Minimizing multimodal functions by simplex coding genetic algorithm*

Abdel-Rahman Hedar and Masao Fukushima
Department of Applied Mathematics and Physics, Graduate School of Informatics,
Kyoto University, Kyoto 606-8501, Japan

September 12, 2002, Revised February 14, 2003

Abstract

Combining meta-heuristics with local search methods is one approach that recently has drawn much attention to design more efficient methods for solving continuous global optimization problems. In this paper, a new algorithm called Simplex Coding Genetic Algorithm (SCGA) is proposed by hybridizing genetic algorithm and simplex-based local search method called Nelder-Mead method. In the SCGA, each chromosome in the population is a simplex and the gene is a vertex of this simplex. Selection, new multi-parents crossover and mutation procedures are used to improve the initial population. Moreover, Nelder-Mead method is applied to improve the population in the initial stage and every intermediate stage when new children are generated. Applying Nelder-Mead method again on the best point visited is the final stage in the SCGA to accelerate the search and to improve this best point. The efficiency of SCGA is tested on some well known functions. Comparison with other meta-heuristics indicates that the SCGA is promising.

Key words: Global optimization, meta-heuristics, genetic algorithm, memetic algorithm, simplex coding, local search methods, Nelder-Mead method.

1 Introduction

Global optimization has drawn much attention recently [8, 17, 18, 19], because of a very broad spectrum of applications in real-world systems. However, the works that deal with the global optimization of functions with continuous variables are still not enough to confront the rapid growth of the applications. Therefore, the global optimization of functions with continuous

*This research was supported in part by a Grant-in-Aid for Scientific Research from the Ministry of Education, Science, Sports and Culture of Japan.

variables is a challenging problem. In this paper, we focus on the case of unconstrained minimization, i.e., the problem is

$$\min_{x \epsilon R^n} f(x),$$

where f is a generally nonconvex, real valued function defined on R^n. Meta-heuristics contribute to a reasonable extent in solving global optimization problems, mainly combinatorial problems [16]. Genetic algorithms (GAs) are one of the most efficient meta-heuristics [5, 12], that have been employed in a wide variety of problems. However, GAs, like other meta-heuristics, suffer from the slow convergence that brings about the high computational cost.

Recently, several new approaches have been developed to furnish meta-heuristics with the ability to simulate the fast convergence of local search methods. Most of these approaches hybridize local search methods with meta-heuristics to obtain more efficient methods with relatively faster convergence. This paper pursues in that direction and proposes a new hybrid method that combines GA with a local search method called Nelder-Mead method [15]. In the combined method, called the simplex coding genetic algorithm (SCGA), we consider the members of the population to be simplices, i.e., each chromosome is a simplex and the gene is a vertex of this simplex. Selection, crossover and mutation procedures are used to improve the initial population. Moreover, Nelder-Mead method is applied to improve the population in the initial stage and every intermediate stage when new children are generated. In the SCGA, we use the linear ranking selection scheme [1] to choose some fit parents to be mated. Then, using a new scheme of a multi-parents crossover, new children are reproduced and a few of them are mutated. Applying Kelley's modification [9, 10] of Nelder-Mead method on the best point visited is the final stage in the SCGA to accelerate the search and to improve this best point.

There have been some attempts to utilize the idea of hybridizing local search methods with GA. Simple hybrid methods use the GA or local search methods to generate the points for the new population and then apply the other technique to improve this new population [6, 23]. Other hybrid methods do some modifications in the GA operations; selection, crossover and mutation using local search methods [14, 20, 21, 22]. However, the method proposed in this paper is different from those hybrid methods as we will see in the next section. Simulated annealing as a meta-heuristic was also considered in hybrid approaches with local search methods [7]. The main idea of these hybrid approaches is to avoid creating random movements by using local information about promising search directions. Moreover, uphill acceptance in simulated annealing makes these hybrid approaches not easily entrapped in local minima.

The next section briefly reviews the basics of the Nelder-Mead method and discuss some hybrid GA methods that use Nelder-Mead method. The description of the proposed method is given in Section 3. Section 4 discusses the experimental results along with the initialization of some parameters and the setting of the control parameters of the proposed method. The conclusion follows the experimental results and makes up Section 5.

2 Simplex-Based Genetic Algorithms

In this section, we review some earlier methods that hybridize GAs and simplex methods. The Nelder-Mead method is the most popular simplex-type method that has been used to design a hybrid simplex-based GA method. First, we highlight the Nelder-Mead method before describing hybrid simplex-based GA methods.

2.1 Nelder-Mead method

The local search method called the Nelder-Mead method [15] is one of the most popular derivative-free nonlinear optimization methods. Instead of using the derivative information of the function to be minimized, the Nelder-Mead method maintains at each iteration a non-degenerate simplex, a geometric figure in n dimensions of nonzero volume that is the convex hull of $n+1$ vertices, $x_1, x_2, \ldots, x_{n+1}$, and their respective function values. In each iteration, new points are computed, along with their function values, to form a new simplex. Four scalar parameters must be specified to define a complete Nelder-Mead method; coefficients of reflection ρ, expansion χ, contraction γ, and shrinkage σ. These parameters are chosen to satisfy

$$\rho > 0,\ \chi > 1,\ 0 < \gamma < 1,\ \text{and } 0 < \sigma < 1.$$

The Nelder-Mead method consists of the following steps:

1. Order. Order and re-label the $n+1$ vertices as $x_1, x_2, \ldots, x_{n+1}$ so that $f(x_1) \le f(x_2) \le \cdots \le f(x_{n+1})$. Since we want to minimize f, we refer to x_1 as the best vertex or point, to x_{n+1} as the worst point.

2. Reflect. Compute the *reflection* point x_r by

$$x_r = \overline{x} + \rho\left(\overline{x} - x_{n+1}\right),$$

where $\overline{x}$ is the centroid of the n best points, i.e., $\overline{x} = \sum_{i=1}^{n} x_i/n$. Evaluate $f(x_r)$. If $f(x_1) \le f(x_r) < f(x_n)$, replace x_{n+1} with the reflected point x_r and go to step 6.

