# COMPARING VARIOUS MARGINAL PROBABILITY MODELS IN EVOLUTIONARY ALGORITHMS

Petr Pošík

Department of Cybernetics, CTU FEE
Technická 2, 166 27 Prague, Czech Republic
e-mail: posik@labe.felk.cvut.cz, phone: +420-2-24357228

**Abstract**

*Evolutionary algorithms based on probabilistic modeling do not use genetic operators anymore. Instead, they learn a probabilistic model and create new population of potential solutions by sampling from this model. This paper presents some results of an empirical comparison of this type of evolutionary algorithm using various marginal probability models (three types of histograms and gaussian mixture) in continuous domain. We found that the equi-height histogram and max-diff histogram are the winners for the type of evaluation functions considered here, although the mixture of gaussians offers several good features.*

**Keywords:** *Evolutionary algorithms, estimation of distribution algorithms, histograms*

## 1 Introduction

Evolutionary algorithms (EA) are known as widely usable, robust optimization and search tool which found its applications in many areas (science, industry, etc.). They search the space of possible solutions on the basis of perturbations of good solutions, and/or on the basis of combining (usually two) good solutions together. In most cases, they take only one or two individuals to produce new, hopefully better, individuals. And here the following question arises: Couldn't we produce better offsprings if we had more information? We can for example use the Brent's method of numeric optimization: randomly choose two points, sample one additional between them, fit a quadratic function to these three points, and create one offspring as the extreme of the fitted parabola. We can also use a higher number of parents, combine them somehow, and produce arbitrary number of offsprings. All evolutionary algorithms based on creating probabilistic models and sampling from them (estimation of distribution algorithms, EDA's) belong to this category. From this point of view, we can consider the model-and-sample part of EA to be a generalized type of multiparent crossover operator.

The easiest EDA's take into account no dependencies between variables. The joint density function is factorized by a product of univariate independent densities. The Univariate Marginal Distribution Algorithm for continuous domains (UMDA$_C$) [2] performs some statistical tests in order to determine which of density functions fits the variable best. Then, it computes the maximum likelihood estimates of parameters for chosen densities. Another example of EDA using models without dependencies is Stochastic Hill Climbing with Learning by Vectors of Normal Distributions [5]. In this algorithm, the joint density is formed as a product of univariate independent normal densities where the vector of means is updated using a kind of Hebbian learning rule and the vector of variances is adapted by a reduction policy (at each iteration it is modified by a factor that is lower than 1). In [7], an extension of the Baluja's Population Based Incremental Learning algorithm for continous domains (PBIL$_C$) was proposed, however this approach is very similar to an instance of evolutionary strategy (see e.g. [6]). Finally, there are several approaches using histogram models [8]. Gallagher et al. [1] proposed a finite Adaptive Gaussian Mixture model (AMix). This model is able to cover multivariate dependencies between variables and includes an adaptive scheme in which the number of mixture components is not constant during the run of the algorithm.

This article clearly builds on and extends the work of Tsutsui, Pelikan and Goldberg [8]. We have repeated several of their experiments and adopted their experimental methodology. However, we track a bit different statistics and look at the data from a bit different point of view.

Section 2 describes the probabilistic models compared in this paper. The results of all conducted tests (along with description of used reference problems and experimental methodology) are presented in section 3. Section 4 summarizes the results and concludes the paper.

## 2 Used Probabilistic Models

All algorithms described in this article fall into class of UMDA. The individual components of promising solutions are supposed to be statistically independent of each other. In turn, this means that the global distribution of promising solutions in the search space can be modeled by a set of univariate marginal distributions. Then, for the global distribution we may write

$$p(\mathbf{x}) = \prod_{i=1}^{n} p_i(x_i), \tag{1}$$

where the $p(\mathbf{x})$ is the global multivariate density and the $p_i(x_i)$'s are the univariate marginal densities.

It's important to stress out that the independence assumption doesn't hold for the most of real world problems. The basic UMDA (which is the simplest among EDA's) is not able to cover any interactions among variables, independently of what type of marginal distribution model the UMDA uses.

In following subsections, several types of marginal densities are described. We want to evaluate the behaviour of UMDA algorithm when using 4 types of marginal probability models. The first two were considered in the work of Tsutsui et al. [8], namely the *equi-width histogram* and *equi-height histogram*. This work expands the set of models to *max-diff histogram* and *univariate mixture of gaussians*.

