# Numerical Optimization with Real–Valued Estimation–of–Distribution Algorithms

Peter A.N. Bosman[1] and Dirk Thierens[2]

[1] Centre for Mathematics and Computer Science (CWI), P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, Peter.Bosman@cwi.nl
[2] Institute of Information and Computing Sciences, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, The Netherlands, Dirk.Thierens@cs.uu.nl

**Summary.** In this chapter we focus on the design of real–valued EDAs for the task of numerical optimization. Here, both the problem variables as well as their encoding are real values. Concordantly, the type of probability distribution to be used for estimation and sampling in the EDA is continuous. In this chapter we indicate the main challenges in this area. Furthermore, we review the existing literature to indicate the current EDA practice for real–valued numerical optimization. Based on observations originating from this existing research and on existing work in the literature regarding dynamics of continuous EDAs, we draw some conclusions about the feasibility of existing EDA approaches. Also we provide an explanation for some observed deficiencies of continuous EDAs as well as possible improvements and future directions of research in this branch of EDAs.

## 1 Introduction

One of the main impulses that triggered the emergence of the EDA field has been the research into the dynamics of discrete GAs. EDAs provide an elegant way of overcoming some of the most important shortcomings of classical GAs. In general, for optimization to proceed efficiently, the induced search bias of the optimizer must fit the structure of the problem under study. From this point of view, the success of the EDA approach in the discrete domain is rather intuitive. Because probability distributions are used to explicitly guide the search in EDAs, the probability distribution itself is an explicit model for the inductive search bias. Estimating the probability distribution from data corresponds to tuning the model for the inductive search bias. Because a lot is known about how probability distributions can be estimated from data (Buntine, 1994; Lauritzen, 1996) the flexibility of the inductive search bias of EDAs is potentially large. In addition, the tuning of the inductive search bias in this fashion also has a rigorous foundation in the form of the well–established field of probability theory.

Estimating probability distributions, especially in the discrete case, is very closely related to the modeling of dependencies between random variables, specific settings for these random variables, or both. Such dependencies are clearly embodied in the use of factorized probability distributions (Edwards, 1995; Lauritzen, 1996; Friedman and Goldszmidt, 1996). As a result, the processing of these dependencies by the EDA in the discrete domain is also explicitly ensured under the assumption of a proper estimation of the (factorized) probability distribution. Dependencies in the discrete domain match exactly with the notion of linkage information or, synonymously, the structure of the problem. Hence, the competent estimation of probability distributions in EDAs allows these algorithms to adequately perform linkage learning and to ensure that such necessary conditions as proper building block mixing (Thierens and Goldberg, 1993) meets with the positive effects of the schema theorem (Holland, 1975) and the building block hypothesis (Goldberg, 1989; Goldberg, 2002a). Although clearly more computational effort is required to tune the inductive search bias by estimating the probability distribution, the payoff has been shown to be worth the computational investment (e.g. polynomial scale–up behavior instead of exponential scale–up behavior) (Pelikan, Goldberg and Cantú-Paz, 1999; Etxeberria and Larrañaga, 1999; Pelikan and Goldberg, 2001; Pelikan and Goldberg, 2003).

The use of factorized probability distributions in an EDA works very well in discrete domain. In general, the EDA approach can be motivated perfectly from the requirement of fitting the inductive search bias to the structure of the optimization problem at hand. It is now interesting to investigate how the approach carries over to the continuous domain. The main important questions to answer are 1) what does the model (i.e. probability distribution) for the inductive search bias look like in the continuous domain and 2) can this model be adapted properly to fit the structure of the problem?

The remainder of this chapter is organized as follows. First, in Section 2 we discuss real–valued numerical optimization problems and point out which sources of problem difficulty typically exist. Next, in Section 3 we consider the EDA approach in the continuous domain and show that in theory real–valued EDAs can be very efficient numerical optimizers. In Section 4 we then present a literature overview that describes the current state of the continuous subfield of EDAs. Subsequently, in Section 5 we reflect on the dynamics of the currently available real–valued EDAs in more depth. We also discuss whether and how improvements can be made over current approaches. Finally, we summarize and conclude this chapter in Section 6.