3. Expand. If $f(x_r) < f(x_1)$, compute the *expansion* point x_e by

$$x_e = \overline{x} + \chi\left(x_r - \overline{x}\right).$$

Evaluate $f(x_e)$. If $f(x_e) < f(x_r)$ replace x_{n+1} with x_e and go to step 6; otherwise replace x_{n+1} with x_r and go to step 6.

4. Contract. If $f(x_r) \ge f(x_n)$, perform a contraction between $\overline{x}$ and the better of x_{n+1} and x_r.

4.1. Outside. If $f(x_n) \le f(x_r) < f(x_{n+1})$ (i.e., x_r is strictly better than x_{n+1}), perform an *outside contraction*: Calculate

$$x_{oc} = \overline{x} + \gamma\left(x_r - \overline{x}\right).$$

Evaluate $f(x_{oc})$. If $f(x_{oc}) \le f(x_r)$, replace x_{n+1} with x_{oc} and go to step 6; otherwise, go to step 5.

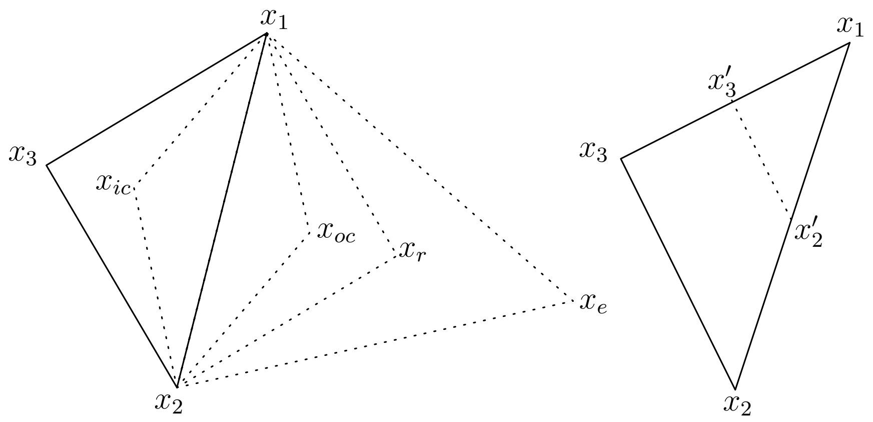

Figure 1: The reflection, expansion, contraction and shrikage points for a simplex in two dimensions.

4.2. Inside. If $f(x_r) \geq f(x_{n+1})$, perform an *inside contraction*: Calculate

$$x_{ic} = \overline{x} + \gamma \left(x_{n+1} - \overline{x}\right).$$

Evaluate $f(x_{ic})$. If $f(x_{ic}) \leq f(x_{n+1})$, replace x_{n+1} with x_{ic} and go to step 6; otherwise, go to step 5.

5. Shrink. Evaluate f at the n new vertices

$$x_i' = x_1 + \sigma \left(x_i - x_1\right), \quad i = 2, \ldots, n+1.$$

Replace the vertices $x_2, \ldots, x_{n+1}$with the new vertices $x_2', \ldots, x_{n+1}'$.

6. Stopping Condition. Order and re-label the vertices of the new simplex as $x_1, x_2, \ldots, x_{n+1}$ such that $f(x_1) \leq f(x_2) \leq \cdots \leq f(x_{n+1})$. If $f(x_{n+1}) - f(x_1) < \varepsilon$, then stop, where $\varepsilon > 0$ is a small predetermined tolerance. Otherwise go to step 2.

Figure 1 shows the effects of reflection, expansion, contraction and shrinkage for a simplex in two dimensions using the standard values of the coefficients

$$\rho = 1, \ \chi = 2, \ \gamma = \frac{1}{2}, \ \text{and} \ \sigma = \frac{1}{2}.$$

After more than thirty years of studying and applying the Nelder-Mead method, McKinnon [11] shows that the Nelder-Mead algorithm can stagnate and converge to a nonoptimal point even for very simple problems. However, Kelley [9, 10] proposes a test for sufficient decrease which, if passed for all iterations, will guarantee convergence of the Nelder-Mead iteration to a stationary point under some appropriate conditions. The Kelley's modification [9, 10] of the Nelder-Mead method will be employed in the final stage of our hybrid method, which will be described in Section 3.

2.2 Hybrid GA methods

There have been several attempts to hybridize GA with simplex-based direct search methods. Remarkable features underlying these hybrid methods are global exploration and parallelism in GA, and local exploitation in direct search methods. Moreover, both GA and direct search methods only use the function values rather than derivatives, which makes those hybrid methods applicable to a broad class of problems. In the following, we briefly summarize some of the hybrid simplex-based GA methods.

Renders and Bersini method [20]. In this method the population is divided into λ groups of $n+1$ chromosomes. Then, one of the following operations is applied to each group with some predetermined probabilities to reproduce exactly one child.

- *Discrete crossover.* Each gene in a child can be chosen from the corresponding gene in a parent which is randomly chosen from the group. This child replaces the worst parent in this group.

- *Average crossover.* The average of all $n+1$ parents in the group replaces the worst parent in this group.

- *Simplex crossover.* Apply the Nelder-Mead method with slight modification to reproduce a new point.

The algorithm terminates if some convergence criterion is reached.

Yang and Douglas method [21]. M points are selected randomly from the search space to form the initial population. GA's reproduction schemes (selection, crossover, and mutation) are used to generate k $(0 < k < M)$ children. The rest of the offspring will be generated by repeating the following procedure $M-k$ times. Using some selection scheme, construct a subcommunity of S points from the M old points. Try to get a better child by applying a simplex method. Otherwise, a child is generated randomly within the search space. If the best point of the new generation is not better than the best one of the old generation, then replace the worst point of the new generation by the best point of the old generation. Moreover, some comparisons are made between the old generation and the new one to copy some of the best points in the old generation into the new one. The algorithm terminates if either a predetermined iteration number is reached or an acceptable objective function value is obtained.

