

Evolutionary Search of Approximated N-Dimensional Landscapes *

Ko-Hsin Liang

School of Computer Science

University College, The University of New South Wales

Australian Defence Force Academy, Canberra, ACT 2600, Australia

Xin Yao

School of Computer Science

The University of Birmingham

Edgbaston, Birmingham B15 2TT, UK

Charles Newton

School of Computer Science

University College, The University of New South Wales

Australian Defence Force Academy, Canberra, ACT 2600, Australia

Abstract

Finding the global optimum on a large, multimodal, complex, and discontinuous (or nondifferentiable) landscape is usually very hard, even using the evolutionary approach. However, some of these complex landscapes can be approximated and smoothed without changing the nature of the problem, i.e., without modifying the global optimum and its location. The approximated and smoothed landscape is often much easier to search than the original one. In this paper, we propose a new algorithm using landscape approximation and hybrid evolutionary and local search. We also list several algorithm design principles. Following the basic algorithm, an example algorithm is given from our previous work of the combination of landscape approximation and local search (LALS). Furthermore, we develop a novel evolutionary algorithm with n -dimensional approximation (EANA), which shares the same rules as the basic algorithm, but remedies some of the drawbacks found in the LALS. Comparisons with evolution strategies and improved fast evolutionary programming are also given in this paper to show the advantages and disadvantages of the proposed algorithms.

1 Introduction

Evolutionary algorithms (EA) have been applied to many optimization problems successfully in recent years. One of the essential applications of EA is global optimization on numerical problems [5, 3, 18, 35]. A global optimization problem can be formalized as a pair (S, f) , where $S \subseteq R^n$ is a bounded set in R^n and $f : S \rightarrow R$ is an n -dimensional real-valued function. The problem is to find a vector $\hat{x} \in S$ such that $f(\hat{x})$ is a global minimum on S . More specifically, it is required to find an $\hat{x} \in S$ such that $\forall x \in S : f(\hat{x}) \leq f(x)$. Here f does not need to be continuous, but it has to be bounded.

A common feature of EA on solving numerical functions is the usage of the step size concept. Evolutionary programming (EP) [12] and evolu-

tion strategies (ES) [27] both use self-adaptation to adjust the step size and improve the search progress. The self-adaptation schemes normally have the trend of tuning the objective variables with smaller step sizes during the latter evolutionary stage. In a multimodal problem, if an individual stays in a local optimum at this time and the landscape between the final local optimum and the global one is really rough or very hilly, it is inevitable that the final solution will become trapped and the probability to jump over the hurdle will be very small.

To overcome this difficulty, recombination operators play a significant role. A good recombination operator provides efficient global search into promising areas. To develop a better recombination operator, approximation and local search techniques are used in this paper. It can help to solve

*Published in *International Journal of Knowledge-Based Intelligent Engineering Systems*, 4(3):172-183, July 2000.

high dimensional multimodal problems more efficiently and effectively.

1.1 Methods with Local Search

The local search techniques have been applied in variant global optimization methods. The simplest method is known as Multistart. This method applies local search from each randomly generated point. However, local searches are the most time consuming parts in the algorithm. Obviously, the inefficiency is caused by using many local searches to find the same minimum several times. To improve the inefficiency, clustering methods [30] and the Multi-Level Single Linkage [25] are designed. They use variant clustering techniques to link points to different groups. Within each group local search is only conducted on one point and the found local optimum is assumed to be the representative of that group. These methods provide impressive results to the benchmark test function in Dixon and Szegö (1978) [11]. However, when solving a problem with a large number of minima, these methods may not be suitable [25, 1].

Using local search techniques can indirectly change the roughness of the fitness landscape. In Figure 1, the effect of local search on the one-dimensional (1D) landscape of the generalized Rastrigin's function is displayed. The difficult problem becomes easier to solve after applying the local search method. Under this concept, it has previously been applied in genetic algorithms and has substantially improved their performance for some multimodal problems [32, 13]. In terms of evolution, the local search can be thought of as a consequence of individual learning during the individual's life time. Combining learning and evolution has the effect of changing the fitness landscape. The fitness of an individual will be changed after applying a learning process (i.e., local search). However, the individual's genetic codes (parameters) may or may not be altered depending on the way that learning and evolution interact. They are known as *Lamarckian evolution* and the *Baldwin effect* [6, 14], respectively. In Hart and Belew [13], the performances of the Baldwin effect and Lamarckian evolution are compared. It should be noticed that the usage rates of local search to the GA have different impacts on the evolutionary process. To design an efficient search algorithm, the application of the local search should be carefully considered. For combinatorial problems, the "STAGE" algorithm using local search and quadratic approximation has obtained many successes [9].

1.2 Methods with Approximation

Using approximation techniques to transform a complex problem into a simpler one is an attrac-

tive idea. Some work has been done on this topic. Two major approaches can be categorized, i.e., the polynomial approximation and computer model approximation. The early work of applying polynomial approximation to optimization dates back to Winfield [33] in the late 1960's. He used a quadratic model to interpolate the next minimal point. The next closed method was developed in 1994, when Powell [21] used linear multivariate interpolation to solve constrained optimization. He also explored the idea further by using a multivariate quadratic interpolation model to solve unconstrained optimization problems [22]. A variant of using multivariate quadratic polynomial techniques was proposed later by Conn and Toint [10]. They also provided some impressive numerical test results.

