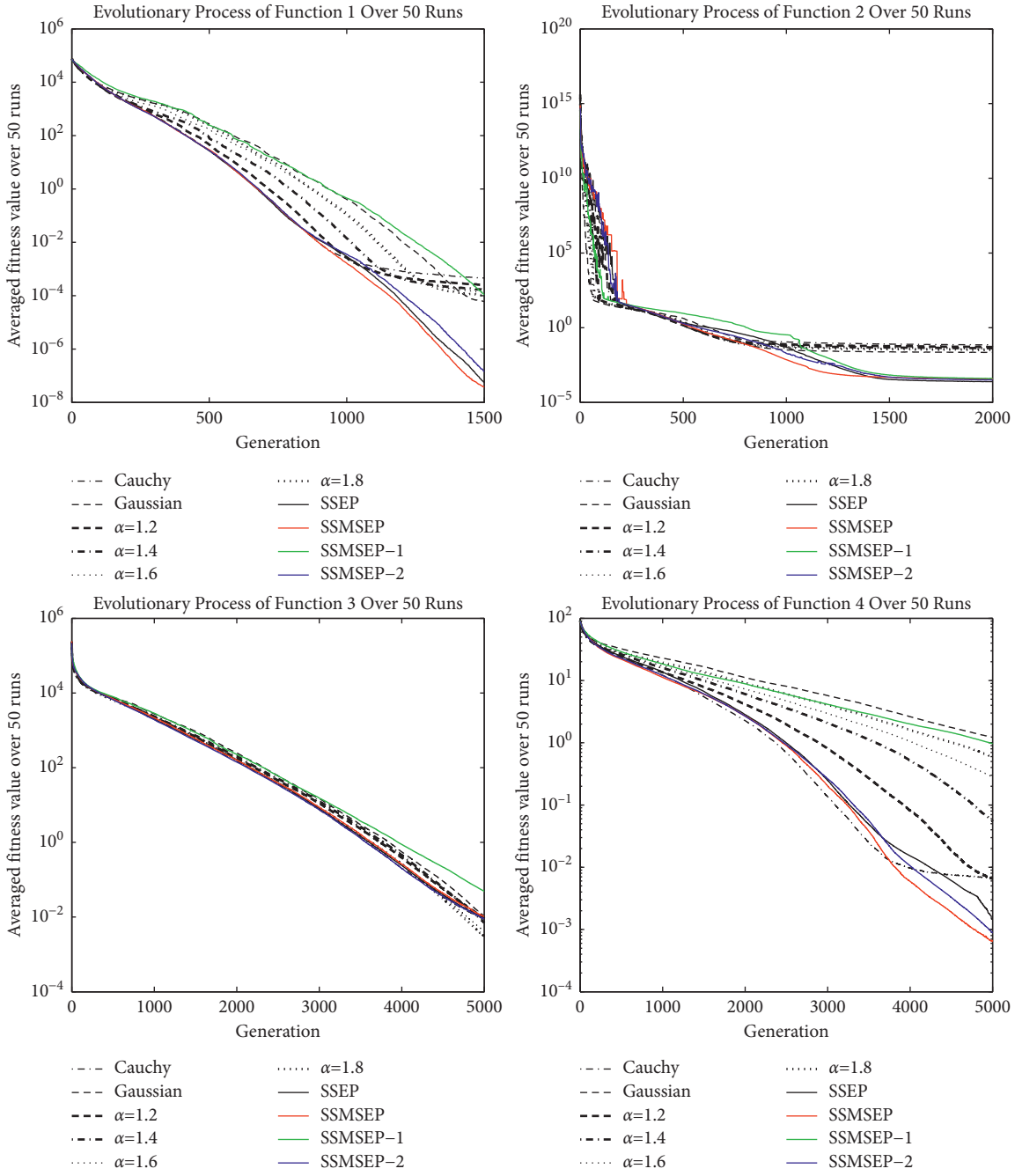

Figure 1: Evolutionary process of $f_1 - f_4$.

by the Lévy distribution with various values of α. The value of α can be updated based on historical information.

(ii) If "step size" and "survival rate" are the first and second keys for our proposed mutation strategy, whether there exists a third key is still worth investigating.

(iii) The mutation strategy is applied at the population level; it is possible to build a mechanism to apply the strategy on an individual level.

(iv) SSMSEP does not show outstanding performance on multimodal functions with a few local optima, and it is worth investigating the reasons to continue to improve SSMSEP.