Yen, Liao, Lee, and Randolph method [22]. This simplex GA hybrid method uses a modification of the Nelder-Mead method called the concurrent simplex method. The initial population consists of M chromosomes and the concurrent simplex method is applied to the top S $(n < S < M)$ chromosomes in the population to produce $S-n$ children. The top n chromosomes are copied to the next generation. The GA's reproduction operations, crossover and mutation, are used to generate the remaining $M-S$ chromosomes. The algorithm terminates when it satisfies a convergence criterion or reaches a predetermined maximum number of fitness evaluations.

Musil, Wilmut, and Chapman method [14]. The initial population consists of M chromosomes generated at random. The cycle starts by selecting $n+1$ random pairs of parents from the population. The binary operations (crossover and mutation) are applied on the parents to reproduce children. One child is selected from each of the $n+1$ pairs of children and this results in $n+1$ new children. The Nelder-Mead method runs for k iterations starting with the simplex that consist of these $n+1$ children. The point that gives the lowest objective function value obtained by Nelder-Mead iterations replaces the one with the highest objective function value in the population. The cycle is terminated when the parameters in the population have converged. At this point, other Nelder-Mead iterations start with the chromosome with the lowest objective function value in the population to refine this chromosome and to get the solution.

Our hybrid method SCGA presented in the next section is different from these hybrid methods in many aspects. One of the main differences lies in the coding representation. We use a simplex coding in which the chromosome is a simplex and its genes are the vertices of this simplex. It is expected that using this coding type and applying the Nelder-Mead method starting from each chromosome in the initial population and from each child chromosome will increase the local exploitation and will improve these chromosomes. The other main difference consists in the crossover operation. We introduce a new kind of multi-parents crossover that gives the chance for more than two parents to cooperate in reproducing children and exploring the region around these parents.

3 Description of SCGA

In this section, we describe the proposed method SCGA. The SCGA uses the main functions of the GA; selection, crossover and mutation, on a population of simplices to encourage the exploration process. Moreover, the SCGA tries to improve the initial members and new children by applying a local search method to enhance the exploitation process. This kind of exploration-exploitation procedure is sometimes called "Memetic Algorithm", see [13]. Finally, the SCGA applies an effective local search method on the best point reached by the previous exploration-exploitation procedure. The purpose of this local search is to accelerate the final stages of the GA procedure. This strategy is expected to be effective because the GA has a difficulty in obtaining some required accuracy although the GA may quickly approach the neighborhood of the global minimum.

3.1 Initialization

The SCGA starts with the following initialization procedure:

1. Generate the initial population P_0 that consists of M chromosomes (simplices), i.e.,

$$P_0 = \left\{ S^j : S^j = \left\{ x^{j,i} \right\}_{i=1}^{n+1} ;\ x^{j,i} \in R^n,\ j = 1, \ldots, M \right\}.$$

2. Order the vertices of each simplex S^j, $j = 1, 2, \ldots, M$, so that

$$f(x^{j,1}) \le f(x^{j,2}) \le \cdots \le f(x^{j,n+1}). \tag{1}$$

3. Apply a small number of iterations of the Nelder-Mead method with each S^j as an initial simplex to improve the chromosomes in the initial population P_0.

4. Order the simplices $S^j = \{x^{j,i}\}_{i=1}^{n+1}$, $j = 1, \ldots, M$ in the improved population P_0 so that

$$f(x^{1,1}) \le f(x^{2,1}) \le \cdots \le f(x^{M,1}). \tag{2}$$

3.2 GA loop

Repeat the following procedures described in 3.2.1–3.2.3 while the stopping conditions are not satisfied.

3.2.1 Selection

We describe how we select the set $Q \subseteq P$ of the members that will be given the chance to be mated from the current population P. For each generation, the size of Q is the same as that of P but more fit members in P are chosen with higher probability to be included in Q. We use Baker's scheme called "linear ranking selection" [1] to select the new members in Q. In this scheme, the chromosomes $S^j \in P$, $j = 1, 2, \ldots, M$, are sorted in the order of raw fitness as in (2), and then the probability of including a copy of chromosome S^j into the set Q is calculated by

$$p_s(S^j) = \frac{1}{M}\left(\eta_{\max} - (\eta_{\max} - \eta_{\min})\frac{j-1}{M-1}\right),$$

where $\eta_{\min} = 2 - \eta_{\max}$ and $1 \le \eta_{\max} \le 2$. Using these probabilities, the population is mapped onto a roulette wheel, where each chromosome S^j is represented by a space that proportionally corresponds to $p_s\left(S^j\right)$. Chromosomes in the set Q are chosen by repeatedly spinning the roulette wheel until all positions in Q are filled.

3.2.2 Crossover and mutation

Choose a random number from the unit interval $[0, 1]$ for each chromosome in Q. If this number is less than the predetermined crossover probability p_c, then this chromosome is chosen as a parent. Repeat the following steps until all parents are mating.

1. Select a number n_c from the set $\{2, \ldots, n+1\}$ randomly to determine the number of parents chosen to be mated together.