## 2.1 Histogram Formalization

Before we describe particular types of histograms, let's formalize the notion of histogram model. From our point of view, the histogram is a parametric model of univariate probability density. The histogram consists of $C$ bins and each $i$-th bin, $i \in 1..C$, is described by three parameters: $b_{i-1}$, $b_i$ are the boundaries of the bin and $p_i$ is the value of probability density function (PDF) if $b_{i-1} < x \leq b_i$. When constructing histogram, we have a set $\{x^j\}_{j=1}^{N}$ of $N$ data points. Let's describe $n_i$ the number of data points which lies between $b_{i-1}$ and $b_i$. Clearly, the values $p_i$ must be proportional to $n_i$. Furthemore, it must be a proper probability density function. If we describe $d_i$ the difference $b_i - b_{i-1}$, we can write:

$$p_i = \alpha n_i = \frac{n_i}{\sum_{i=1}^{C} n_i d_i} \tag{2}$$

### 2.1.1 Equi-Width Histogram

This type of histogram is probably the best known and the most used one, although it exhibits several unpleasant features. The domain of respective variable is divided to $C$ equally large bins. Since the size of each bin can be written as $d_i = d = \frac{b_C - b_0}{C}$, the equation for all the values of PDF reduces to

$$p_i = \frac{n_i}{dN} \tag{3}$$

When constructing the histogram, first determine the boundaries of each bin, count the respective number of data points in each bin, and use the equation 3 to compute the values of PDF for this histogram model.

### 2.1.2 Equi-Height Histogram

This type of histogram was used in the paper of Tsutsui et al. [8] too. Instead of fixing the width of bins, it fixes their heights. Each bin then contains the same number of data points. This means that in areas where the data points are sparse, the bin boundaries are far from each other, while in regions where the data points are dense, the boundaries are pretty close to each other. Since the probability of each bin is constant, we can write

$$p = p_i = \frac{1}{b_C - b_0} \tag{4}$$

In real cases we could hardly expect that the number of data points can be divided by the number of bins without any reminder. One simple method how to construct this type of histogram is to compute the number of data points which should be contained in each bin. Then we put the boundary of consecutive bins to the half of the distance of a related pair of neighbouring data points.

### 2.1.3 Max-Diff Histogram

This type of histogram is reported to have several good features (small errors, short construction time), see [4]. Max-diff histogram doesn't put any 'equi-something' constraint on the model. Instead, it is defined by a very simple rule. If we construct max-diff histogram with $C$ bins, we need to specify $C - 1$ boundaries (the first and the last boundaries are equal to minimum and maximum of the respective variable domain). If we compute all the distances between all two neighbouring data points, we can select the $C - 1$ largest of them and the boundaries put to the half of the respective intervals.

This procedure has a natural explanation. We try to find consistent clusters in data, and all data points in such a cluster are put to the same bin. It is just an approximation and we can argue what results this procedure gives when the data doesn't show any structure, but it turned out to be very precise and useful in many situations [4].

| Function | Domain | Number of bins (components) |
|---|---|---|
| $F_{TwoPeaks} = 5n - \sum_{i=1}^{n} f_i$ | $\langle 0, 12 \rangle^n, n = 20$ | $Low = 60(3), High = 120(6)$ |
| $F_{Griewangk} = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} + \prod_{i=1}^{n} \cos \frac{x_i}{\sqrt{i}}$ | $\langle -5, 5 \rangle^n, n = 10$ | $Low = 50(3), High = 100(6)$ |

**Table 1:** Test functions and related parameters. The $f_i$ function can be described as polyline going through points $(0, 0)$, $(1, 5)$, $(2, 0)$, $(7, 4)$, $(12, 0)$.

### 2.2 Univariate Mixture of Gaussians

Univariate mixture of gaussians (MOG) is the last marginal probability model compared in this part. It is included in this comparison for several reasons. We can consider all the histogram models to be certain kind of mixture models too — mixture of uniforms. From this point of view, the MOG can give interesting results when compared to histograms. Since the mixture of gaussians can describe the data with more 'fidelity', we can drastically decrease the number of components (when compared to the number of bins needed by histograms). We have also to say that MOG requires much more effort to construct because it must be trained by an iterative learning scheme — the Expectation Maximization algorithm (see e.g. [3]). Figure 1 shows examples of probability densities presented by means of the above mentioned models.