The computer model approximation is a method to model the deterministic output of computer experiments as a stochastic process. The popular approach is called "Design and Analysis of Computer Experiments" (DACE) [26, 16]. This approximation model, also named "surrogate", has had some successful applications in engineering design [8, 29]. Recently, an efficient global optimization algorithm was developed using the DACE model by Jones et al. [15].

Using approximation and/or local search techniques in evolutionary algorithms have provided many successful applications, for example the evolutionary algorithms with kriging approximation [23, 24], the evolutionary algorithms with local search [32, 13, 7], the crossover operators with approximation concepts [28, 2] and the evolutionary algorithm with both landscape approximation and local search (LALS) [17]. The LALS algorithm demonstrated high reliability in finding the global optimum of the benchmark multimodal problems.

In this paper, we propose a new algorithm using landscape approximation and hybrid evolutionary and local search. We list several algorithm design principles. Following the basic algorithm, an example algorithm, LALS, is given. In the meantime, a novel algorithm is developed and shares the same rules as the basic algorithm, but remedies some of the drawbacks found in the LALS. The experimental results show that the new algorithm has good global reliability and better efficiency than traditional evolution strategies and evolutionary programming.

The rest of this paper is organized as follows. Section 2 describes the basic evolutionary algorithms using approximation and local search. An example algorithm is also given. Section 3 introduces an enhanced algorithm which improves the drawbacks of the LALS. The empirical results and discussions are presented in Section 4. Finally, Section 5 concludes the paper with some remarks and future research directions.

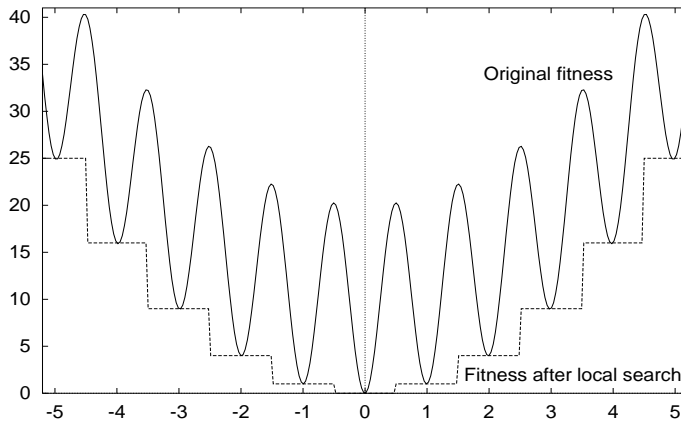


Figure 1: The effect of local search on the 1D landscape of the generalized Rastrigin’s function.

2 Evolutionary Algorithms with Approximation and Local Search

When designing a global optimization algorithm, two basic aspects should be considered, global approach and local approach. A common concept used in many optimization algorithms is the step size control. To ensure the current optimum located in certain neighbour areas, the small step size is needed. To prevent the search being trapped in a local optimum, the large step size can help to jump out and search somewhere else. For example, the Cauchy mutation used in evolutionary programming combines both large and small step sizes in one operator, and makes the tradeoff between global and local search via the Cauchy distribution [35, 36].

An intuitive way for the local approach is applying some hill-climbing or local search techniques to the starting points. Therefore, using randomly generated starting points as the global approach is the so called “Multistart” method. Many global optimization methods using local search techniques have shown promising results in solving difficult numerical problems.

Applying approximation to solve unimodal problems may be an efficient approach [21]. Although the approximation can provide suggestions for the next search regions, it may fail if too little information is available on a very rugged landscape. Especially, when tackling high dimensional multimodal problems, implementing a precise enough approximation becomes too costly. However, we are not interested in how hilly the landscape is. We are only concerned with where the deepest valley could be. Hence using information from local minima, we expect the approximated results may provide some promising search directions.

To illustrate the basic idea, Figure 2 shows a simple way to use quadratic approximation and local search on one dimensional case of Ackley’s function. In this example, P_1, P_2, P_3 are three local minimal points, P_4 is the estimated point and is mapped to the fitness point P'_4 . After proceeding a local search from P'_4 , the potential of finding better optima can be strongly expected.

In the following, we propose a basic algorithm using local search and approximation techniques into evolutionary algorithms.

Algorithm A0: Basic algorithm

- Step 1:** Generate μ individuals and their local search information.
- Step 2:** Apply λ individuals to provide ρ predicted minimum points using approximation techniques.
- Step 3:** Obtain local search information for ρ individuals.
- Step 4:** Select next generation from $\mu, \lambda,$ and ρ individuals.
- Step 5:** Go to Step 2 if termination criteria are not met.

The Algorithm A0 provides a skeleton to develop global optimization algorithms using evolutionary algorithms with approximation and local search techniques. To implement a practical algorithm, some principles are discussed in the following.

- **Select an efficient local search technique.** We can categorize local search into gradient or non-gradient based methods. Many real world problems have difficulties to compute any associated derivatives (gradient or Hessian). For non-gradient based local search methods, direct search methods [20,

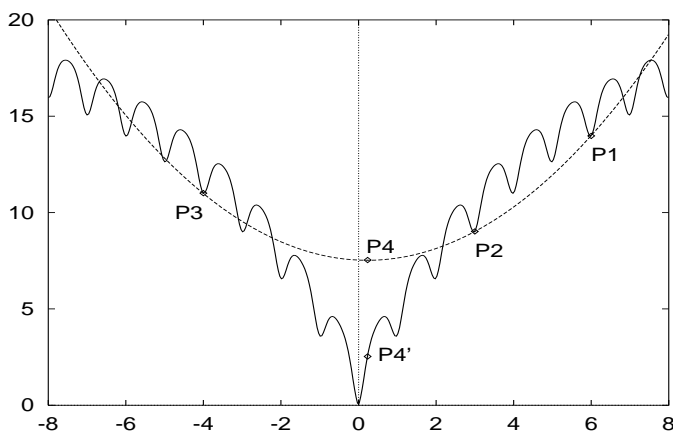


Figure 2: The 1D landscape approximation with local search result on Ackley’s function

19], linear approximation method [21], and local evolutionary search [31] are all very good choices.