2. Compute new children $C^i = \left\{x_c^{i,k}\right\}_{k=1}^{n+1}$, $i = 1, \ldots, n_c$ by

$$x_c^{i,k} = \overline{x}^k + d\,r^i,\ k = 1, \ldots, n+1, \tag{3}$$

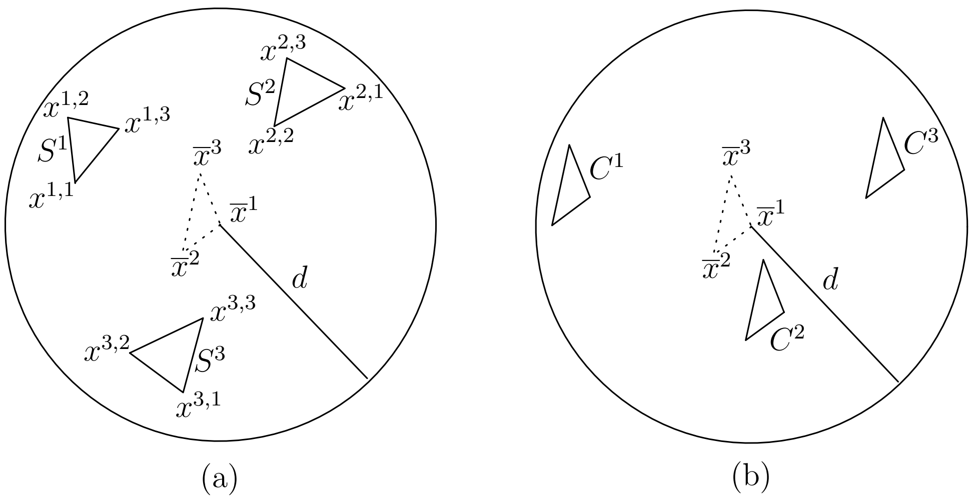

Figure 2: An example of SCGA crossover in two dimensions.

where $r^i,\ i = 1, \ldots, n_c$, are random vectors of length less than 1, d is the maximum distance between pairs of parents and $\overline{x}^k$ is the average of the kth vertices of all parents, i.e.,

$$\overline{x}^k = \frac{1}{n_c} \sum_{i=1}^{n_c} x^{i,k},\ k = 1, \ldots, n+1. \tag{4}$$

Figure 2 shows an example of crossover in two dimensions. In Figure 2(a), we use Equations (4) to compute the dotted simplex whose vertices are the average of the vertices of the parents S^1, S^2 and S^3. By using Equations (3), we move this dotted simplex randomly inside the circle to create the children C^1, C^2 and C^3, as in Figure 2(b).

3. Choose a random number from the unit interval $[0, 1]$ for each child $C^i,\ i = 1, \ldots, n_c$. If this number is less than the predetermined mutation probability p_m, then this child is mutated. Let I_m be the index set of those children who are mutated.

4. Apply the following procedure for each child $C^i = \left\{x_c^{i,k}\right\}_{k=1}^{n+1},\ i \in I_m$. Select a number n_i from the set $\{1, 2, \ldots, n+1\}$ randomly to determine the vertex that is reflected as a mutation. Compute the mutated child $\widetilde{C}^i = \left\{x_m^{i,k}\right\}_{k=1}^{n+1}$ by

$$\begin{aligned} x_m^{i,k} &= x_c^{i,k},\ k = 1, \ldots, n_i - 1, n_i + 1, \ldots, n+1, \\ x_m^{i,n_i} &= \overline{x} + u\left(\overline{x} - x_c^{i,n_i}\right), \end{aligned}$$

where u is a random number in the interval $[0.5, 1.5]$ and $\overline{x}$ is the average of vectors $x_c^{i,1}, \ldots, x_c^{i,n_i-1}, x_c^{i,n_i+1}, \ldots, x_c^{i,n+1}$. Replace the child C^i by the mutated one $\widetilde{C}^i$. Figure 3 shows an example of mutation in two dimensions, where the mutated simplex consists of the vertices $x_m^{1,1}$, $x_m^{1,2}$ and $x_m^{1,3}$, where the vertex $x_m^{1,2}$ is randomly chosen on the line segment $\overline{p_1 p_2}$.

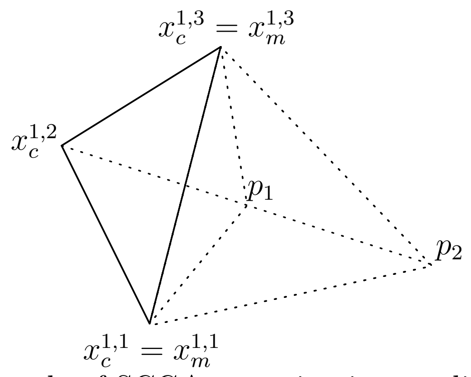

Figure 3: An example of SCGA mutation in two dimensions.

5. Apply a small number of iterations of the Nelder-Mead method with each child C^i, $i = 1, \ldots, n_c$ as an initial simplex to improve the chromosomes.

6. The population in the next generation consists of the M best ones from the set $P \cup \{C^i\}_{i=1}^{n_c}$. Re-order the chromosomes in the new population so that (1) and (2) hold.

3.2.3 Reduction of the population

After every predetermined number of generations, remove some of the worst members in the population P.

3.2.4 Acceleration in the final stage

From the best point obtained by the above procedures, construct a small simplex. Then, apply Kelley's modification [9, 10] of the Nelder-Mead method on this simplex to obtain the final solution.

4 Experimental Results

4.1 Parameter setting

We specify suggested values of the initial and control parameters.

4.1.1 Generating the initial population

Let $[L, U] = \{x \in R^n : \ l_i \leq x_i \leq u_i, \ i = 1, 2, \ldots, n\}$ be the domain in which the initial points are chosen and let this domain be divided into an equal distance grid. The population consists of simplices distributed on this grid and in each coordinate direction there are μ simplices, i.e., the size M of population equals μ^n. We distribute the simplices in such a way that each simplex is put in a neighborhood of one of the knots in the grid. We consider all possible positions of simplices if $n = 2$, and some best positions of them if $3 \leq n \leq 10$.

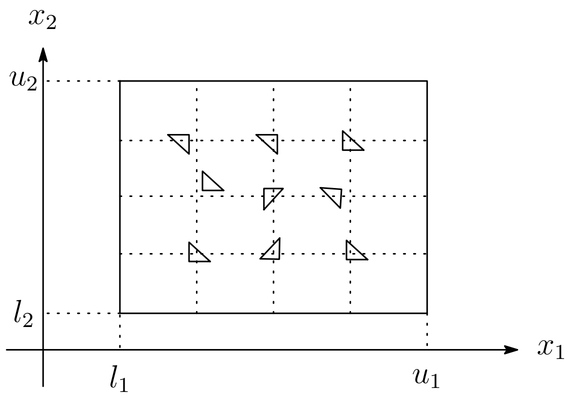

Figure 4: Initial population in two dimensions.