## 2 Problem difficulty

The degree of difficulty of a numerical optimization problem is in a certain sense unbounded. Because the search space is by definition infinitely large, the number of substructures, i.e. the number of local optima and areas with var-

ious types of dependency, can also be infinite. The structure of a continuous problem can most generally be seen as the composition of its contour lines. The shape of these contour lines is unbounded and there are infinitely many of them. Hence, anything is possible. Fortunately, we are typically not interested in all possible problems. We assume that there is some form of "logical" structure, similar to assuming in the binary domain that the function is not a needle–in–a–haystack function. Hence, the problem structure we typically expect to be asked to tackle is a composition of simpler structures such as local optima and some form of dependency between the problem variables.

Still, even assuming "simpler" substructures are present that we can tailor our EDA to, it should be noted that in the continuous case the actual difficulty of the problem can still be arbitrary in the sense that the number of substructures that can be present in the problem is arbitrarily large, even if the problem has only one variable. This arbitrary problem difficulty in terms of presence of multiple substructures throughout the search space will play important role in evaluating the sensibility of the EDA approach as a whole as well as existing actual EDA instances in the continuous domain.

At the most general level the (sub)structures that are typically to be expected in the continuous domain are similar to the ones in the discrete domain. The two basic substructures are multi–modality and dependency. The main difference with the discrete case is that in the continuous case these structures can also be viewed upon from a different perspective, namely one that entails slopes and peaks (i.e. hills or valleys). Since nothing is said about the number of slopes and peaks or their actual configuration with respect to orientation and location, this essentially allows for the construction of the same search spaces with local optima (i.e. multimodality) and dependencies.

Various sources of problem difficulty in continuous spaces are often explained most effectively by the design of an actual problem and its visualization. For the purpose of illustration of problem difficulty and because the discussion of the results obtained with various continuous EDAs is often related to existing benchmark problems, we now describe a set of five optimization problems. These problems represent a variety of difficulties in numerical optimization. The definitions of the numerical optimization problems are given in Table 1. For an intuitive impression of the characteristics of the problems, two–dimensional surface plots (with the exception of the readily–imaginable sphere function) are provided in Figure 1.

### Sphere

The sphere function is probably the most standard unimodal benchmark problem for numerical optimization. It involves the minimization of a single hyperparabola. The function is unimodal, has no dependencies and has smooth gradient properties. The sphere function is often used to study convergence. The minimum value for any dimensionality is 0 which is obtained if all $y_i$ are set to a value of 0.

| Name | Definition | Range |
|---|---|---|
| Sphere | **Minimize** $\sum_{i=0}^{l-1} y_i^2$ | $y_i \in [-5, 5]$ $(0 \leq i < l)$ |
| Griewank | **Minimize** $\frac{1}{4000} \sum_{i=0}^{l-1} (y_i - 100)^2 - \prod_{i=0}^{l-1} \cos\left(\frac{y_i - 100}{\sqrt{i+1}}\right) + 1$ | $y_i \in [-600, 600]$ $(0 \leq i < l)$ |
| Michalewicz | **Minimize** $-\sum_{i=0}^{l-1} \sin(y_i) \sin^{20}\left(\frac{(i+1) y_i^2}{\pi}\right)$ | $y_i \in [0, \pi]$ $(0 \leq i < l)$ |
| Rosenbrock | **Minimize** $\sum_{i=0}^{l-2} 100(y_{i+1} - y_i^2)^2 + (1 - y_i)^2$ | $y_i \in [-5.12, 5.12]$ $(0 \leq i < l)$ |
| Summation Cancellation | **Maximize** $100/(10^{-5} + \sum_{i=0}^{l-1} \lvert\gamma_i\rvert)$ **where** $\gamma_0 = y_0, \gamma_i = y_i + \gamma_{i-1}$ | $y_i \in [-3, 3]$ $(0 \leq i < l)$ |

**Table 1.** Numerical optimization test problems.

### Griewank

Griewank's function is a function with many local optima. Basically, it is a parabola superimposed with a sine function to obtain many local optima. As a result, if large steps are taken in Griewank's function, the so observed coarse–grained gradient information will quickly lead to a region close to the minimum of the parabola. However, if only small steps are taken, the many local optima will prevent efficient optimization of this function, even when a random restart strategy is used. Furthermore, for a fixed precision, Griewank's function becomes easier to optimize as $l$ increases if large steps are taken. The minimum value for Griewank's function for any dimensionality is 0, which is obtained if all $y_i$ are set to a value of 100.