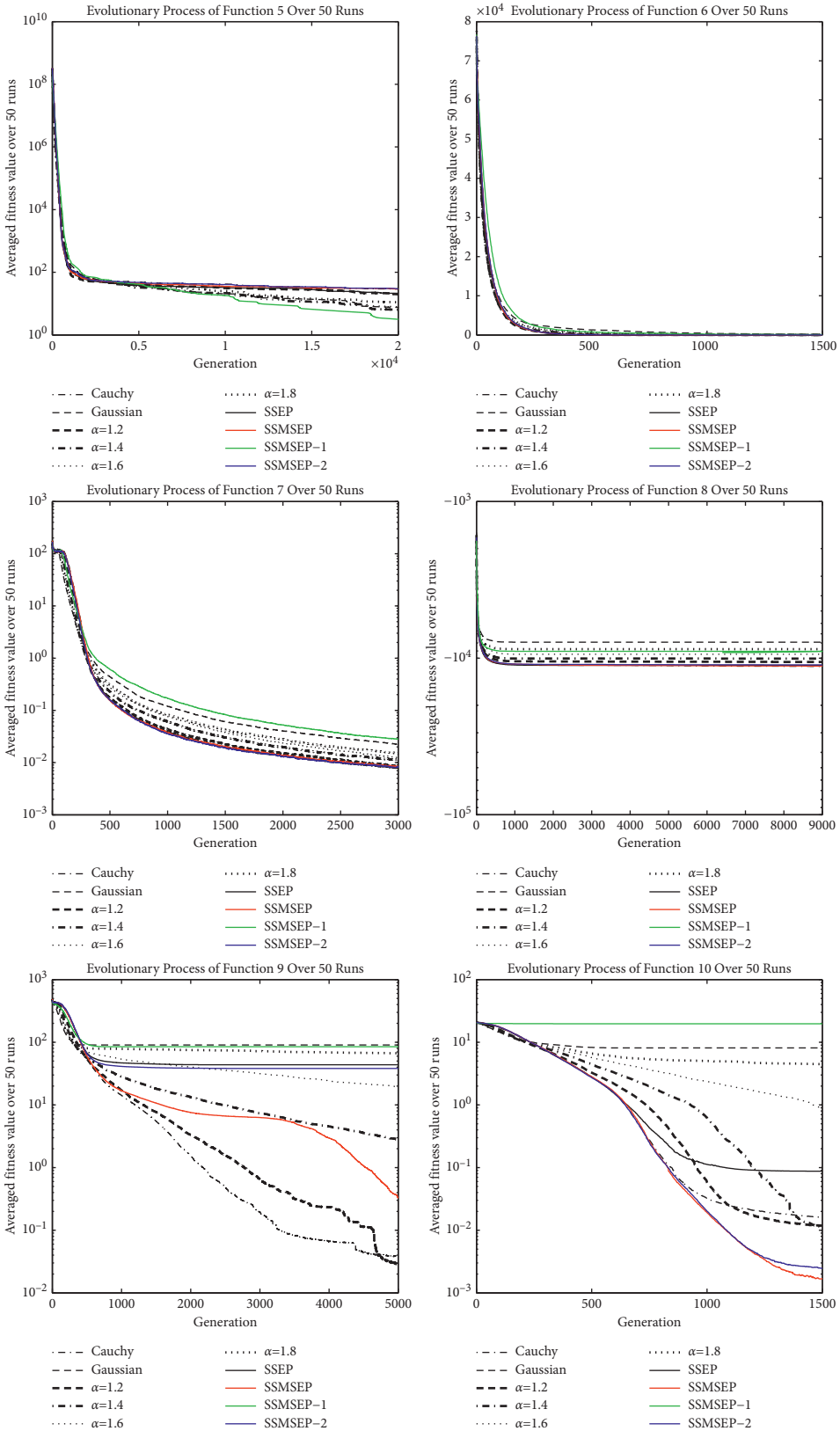

Figure 2: Evolutionary process of $f_5 - f_{10}$.

Table 7: Mean and standard deviations of SSMSEP, SSMSEP-1, SSMSEP-2, SSEP, MSEP, and LMSEP ($f_1 - f_{15}$ averaged over 50 runs, and $f_{16} - f_{23}$ averaged over 1000 runs; the mean best fitness values are shown in bold).