However, for $n > 10$, we employ another procedure for choosing the initial simplices as will be discussed later. From each one of these main vertices we construct a right-angled simplex by taking a step in each coordinate direction. This step size is called edge length. Figure 4 shows an example of the distribution of the population in two dimensions. The values of the parameters used in generating the initial population are given as follows.

1. The number μ of simplices per coordinate direction is varied from 2 to 5 according the estimated density of the local minima of the test function and the number n of variables.

2. The edge length is set equal to $\min_{1 \le i \le n} (u_i - l_i)/10$.

3. The number of Nelder-Mead iterations in the local search for the initial population is set equal to 2.

In the case of $n > 10$, the main vertices in the initial population are generated in a different way. First, we choose a random point $x^{1,1} \in [L, U]$. Then we generate the other main vertices $x^{j,1}$, $j = 2, \ldots, M$, by using the following procedure:

1. Generate $x^{j,1}$.

2. If $\min_{1 \le k \le j-1} \max_{1 \le i \le n} |x_i^{j,1} - x_i^{k,1}| / (u_i - l_i) \ge h$ for some prescribed $h \in (0, 1)$, then accept this vertex $x^{j,1}$. Otherwise, return to step 1.

We set the probability of accepting the new main vertex equal to 0.7. From these main vertices, we construct the simplices of the initial population as in the case of $n \le 10$.

4.1.2 GA loop parameters

The steps of the GA loop have been described in the previous section. Here we specify the values of the parameters used in this loop.

1. The control parameter $\eta_{\max}$ in the selection procedure is chosen to be 1.1 according to the original setting in [1].
2. The crossover probability p_c and the mutation probability p_m are set equal to 0.6 and 0.1, respectively.
3. The number of Nelder-Mead iterations in the local search for the new children is fixed at 2.
4. At every $3n$ generations, we remove the n worst chromosomes from the population unless the number of its chromosomes is less than $2n$.

4.1.3 Termination criteria

The SCGA is terminated when one of the following termination conditions is satisfied.

1. The function values at all vertices of the simplex that contains the best point become close to each other, i.e.,
$$f(x^{1,n+1}) - f(x^{1,1}) \leq \epsilon,$$
where the tolerance ϵ is a small positive number and set equal to 10^{-8}.
2. The number of generations exceeds the predetermined number that is set equal to $\min(10n, 100)$.

4.2 Numerical results

The performance of the SCGA was tested on a number of well known functions [2, 3, 7, 21, 22], which are given in the Appendix. The behavior of these test functions varies to cover many kinds of difficulties that face global optimization problems. For each function we made 100 trials with different initial populations. To judge the success of a trial, we used the condition

$$|f^* - \widehat{f}| < \epsilon_1 |f^*| + \epsilon_2, \tag{5}$$

where $\widehat{f}$ refers to the best function value obtained by SCGA, f^* refers to the known exact global minimum, and ϵ_1 and ϵ_2 are small positive numbers. We set ϵ_1 and ϵ_2 equal to 10^{-4} and 10^{-6}, respectively, when $n \leq 10$, but for $n > 10$ we relax this test condition by increasing the value of ϵ_2 to 10^{-4}. The results are shown in Table 1, where the average number of function evaluations and the average error are related to only successful trials. Table 1 shows that the SCGA reached the global minima in a very good success rate for the majority of the tested

Table 1: Results of SCGA

Function	Rate of success	Average number of function evaluations	Average error
RC	100	173	3.62e-07
ES	100	715	4.97e-09
GP	100	191	4.81e-09
HM	100	176	5.23e-08
SH	98	742	8.83e-06
MZ	100	179	3.40e-06
B_1	99	460	5.11e-09
B_2	99	471	5.43e-09
B_3	100	468	5.14e-09
R_2	100	222	4.60e-09
Z_2	100	170	4.68e-09
DJ	100	187	5.12e-09
$H_{3,4}$	100	201	2.14e-06
$S_{4,5}$	79	1086	3.28e-07
$S_{4,7}$	81	1087	4.06e-05
$S_{4,10}$	84	1068	9.81e-06
R_5	90	3629	5.88e-09
Z_5	100	998	7.10e-09
$H_{6,4}$	99	989	2.00e-06
GR	100	906	8.46e-09
R_{10}	90	6340	1.85e-08
Z_{10}	100	1829	1.76e-08
R_{20}	90	33134	7.59e-05
Z_{20}	100	33106	5.79e-07

functions. Moreover, the numbers of function evaluations and the average errors show the efficiency of the method.

In Table 2, we compare the results of the SCGA with those of three other meta-heuristic methods. These methods are:

1. Real-value Coding Genetic Algorithm (RCGA) [2].
2. Continuous Genetic Algorithm (CGA) [4].
3. Direct Search Simulated Annealing (DSSA) [7].

Table 2: Average number of function evaluations in SCGA and other meta-heuristics

Function	SCGA	RCGA [2]	CGA$^{\otimes}$ [4]	DSSA [7]
RC	173	490	620	118
ES	715	642	1504	1442 (93%)
GP	191	270	410	261
HM	176	-	-	225
SH	742 (98%)	946	575	457 (94%)
MZ	179	452	-	-
B_1	460 (99%)	-	430	252
B_2	471 (99%)	493	-	-
R_2	222	596	960	306
Z_2	170	437	620	186
DJ	187	395	750	273
$H_{3,4}$	201	342	582	572
$S_{4,5}$	1086 (79%)	1158 (62%)	610 (76%)	993 (81%)
$S_{4,7}$	1087 (81%)	1143 (70%)	680 (83%)	932 (84%)
$S_{4,10}$	1068 (84%)	1235 (58%)	650 (81%)	992 (77%)
R_5	3629 (90%)	4150 (60%)	3990	2685
Z_5	998	1115	1350	914
$H_{6,4}$	989 (99%)	973	970	1737 (92%)
GR	906	-	-	1830 (90%)
R_{10}	6340 (90%)	8100 (70%)	21563 (80%)	16785
Z_{10}	1829	2190	6991	12501

The figures for these methods in Table 2 are taken from the original references. For those results of the CGA which are marked by ($^{\otimes}$) in Table 2, their original corresponding data in Table 1 in [4] seem to contain some inconsistencies. In fact, since the same condition as (5) is used in CGA [4] to test the successful trials, the average errors for the tested functions must be less than the right-hand side of (5) for all these functions. However, the average errors corresponding to the tested functions in [4] are reported to be greater than the right-hand side of (5). The comparison given in Table 2 shows the SCGA outperforms the others for many of those functions.

