# A Robust Real-Coded Genetic Algorithm using Unimodal Normal Distribution Crossover Augmented by Uniform Crossover : Effects of Self-Adaptation of Crossover Probabilities

**Isao Ono**
Faculty of Engineering,
University of Tokushima,
2-1, Minami-Josanjima,
Tokushima, 770-8506, JAPAN.
e-mail: isao@is.tokushima-u.ac.jp
Phone: +81-88-656-9139

**Hajime Kita**
Graduate School of
Interdisciplinary Sci. & Eng.,
Tokyo Institute of Technology,
4259, Nagatsuta, Yokohama,
226-8502, JAPAN.
e-mail: kita@dis.titech.ac.jp
Phone: +81-45-924-5680

**Shigenobu Kobayashi**
Graduate School of
Interdisciplinary Sci. & Eng.,
Tokyo Institute of Technology,
4259, Nagatsuta, Yokohama,
226-8502, JAPAN.
e-mail: kobayasi@dis.titech.ac.jp
Phone: +81-45-924-5648

## Abstract

This paper presents a robust real-coded genetic algorithm using the Unimodal Normal Distribution Crossover (UNDX) enhanced by the Uniform Crossover (UX). The UNDX has an advantage, which most other crossover operators do not have, that it can efficiently optimize functions with strong epistasis among parameters. However, the UNDX has a disadvantage that there can be some areas where the UNDX cannot sufficiently generates individuals by using a given initial population. Contrary to this, the UX has an advantage that it can make individuals in areas where the UNDX cannot with the same initial population and has a disadvantage that it cannot efficiently optimize functions with epistasis among parameters. To make use of the advantages of the UNDX and the UX that are complementary to each other, we introduce a mechanism of adapting the operator probabilities according to the characteristics of a given function. Through some experiments, we show the robustness of the proposed method by demonstrating that the proposed method can solve more various functions than a GA using only the UNDX.

## 1 Introduction

There have been proposed various real-coded genetic algorithms for function optimization so far [Davis 90], [Radcliffe 90], [Wright 91], [Janikow 91], [Michalewicz 92], [Eshelman 93], [Voigt 95], [Ono 96], [Eshelman 97], [Ono 97]. Functions that we have to optimize in real world problems often have strong epistasis among parameters. However, the performance of most real-coded GAs deteriorates considerably on functions with epistasis among parameters [Salomon 96], [Ono 96]. Against this problem, the authors have proposed the Unimodal Normal Distribution Crossover (UNDX) for real-coded GAs, considering epistasis among parameters, and have shown that the UNDX can efficiently optimize some benchmark functions with strong epistasis among parameters [Ono 97]. Eshelman et al. have independently proposed a similar crossover operator named the directional-BLX [Eshelman 97].

The UNDX generates offsprings mainly on the line segment connecting two parents. The authors have shown that the UNDX preserves the correlation among parameters well[Kita 98]. The advantage of the UNDX in optimization of epistatic functions is attributed to this feature.

However, at the same time, this feature of the UNDX brings about the following two disadvantages. The first problem is that there can be some areas where the UNDX cannot generates offsprings with a given initial population when the population size is small relative to the search space. If one of such areas is the promising area including the optimal point, the UNDX fails in finding the optimal point. The second problem is that the UNDX has the difficulty in finding the optimal point(s) near the boundaries.

In this paper, we propose a robust real-coded genetic algorithm employing both the UNDX and the UX, complementary crossover operators. To choose a crossover operator efficiently, we introduce a self-adaptive mechanism of crossover probabilities into the algorithm. To examine the effectiveness of the proposed method, we apply it to optimization problems of various benchmark functions such as multimodal functions, epistatic functions and mixed variables functions including discrete parameters as well as continuous ones.

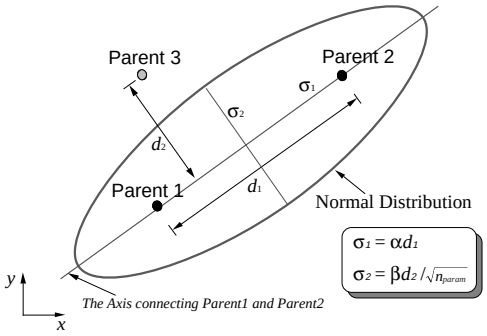



Figure 1: UNDX ($n_{\rm param} = 2$) [Ono 97]

In section two, we briefly review the UNDX and the MGG [Satoh 96], which is a generation-alternation model used with the UNDX in [Ono 97]. We discuss some advantages and disadvantages of the UNDX+MGG [Ono 97] in section three. In section four, we propose a robust real-coded genetic algorithm with adapting probabilities of applying the UX and the UNDX. The proposed method is applied to some benchmark functions to show its robustness in section five. Section six is a discussion and section seven concludes this paper.