- **Decide the population size.** Since we use the local minimum information to perform approximation, each individual will more and less possess such information to make it useful. The larger the population size, the more local search will be executed. Local search is expensive (computationally), especially when non-gradient based local search methods are used. However, the larger population can support better search diversity and increase the probability to find the global optimum.
- **Use a simple approximation method.** A basic principle of applying approximation techniques is that the approximated function can be easily calculated to get the approximated minimum location. A quadratic polynomial with least squares approximation is a good example and has been applied in some optimization methods [33, 22, 10, 17]. The DACE model based on Bayesian statistics can give more precise approximation about the landscape, however other auxiliary global optimization algorithms are needed to obtain the approximated minimum from the approximation function. Different approximation methods also demand different λ individuals. The larger λ is, the more precise approximation can be obtained.
- **Maintain the local search information.** It is possible that the approximated minimum is poor or out of variable boundaries. If any of these happen, either discarding or modifying the individual has to be made.
- **Consider that ρ is more than one.** Using more approximated minima in one gener-

ation can increase the search diversity. When the problems have more than one rugged valley, performing approximation search at different areas at the same time could be meritorious.

- **Make the algorithm stop.** Stopping criteria are an important subject in stochastic algorithms. For the algorithms using local search, suitable stopping rules can make it more efficient. In most test problems the global optima are known, we can stop the algorithms once the global optima are found. If the optima are unknown, the algorithms stop when the computation resources are all used or some convergence situations are met.
- **Add other evolutionary operators.** Since Algorithm A0 is derivative from evolutionary algorithms, any evolutionary operators can be freely added to increase its effectiveness and efficiency.

2.1 An Example

In our previous work, we developed a method using landscape approximation and local search (LALS) with EA to solve multimodal problems [17]. The empirical results of 18 benchmark problems show that LALS has very good global reliability. The LALS algorithm is stated as follows.

Algorithm A1: LALS

- Step 1:** Generate μ individuals and their local search information.
- Step 2:** Obtain 3 individuals from global discrete recombination and use the quadratic approximation technique to provide 1 predicted minimum point.

Step 3: Perform local search for the approximated individual.

Step 4: Make $(\mu + 4)$ selections for next generation.

Step 5: Go to Step 2 if termination criteria are not met.

The principles and techniques used in LALS are as follows:

- The local search technique is the Local Evolutionary Search with Random Memorizing (LESRM) from Voigt and Lange (1998) [31].
- Three population sizes are used to deal with different difficulties of the problems, 50,30, and 10.
- The LALS uses one-dimensional quadratic polynomial approximation.
- The approximated minimum is always used to do local search.
- One approximated minimum is generated.
- The stopping criteria are the maximum number of function evaluations is reached or the differences of the fitness values of P_1, P_2, P_3 are less than $1e-6$.
- The global discrete recombination is added.

3 The Evolutionary algorithm with n -dimensional approximation

There are three weaknesses in LALS. Firstly, the application of local search in LALS is inefficient. In examples of solving high dimensional problems, the good results can only be obtained by increasing the population size, which means the unconditional uses of local search are questionable for high dimensional problems. Furthermore, the approximated point in LALS also unconditionally performs a local search, no matter how bad the point is.

The second weakness of LALS is obviously the approach of the one-dimensional approximation. When solving the problems with non-separable variables, the LALS could not correctly predict the optimal location and thereby degraded its performance.

The third improvement is related to the LALS's stopping rule. In some cases, the 3 obtained individuals from global discrete recombination never have the same fitness values and the LALS fails to stop.

In this section, we propose an enhanced algorithm to improve the design of the LALS. This is the evolutionary algorithm with n -dimensional approximation (EANA) stated as follows.

Algorithm A2: EANA

Step 1: Generate μ individuals and their local search information.

Step 2: Obtain λ individuals from global discrete recombination and use the n -dimensional quadratic approximation technique to provide 1 predicted minimum point.

Step 3: Perform local search for the approximated individual with a probability rule.

Step 4: Make $(\mu + \lambda + 1)$ selections for next generation.

Step 5: Go to Step 1 if termination criteria are not met.

We describe the improved design in EANA following the same sequence as the LALS in the last section.

- The same local search technique is used.
- The population size is set to $2n + 1$, where n is the problem dimension. This design is to match the approximation method described in the next section.
- In EANA, the n -dimensional approximation is implemented by using a quadratic polynomial model created with the least squares method. Hence λ is equal to $2n + 1$.
- Perform local search for the approximated minimum if the approximated minimum is less than any one of the μ parents, otherwise perform local search with a probability of 0.5.
- One approximated minimum is generated.
- The algorithm terminates when 1000 generations have evolved, or all the step sizes of each parent are less than $1e - 6$, or the best three parents are too close to each other, which means it stops if $|x_{1,j} - x_{2,j}| < 0.0001$ and $|x_{2,j} - x_{3,j}| < 0.0001, \forall j \in \{1, \dots, n\}$.
- The same global discrete recombination is added.