### Michalewicz

Michalewicz's function is also a function with many local optima, albeit to a lesser degree than Griewank's function. An important difference is that Michalewicz's function has many long channels along which the minimum value throughout the channel is the same. The gradient information in such a channel therefore does not lead to the better local optima which are found at the intersections of the channels. Proper optimization of Michalewicz's function is therefore only possible if the channels of equal optimization value are explored or covered fully to find the intersections. The minimum value for Michalewicz's function depends on its dimensionality. A description of its solutions at which the minimum value is obtained for any dimensionality has not been reported in the literature.

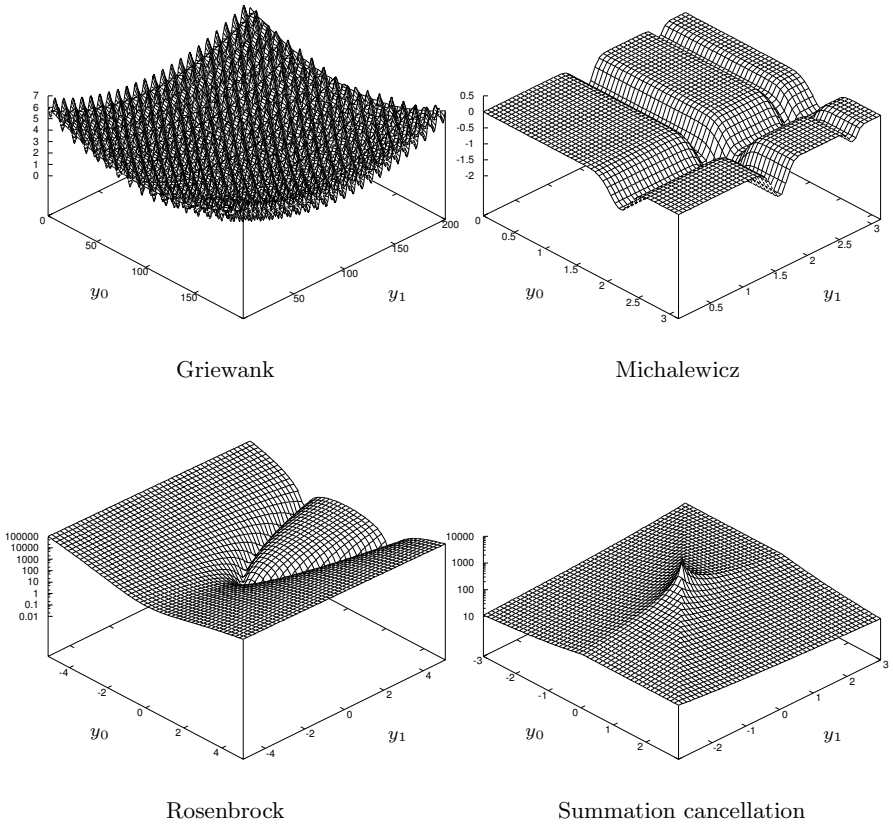


**Fig. 1.** Two–dimensional surface plots for four numerical optimization problems. The ranges for Griewank's function were zoomed to get a better indication of the many local optima. Rosenbrock's function and the summation cancellation function are shown on a logarithmic scale for a better impression of their problem features. Note that the summit of the peak for summation cancellation is actually $10^7$, but the drawing resolution prohibits accurate visualization thereof.

### Rosenbrock

Rosenbrock's function is highly non–linear. It has a curved valley along which the quality of the solutions is much better than in its close neighborhood. Furthermore, this valley has a unique minimum of 0 itself for any dimensionality of Rosenbrock's function, which is obtained if all $y_i$ are set to a value of 1. Rosenbrock's function has proved to be a real challenge for any numerical optimization algorithm. The gradient along the bottom of the non–linear valley is very slight. Any gradient approach is therefore doomed to follow the long road along the bottom of the valley, unless a starting point is provided

in the vicinity of the optimum. Furthermore, since the valley is non–linear, simple gradient based approaches will oscillate from one side of the valley to the other, which does not result in efficient gradient exploitation. For an IDEA, capturing the valley in a probabilistic model is difficult, even if all of the points within the valley are known. The reason for this is that the valley is non–linear in the coding space.