## 2 A Brief Review of UNDX+MGG

### 2.1 UNDX [Ono 97]

The UNDX generates two offsprings by using the normal distribution which is defined by three parents, as shown in Fig. 1. Offsprings are made around the line segment connecting the two parents, Parent 1 and Parent 2. One of the standard deviation values of the normal distribution which corresponds to the axis connecting Parent 1 and Parent 2 is proportional to the distance between Parent 1 and Parent 2. The others are proportional to the distance of the third parent, Parent 3, from the line connecting Parent 1 and Parent 2 and are multiplied by $1/\sqrt{n_{\rm param}}$, where $n_{\rm param}$ is the number of parameters. The effect of $1/\sqrt{n_{\rm param}}$ is to keep the distribution of individuals unchanged in the process of applying the UNDX repetitively [Kita 98]. Offsprings are generated as the followings:

$$\vec{C}_1 = \vec{m} + z_1\vec{e}_1 + \sum_{k=2}^{n_{\rm param}} z_k\vec{e}_k, \qquad (1)$$

$$\vec{C}_2 = \vec{m} - z_1\vec{e}_1 - \sum_{k=2}^{n_{\rm param}} z_k\vec{e}_k, \qquad (2)$$

where $\vec{C}_1$, $\vec{C}_2$ are offspring vectors respectively, $\vec{m} = (\vec{P}_1 + \vec{P}_2)/2$, $\vec{P}_1$ and $\vec{P}_2$ are parent vectors of Parent 1 and Parent 2 respectively, $z_1 \sim N(0, \sigma_1^2)$, $z_k \sim N(0, \sigma_2^2)$ $(k = 2, \cdots, n_{\rm param})$, $\sigma_1 = \alpha d_1$, $\sigma_2 = \beta d_2/\sqrt{n_{\rm param}}$, $\vec{e}_1 = (\vec{P}_2 - \vec{P}_1)/\left|\vec{P}_2 - \vec{P}_1\right|$, vectors $\vec{e}_2, \cdots \vec{e}_{n_{param}}$ are the orthogonal basis vectors spanning the subspace perpendicular to vector $\vec{e}_1$, $d_1$ is the distance between Parent 1 and Parent 2, $d_2$ is the distance of the third parent, Parent 3, from the line connecting Parent 1 and Parent 2. $\alpha$ and $\beta$ are constants and $\alpha = 0.5$ and $\beta = 0.35$ are recommended [Ono 97], [Kita 98].

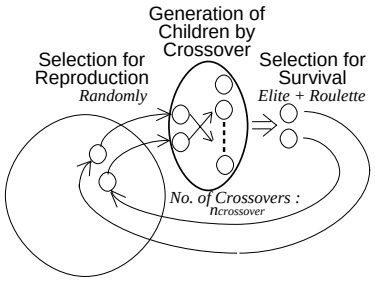



Figure 2: MGG [Satoh 96]

### 2.2 MGG [Satoh 96]

The Minimal Generation Gap (MGG) model has been proposed in [Satoh 96] and has shown good performance [Satoh 96]. The MGG was used in [Ono 97] as a generation-alternation model. Fig.2 illustrates the MGG. The algorithm of the MGG is described as the followings:

1. *Generation of Initial Population*
   Make an initial population that is composed of random real number vectors.
2. *Selection for Reproduction*
   Choose a pair of individuals by random sampling without replacement from the population.
3. *Generation of Offsprings*
   Generate $2n_{\rm crossover}$ offsprings by applying the UNDX to the chosen pair of individuals $n_{\rm crossover}$ times. The third parent which is used for calculating the standard deviation value, $\sigma_2$, is randomly chosen from the population.
4. *Selection for Survival*
   Choose two individuals from the family containing the parents and their offsprings; one is the best individual and the other is selected from $2n_{\rm crossover} + 1$ individuals other than the best one by the rank-based roulette wheel selection [Goldberg 89]. Replace the parents chosen in Step 2 in the population with the two individuals.
5. Repeat the above procedures from step 2 to step 4 until a certain stop condition is satisfied.

## 3 Features of UNDX+MGG : Advantages and Disadvantages

The feature of the UNDX is that the distribution of offsprings generated by the UNDX with an appropriate parameter setting preserves the mean vector and the covariance matrix of the distribution of the parent population, according to a theoretical analysis [Kita 98]. In other word, the UNDX generates offsprings distributed similarly to the parent population. This feature gives the UNDX an advantage as well as a disadvantage as described in the followings.

The feature becomes an advantage when parents distribute on some promising areas, especially on valleys that are not parallel to the coordinate axes. Such valleys are caused by epistasis among parameters. This case often happens in the middle search phases or later. In this case, the UNDX can efficiently search along the valley without making children away from the valley. If the optimal point lies on the valley, the UNDX can find the optimal point efficiently. Therefore, for the UNDX to perform successfully, it is important to sufficiently sample points from all over the search space in the early search phases for correctly finding promising areas and fitting the distribution of parents to the shape of the promising areas.