f_n	NGs	SSMSEP	SSMSEP-1	SSMSEP-2	SSEP	LMSEP [7]	MSEP [7]
f_1	1500	**3.7814E − 08**	1.0901E − 04	1.4066E − 07	5.5197E − 08	3.803E − 05	6.209E − 05
		(3.78E − 08)	(2.34E − 04)	(6.14E − 07)	(6.55E − 08)	(7.97E − 05)	(1.61E − 04)
f_2	2000	3.6408E − 04	4.0167E − 04	3.2274E − 04	**2.4660E − 04**	1.556E − 03	8.226E − 02
		(3.01E − 05)	(3.08E − 05)	(2.86E − 05)	(1.56E − 05)	(9.37E − 04)	(4.35E − 01)
f_4	5000	**6.4065E − 04**	9.7131E − 01	8.8194E − 04	1.4405E − 03	0.767	0.629
		(1.80E − 03)	(8.19E − 01)	(1.49E − 03)	(4.03E − 03)	(1.09)	(1)
f_6	1500	**0**	64.98	**0**	**0**	**0**	43.8
		(0)	(128)	(0)	(0)	(0)	(126)
f_7	3000	8.4609E − 03	2.8136E − 02	8.1618E − 03	**7.9013E − 03**	3.514E − 02	3.56E − 02
		(2.68E − 03)	(9.89E − 03)	(2.18E − 03)	(2.13E − 03)	(1.85E − 02)	(1.74E − 02)
f_9	5000	**3.3160E − 01**	84.77	38.16	43.69	61.39	63.44
		(1.10)	(24.67)	(23.95)	(29.72)	(13.18)	(13.8)
f_{10}	1500	**1.6879E − 03**	19.69	2.4643E − 03	8.7447E − 02	1.956E − 03	6.498
		(4.80E − 03)	(1.86)	(5.95E − 03)	(4.19E − 01)	(1.76E − 03)	(2.49)
f_{11}	2000	2.2300E − 02	7.4742E − 02	2.2230E − 02	**1.7921E − 02**	5.0E − 02	7.768E − 02
		(2.7383E − 02)	(5.9743E − 02)	(3.4994E − 02)	(1.9481E − 02)	(4.843E − 2)	(11.198)
f_{12}	1500	4.1633E − 03	1.3546E + 00	**4.1507E − 03**	4.1515E − 03	1.282	9.36E − 01
		(2.0519E − 02)	(2.3403E + 00)	(2.0521E − 02)	(2.0520E − 02)	(2.065)	(2.46E − 02)
f_{15}	4000	4.3569E − 04	4.5400E − 04	4.3569E − 04	5.0894E − 04	**2.247E − 04**	3.077E − 04
		(3.2096E − 04)	(3.3910E − 04)	(3.2096E − 04)	(3.8317E − 04)	(1.885E − 04)	(1.090E − 06)
f_{16}	100	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
		(2.34E − 09)	(3.77E − 10)	(4.70E − 11)	(2.00E − 12)	(4.82E − 09)	(3.42E − 08)
f_{18}	100	**3**	**3**	**3**	**3**	**3**	**3**
		(3.38E − 07)	(8.35E − 06)	(3.62E − 07)	(6.04E − 08)	(4.43E − 08)	(0)
f_{19}	100	**−3.863**	**−3.863**	**−3.863**	**−3.863**	**−3.863**	**−3.863**
		(1.63E − 06)	(8.19E − 08)	(2.49E − 04)	(1.68E − 06)	(7.22E − 08)	(4.568E − 07)
f_{21}	100	−6.586	−5.480	−6.755	−6.683	**−8.744**	−8.62
		(2.41)	(1.55)	(2.62)	(2.54)	(2.744)	(2.59)
f_{22}	100	−7.76	−6.26	−8.17	−8.03	**−9.585**	−9.37
		(2.88)	(2.48)	(2.89)	(2.91)	(2.136)	(2.32)
f_{23}	100	−8.35	−7.53	−8.63	−8.62	**−9.483**	−9.63
		(3.06)	(3.28)	(3.06)	(3.05)	(2.466)	(2.38)

7. Conclusions and Summary

Evolutionary programming is a popular evolutionary computation algorithm used to solve numerical optimization problems. It has been applied in various domains [16–21]. Probability distributions (Gaussian, Cauchy, Lévy, and double exponential, among others) were used as unique mutation operators for evolutionary programming early on. Over the last two decades, several mutation strategies have been adopted in the EP algorithm, such as combining mutation operators [4], which improved FEP [5], MSEP [6], LMSEP [7], and SSEP [9].

The SSMSEP is inspired by the drawback of the SSEP. SSEP only uses "step size" to control the selection of the proper mutation operator with parameters; "step size" is imported as a unique key to control the mutation type and parameters of probability distributions. The SSMSEP imports the "survival rate," and our proposed method updates the parameters at each generation to select mutation operators throughout the evolutionary process. The mutation strategy proposed herein can maintain a high convergence rate in both the earlier and later generations of EP on unimodal benchmark functions and multimodal benchmark functions with many local optima, thereby demonstrating excellent performance on most benchmark functions in both the earlier and later generations of EP. In SSMSEP, the "survival rate" is imported as a second key to control the selection of mutation types and can update the parameters for the mutation operator. SSMSEP-1 and SSMSEP-2 are two variants of SSMSEP, with either one using only "survival rate" or "step size." Neither SSMSEP-1 nor SSMSEP-2 demonstrated an impressive performance in the experiments. SSMSEP can successfully overcome the defects of SSEP, which is the loss of step size control in the earlier generation of evolution. The new algorithm demonstrates its robustness and effectiveness in most tests although it does not demonstrate its best performance on some tests, only second-best or third-best. SSMSEP neither displays outstanding performance nor worst performance on average on multimodal benchmark functions with a few local optima. The research direction for future work is proposed as it is necessary to continue to discover more factors and investigate the internal connections among collected historical information.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] X. Yao and Y. Liu, "Fast evolutionary programming," in *Proceedings of the 5th Annual Conference on Evolutionary Programming*, pp. 451–460, San Diego, CA, USA, March 1996.