Next, we try to compare SCGA with some of the other simplex-based GA methods described in Section 2. Actually, for many reasons, it is not so easy to make clear comparisons between SCGA and other simplex-based GA methods of [14, 20, 21, 22]. In fact, some of these hybrid methods such as [14, 22] are concentrated on a certain complicated specific problem. Moreover, for some of these methods, computational experiments reported in their original references do not show much helpful information for comparison. For instance, the

Table 3: The results for F_1 function

Function evaluations	SCGA	Simplex GA [21]
Average	351	660
Min	259	32
Max	452	9538

Table 4: The results for De Joung F5 function

Method	Average number of function evaluations
SCGA	1570
Simplex GA [20]	14924
Simplex GA [22]	1695

successful trial test is not mentioned in [21, 22] and the number of the test problems is very small in [14, 20, 21]. Nevertheless, in Tables 3 and 4, we give the available comparisons between SCGA and the other Simplex GA methods of [20, 21, 22]. First, we compare SCGA with the Simplex GA [21] using two functions F_1 ($n = 2$) and F_2 ($n = 10$), see A.15 and A.16 in the Appendix. The results for F_1 are shown in Table 3 and the results for both SCGA and the Simplex GA [21] are taken over 100 trials. It is seen that for this function, SCGA outperforms the Simplex GA [21] with regard to the average function evaluations. For the other function F_2, many results for the Simplex GA are reported in [21] and the best of them is 0.0002 for the best function value with 6400 function evaluations. On the other hand, the results of SCGA for this function are slightly worse, that is, 0.0008 for the best function value with 8127 function evaluations. However, there is another function with $n = 10$ studied in [21] for which the Simplex GA [21] needed to generate 640 generations to obtain the accuracy 10^{-3}, whereas SCGA needed only 60 generation to obtain the accuracy 10^{-9}.

SCGA is also compared with the Simplex GA methods of [20, 22] using De Joung F5 function, see A.17 in the Appendix. All these methods have the same rate of success (100%), but the Simplex GA [20] required a large number of function evaluations, as shown in Table 4. We note that the results for the Simplex GA methods of [20, 22], which are cited from [22], are the average taken over 10 trials. However, the reference [22] does not give the condition used to judge the success of trials for both of the Simplex GA methods of [20, 22], whereas the results for SCGA are taken over 100 trials and we use condition (5) with $\epsilon_1 = 10^{-4}$ and $\epsilon_2 = 10^{-6}$ to judge the success of trials. It is noteworthy that the average error obtained by SCGA for De Joung function F5 is 1.6×10^{-7}.

5 Conclusion

In this paper, we have introduced a simplex coding genetic algorithm that uses a set of simplices as the population. Applying the Nelder-Mead local search method on these simplices in addition to the ordinary GA operations such as selection, crossover and mutation enhances the exploration process and accelerates the convergence of the GA. We also have introduced a new kind of multi-parents crossover that gives more than two parents the chance to cooperate in reproducing children and exploring the region around these parents. Moreover, using a local search method again in the final stage helps the GA in obtaining good accuracy quickly. Finally, the computational results show that the SCGA works successfully on some well known test functions.

References

[1] J. E. Baker (1985). Adaptive selection methods for genetic algorithms. In: J. J. Grefenstette (Ed.), *Proceedings of the First International Conference on Genetic Algorithms*, pp. 101–111. Lawrence Erlbaum Associates, Hillsdale, MA.

[2] M. Bessaou and P. Siarry (2001). A genetic algorithm with real-value coding to optimize multimodal continuous functions, *Struct. Multidisc. Optim.,* **23**, 63–74.

[3] I. Bohachevsky, M. E. Johnson and M. L. Stein (1986). Generalized simulated annealing for function optimization, *Technometrics,* **28**, 209–217.

[4] R. Chelouah and P. Siarry (2000). A continuous genetic algorithm designed for the global optimization of multimodal functions, *J. Heuristics,* **6**, 191–213.

[5] D. E. Goldberg (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley.

[6] T. Günal (2000). A hybrid approach to the synthesis of nonuniform lossy transmission-line impedance-matching sections, *Microwave and Optical Technology Letters*, **24**, 121–125.

[7] A. Hedar and M. Fukushima (2002). Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization, *Optim. Methods and Software*, **17**, 891–912.

[8] R. Horst and P. M. Pardalos (Eds.) (1995). *Handbook of Global Optimization.* Kluwer Academic Publishers, Boston, MA.

[9] C. T. Kelley (1999). Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition, *SIAM J. Optim.*, **10**, 43–55.

[10] C. T. Kelley (1999). *Iterative Methods for Optimization.* Frontiers Appl. Math. 18, SIAM, Philadelphia, PA.

[11] K. I. M. McKinnon (1999). Convergence of the Nelder-Mead simplex method to a non-stationary point*, SIAM J. Optim.,* **9**, 148-158.

[12] Z. Michalewicz (1996). *Genetic algorithms + Data structures = Evolution programs.* Springer, Berlin, Heidelberg, New York.

[13] P. Moscato (1999). Memetic algorithms: An introduction. In: D. Corne, M. Dorigo and F. Glover (Eds.), *New Ideas in Optimization.* McGraw-Hill, London, UK.