On the other hand, the feature becomes a disadvantage when parents do not distribute on any promising areas and exploring the search space is required. This case often happens in the early search phases. In this case, there can be some areas where the UNDX cannot generates offsprings with a given initial population when the population size is small relative to the search space size, which often happens when the number of parameters to be optimized is large. Suppose that we optimize a function with $n_{\mathrm{param}}$ parameters by using a UNDX-based GA with the population size of 500 and that we divide the search space into $2^{n_{\mathrm{param}}}$ subspaces by splitting each axis into two parts. If $n_{\mathrm{param}} > 9$, we have more than 512 subspaces and the expected number of parents in each subspace becomes less than one. This means that there are many subspaces including no parents. In situations like this, it is difficult to generate offsprings in the subspaces including no parents because the UNDX basically makes offsprings along the line connecting two parents.

Furthermore, the UNDX has another problem. The UNDX tends to sample more points near the center of the search space than ones near the boundaries even if parents are uniformly distributed over the search space. As the result, the UNDX has the difficulty in finding the optimal point(s) near the boundaries.

The feature of the MGG is that it can maintain a diversity in the population during the search very well though it works based on only the order of fitness. An advantage of the MGG is that the algorithm is very simple and it can be applied to any type of problems. A disadvantage of the MGG is that the convergence speed is slow. The reason is that it needs to choose a bad individual in the step of *selection for survival* for maintaining a diversity in genotype because the selection is based on only fitness.

## 4 A Robust Real-Coded GA : (UX,UNDX)+EMGG

### 4.1 Uniform Crossover for Augmenting UNDX

To compensate for insufficient capability of the UNDX in exploration of the search space, we introduce the Uniform Crossover (UX) that has complementary characteristics to the UNDX. The UX has been used in several real-coded GAs [Davis 90],[Wright 91],[Janikow 91], [Michalewicz 92] and in Evolution Strategies (ESs) [Bäck 96]. In ESs, the UX is called the Discrete Recombination and has been report to accelerate the search process according to [Bäck 96].

In the UX, for each component of the vectors, it is decided at random from which of both parents the component is copied to the offsprings.

$$c_{1i} = p_{xi},\ c_{2i} = p_{yi} \tag{3}$$

where $\vec{P}_1 = (p_{11},\ \cdots,\ p_{1n_{\mathrm{param}}})$ and $\vec{P}_2 = (p_{21},\cdots,\ p_{2n_{\mathrm{param}}})$ are parent vectors, $\vec{C}_1 = (c_{11},\ \cdots,\ c_{1n_{\mathrm{param}}})$ and $\vec{C}_2 = (c_{21},\ \cdots,\ c_{2n_{\mathrm{param}}})$ are offspring vectors and $n_{\mathrm{param}}$ is the number of parameters. Index $x$ is a uniform random variable which takes 1 or 2. Index $y$ is a complement of $x$ given by $y = 3 - x$.

The UX can generate various offsprings that the UNDX cannot make from the same pair of parents. Moreover, the UX uniformly samples points from the search space when parents distribute uniformly. We expect that the UX widely samples points from various areas to correctly adapt the parent distribution to the shapes of promising areas and, as the result, the UNDX efficiently searches in the promising areas exploiting the parent distribution.

However, the UX has a disadvantage that the search efficiency considerably deteriorates when applied to functions with epistasis among parameters.

### 4.2 Self-Adaptation of Crossover Probabilities

Because the UX and the UNDX are complementary to each other, a mechanism of utilizing a suitable one to the situation is required to achieve efficient search.

We introduce a mechanism for adapting operator probabilities into the MGG. Some studies on adapting op-

erator probabilities have been done [Davis 89], [Julstrom 95], [Tuson 96]. In this paper, we employ a very simple mechanism. The basic idea of the mechanism is that the probabilities of applying each operator are adapted based on the performance of the offsprings generated by the operator, which was originally proposed in [Davis 89]. Similar idea has been proposed by Schwefel for mutation operator[Schwefel 95], [Bäck 96]. In our algorithm, the probabilities of applying the UX and the UNDX are calculated every generation of $n_{\rm kid} \times (n_{\rm pop}/2)$ pairs of offsprings by these operators as follows:

$$P^{\rm apply}_{\rm UNDX} = P^{\rm success}_{\rm UNDX}/(P^{\rm success}_{\rm UX} + P^{\rm success}_{\rm UNDX}) \quad (4)$$

$$P^{\rm apply}_{\rm UX} = 1 - P^{\rm apply}_{\rm UNDX} \quad (5)$$

where $n_{\rm kid}$ is a constant value given by a user, $n_{\rm pop}$ is the population size, $P^{\rm apply}_{\rm UNDX}$ and $P^{\rm apply}_{\rm UX}$ are the probabilities of the UNDX and the UX respectively, $P^{\rm success}_{\rm UNDX}$ and $P^{\rm success}_{\rm UX}$ are the rate at which each crossover operator succeeds in generating a better offspring than the both parents after the last update of $P^{\rm apply}_{\rm UNDX}$ and $P^{\rm apply}_{\rm UX}$. We limit the range of $P^{\rm apply}_{\rm UNDX}$ to 0.05 to 0.95 because $P^{\rm success}_{\rm UNDX}$ (or $P^{\rm success}_{\rm UX}$) cannot be calculated if $P^{\rm apply}_{\rm UNDX}$ is 0.0 (or 1.0).