A special design of the EANA not mentioned above is that the EANA does not initialize the population with any local search in Step 1. Instead, each parent proceeds with a single downward step (1+1)-EA in each iteration and then an approximation is made of those parents. That is, the approximation does not need to wait for the results of local searches. The rough tendency of the landscape will be getting smoother when all the parents reach their valley bottoms. Therefore, the iteration in Step 5 goes back to Step 1.

The local search method used in both Algorithm A1 and A2 is the Local Evolutionary Search with Random Memorizing (LESRM) from Voigt

and Lange (1998) [31]. LESRM generates promising new search directions from pre-stored solutions to the current point. A generated good solution is stored in a sequential memory which is randomly accessed later. Very impressive results of solving some unimodal functions are shown in [31]. When applying LESRM, we modify it with different stopping criteria. The search is terminated if 5000 function evaluations are used or the search step size is less than $1e - 6$.

The global discrete recombination is that each new individual's objective variable is decided at random and copied from one of the parents:

$$x_{i,j} = x_{\chi,j}, \forall j \in \{1, \dots, n\},$$

where $x_{i,j}$ denotes the j -th component of the vectors x_i , $\forall i \in \{1, \dots, \mu\}$, n is the dimensionality. χ denotes a uniform distributed random integer in $\{1, \dots, \mu\}$, and μ is the population size.

A brief introduction of the n -dimensional approximation is described in the next subsection.

3.1 Landscape Approximation — The Polynomial Model

The polynomial approximation model is used to approximate an n -dimensional landscape in the EANA algorithm.

A quadratic polynomial model has the form:

$$y(\mathbf{x}) = c_0 + \sum_{i=1}^n c_i x_i + \sum_{i=1}^n c_{i+n} x_i^2 + \sum_{i=1}^{n(n-1)/2} c_{i+2n} x_j x_k, \quad (1)$$

where n is the number of variables, c_i is the i -th polynomial coefficient, and $1 \leq j < k \leq n$. There are $t = (n + 1)(n + 2)/2$ unknown coefficients. Thus, we need at least t sample points to obtain a unique solution. This polynomial model with t sample points expressed in matrix notation is

$\mathbf{y} = \mathbf{X}\mathbf{c}$, where \mathbf{y} is the vector formed by t fitness values of the sample points, $\mathbf{y} = [y^1, y^2, \dots, y^t]^T$, and \mathbf{c} is the vector of unknown coefficients, $\mathbf{c} = [c_0, c_1, \dots, c_{t-1}]^T$, and \mathbf{X} is the matrix expressed as

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_n^{(1)} & (x_1^{(1)})^2 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ 1 & x_1^{(t)} & \dots & x_n^{(t)} & (x_1^{(t)})^2 & \dots \\ \\ (x_n^{(1)})^2 & x_1 x_2^{(1)} & \dots & x_{n-1} x_n^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ (x_n^{(t)})^2 & x_1 x_2^{(t)} & \dots & x_{n-1} x_n^{(t)} \end{bmatrix}$$

¹The matrix computation is implemented with the public domain package LinAlg available from <http://www.lh.com/~oleg/ftp/packages.html>

The unique least squares solution of \mathbf{c} is

$$\mathbf{c} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y},$$

if $\mathbf{X}^T \mathbf{X}$ is non-singular.

Using the quadratic polynomial model of Eq. 1 to obtain an approximated point will need $(n + 1)(n + 2)/2$ points. This will be a major challenge coping with high dimensional problems. For a 30-dimensional problem, for example, we need 496 individuals to accomplish this job. An alternative and simplified approach is to discard the third summation in Eq. 1. Therefore, only $2n + 1$ individuals are needed to perform an approximation¹. To understand the impact of the simplified version of quadratic approximation, we ran a preliminary experiment to compare the difference on computation time and performance between the original and the simplified quadratic approximation. The experiments are tested on the generalized Rastrigin function with various dimensions using Algorithm A2. On the left of Figure 3 shows the computation time used to accomplish 20 runs in the experiments. The time used on the 30 dimensional problem is 57 hours from the original version, and 10 minutes from the simplified version. However, the performances on the successful rate of finding global minima from both versions are quite close as referred to the right of Figure 3. The worse outcome can be expected from the simplified version, since we may lose some information from the approximation.

Even though the original quadratic approximation supports better approximation, the computation cost from both the population and matrix size will exponentially increase with the function dimension. In this paper, only the simplified quadratic approximation is applied in Algorithm A2.

4 Results

The EANA algorithm is designed to solve multimodal problems. We use all the multimodal functions tested by LALS [17] and two more functions from [4] in our experimental studies. Table 1 lists all the test functions. To compare the performance of EANA, we also list the results of ES and Improve Fast EP (IFEP) [34] in the tables in the next two sections. The ES uses n self-adaptive strategy parameters and no correlated mutations. Recombination is discrete on object variables and global intermediate on strategy parameters. The selection uses a (15,100) mechanism. In IFEP each parent generates two offspring, one with Gaussian