[14] M. Musil, M. J. Wilmut and R. Chapman (1999). A hybrid simplex genetic algorithm for estimating geoacoustic parameters using matched-field inversion, *IEEE J. Oceanic Eng.*, **24**, 358–369.

[15] J. A. Nelder and R. Mead (1965). A simplex method for function minimization, *Comput. J.*, **7**, 308–313.

[16] I. H. Osman and J. P. Kelly (Eds.) (1996). *Meta-Heuristics: Theory and Applications.* Kluwer Academic Publishers, Boston, MA.

[17] P. M. Pardalos and M. G. C. Resende (Eds.) (2002). *Handbook of Applied Optimization.* Oxford University Press, Oxford.

[18] P. M. Pardalos, H. E. Romeijn and H. Tuy (2000). Recent developments and trends in global optimization, *J. Comput. Appl. Math.*, **124**, 209–228.

[19] P. M. Pardalos and H. E. Romeijn (Eds.) (2002). *Handbook of Global Optimization.* Kluwer Academic Publishers, Boston, MA.

[20] J. M. Renders and H. Bersini (1994). Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways. In: Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel. D. B. Fogel, and H. Kitano (Eds.), *Proceedings of the First IEEE Conference on Evolutionary Computation*, pp. 312–317. IEEE Press.

[21] R. Yang and I. Douglas (1998). Simple genetic algorithm with local tuning: Efficient global optimizing technique, *J. Optim. Theory Appl.,* **98**, 449–465.

[22] J. Yen, J. C. Liao, B. Lee and D. Randolph (1998). A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method, *IEEE Trans. on Syst., Man, and Cybern. B*, **28**, 173–191.

[23] R. Zentner, Z. Sipus and J. Bartolic (2001). Optimization synthesis of broadband circularly polarized microstrip antennas by hybrid genetic algorithm, *Microwave and Optical Technology Letters*, **31**, 197–201.

A List of test functions

A.1 Branin RCOS function (RC)

- Number of variables: $n = 2$.
- Definition: $RC(x_1, x_2) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$.
- Range of initial points: $-5 < x_1 < 10,\ 0 < x_2 < 15$.
- Number of local minima: no local minimum[1].
- Global minima: $(x_1, x_2)^* = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)$;

$RC((x_1, x_2)^*) = 0.397887$.

A.2 Easom function (ES)

- Number of variables: $n = 2$.
- Definition: $ES(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$.
- Range of initial points: $-10 < x_j < 10,\ j = 1, 2$.
- Number of local minima: several local minima.
- The global minimum: $(x_1, x_2)^* = (\pi, \pi)$; $ES((x_1, x_2)^*) = -1$.

A.3 Goldstein and Price function (GP)

- Number of variables: $n = 2$.
- Definition: $GP(x_1, x_2) = u * v$,

where $u = 1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)$,
and $v = 30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)$.

- Range of initial points: $-2 < x_j < 2,\ j = 1, 2$.
- Number of local minima: 4 local minima.
- The global minimum: $(x_1, x_2)^* = (0, -1)$; $GP((x_1, x_2)^*) = 3$.

[1]Throughout the Appendix, ‘local minimum’ means a mere local minimum that is not a global minimum

A.4 Hump function (HM)

- Number of variables: $n = 2$.
- Definition: $HM(x_1, x_2) = 1.0316285 + 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$.
- Range of initial points: $-5 < x_j < 5,\ j = 1, 2$.
- Number of local minima: no local minima.
- Global minima: $(x_1, x_2)^* = (0.0898, -0.7126),\ (-0.0898, 0.7126)$;

$HM((x_1, x_2)^*) = 0$.

A.5 Shubert function (SH)

- Number of variables: $n = 2$.
- Definition: $SH(x_1, x_2) = \left(\sum_{j=1}^{5} j\cos\left((j+1)\,x_1 + j\right)\right)\left(\sum_{j=1}^{5} j\cos\left((j+1)\,x_2 + j\right)\right)$.
- Range of initial points: $-10 < x_j < 10,\ j = 1, 2$.
- Number of local minima: 760 local minima.
- Global minima: 18 global minima and $SH((x_1, x_2)^*) = -186.7309$.

A.6 Michalewicz function (MZ)

- Number of variables: $n = 2$.
- Definition: $MZ(x_1, x_2) = -\sum_{j=1}^{2} \sin(x_j)\left(\sin\left(jx_j^2/\pi\right)\right)^{2m};\ m = 10$.
- Range of initial points: $0 \le x_j \le \pi,\ j = 1, 2$.
- Number of local minima: many local minima.
- Global minima: $MZ((x_1, x_2)^*) = -1.8013$.

A.7 Bohachevsky functions (B_m)

- Number of variables: $n = 2$.
- Three functions were considered: B_1, B_2 and B_3.
- Definitions: $B_1(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$.

$$B_2(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3.$$
$$B_3(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3.$$

Table 5: The coefficients of Hartmann function $H_{3,4}$

i	a_{ij}			c_i	p_{ij}		
1	3.0	10.0	30.0	1.0	0.689	0.1170	0.2673
2	0.1	10.0	35.0	1.2	0.4699	0.4387	0.7470
3	3.0	10.0	30.0	3.0	0.1091	0.8732	0.5547
4	0.1	10.0	35.0	3.2	0.0381	0.5743	0.8828

- Range of initial points: $-10 \leq x_j \leq 10,\ j = 1, 2$.
- Number of local minima: many local minima.
- Global minima: $(x_1, x_2)^* = (0, 0);\ B_m((x_1, x_2)^*) = 0;\ m = 1, 2, 3$.

A.8 De Joung function (DJ)

- Number of variables: $n = 3$.
- Definition: $DJ(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2$.
- Range of initial points: $-5 < x_j < 5,\ j = 1, 2, 3$.
- Number of local minima: no local minima.
- The global minimum: $(x_1, x_2, x_3)^* = (0, 0, 0);\ DJ((x_1, x_2, x_3)^*) = 0$.