### 4.3 Enhanced-MGG (EMGG)

The MGG has a problem with the way of *selection for survival* as described in the previous section. To overcome the problem, we introduce a new selection scheme, in which individuals are chosen based on not only fitness but also the distance between individuals, into the MGG.

In the new scheme for *selection for survival*, if an offspring is better than the both parents, the offspring replaces one of its parents nearer to the offspring. The other parent is replaced with the other offspring if the offspring is better than the parent.

We expect that the new scheme improves the convergence speed, keeping the possibility of finding the optimum. In this paper, we call the new model the Enhanced-MGG (EMGG).

### 4.4 Algorithm

In this section, we design a real-coded GA with adapting probabilities of the UX and the UNDX, named (UX,UNDX)+EMGG, based on the above discussions. The algorithm is as the followings:

1. *Generation of Initial Population*
   Make $n_{\rm pop}$ real-number vectors randomly and let them be an initial population. Let $P^{\rm apply}_{\rm UNDX}$ be $InitP^{\rm apply}_{\rm UNDX}$ given by a user and $P^{\rm apply}_{\rm UX}$ be $(1 - InitP^{\rm apply}_{\rm UNDX})$.
2. *Initialization of parameters for adaptation*
   Let $N_{\rm UX}$, $N_{\rm UNDX}$, $N^{\rm success}_{\rm UX}$ and $N^{\rm success}_{\rm UNDX}$ be zero respectively.
3. *Generation-alternation Cycle #1*
   Repeat step 3.1 to 3.3 $n_{\rm kid} \times (n_{\rm pop}/2)$ times.
   3.1 *Selection for Reproduction #1*
   Choose a pair of individuals, Parent 1 and Parent 2, by random sampling without replacement from the population.
   3.2 *Generation of Offsprings #1*
   Choose the UX or the UNDX according to operator probabilities, $P^{\rm apply}_{\rm UX}$ and $P^{\rm apply}_{\rm UNDX}$. Here, let the chosen crossover operator be CROSSOVER. Increment $N_{\rm CROSSOVER}$. Apply the CROSSOVER to Parent 1 and Parent 2 to make two offsprings, Child 1 and Child 2, where the offspring nearer to Parent 1 is named Child 1 and the other Child 2. In applying the UNDX, the third parent which is used for calculating the standard deviation value, $\sigma_2$, is randomly chosen from the population. If Child 1 or Child 2 is better than the both parents, increment $N^{\rm success}_{\rm CROSSOVER}$.
   3.3 *Selection for Survival #1*
   If Child 1 is better than the both parents,
   - Replace Parent 1 with Child 1.
   - If Child 2 is better than Parent 2, then replace Parent 2 with Child 2.

   If Child 2 is better than the both parents,
   - Replace Parent 2 with Child 2.
   - If Child 1 is better than Parent 1, then replace Parent 1 with Child 1.
4. *Generation-alternation Cycle #2*
   If $n_{\rm pop}/2 > N^{\rm success}_{\rm UX} + N^{\rm success}_{\rm UNDX}$ then repeat step 4.1 to 4.3 $(n_{\rm pop}/2 - (N^{\rm success}_{\rm UX} + N^{\rm success}_{\rm UNDX}))$ times.
   4.1 *Selection for Reproduction #2*
   Do the same procedure as step 3.1.
   4.2 *Generation of Offsprings #2*
   Do the same procedure as step 3.2.
   4.3 *Selection for Survival #2*
   Choose two individuals from the family containing the parents and their offsprings; one is the best individual and the other is selected from three individuals other than the best one by the rank-based roulette wheel selection. Replace the parents chosen in Step 4.1 in the population with the individuals.
5. *Update of the operator probabilities*
   Update the operator probabilities according to eq. (4) and (5). Here, $P^{\rm success}_{\rm UX} = N^{\rm success}_{\rm UX}/N_{\rm UX}$ and $P^{\rm success}_{\rm UNDX} = N^{\rm success}_{\rm UNDX}/N_{\rm UNDX}$.
6. Repeat the above procedures from step 2 to step 5 until a certain stop condition is satisfied.

## 5 Experiments

To examine the robustness of the proposed method, we performed some experiments. We compared the (UX,UNDX)+EMGG with the UNDX+EMGG in which the UX was not used (i.e. $P_{\mathrm{UX}}^{\mathrm{apply}} = 0.0$, $P_{\mathrm{UNDX}}^{\mathrm{apply}} = 1.0$), the UX+EMGG in which the UNDX was not used (i.e. $P_{\mathrm{UX}}^{\mathrm{apply}} = 1.0$, $P_{\mathrm{UNDX}}^{\mathrm{apply}} = 0.0$) and the UNDX+MGG [Ono 97]. $InitP_{\mathrm{UNDX}}^{\mathrm{apply}}$ was set to be 0.1 and $n_{\mathrm{kid}}$ to be 100 in the (UX,UNDX)+EMGG. $n_{\mathrm{crossover}}$ was set to be 100 in the UNDX+MGG. In all experiments, the population size was set to be 500 and ten independent runs were performed.