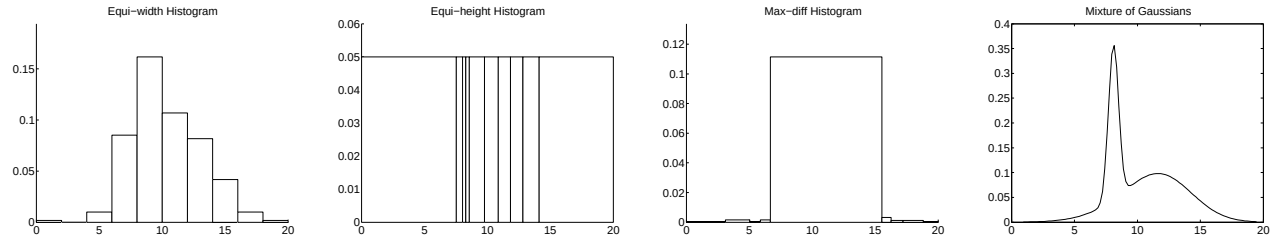


**Figure 1:** Equi-width, equi-height, and max-diff histograms, and mixture of gaussians

### 2.3 Sampling

We use the sampling method which was described in the Tsutsui paper [8] as Baker's stochastic universal sampling (SUS). It uses the same principles as the (in EC community perhaps more known) reminder stochastic sampling (RSS).

First, we compute the expected number of data points which should be generated by each bin. It is computed according the bin (or component) probabilities given by equation 2. Then we generate the integer part of expected number by sampling from uniform (in case of histograms) or normal (in case of mixture of gaussians) distribution with proper parameters. The rest of points is then sampled by stochastically selecting bins (or components) with probability given by the fractional parts of expected numbers for each bin (component).

## 3 Empirical Comparison

### 3.1 Evolutionary model

Let the population size be $N$. In each iteration we make the following: (1) model the population with histograms (or mixtures of gaussians), (2) sample $N$ new data points from the model, (3) join the old and new population to get data pool of size $2N$, and (4) use truncation selection to select the better half of data points.

### 3.2 Test Suite

For empirical comparison we chose only two of the four functions used in the work of Tsutsui [8], namely the Two Peaks function and the Griewangk function. It is very small test set but this comparison is aimed on the suitability and behaviour of individual types of probabilistic models for modeling of marginal probabilities rather than comparing the efficiency of the resulting algorithms on various problems. The functions are defined in table 1. The Two Peaks function is separable and there exists very easy algorithm which solves separable problems very efficiently. It is so called *line search* algorithm (see section 3.3). The form of multidimensional Two Peaks function is shown in figure 2. The Griewangk function represents non-separable function with interactions, and is a bit harder to optimize.

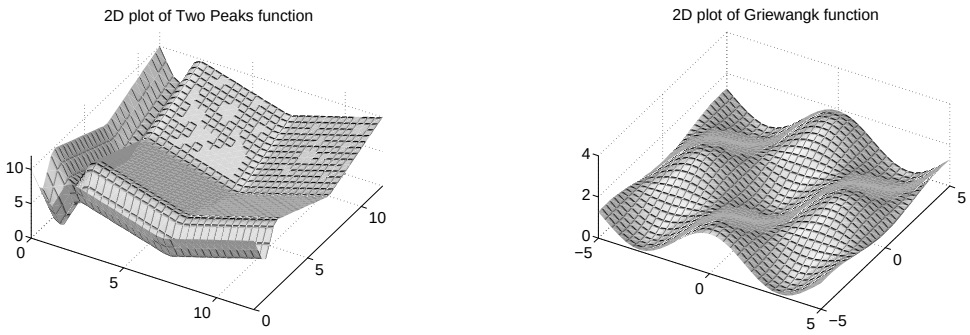


**Figure 2:** Fitness landscape of 2D Two Peaks function and 2D Griewangk Function.

### 3.3 Experimental Setup

For both test problems, Two Peaks and Griewangk, we compared four UMDA's — each using different model of marginal probability. The factor *model* has the following levels: (1) *Equi-Width histogram* (HEW, in short), (2) *Equi-Height histogram* (HEH), (3) *Max-Diff histogram* (HMD), and (4) *mixture of gaussians* (MOG).