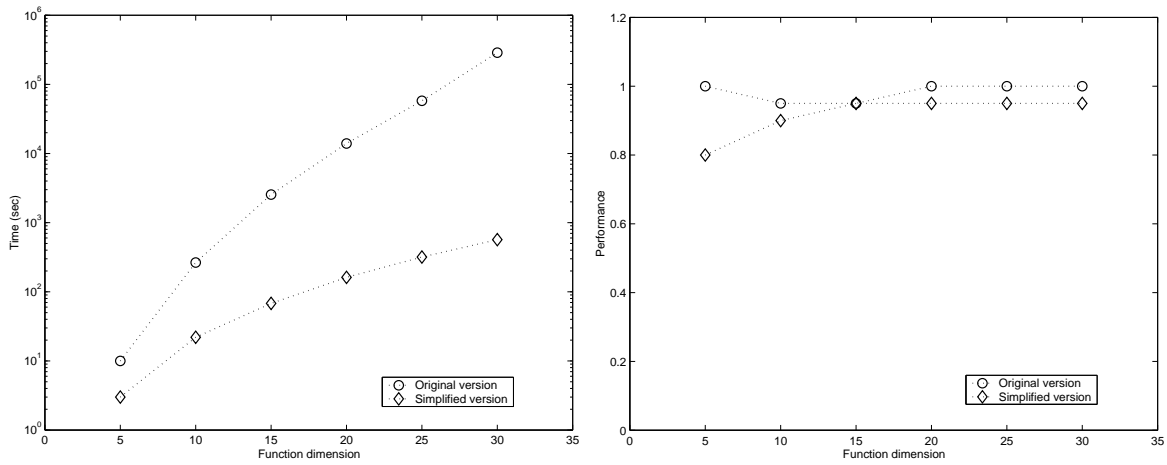


Figure 3: The difference on computation time (left) and performance (right) between the original and simplified quadratic approximation on solving Generalized Rastrigin function.

and the other with Cauchy mutation. It is a very simple and effective algorithm for function optimization. The parameter setup follows the suggestions from [34]. Each problem for each algorithm is run 50 times. The average function evaluations and best values are recorded. The number of the global minimum found is also noted and named as “global hit”.

To compare the three algorithms, we regard the global hits as the primary objective achieved by the algorithm and the function evaluations as the second objective. Although the qualities of the problem’s solutions are important, their values obtained are highly relevant to the function evaluations used. For comparisons, the following rules are used sequentially:

- Rule 1:** If both algorithms have 100% global hits, we use rule 5.
- Rule 2:** If only one algorithm has 100% global hits, we infer this algorithm outperforms the other.
- Rule 3:** If no algorithms have 100% global hits and both results of the global hits are close, we use rule 5.
- Rule 4:** If no algorithms have 100% global hits and the results of the global hits have over 25% difference, we infer the algorithm with higher global hits outperforms the other.
- Rule 5:** The function evaluations are used to compare the efficiency. If the relative difference is larger than 0.1, the algorithm with the less function evaluations outperforms the other. Otherwise, a t -test is needed to make a comparison.

In order to facilitate comparisons, we have used the same numbering system for the benchmark functions as that used in previous studies [36, 17, 34, 35]. That is why the benchmark functions were not numbered from 1.

4.1 High Dimensional Multimodal Functions($f_8-f_{13}, f_{26}, f_{27}$)

Functions $f_8-f_{13}, f_{26}, f_{27}$ are high dimensional multimodal functions with many local minima. The number of local minima increases exponentially as the function dimension increases. These functions appear to be very “rugged” and difficult to optimize. Table 2 lists the experimental results for ES, IFEP, and EANA. The problem results are made bold for those algorithms which have better performance.

For functions f_8, f_9, f_{11} , and f_{27} , the EANA outperforms ES and IFEP from its effectiveness and/or efficiency. The ES is better on functions f_{10} and f_{12} , where it provides better efficiency. The IFEP outperforms the others on function f_{13} with better efficiency than the EANA, and better effectiveness than the ES. The case of the function f_9 is worth noticing where the EANA has a much more outstanding performance. A further understanding about the causes of the different results will be discussed in Section 4.3.

The f_{26} is called the Fletcher-Powell function, which no algorithm is clearly preferred. Since the EANA has difficulty to terminate f_{26} properly, a lower mean best does not mean it is better. No algorithm could find any global minimum. We redo the EANA and make it stop if the number of function evaluations is over 200000 to make fair comparisons. We find that the mean best is equal to 1195.08, which shows that the EANA outperforms the others. From the results of f_{26} , we observe that if the approximated point is not good we then proceed with a local search with probability 0.5, which may use unnecessary searches before all the individuals stop. This is a difficult tradeoff between the search diversity and the search effort.

Table 1: The 20 test functions used in our experimental studies, where n is the dimension of the function, f_{min} is the minimum value of the function, N_{min} is the number of the function's minima, and $S \subseteq R^n$.