We used ten 20-dimensional functions as benchmark problems here. These functions can be categorized into three types as follows:

- *Continuous function*
  All variables are continuous.
- *Discrete function*
  All variables are discrete. Each variable takes a value from the value set that contains values discretized at the interval of 0.2 within a given range of each variable.
- *Mixed function*
  The first ten variables are continuous and the remains are discrete. Each discrete variable takes values discretized at the interval of 0.2 within a given range of each variable.

In discrete functions and mixed ones, the UNDX does not always generate feasible offsprings whose all values are contained in a given value set. Illegal values in offsprings are modified to the nearest values respectively before the offsprings are evaluated.

The ten benchmark functions used here are follows:

- *Continuous/Discrete/Mixed Rosenbrock function*
  The Rosenbrock function is given by

$$f(\vec{x}) = \sum_{i=2}^{n} [100(x_1 - x_i^2)^2 + (x_i - 1)^2], \qquad (6)$$

  where $-2.048 \leq x_i \leq 2.048$ and $n = 20$. This is a unimodal function with strong epistasis among parameters. It has a parabolic valley along the curve $x_1 = x_i^2 \, (i = 2, \cdots n)$ with the unique minimum of zero at the point $(1, \cdots, 1)$. Fig.3(a), (b) and (c) show the online performance of each method on the Continuous, Discrete and Mixed Rosenbrock functions, respectively. Each curve shows the average of function values of the best individuals in each run.
- *Continuous/Discrete/Mixed Rastrigin function*
  The Rastrigin function is given by

$$f(\vec{x}) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)], \qquad (7)$$

  where $-5.12 \leq x_i \leq 5.12$ and $n = 20$. It is a multimodal function with no epistasis among parameters and has a unique global minimum of zero at the origin surrounded by many local minima. Fig.3(d), (e) and (f) show the online performance of each method on the Continuous, Discrete and Mixed Rastrigin functions, respectively.
- *Continuous/Discrete/Mixed Rotated-Rastrigin function*
  The Rotated-Rastrigin function is the one made by randomly rotating the Rastrigin function around the origin. It is a multimodal function with epsitasis among parameters. Fig.3(g), (h) and (i) show the online performance of each method on the Continuous, Discrete and Mixed Rotated-Rastrigin functions, respectively.
- *Continuous Schwefel function*
  The Schwefel function is given by

$$f(\vec{x}) = 418.9828873n + \sum_{i=1}^{n} x_i \sin\sqrt{|x_i|}, \qquad (8)$$

  where $-500 \leq x_i \leq 500$ and $n = 20$. This function is a multimodal one with no epistasis among parameters. It has a unique global minimum of zero at the point $(-420.968746, \cdots, -420.968746)$ near the boundaries and has many local minima. Fig.3(j) shows the online performance of each method on the function.

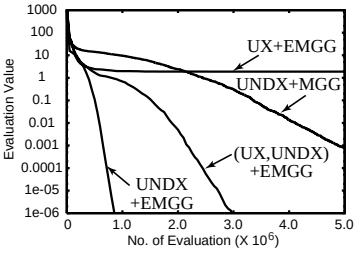
Fig.4 shows the changes of the probabilities of the UNDX, $P_{\mathrm{UNDX}}^{\mathrm{apply}}$, during the search when the (UX,UNDX)+EMGG is applied to each function. Each curve is the average of ten runs.

Fig.5 summarizes the results shown in Fig.3. In the table, the numbers show the ranking of each method on each function in convergence speed to the optimum. *Failed* means that corresponding method failed in converging to the optimum.

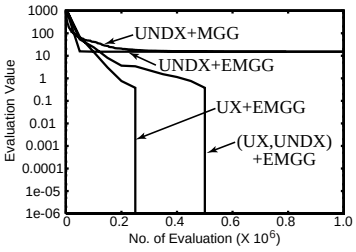
## 6 Discussions

**[Robustness and convergence speed]**

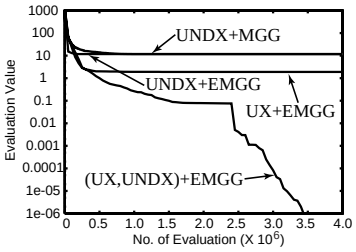
As shown in Fig.3 and Fig.5, the (UX,UNDX)+EMGG succeeds in finding the optimum in all ten functions while the UX+EMGG only two functions and the UNDX+EMGG seven ones. This result shows that the (UX,UNDX)+ EMGG is a robust search algorithm compared to the UX+EMGG and the UNDX+EMGG.