For each of the models, we tried to vary also the number of bins (components). Factor *Number of Bins* has two levels, *High* and *Low*. The exact number of bins varies and it was set in accordance with Tsutsui et al. [8]. The actual value of level *High* for histogram models and Two Peaks function is 120. The value of level *Low* is one half of level *High*, i.e. 60 for histograms and Two Peaks function. For the Griewangk function, the *High* number of bins presents 100, and the *Low* number of bins presents 50. The mixture models need less components than histograms. The levels were set to 6 and 3 for the *High* and *Low* number of components, respectively, and were determined empirically. The initial values for the centers of individual components of mixture were constructed using the k-means clustering method.

Furthermore, we varied the *Population size* in our experiments. Levels of this factor include populations of sizes 200, 400, 600, and 800 data points.

In order to give some indication of how complex the test problems are, we tried to optimize them also with so called *line search* (LS) method (see [9]). This method is rather a heuristic which is pretty effective for separable problems and thus it can give you some insight in the complexity of the evaluation function. Its principle is very easy: start from randomly generated data point, randomly select a dimension, discretize and enumerate it, move to the best data point in the unidimensional enumerated space, and repeat for further dimension. We have set the discretization length equal to 0.1. If the LS gets stuck (no further improvement possible), it is restarted from another randomly generated point.

### 3.4 Monitored statistics

For each of possible factor combination we run the algorithm 20 times. Each run was allowed to continue until the maximum number of 50000 evaluations was reached. We say, that the algorithm found the global optimum if for each variable $x_i$ the following relation holds: $\left|x_i^{best} - x_i^{opt})\right| < 0.1$ (if the difference of the best found solution $x^{best}$ from the optimal solution $x^{opt}$ is lower then 0.1 in each of coordinates).

In all experiments, we track three statistics:

1. The number of runs in which the algorithm succeeded in finding the global optimum (*NoFoundOpts*).
2. The average number of evaluations needed to find the global optimum computed from those runs in which the algorithm really found the optimum (*AveNoEvals*).
3. The average fitness of the best solution the algorithm was able to find in all 20 runs (*AveBest*).

### 3.5 Results and Discussion

The results of our experiments are presented in table 2. The results of the line search heuristic indicate that the Two Peaks function could be solved much more effectively by other types of algorithms. Nevertheless, EDA's should be able to solve this easy type of problem. The Griewangk function is more complex and the line search heuristic was not able to solve it in all experiments.

Let's first consider the results for the HEW model. In accordance with the results of Tsutsui [8], we confirm that the HEW model is the least flexible one. Furthermore, it can be characterized by one important feature: if the population is initialized in such a way that the resulting probabilistic density will be equal to zero somewhere in the search space, the algorithm has no possibilities to overcome it and to start searching in these 'zero areas'.