Test function	n	N_{min}	S	f_{min}
$f_8(x) = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	30	8^n	$[-500, 500]^n$	-12569.5
$f_9(x) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	30	11^n	$[-5.12, 5.12]^n$	0
$f_{10}(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos 2\pi x_i\right) + 20 + e$	30	65^n	$[-32, 32]^n$	0
$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	1.64e52	$[-600, 600]^n$	0
$f_{12}(x) = \frac{\pi}{n} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} + \sum_{i=1}^n u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a \leq x_i \leq a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases}$	30	5^n	$[-50, 50]^n$	0
$f_{13}(x) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \right\} + \sum_{i=1}^n u(x_i, 5, 100, 4)$	30	30^n	$[-50, 50]^n$	0
$f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1}$	2	25	$[-65.536, 65.536]^n$	1
$f_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	∞	$[-5, 5]^n$	0.0003075
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	6	$[-5, 5]^n$	-1.0316285
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10 \left(1 - \frac{1}{8\pi}\right) \cos x_1 + 10$	2	3	$[-5, 10] \times [0, 15]$	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	4	$[-2, 2]^n$	3
$f_{19}(x) = -\sum_{i=1}^4 c_i \exp\left[-\sum_{j=1}^4 a_{ij}(x_j - p_{ij})^2\right]$	3	4	$[0, 1]^n$	-3.86
$f_{20}(x) = -\sum_{i=1}^4 c_i \exp\left[-\sum_{j=1}^6 a_{ij}(x_j - p_{ij})^2\right]$	6	4	$[0, 1]^n$	-3.32
$f_{21}(x) = -\sum_{i=1}^5 [(x - a_i)^T(x - a_i) + c_i]^{-1}$	4	5	$[0, 10]^n$	$-1/c_1$
$f_{22}(x) = -\sum_{i=1}^7 [(x - a_i)^T(x - a_i) + c_i]^{-1}$	4	7	$[0, 10]^n$	$-1/c_1$
$f_{23}(x) = -\sum_{i=1}^{10} [(x - a_i)^T(x - a_i) + c_i]^{-1}$ where $c_1 = 0.1$	4	10	$[0, 10]^n$	$-1/c_1$
$f_{24}(x) = 0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2 - 0.5}}{[1.0 + 0.001(x_1^2 + x_2^2)]^2}$	2	∞	$[-100, 100]^n$	0
$f_{25}(x) = (x_1^2 + x_2^2)^{0.25} [\sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1.0]$	2	∞	$[-100, 100]^n$	0
$f_{26}(x) = \sum_{i=1}^n (A_i - B_i)^2$ $A_i = \sum_{j=1}^n (a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j)$ $B_i = \sum_{j=1}^n (a_{ij} \sin x_j + b_{ij} \cos x_j)$	30	2^n	$[-\pi, \pi]^n$	0
$f_{27}(x) = -\sum_{i=1}^{30} c_i \left[\exp\left(-\frac{1}{\pi} A\right) \cdot \cos(\pi A) \right]$ $A = \sum_{j=1}^n (x_j - a_{ij})^2$	30	$> 10^n$	$[0, 10]^n$	unknown

f_{27} is the generalized Langerman function². We do not know the fitness value of its global minimum, hence the global hits are not available.

From functions f_{10} , f_{12} , and f_{13} , although the EANA has good global hit, it still spent many function evaluations. We observe the results are caused because the stopping conditions are too late to be met. A good stopping rule is needed in our future work.

4.2 Low Dimensional Multimodal Functions(f_{14} - f_{25})

The final results of ES, IFEP, and EANA on functions f_{14} - f_{25} are summarized in Table 3. For functions f_{15} - f_{24} , the EANA has better performances than ES and IFEP from the global hit and/or the function evaluations. The EANA on function f_{14} is the only case among all 20 benchmark problems that has a much worse global hit than both ES and IFEP. Further discussions about the results of function f_{14} are deferred to the next section. For function f_{25} , IFEP is better at finding global minimum in all 50 runs.

Although the EANA has better performance on functions f_{20} - f_{24} , it still misses global optima in some case of these problem. We notice that the search diversity is contributed mainly by the approximated individual in every generation through the evolutionary process. If bad approximation keeps happening, we probably need other strategies to compensate this drawback. That is, more than one approximated minima should be considered to enhance the search diversity.

4.3 Discussions

According to Table 2, the EANA algorithm has better reliability to find global optima than both ES and IFEP on high dimensional problems with many local minima. From Table 3, EANA can maintain good search diversity to find the global minima, but with less search effort on most problems. However, EANA still needs improvement on global hits on some low dimensional problems.

The advantages and disadvantages are discussed using functions f_9 and f_{14} as examples where the large discrepancies were found. Figure 4 shows the two dimensional landscape of functions f_9 and f_{14} . In the beginning we inferred that the better performance of EANA on f_9 was brought about by its bigger population size. However, by redoing the experiment with $\mu = 10$, we obtained the same global hit (50/50), and the average function evaluations were even less (21358). Therefore, we conjecture that the n -dimensional approximation may be the direct contributor. Some randomly distributed points can easily be approximated to a

spherical shape which matches the quadratic polynomial model and can efficiently provide a good estimation of the global optimum. Finding more evidence to support our assumption will be an aim in our future research.

Function f_{14} is one of the examples where EANA's performance was worse. This could be caused by the small population size of EANA ($\mu = 5$). By increasing the population sizes of EANA, the experimental results show that the increased population size does help to improve the performance. However the computational effort is too costly. The population size is an important key to LALS [17], but not to EANA. There are 25 holes in function f_{14} (see Figure 4). Suppose an individual in the initial population of EANA is located in the region of attraction of the global optimum. Before it reaches the bottom, we still have chances to lose it through the selection procedure. If we do not have such an individual, the approximation will be the only chance to generate one.

From the discussions of the results of functions f_9 and f_{14} , it is apparent that the population size is no longer a key parameter in EANA. Tuning the population parameter becomes unnecessary. This can be an important advantage in the application of EANA by users who may not have expertise in evolutionary computation. The n -dimensional approximation can be regarded as a form of recombination operator and an important merit to the EANA algorithm. The disadvantage is that the single approximation will stop providing the search diversity when all the individuals reach the local minima and have no more variations.