A.9 Hartmann function ($H_{3,4}$)

- Number of variables: $n = 3$.
- Definition: $H_{3,4}(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2\right]$. The coefficients are shown in Table 5.
- Range of initial points: $0 < x_j < 1,\ j = 1, 2, 3$.
- Number of local minima: 4 local minima.
- The global minimum: $\mathbf{x}^* = (0.114614, 0.555649, 0.852547);\ H_{3,4}(\mathbf{x}^*) = -3.86278$.

Table 6: The coefficients of Shekel function $S_{4,m}$

i	a_{ij}				c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

A.10 Shekel functions ($S_{4,m}$)

- Number of variables: $n = 4$.
- Definition: $S_{4,m}(\mathbf{x}) = -\sum_{i=1}^{m}\left[\sum_{i=1}^{4}(x_i - a_i)^2 + c_i\right]^{-1}$. The coefficients are shown in Table 6.
- Three functions were considered: $S_{4,5}, S_{4,7}$ and $S_{4,10}$.
- Range of initial points: $0 < x_j < 10,\ j = 1, \ldots, 4$.
- Number of local minima: m local minima.
- Same global minimum for three functions $S_{4,5},\ S_{4,7}$ and $S_{4,10}$: $\mathbf{x}^* = (4, 4, 4, 4)$;

$S_{4,5}(\mathbf{x}^*) = -10.1532,\ S_{4,7}(\mathbf{x}^*) = -10.4029$ and $S_{4,10}(\mathbf{x}^*) = -10.5364$.

A.11 Hartmann function ($H_{6,4}$)

- Number of variables: $n = 6$.
- Definition: $H_{6,4}(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$. The coefficients are ahown in Table 7.
- Range of initial points: $0 < x_j < 1,\ j = 1, \ldots, 6$.
- Number of local minima: 6 local minima.
- The global minimum:

Table 7: The coefficients of Hartmann function $H_{6,4}$

i	a_{ij}						c_i
1	10.00	3.00	17.00	3.50	1.70	8.00	1.0
2	0.05	10.00	17.00	0.10	8.00	14.00	1.2
3	3.00	3.50	1.70	10.0	17.00	8.00	3.0
4	17.00	8.00	0.05	10.00	0.10	14.00	3.2

i	p_{ij}					
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

$\mathbf{x}^* = (0.201690, 0.150011, 0.476874, 0.275332, 0.311652, 0.657300)$;
$H_{6,4}(\mathbf{x}^*) = -3.32237$.

A.12 Griewank function (GR)

- Number of variables: $n = 6$.
- Definition: $GR(\mathbf{x}) = \sum_{j=1}^{6} \frac{x_j^2}{4000} - \prod_{j=1}^{6} \cos\left(\frac{x_j}{\sqrt{j}}\right) + 1$.
- Range of initial points: $-1 < x_j < 1,\ j = 1, 2, \ldots, 6$.
- Number of local minima: many local minima.
- The global minimum: $\mathbf{x}^* = (0, \ldots, 0)$, $GR(\mathbf{x}^*) = 0$.

A.13 Rosenbrock functions (R_n)

- Number of variables: $n = 2, 5, 10$.
- Definition: $R_n(\mathbf{x}) = \sum_{j=1}^{n-1} \left[100\left(x_j^2 - x_{j+1}\right)^2 + (x_j - 1)^2\right]$.
- Range of initial points: $-5 < x_j < 10,\ j = 1, 2, \ldots, n$.
- Number of local minima: no local minimum.
- The global minimum: $\mathbf{x}^* = (1, \ldots, 1)$, $R_n(\mathbf{x}^*) = 0$.

A.14 Zakharov functions (Z_n)

- Number of variables: $n = 2, 5, 10$.
- Definition: $Z_n(\mathbf{x}) = \sum_{j=1}^{n} x_j^2 + \left(\sum_{j=1}^{n} 0.5jx_j\right)^2 + \left(\sum_{j=1}^{n} 0.5jx_j\right)^4$.
- Range of initial points: $-5 < x_j < 10,\ j = 1, 2, \ldots, n$.
- Number of local minima: no local minimum.
- The global minimum: $\mathbf{x}^* = (0, \ldots, 0)$, $Z_n(\mathbf{x}^*) = 0$.

A.15 Function (F_1)

- Number of variables: $n = 2$.
- Definition: $F_1(x_1, x_2) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$.
- Range of initial points: $-1 < x_j < 1,\ j = 1, 2$.
- Number of local minima: many local minima.
- The global minimum: $(x_1, x_2)^* = (0, 0);\ F_1((x_1, x_2)^*) = -2$.

A.16 Function (F_2)

- Number of variables: $n = 10$.
- Definition: $F_2(\mathbf{x}) = \sum_{j=1}^{10} \min\left\{|x_j - 0.2| + a, |x_j - 0.4|, |x_j - 0.7| + a\right\},\ a = 0.05$.
- Range of initial points: $0 < x_j < 1,\ j = 1, \ldots, 10$.
- Number of local minima: many local minima.
- The global minimum: $\mathbf{x}^* = (0.4, 0.4, \ldots, 0.4);\ F_2(\mathbf{x}^*) = 0$.

A.17 De Joung F5 function

- Number of variables: $n = 2$.
- Definition: $F5(x_1, x_2) = \left(0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$,

  $a_{1j} = -32, -16, 0, 16, 32$ for $j = 1, 2, \ldots, 5$,
  $a_{1k} = a_{1j}$ for $k = j + 5, j + 10, j + 15, j + 20$, and $j = 1, 2, \ldots, 5$,
  $a_{2j} = -32, -16, 0, 16, 32$ for $j = 1, 6, 11, 16, 21$,
  $a_{2k} = a_{2j}$ for $k = j + 1, j + 2, j + 3, j + 4$, and $j = 1, 6, 11, 16, 21$.

- Range of initial points: $-65.536 < x_i < 65.536,\ i = 1, 2$.
- Number of local minima: many local minima.
- The global minimum: $(x_1, x_2)^* = (-32, -32);\ F5((x_1, x_2)^*) = 0.998004$.