**Summation cancellation**

The summation cancellation problem was proposed by Baluja and Caruana (1995). This optimization problem has multivariate linear interactions between the problem variables. This should allow algorithms that are capable of modeling linear dependencies to outperform algorithms that are not capable of doing so. Furthermore, the degree of multivariate interaction is as large as possible since each $\gamma_i$ in the problem definition is defined in terms of all $y_j$ with $j < i$. Finally, the optimum is located at a very sharp peak, which implies that the optimization algorithm needs to have a large precision and needs to be able to prevent premature convergence in order to reach the global optimum. The minimum value for this function for any dimensionality is $10^7$, which is obtained if all $y_i$ are set to a value of 0.

## 3 Optimal EDAs

The definition of an optimal algorithm is subject to a certain viewpoint. In search and optimization, a common definition of optimality is that the algorithm is capable of finding an optimal solution (Russel and Norvig, 2003). A more refined definition of optimality in the context of a specific class of optimization problem is that for any problem instance there exists no algorithm that is able to find the optimal solution faster than the optimal algorithm. The EDA method is driven mainly by the estimation of a probability distribution. The estimated probability distribution is meant to approximate the true probability distribution. The true probability distribution is the one that describes perfectly the set of the selected solutions in the case of an infinitely large population size. If the method of selection is Boltzmann selection, it can be proved that the resulting EDA using the true probability distribution is optimal in the sense that it will indeed converge to the optimal solution (Mühlenbein and Höns, 2005). Hence, we can call an EDA optimal if the estimated probability distribution equals the true probability distribution.

Drawing samples from the Boltzmann distribution requires exponential computation time and hence, the optimal Boltzmann EDA is not a practical search algorithm. In practice, selection methods such as tournament selection and truncation selection are commonly used. Without loss of generality, assume that the goal of the EDA is to minimize fitness. Now, the optimal probability distribution associated with truncation selection is a distribution

that is uniform over all solutions $\boldsymbol{y}$ that have a fitness $\mathfrak{F}(\boldsymbol{y})$ that is at most the value of the fitness of the worst selected solution. This essentially corresponds to pruning the search space to only those regions that are at least as interesting as the currently available worst selected solution. The probability distribution can be imagined as essentially maintaining exactly all solutions that have a fitness of at most the value of the fitness of the worst selected solution. Hence, sampling from this probability distribution entails nothing more than just returning a single solution from the (infinitely large) set of maintained solutions.

Now observe the series of the fitness values of the worst selected solution in subsequent generations. Using truncation selection this series is monotonously decreasing. Moreover, the series becomes strictly monotonously decreasing if in generation $t$ all solutions with a fitness at least equal to the worst selected fitness in generation $t-1$ are pruned from the selected solutions. To prevent obtaining an empty selection set, this additional discarding should not be done for the solution(s) with the best fitness. Using this slight variation to the truncation selection scheme the EDA approach with the true probability distribution will clearly converge to the optimum with probability 1 because optimal solutions are never discarded from the probability distribution and the worst selected fitness values strictly decreases every generation.

The size of the search space that is still to be regarded, is encoded in the probability distribution. Using truncation selection and an infinitely large population, $100(1-\tau)\%$ of all solutions that make up this remaining search space are discarded. Hence, the size of the search space still to be explored is reduced exponentially as a function of the generations passed. In practice however, we don't have an infinite population size. Still the use of the optimal probability distribution is likely to be extremely efficient in terms of the number of evaluations actually required to find a solution of a certain quality. To illustrate the potential effectiveness of EDAs in continuous spaces, Figure 2 shows the convergence behavior on three of all optimization problems for a dimensionality of $l = 5$ if the optimal probability distribution is known and used in the EDA. The optimal probability distribution is hard to formalize analytically, hence the method used uses rejection sampling. In other words, the method generates random solutions and only accepts those that have a fitness value smaller or equal to the worst selected solution. For Michalewicz' function and the summation cancellation function we plotted the difference with the optimal fitness value to be able to plot their convergence graphs as a minimization procedure that searches for the minimum value of 0. From the results in Figure 2 we indeed find extremely efficient optimization behavior. The fitness improves exponentially fast with the number of evaluations required. Hence, only extremely few evaluations are actually needed to obtain close–to–optimal results, regardless of the actual optimization problem. Concluding, in theory, as a result of the solid background, EDAs for continuous domains work well also next to EDAs for discrete domains as long as the true probability distribution can be closely approximated.