5 Conclusions

In this paper we have proposed a basic evolutionary algorithm using approximation and local search to tackle global optimization problems. A novel evolutionary algorithm with n -dimensional approximation, EANA, was developed following our design principles. EANA improves some of the drawbacks of our previous work, LALS. The empirical comparisons were made among the traditional ES and improved fast EP. The comparisons of the algorithms' performances are made using five rules considering the global hits, the function evaluations, and solution values. EANA can provide more efficient performance with equivalent reliability on some high dimensional multimodal problems. However, there are some experimental results raising the issue that EANA may not have enough search diversity during later evolutionary stages. Hence, using two or more approximations in one generation to increase EANA's exploration ability will be included in our future work. Some analysis work is also needed to explain the ideas

²The matrix data (a_{ij}) can be found at <http://www.wi.leidenuniv.nl/CS/ALP/alea.html>.

Table 2: The results of ES, IFEP and EANA on $f_8-f_{13}, f_{26}, f_{27}$. “Eval.” is the number of evaluations. “G. hit” is the global hits.

Prob	ES			IFEP			EANA		
	Eval.	Mean	G. hit	Eval.	Mean	G. hit	Eval.	Mean	G. hit
8	241500	-10091.26	0/50	757500	-10640.18	0/50	102470	-12191.21	0/50
9	500000	344.44	0/50	500000	2.89	1/50	83483	1.99e-2	49/50
10	100000	2.66e-10	50/50	150000	6.33e-4	50/50	181578	3.30e-5	50/50
11	100000	0	50/50	176500	1.27e-1	3/50	9372	2.05e-10	50/50
12	50000	1.06e-9	50/50	476245	2.49e-2	40/50	185318	8.14e-11	50/50
13	50000	4.40e-4	48/50	150000	4.42e-8	50/50	349059	1.88e-9	50/50
26	200000	5223.03	0/50	200000	23438.64	0/50	1139316	222.81	0/50
27	200000	-9.20e-3	N/A	200000	-8.55e-2	N/A	49839	-0.24325	N/A

Table 3: The results of ES, IFEP and EANA on $f_{14}-f_{25}$. “Eval.” is the number of evaluations. “G. hit” is the global hits.

Prob	ES			IFEP			EANA		
	Eval.	Mean	G. hit	Eval.	Mean	G. hit	Eval.	Mean	G. hit
14	1000	4.18	33/50	10000	1.56	33/50	1685	7.36	4/50
15	400000	3.92e-3	44/50	235500	4.17e-4	44/50	31645	3.07e-4	50/50
16	2500	-1.031628	50/50	3500	-1.031628	50/50	863	-1.031628	50/50
17	3000	0.3979	50/50	4500	0.3979	50/50	945	0.3979	50/50
18	3500	3	50/50	4500	3	50/50	822	3	50/50
19	4000	-3.86	50/50	6000	-3.86	50/50	1297	-3.86	50/50
20	10500	-3.31	43/50	17500	-3.26	24/50	8357	-3.3128	46/50
21	5000	-6.71	24/50	10000	-7.02	21/50	2958	-9.6512	46/50
22	5000	-9.14	41/50	9500	-8.42	34/50	2880	-9.1430	41/50
23	5000	-9.32	42/50	7500	-9.09	39/50	3118	-9.5767	43/50
24	4000	1.81e-2	1/50	8500	8.86e-3	4/50	1181	2.97e-2	30/50
25	10000	1.42e-1	48/50	10000	2.47e-3	50/50	1687	3.19e-1	44/50

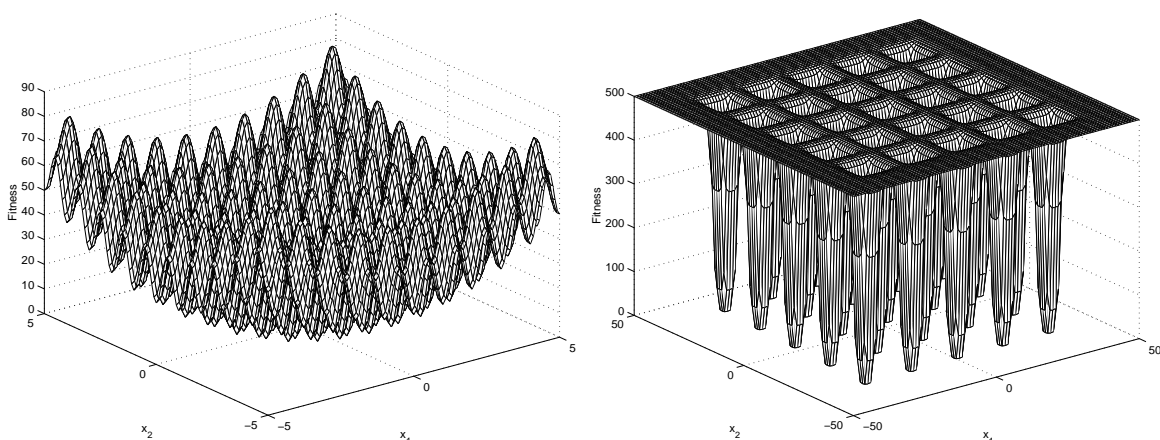


Figure 4: The 2D landscapes of Generalized Rastrigin function f_9 (left) and Shekel's foxholes function f_{14} (right).

of combining approximation and local search with evolutionary algorithms.