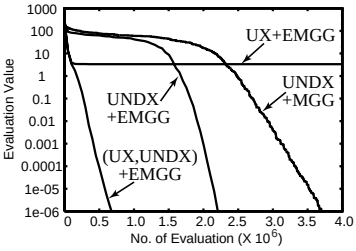

(a) Continuous Rosenbrock function



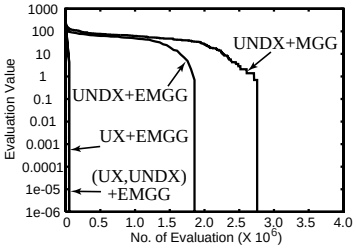

(b) Discrete Rosenbrock function



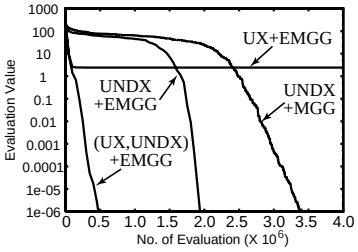

(c) Mixed Rosenbrock function



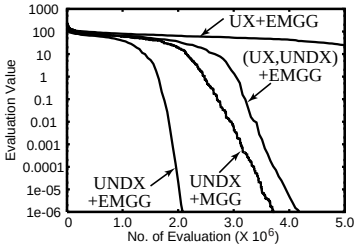

(d) Continuous Rastrigin function



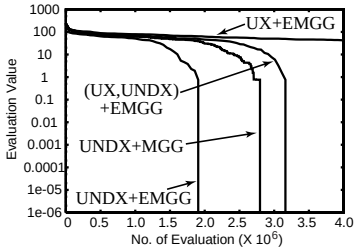

(e) Discrete Rastrigin function



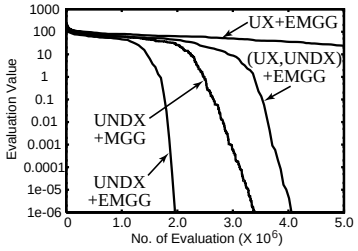

(f) Mixed Rastrigin function



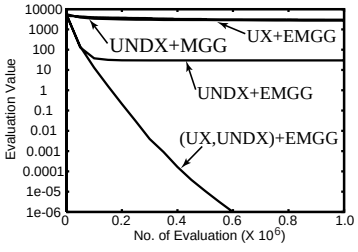

(g) Continuous Rotated-Rastrigin function




(h) Discrete Rotated-Rastrigin function

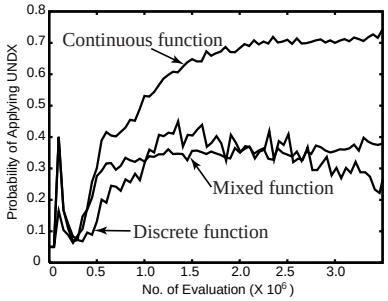



(i) Mixed Rotated-Rastrigin function

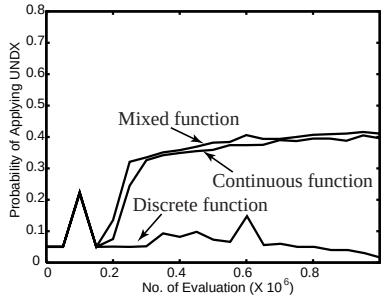



(j) Continuous Schwefel function

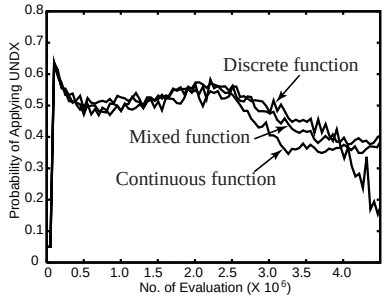
Figure 3: Performance Curves



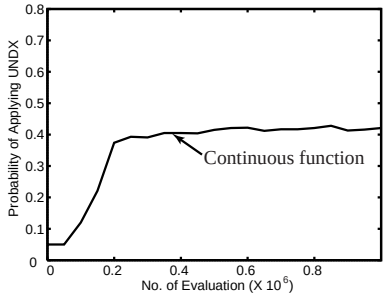

(a) Rosenbrock function




(b) Rastrigin function




(c) Rotated-Rastrigin function




(d) Schwefel function

Figure 4: $P_{\mathrm{UNDX}}^{\mathrm{apply}}$ Curves

| | (UX,UNDX) +EMGG | UX +EMGG | UNDX +EMGG | UNDX +MGG |
|---|---|---|---|---|
| Continuous Rosenbrock | 2 | Failed | 1 | 3 |
| Discrete Rosenbrock | 2 | 1 | Failed | Failed |
| Mixed Rosenbrock | 1 | Failed | Failed | Failed |
| Continuous Rastrigin | 1 | Failed | 2 | 3 |
| Discrete Rastrigin | 1 | 1 | 2 | 3 |
| Mixed Rastrigin | 1 | Failed | 2 | 3 |
| Continuous Rotated-Rastrigin | 3 | Failed | 1 | 2 |
| Discrete Rotated-Rastrigin | 3 | Failed | 1 | 2 |
| Mixed Rotated-Rastrigin | 3 | Failed | 1 | 2 |
| Countinuous Schwefel | 1 | Failed | Failed | Failed |