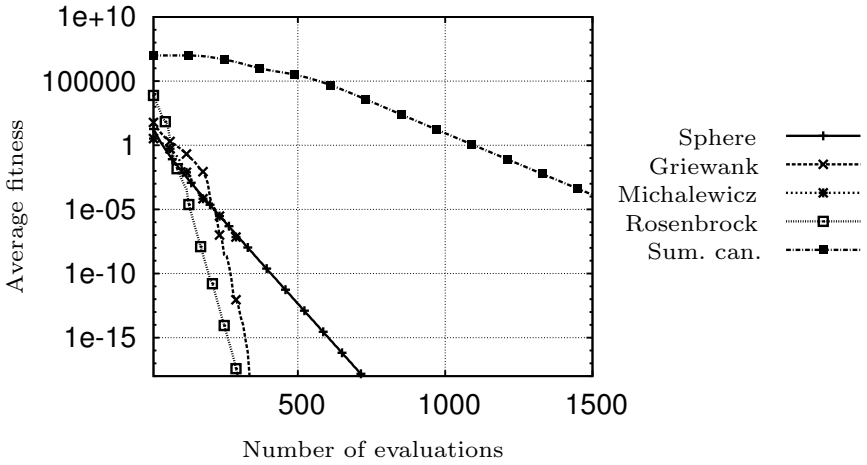


**Fig. 2.** Average convergence behavior (100 runs) of the EDA with true density on a selection of problems and a dimensionality of $l = 5$.

In practice we in general do not have access to the optimal probability distribution. Moreover, in the general sense the optimal probability distribution is arbitrarily complex as the problem itself can be arbitrarily complicated (see Section 2). This makes that sampling from the true probability distribution can take up an arbitrary long time. Hence, to obtain an actual EDA to work with, we must approximate the probability distribution using practical techniques. In the continuous case, the most common approaches are the normal probability density function (pdf) or combinations thereof. It is not surprising that the first EDAs in continuous spaces were based exactly on these common parametric pdfs. The most important question is of course how the extremely efficient optimization behavior of the EDA using the true probability distribution changes in the continuous domain if we use approximated probability distributions instead.

## 4 An overview of existing EDAs

In this section we provide a literature survey of EDAs for continuous domains. In Section 4.1 we consider factorized probability distributions. In Section 4.2 we consider mixtures of factorized probability distributions. The majority of the literature concerning EDAs is based on these two types of probability distribution. We end with Section 4.3 where we consider other classes of probability distribution.

### 4.1 Factorized probability distributions

Similar to the discrete case, estimating a factorization, typically a Bayesian one as introduced in the previous chapter, makes sense since at the top level it assists in finding dependencies between the variables (i.e. the use of a joint distribution versus the use of separate marginal distributions). The same greedy arc–addition algorithm and scoring metrics such as BIC metric can be used as in the discrete case. Note that in the continuous domain the nature or shape of the dependency depends on the actual pdf that is factorized.

#### Normal pdf

The first real–valued EDAs used maximum–likelihood estimations of the normal pdf. The use of the normal pdf was a logical first choice since this pdf is widely used, relatively simple and unimodal.