References

- [1] M.M. Ali, C. Storey, and A. Törn. Application of stochastic global optimization algorithms to practical problems. *Journal of Optimization and Application*, 95:545–563, 1997.
- [2] K.S. Anderson and Y. Hsu. Genetic crossover strategy using an approximation concept. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 1, pages 257–533, Piscataway, NJ, 1999. IEEE Press.
- [3] Th. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University, New York, 1996.
- [4] Th. Bäck and A.E. Eiben. Generalizations of intermediate recombination in evolution strategies. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 2, pages 1566–1573, Piscataway, NJ, 1999. IEEE Press.
- [5] Th. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.
- [6] J.M. Baldwin. A new factor in evolution. *American Naturalist*, 30:441–451, 1896.
- [7] H.K. Birru, K. Chellapilla, and S.S. Rao. Local search operators in fast evolutionary programming. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 2, pages 1506–1513, Piscataway, NJ, 1999. IEEE Press.
- [8] A.J. Booker, Jr. J.E. Dennis, P.D. Frank, D.B. Serafini, V. Torczon, and M.W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13, 1999.
- [9] J.A. Boyan and A.W. Moore. Learning evaluation functions for global optimization and boolean satisfiability. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 3–10, 1998.
- [10] A.R. Conn and Ph.L. Toint. An algorithm using quadratic interpolation for unconstrained derivative free optimization. In G. Di Pillo and F. Gianessi, editors, *Nonlinear Optimization and Applications*, pages 27–47, New York, 1996. Plenum Publishing.
- [11] L.C.W. Dixon and G.P. Szegö. The global optimization problem: An introduction. In L.C.W. Dixon and G.P. Szegö, editors, *Towards Global Optimization 2*, pages 1–15, Amsterdam, 1978. North-Holland.
- [12] D.B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, New York, 1995.
- [13] W.E. Hart and R.K. Belew. Optimization with genetic algorithm hybrids that use local search. In R.K. Belew and M. Mitchell, editors, *Adaptive Individuals in Evolving Populations: Models and Algorithms*, volume 26 of *SFI Studies in the Sciences of Complexity*, pages 483–496, Reading, MA, 1996. Addison-Wesley.
- [14] G.E. Hinton and S.J. Nolan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.
- [15] D.R. Jones, M. Schonlau, and W.J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.
- [16] J.R. Koehler and A.B. Owen. Computer experiments. *Handbook of Statistics*, 13:261–308, 1996.
- [17] K.-H. Liang, X. Yao, and C. Newton. Combining landscape approximation and local search in global optimization. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 2, pages 1514–1520, Piscataway, NJ, 1999. IEEE Press.
- [18] H. Mühlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm I. Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49, 1993.
- [19] J.A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1965.
- [20] M.J.D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7:155–162, 1964.
- [21] M.J.D. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In S. Gomez and J.-P. Hennart, editors, *Advances in Optimization and Numerical Analysis, Proceedings of the Sixth Workshop on Optimization and Numerical Analysis, Oaxaca, Mexico*, volume 275, pages 51–67, Dordrecht, NL, 1994. Kluwer Academic Publishers.

- [22] M.J.D. Powell. A direct search optimization method that models the objective by quadratic interpolation. Presentation at the 5th Stockholm Optimization Days, 1994.
- [23] A. Ratle. Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In A.E. Eiben, Th. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature-PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 87–96, Berlin, 1998. Springer.
- [24] A. Ratle. Optimal sampling strategies for learning a fitness model. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 3, pages 2078–2085, Piscataway, NJ, 1999. IEEE Press.
- [25] A.H.G. Rinnooy Kan and G.T. Timmer. Stochastic global optimization methods part II: Multi level methods. *Mathematical Programming*, 39:57–78, 1987.
- [26] J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435, 1989.
- [27] H.-P. Schwefel. *Evolution and Optimum Seeking*. Wiley, New York, 1995.
- [28] T.J. Stidsen, O. Caprani, and Z. Michalewicz. A parabolic operator for parameter optimization problems. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 2, pages 1494–1500, Piscataway, NJ, 1999. IEEE Press.
- [29] V. Torczon and M.W. Trosset. Using approximations to accelerate engineering design optimization. In *The 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, pages 738–748, St. Louis, Missouri, 1998.
- [30] A.A. Törn. A search-clustering approach to global optimization. In L.C.W. Dixon and G.P. Szegö, editors, *Towards Global Optimization 2*, pages 49–62, Amsterdam, 1978. North-Holland.
- [31] H.-M. Voigt and J.M. Lange. Local evolutionary search enhancement by random memorizing. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation (ICEC'98)*, pages 547–552, Piscataway, NJ, 1998. IEEE Press.
- [32] D. Whitley, V.S. Gordon, and K. Mathias. Lamarckian evolution, the baldwin effect and function optimization. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, *Parallel Problem Solving from Nature-PPSN III*, volume 866 of *Lecture Notes in Computer Science*, pages 6–15, Berlin, 1994. Springer-Verlag.
- [33] D. Winfield. Function minimization by interpolation in a data table. *Journal of the Institute of Mathematics and its Applications*, 12:339–347, 1973.
- [34] X. Yao, G. Lin, and Y. Liu. An analysis of evolutionary algorithms based on neighbourhood and step sizes. In P.J. Angeline, R.G. Reynolds, J.R. McDonnell, and R. Eberhart, editors, *Evolutionary Programming VI: Proc. of the Sixth Annual Conference on Evolutionary Programming*, volume 1213 of *Lecture Notes in Computer Science*, pages 297–307, Berlin, 1997. Springer.
- [35] X. Yao and Y. Liu. Fast evolutionary programming. In L.J. Fogel, P.J. Angeline, and Th. Bäck, editors, *Evolutionary Programming V: Proc. of the Fifth Annual Conference on Evolutionary Programming*, pages 451–460, Cambridge, MA, 1996. MIT Press.
- [36] X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, 3(2):82–102, 1999.