Figure 5: Summary of the results shown in Fig.3

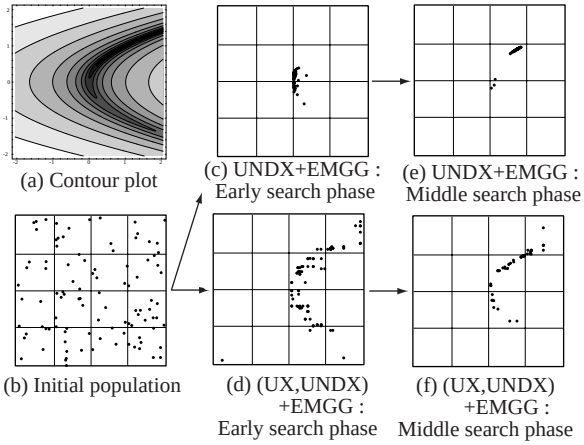



Figure 6: A contour plot of Rosenbrock function and behavior of UNDX+EMGG and (UX, UNDX)+EMGG on Rosenbrock function

As shown in Fig.5, the (UX,UNDX)+EMGG succeeds in finding the optimal point in the Mixed Rosenbrock function and the Continuous Schwefel function where both of the UX+EMGG and the UNDX+EMGG fail. This shows that the UX and the UNDX complementarily work each other during a search process. It is thought that the UX contributes to making parents distribute along the parabolic valley in the Mixed Rosenbrock function and near the boundaries in the Continuous Schwefel function.

In the Continuous Rastirigin function and the Mixed Rastrigin function, the convergence speed of the (UX,UNDX)+ EMGG is faster than that of the UNDX+EMGG although the UX+EMGG fails in finding the optimum as shown in Fig.5. This can be explained by that the UX+EMGG quickly converges to solutions near the optimum as shown in Fig.3(d) and (f).

The convergence speed of the (UX,UNDX)+EMGG is slower than that of the UNDX+EMGG in the Continuous Rosenbrock function, the Continuous Rotated-Rastrigin function, the Discrete Rotated-Rastrigin function and the Mixed Rotated-Rastrigin function.

Fig.6 shows search processes by the (UX,UNDX)+ EMGG and the UNDX+EMGG on the Continuous Rosenbrock function. In the search process by the UNDX+EMGG, first, parents do not widely distribute on the parabolic valley but get together around the origin and stay there for a while (Fig.6(c)). After that, the parents find the direction of the valley and gradually moves along the valley down to the optimal point (Fig.6(e)). On the other hand, in the search process by the (UX,UNDX)+EMGG, first, parents widely distribute on the parabolic valley (Fig.6(d)). After that, the parents get together at good area on the parabolic valley and move to the optimal point (Fig.6(f)). While the UNDX generates improved offsprings on the parabolic valley with high probability in a situation such as Fig.6(c) [1], it generates offsprings in areas other than the parabolic valley with high probabilities in a situation such as Fig.6(d) and the search becomes inefficient. This is the reason the convergence speed of the (UX,UNDX)+EMGG is slow on the Continuous Rosenbrock function.

In the Rotated-Rastrigin function, it is required to search along lines connecting local optimal points for efficient search from the early search phases. In such situations, the UNDX works well from the early search phases and the UX does not attribute to the search very much. As the result, the convergence speed of the (UX,UNDX)+EMGG is slower than the UNDX+EMGG on the Rotated-Rastrigin function.

**[Effects of the adapting operator probabilities]**

In the Continuous and Mixed Rosenbrock functions, it is advantageous to choose the UNDX after parents distribute on the parabolic valley, especially when all the variables are continuous and the parabolic valley is smooth. The curves of $P_{\mathrm{UNDX}}^{\mathrm{apply}}$ begin to rise at the number of evaluation of about $0.25 \times 10^6$ as shown in Fig.4(a). At this time, the performance curves are at around the evaluation value of 7.0 as shown in Fig.3(a) and (c), which means that parents distribute on the parabolic valley. The curve of the Continuous Rosenbrock function rise higher than that of the Mixed Rosenbrock function since the Continuous Rosenbrock

[1] In this situation, the UNDX cannot make improved offsprings and cannot find the direction of the parabolic valley in the Discrete Rosenbrock function and the Mixed Rosenbrock function because the landscape of the area around the origin is flat. This is the reason the UNDX+EMGG and the UNDX+MGG fail in finding the optimum in these functions.

function has a smoother parabolic valley.

In the Continuous Rastrigin function, the Mixed Rastrigin function and the Continuous Schwefel function, it is advantageous to choose the UNDX after parents distribute on the *hollow* where the global optimal point exists. As shown in Fig.4, the curves of $P_{\mathrm{UNDX}}^{\mathrm{apply}}$ rises around the number of evaluation at which the best individual lies in the *hollow* where the global optimal point exists.