| Function | Model | Number of Bins (Components) | | | | | | | | Statistics |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Low | | | | High | | | | |
| | | Population Size | | | | | | | | |
| | | 200 | 400 | 600 | 800 | 200 | 400 | 600 | 800 | |
| Two Peaks | LS | 20 | | | | | | | | NoFoundOpts |
| | | 2421 | | | | | | | | AveNoEvals |
| | | 0,00000 | | | | | | | | AveBest |
| | HEW | 0 | 1 | 0 | 1 | 1 | 17 | 19 | 20 | NoFoundOpts |
| | | | 13200 | | 47200 | 5600 | 10306 | 15189 | 19600 | AveNoEvals |
| | | 5,00560 | 4,82110 | 5,05280 | 4,95510 | 3,48040 | 2,51180 | 2,50030 | 2,52340 | AveBest |
| | HEH | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | NoFoundOpts |
| | | 6530 | 11080 | 16050 | 20760 | 7720 | 10620 | 15570 | 20400 | AveNoEvals |
| | | 0,10710 | 0,02070 | 0,01040 | 0,07490 | 0,03740 | 0,00690 | 0,00430 | 0,03520 | AveBest |
| | HMD | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | NoFoundOpts |
| | | 6270 | 14920 | 27960 | 47080 | 6770 | 10980 | 17490 | 25280 | AveNoEvals |
| | | 0,00003 | 0,00008 | 0,04990 | 2,20380 | 0,00260 | 0,00001 | 0,00230 | 0,05740 | AveBest |
| | MOG | 7 | 20 | 20 | 19 | 16 | 20 | 20 | 20 | NoFoundOpts |
| | | 6571 | 11860 | 17130 | 23032 | 5613 | 10480 | 15780 | 20720 | AveNoEvals |
| | | 0,87390 | 0,00011 | 0,00930 | 0,13970 | 0,19650 | 0,00008 | 0,00660 | 0,06370 | AveBest |
| Griewangk | LS | 18 | | | | | | | | NoFoundOpts |
| | | 14056 | | | | | | | | AveNoEvals |
| | | 0,00250 | | | | | | | | AveBest |
| | HEW | 13 | 17 | 16 | 14 | 1 | 15 | 18 | 19 | NoFoundOpts |
| | | 15954 | 20353 | 25763 | 27886 | 5800 | 12320 | 18867 | 24926 | AveNoEvals |
| | | 0,00370 | 0,00210 | 0,00230 | 0,00320 | 0,00500 | 0,00081 | 0,00095 | 0,00083 | AveBest |
| | HEH | 18 | 18 | 18 | 20 | 16 | 18 | 20 | 20 | NoFoundOpts |
| | | 6667 | 12867 | 18967 | 25000 | 6650 | 12711 | 18270 | 23920 | AveNoEvals |
| | | 0,00074 | 0,00074 | 0,00074 | 0,00000 | 0,00150 | 0,00074 | 0,00000 | 0,00000 | AveBest |
| | HMD | 17 | 17 | 18 | 16 | 18 | 17 | 19 | 20 | NoFoundOpts |
| | | 6482 | 12376 | 19200 | 25850 | 6456 | 12235 | 18221 | 23720 | AveNoEvals |
| | | 0,00110 | 0,00110 | 0,00074 | 0,00140 | 0,00074 | 0,00110 | 0,00037 | 0,00000 | AveBest |
| | MOG | 15 | 15 | 18 | 19 | 19 | 19 | 17 | 18 | NoFoundOpts |
| | | 5693 | 10613 | 16100 | 21726 | 5894 | 12084 | 17682 | 23644 | AveNoEvals |
| | | 0,00190 | 0,00180 | 0,00074 | 0,00037 | 0,00037 | 0,00037 | 0,00110 | 0,00074 | AveBest |

**Table 2:** Results of carried out experiments

This is also documented by the very weak results obtained by the UMDA with HEW model. If the 'resolution' (the number of bins) of the histogram is lower than the precision required, the algorithm becomes very unprecise. If the 'resolution' is sufficient, the efficiency of the algorithm is very dependent on the population size. This can be seen especially in the case of the Two Peaks evaluation function.

The other three models, HEH, HMD, and MOG, seem to be much more robust with respect to the parameters. All of them also don't exhibit the unpleasant feature of HEW model ('zero areas') described in previous paragraph. The models are constructed in such a way that they do not allow for a zero values of PDF.

HEH and HMD models showed very similar behaviour, although on these two test functions the HEH model seems to work better. However, the differences are very insignificant. Both are very successful in the number of runs they succeed to find the global optima. The average number of evaluations they need to find the optima is comparable for both models. However, looking at the results table, we can see that the achieved average fitness scores of best individuals are still highly dependent on the size of population.

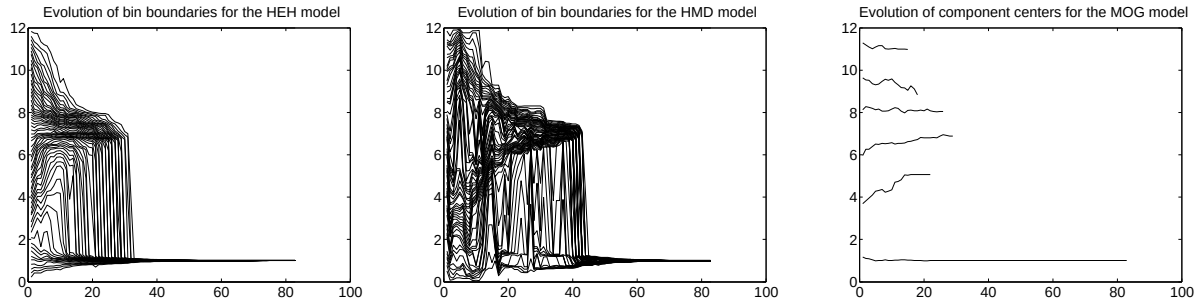


**Figure 3:** Two Peaks function — Evolution of bin boundaries for HEH and HMD models and evolution of component centers for MOG model.