*Definition*

The normal pdf $P^{\mathcal{N}}_{(\boldsymbol{\mu_i},\boldsymbol{\Sigma^i})}(Y_{\boldsymbol{i}})$ is parameterized by a vector $\boldsymbol{\mu_i}$ of means and a symmetric covariance matrix $\boldsymbol{\Sigma^i}$ and is defined by

$$P^{\mathcal{N}}_{(\boldsymbol{\mu_i},\boldsymbol{\Sigma^i})}(Y_{\boldsymbol{i}})(\boldsymbol{y}) = \frac{(2\pi)^{-\frac{|\boldsymbol{i}|}{2}}}{(\det\ \boldsymbol{\Sigma^i})^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu_i})^T(\boldsymbol{\Sigma^i})^{-1}(\boldsymbol{y}-\boldsymbol{\mu_i})} \tag{1}$$

*Parameter estimation*

A maximum likelihood estimation for the normal pdf is obtained from a vector $\boldsymbol{\mathcal{S}}$ of samples if the parameters are estimated by the sample average and the sample covariance matrix (Anderson, 1958; Tatsuoka, 1971):

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{i}} = \frac{1}{|\boldsymbol{\mathcal{S}}|}\sum_{j=0}^{|\boldsymbol{\mathcal{S}}|-1}(\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{i}}, \qquad \hat{\boldsymbol{\Sigma}}^{\boldsymbol{i}} = \frac{1}{|\boldsymbol{\mathcal{S}}|}\sum_{j=0}^{|\boldsymbol{\mathcal{S}}|-1}((\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{i}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{i}})((\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{i}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{i}})^T \tag{2}$$

To estimate the conditional pdfs $P^{\mathcal{N}}(Y_i|Y_{\boldsymbol{\pi}_i})$ required when constructing Bayesian factorizations, let $\boldsymbol{W^j}$ be the inverse of the symmetric covariance matrix, that is $\boldsymbol{W^j} = (\boldsymbol{\Sigma^j})^{-1}$. Matrix $\boldsymbol{W^j}$ is commonly called the *precision matrix*. It can be shown that for a maximum likelihood estimate of $P^{\mathcal{N}}(Y_i|Y_{\boldsymbol{\pi}_i})$ the maximum–likelihood estimations in equation 2 can be used (Bosman and Thierens, 2000b):

$$\hat{P}^{\mathcal{N}}(Y_i|Y_{\boldsymbol{\pi}_i})(y_{(i,\boldsymbol{\pi}_i)}) = \frac{1}{(\breve{\sigma}_i\sqrt{2\pi})} e^{\frac{-(y_i-\breve{\mu}_i)^2}{2\breve{\sigma}_i^2}} \tag{3}$$

$$\text{where } \begin{cases} \breve{\sigma}_i = \frac{1}{\sqrt{\hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)}}} \\ \breve{\mu}_i = \frac{\hat{\boldsymbol{\mu}}_i\hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)} - \sum_{j=0}^{|\boldsymbol{\pi}_i|-1}(y_{(\boldsymbol{\pi}_i)_j} - \hat{\boldsymbol{\mu}}_{(\boldsymbol{\pi}_i)_j})\hat{\boldsymbol{W}}_{(j+1)0}^{(i,\boldsymbol{\pi}_i)}}{\hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)}} \end{cases}$$

*Properties*

The number of parameters to be estimated equals $\frac{1}{2}|\boldsymbol{i}|^2 + \frac{3}{2}|\boldsymbol{i}|$. Different from the discrete case, the number of parameters to be estimated therefore does not grow exponentially with $|\boldsymbol{i}|$ but quadratically.

The density contours of a normal factorized probability distribution are ellipsoids. Depending on the dependencies modeled by the factorization, these ellipsoids can be aligned along any axis. If there is no dependency between a set of random variables, the projected density contours in those dimensions are aligned along the main axes. In either case, a normal pdf is only capable of efficiently modeling *linear* dependencies.

*EDAs*

The first EDA for real–valued random variables was an adaptation of the original binary PBIL algorithm. The algorithm uses $l$ normal pdfs, one for each of the $l$ random variables (Rudlof and Köppen, 1996). To accommodate for these normal pdfs, the probability vector from the original PBIL algorithm is replaced with a vector that specifies for each variable the mean and variance of the associated normal pdf. The means are updated using a similar update rule as in the original binary PBIL. The variances are initially relatively large and are annealed down to a small value using a geometrically decaying schedule. New solutions are generated by drawing values from the normal pdfs for each variable separately.

A second adaptation of PBIL to the continuous case was introduced by Sebag and Ducoulombier (1998). Similar to the approach by Rudlof and Köppen (1996), they proposed to use a normal pdf for each variable. However, the variance is now updated using the same update rule as for the mean.