In the Continuous and Mixed Rotated-Rastrigin functions, it is recommended to choose the UNDX from the early search phases. As shown in Fig.4(c), $P_{\mathrm{UNDX}}$ is large from the early search phases.

**[Effects of the new selection scheme]**

As shown in Fig.3 and Fig.5, the EMGG always outperforms the MGG, comparing the UNDX+EMGG with the UNDX+MGG. The EMGG always not only obtains the same solution or a better one but also finds it faster. It is thought that this is because the new selection scheme works successfully.

## 7 Conclusions

In this paper, we proposed a robust real-coded GA named (UX,UNDX)+EMGG. The GA employs both the UNDX and the UX, complementary crossover operators. To choose crossover operator efficiently, (UX,UNDX)+EMGG has a self-adaptive mechanism of crossover probabilities. Through some experiments, we showed the robustness of the proposed method by demonstrating that the proposed method can solve more various functions than a GA using only the UNDX. We also confirmed that the self-adaptive mechanism of crossover probabilities worked successfully by examining the changes of probabilities of the UNDX during the search.

In this paper, we used the UX as a complementary operator to the UNDX for exploration. From this view points, the generalized discrete recombination (GDR), used in ESs [Bäck 96], may be better than the UX since the GDR can produce more kinds of offsprings. We have a plan to try the GDR. Furthermore, we have a plan to apply the proposed method to practical applications such as the lens design problem known as a very difficult real-world problem.

### References


[Bäck 96] Bäck, T.: Evolutionary Algorithms in Theory and Practice, Oxford (1996).

[Davis 89] Davis, L.: Adapting operator probabilities in genetic algorithms, Proc. 3rd ICGA, pp.61-69 (1989).

[Davis 90] Davis, L.: The Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York (1990).

[Eshelman 93] Eshleman, L. J. and Schaffer, J. D.: Real-Coded Genetic Algorithms and Interval-Schemata, FOGA 2, pp.187-202 (1993).

[Eshelman 97] Eshelman, L. J., Mathias, K. E. and Schaffer, J. D.: Crossover Operator Biases: Exploiting the Population Distribution, Proc. 7th ICGA, pp.354-361 (1997).

[Goldberg 89] Goldberg, D. E. : Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wsley Publishing Company Inc. (1989).

[Jonikow 91] Jonikow, C. Z. and Michalewicz, Z.: An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms, Proc. 4th ICGA, pp.31-36 (1991).

[Julstrom 95] Julstrom, B. A.: What Have You Done for Me Lately? Adapting Operator Probabilities in a Steady-State Genetic Algorithm, Proc. 6th ICGA, pp.81-87 (1995).

[Kita 98] Kita, H., Ono, I. and Kobayashi, S. : Theoretical Analysis of the Unimodal Normal Distribution Crossover for Real-coded Genetic Algorithms, Proc. 1998 IEEE ICEC, pp.529-534 (1998).

[Michalewicz 92] Michalewicz, Z. : Genetic Algorithms+Data Structures=Evolution Programs, Springer-Verlag, Berlin (1992).

[Mühlenbein 93] Mühlenbein, H. and Schlierkamp-Voosen, D. : Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization, Evolutionary Computation, Vol.1, pp.25-49 (1993).

[Ono 96] Ono, I., Yamamura, M., Kobayashi, S. : A Genetic Algorithm with Characteristic Preservation for Function Optimization, Proc. IIZUKA '96, pp.511-514 (1996).

[Ono 97] Ono, I. and Kobayashi, S. : A Real-coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover, Proc. 7th ICGA, pp.246-253 (1997).

[Radcliffe 91] Radcliffe, N. J.: Forma Analysis and Random Respectful Reconbination, Proc. 4th ICGA, pp222-229 (1991).

[Salomon 96] Salomon, R. : Performance Degradation of Genetic Algorithms Under Coordinate Rotation, Proc. 5th Annual Conf. on EP, pp.155-161 (1996).

[Schwefel 95] Schwefel, H.-P.: Evolution and Optimum Seeking, John Wiley & Sons, Inc. (1995).

[Satoh 96] Satoh, H., Yamamura, M. and Kobayashi, S. : Minimal Generation Gap Model for GAs Considering Both Exploration and Exploitation, Proc. IIZUKA'96, pp.494-497 (1996).

[Tuson 96] Tuson, A., Ross, P.: Cost Based Operator Rate Adaptation: An Investigation, PPSN IV, pp.461-469 (1996).

[Voigt 95] Voigt, H. -M., Mühlenbein, H. and Gvetkovic, D. : Fuzzy Recombination for the Breeder Genetic Algorithm, Proc. 6th ICGA, pp104-111 (1995).

[Wright 91] Wright, A.: Genetic Algorithms for Real Parameter Optimization, FOGA, pp.205-218 (1991).