In figure 3, the typical 'tracks' of evolution of bin boundaries (for histogram models) and component centers (for mixture of gaussians model) are presented. Similar results can be obtained for the Griewangk function. These 'typical' representats show cases in which all the models converged to the right solution (for this particuar dimension) in both cases of evaluation function. However, to find the global optimum the algorithm had to converge to the right solution in all the dimensions and that's often not the case for the MOG model.

For the MOG model, we can see that several components of the mixture simply stopped evolving. The reason for this is very simple — their mixture coefficients dropped to zero and the components simply vanished. Generally, this should be a good feature, however, we have experienced several runs in which a particular component vanished before it could take over the whole population although it discovered the right solution (for particular dimension). This can be thought of as an analogy of the 'premature convergence' in genetic algorithms.

We have to say a few words on the running times of the algorithms. The evolution of MOG model took approximately two times more time than the evolution of histogram models. However, this needn't be due to the complexity of the models, rather it can be caused by the complexity of sampling (a normal random generator is much more slower then a uniform random generator). This feature deserves further investigation.

# 4 Summary and Conclusions

In this article we have described some results of a small empirical comparison of the UMDA type of estimation of distribution algorithm using several types of marginal probability models. We have viewed the algorithms also from the robustness point of view, i.e. from the view of their (non-)dependancy on the algorithm parameters setting. We used only two test functions that were not too difficult. We do not expect that the efficiency of this type of EDA (with marginal probability modeling) should be very high for more complex functions. To solve them reliably, we will have to use more general probabilistic models.

The UMDA with the equi-width histogram turned out to be very unreliable and non-robust algorithm. UMDA with the other two types of histogram models, equi-height and max-diff, seem to be effective and robust for separable functions, and for functions with small amount of interactions between variables. In the latter case (for a Griewangk function), they outperformed the simple line search heuristic.

The mixture of gaussians model is a representant of qualitativelly different type of model. Although it exhibits a bit worse behaviour than the two above mentioned types of histograms, it is fair to say that it was used with a considerably smaller number of components and that it offers a few additional advantages. E.g., with almost no effort, it can be easilly extended from a set of independent marginal mixtures of gaussians to a general multidimensional mixture of gaussians which will be able to cover some interactions among variables. This is also a topic for further research.

# Acknowledgments

The work on this project was funded by Czech Ministry of Education No. MSM 212300014 entitled "Research in Information Technologies and Telecommunications", and was supervised by Doc. Ing. Jiří Lažanský, CSc. The author also would like to thank to Marcus Gallagher for many valuable comments.

# References

[1] Gallagher, M.R., Frean, M., Downs, T. *Real-Valued Evolutionary Optimization Using a Flexible Probability Density Estimator.* In Proc. GECCO-1999, p. 840-864. Morgan Kaufmann, 1999.

[2] Larranaga, P. et al. *A Review of the Cooperation Between Evolutionary Computation and Probabilistic Graphical Models.* In Second Symp. on AI, Adaptive Systems. CIMAF 99, p. 314-324. La Habana, 1999.

[3] Moerland, P. *Mixture Models for Unsupervised and Supervised Learning.* PhD Thesis, Computer Science Department, Swiss Federal Institute of Technology, Lausanne, 2000.

[4] Poosala, V. et al. *Improved histograms for selectivity estimation of range predicates.* In Proc. 1996 ACM SIGMOD Int. Conf. Management of Data, p. 294–305. ACM Press, 1996.

[5] Rudlof, S., Koeppen, M. *Stochastic Hill Climbing by Vectors of Normal Distributions.* In Proc. First Online Workshop on Soft Computing (WSC1). Nagoya, Japan, 1996.

[6] Schwefel, H.-P. *Evolution and Optimum Seeking.* Wiley, New York, 1995.

[7] Sebag, M., Ducoulombier, A. *Extending Population Based Incremental Learning to Continuous Search Spaces.* In Parallel Problem Solving From Nature - PPSN V, pages 418-427. Springer Verlag, Berlin, 1998.

[8] Tsutsui, S., Pelikan, M., Goldberg, D.E. *Evolutionary Algorithm Using Marginal Histogram Models in Continuous Domain.* IlliGAL Report No. 2001019. University of Illinois at Urbana-Champaign. 2001.

[9] Whitley, D., Mathias, K., Rana, S., Dzubera, J. *Evaluating Evolutionary Algorithms.* Artificial Intelligence Volume 85, p. 245-276, 1996.