For real–valued random variables, Bayesian factorizations using normal pdfs were proposed simultaneously by Bosman and Thierens (2000b) within the probabilistic model–building EA framework and by Larrañaga, Etxeberria, Lozano and Peña (2000) in a variant of MIMIC that uses normal pdfs, termed $\text{MIMIC}_\text{C}$, and in the Estimation of Gaussian Network Algorithm (EGNA). As a first approach, Bosman and Thierens (2000b) used an algorithm by Edmonds (1967) to find a Bayesian factorization of minimal entropy in which each variable has at most one parent. Also, the optimal dependency–tree algorithm used in COMIT and the greedy chain–learning algorithm used in MIMIC were used (Bosman and Thierens, 2000a; Bosman and Thierens, 2000b). In a later publication (Bosman and Thierens, 2001a), the BIC metric was proposed in combination with a greedy factorization–learning algorithm. In the work by Larrañaga et al. (2000), finding a Bayesian factorization starts with a complete factorization graph. A likelihood–ratio test is then performed for each arc to determine whether or not that arc should be excluded from the graph. A greedy factorization–learning algorithm based on the BIC metric that starts from the univariate factorization was also used.

The use of univariate factorizations for real–valued random variables was studied and compared against the use of Bayesian factorizations by various researchers (Bosman and Thierens, 2000a; Bosman and Thierens, 2000b; Larrañaga et al., 2000; Bosman and Thierens, 2001a). In these studies, the use of univariately factorized normal probability distributions was shown to be inferior to the use of multivariate factorized normal probability distributions for optimization problems that have linear interactions between the problem variables. Specifically, far better results were obtained on the summation cancellation function. Although on this particular problem the EDAs even outperformed evolution strategies (ES), improvement over ES was not observed on all problems described in Section 2. On both the Michalewicz and the Rosenbrock function the EDA approach was even strongly inferior to ES. In general, the EDA approach was observed to have good optimization performance on problems with linear dependencies and even on problems with many local optima, in the likes of the Griewank function, of both lower and higher dimensionality. However, the EDAs cannot efficiently solve optimization problems with non–linear interactions between their variables. The main reason is that the interactions that can be modeled using the normal pdf are just linear. Hence, the structure of the problem is not fit well by the probability distribution. As a result, points surrounding the optimum get lost during optimization and the EDA is left only with solutions far away from the optimum. However, by estimating the probability distribution of these points with maximum likelihood, the EDA suffers from the drawback that it disregards gradient information. Instead, it keeps sampling points in the same area where the current selected solutions are while selection continuously decreases the size of this area. The EDA can therefore converge even while on a slope towards the optimum. Hence, even if the problem is unimodal, the EDA using maximum–likelihood estimations of the normal pdf can fail in finding an optimal solution. This happens for instance on the Rosenbrock problem, even for very large population sizes as is illustrated in Figure 3. The success rate of the EDA using the maximum–likelihood normal pdf was 0% for both dimensionalities. Conversely however, if the solutions do continue to surround the optimum, the use of the maximum–likelihood normal pdf is very efficient as is for instance the case on the sphere function where the initial variable ranges are centered around the optimum.

### Normal kernels pdf

*Definition*

The normal kernels pdf is obtained by centering one multivariate normal pdf at each point in the sample vector and by normalizing the sum of the densities:

$$P^{\mathcal{N}_K}_{(\boldsymbol{\Sigma}^{\boldsymbol{i}})}(Y_{\boldsymbol{i}})(\boldsymbol{y}) = \frac{1}{|\boldsymbol{\mathcal{S}}|} \sum_{j=0}^{|\boldsymbol{\mathcal{S}}|-1} P^{\mathcal{N}}_{((\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{i}},\boldsymbol{\Sigma}^{\boldsymbol{i}})}(Y_{\boldsymbol{i}})(\boldsymbol{y}) \tag{4}$$