[2] C.-Y. Lee and X. Yao, "Evolutionary programming using mutations based on the lévy probability distribution," *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 1–13, 2004.

[3] H. Narihisa, K. Kohmoto, T. Taniguchi, M. Ohta, and K. Katayama, "Evolutionary programming with only using exponential mutation," in *Proceedings of the 2006 IEEE International Conference on Evolutionary Computation*, Vancouver, BC, Canada, July 2006.

[4] C. Kumar, "Combining mutation operators in evolutionary programming," *IEEE Transactions on Evolutionary Computation*, vol. 2, no. 3, pp. 91–96, 1998.

[5] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 82–102, 1999.

[6] H. Dong, J. He, H. Huang, and W. Hou, "Evolutionary programming using a mixed mutation strategy," *Information Science*, vol. 177, pp. 312–327, 2007.

[7] L. Shen and J. He, "A mixed strategy for evolutionary programming based on local fitness landscape," in *Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Barcelona, Spain, July 2010.

[8] R. Mallipeddi, S. Mallipeddi, and P. N. Suganthan, "Ensemble strategies with adaptive evolutionary programming," *Information Sciences*, vol. 180, no. 9, pp. 1571–1581, 2010.

[9] L. Hong, J. H. Drake, and E. Özcan, "A step size based self-adaptive mutation operator for evolutionary programming," in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation Conference*, pp. 1381–1388, Vancouver BC Canada, July 2014.

[10] L. Hong, J. Woodward, J. Li, and E. Özcan, "Automated design of probability distributions as mutation operators for evolutionary programming using genetic programming," in *Proceedings of the 16th European Conference, EuroGP 2013*, pp. 85–96, Vienna, Austria, April 2013.

[11] L. Hong, J. H. Drake, J. R. Woodward, and E. Özcan, "A hyper-heuristic approach to automated generation of mutation operators for evolutionary programming," *Applied Soft Computing*, vol. 62, pp. 162–175, 2018.

[12] L. Hong, J. R. Woodward, E. Özcan, and F. Liu, "Hyper-heuristic approach: automatically designing adaptive mutation operators for evolutionary programming," *Complex and Intelligent Systems*, pp. 1–29, 2021.

[13] K.-H. Liang, X. Yao, Y. Liu, C. Newton, and D. Hoffman, "An experimental investigation of self-adaptation in evolutionary programming," in *Proceedings of the 7th International Conference on Evolutionary Programming VII*, pp. 291–300, San Diego, CA, USA, March 1998.

[14] K. H. Liang, Y. Xin, and C. S. Newton, "Adapting self-adaptive parameters in evolutionary algorithms," *Applied Intelligence*, vol. 15, no. 3, pp. 171–180, 2001.

[15] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, vol. 1, pp. 1–23, 1993.

[16] B. F. David, "Applying evolutionary programming to selected traveling salesman problems," *Journal of Cybernetics*, vol. 24, no. 1, pp. 27–36, 1993.

[17] J. Yuryevich and K. Po Wong, "Evolutionary programming based optimal power flow algorithm," *Power Systems, IEEE Transactions on*, vol. 14, no. 4, pp. 1245–1250, 1999.

[18] N. Sinha, R. Chakrabarti, and P. K. Chattopadhyay, "Evolutionary programming techniques for economic load dispatch," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 1, pp. 83–94, 2003.

[19] W. Yong, Z. Gang, and P. C. Chang, "Improved evolutionary programming algorithm and its application research on the optimization of ordering plan," *Systems Engineering—Theory and Practice Online*, vol. 29, no. 6, pp. 172–177, 2010.

[20] S. Salcedo-Sanz, A. Pérez-Bellido, E. Ortiz-García, A. Portilla-Figueras, and S. Jiménez-Fernĺćndez, "A hybrid evolutionary programming approach for optimal worst case tolerance design of magnetic devices," *Applied Soft Computing*, vol. 12, pp. 2425–2434, 2012.

[21] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile robot path planning using artificial bee colony and evolutionary programming," *Applied Soft Computing*, vol. 30, pp. 319–328, 2015.