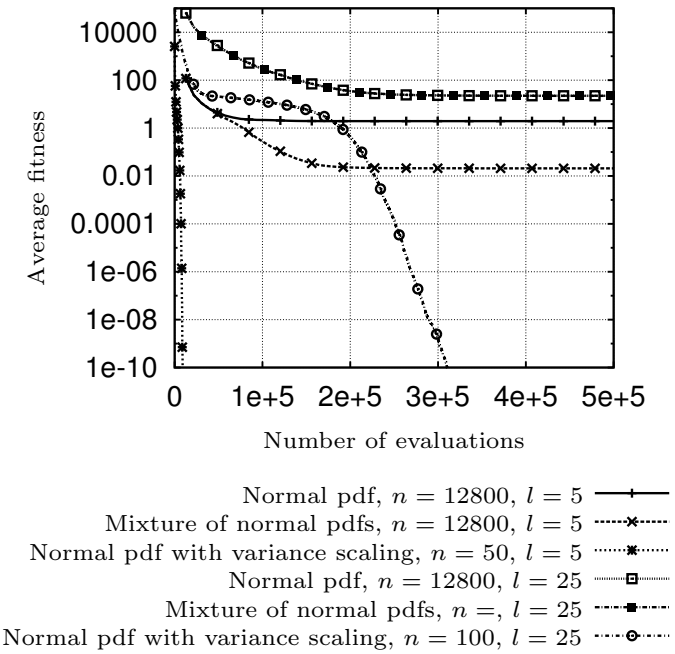


**Fig. 3.** Average convergence behavior (100 runs) of various continuous EDAs on the Rosenbrock function with a dimensionality of $l \in \{5, 25\}$. The results shown are the best results from all test settings: $n \in \{25, 50, 100, 200, 400, 800, 1600, 3200, 6400, 12800\}$, maximum number of evaluations $= 10^6$, number of clusters in mixture distribution $= 5$. Rates of success for $l = 5$: normal pdf 0%, mixture of normal pdfs 72%, normal pdf with variance scaling 100%. Rates of success for $l = 25$: normal pdf 0%, mixture of normal pdfs 0%, normal pdf with variance scaling 100%.

*Parameter estimation*

Deciding how to choose the covariance matrix for each normal pdf is hard. A maximum–likelihood estimate is undesirable because in that estimate the variances are zero, corresponding to a density of infinity at the mean of each normal pdf. Therefore, the variances in the covariance matrix that is used for each normal pdf in the normal kernels pdf should be set in a different manner.

One way of doing so, is to compute the range of the samples in $\boldsymbol{\mathcal{S}}$ in each dimension and to set the variance in the $i$–th dimension to a valued based on the range such that it decreases as the number of samples increases, e.g. $\alpha \cdot \text{range}_i / |\boldsymbol{\mathcal{S}}|$. The value of $\alpha$ controls the smoothness of the resulting density estimation.

Although formulas exist for the univariate conditional form of the normal kernels pdf that is required to construct Bayesian factorizations (Bosman, 2003), both their computational complexity and the lack of sensibility of using

maximum likelihood estimates have prevented the use of such factorizations in continuous EDAs using the normal kernels pdf.

*Properties*

The main advantage of the normal kernels pdf is that it has a natural tendency to fit the structure of the sample vector and is thereby capable of expressing complicated non–linear dependencies. A related disadvantage however is that the quality of the density estimation heavily depends on the value of $\alpha$. Intuitively, a larger $\alpha$ results in a smoother fit, but it is hard to predict beforehand what a good value for $\alpha$ would be. The normal kernels pdf has a tendency to *overfit* a sample collection. Without proper model selection and model fitting, the normal kernels pdf is hard to handle although it may be relatively fast in its use. One other possibility is to set the variances, or even covariances, for the normal kernels pdf adaptively. If the adaptation is done for each normal kernel separately, the resulting approach is equivalent to the use of evolution strategies (Bäck and Schwefel, 1993). Concluding, the normal kernels pdf certainly has interesting properties and potential to be used in IDEAs, but it is likely to be hard to handle when perceived as a manner of describing the structure of the data in a sample set.

*EDAs*

The normal kernels pdf was initially tried in an EDA by Bosman and Thierens (2000a) (see also (Bosman and Thierens, 2000b; Bosman, 2003)). The range–based approach based on the $\alpha$ parameter as mentioned above was used to set the variances of the normal kernels. Fine–tuning $\alpha$ was found to be highly problem–dependent. Not only is it difficult to set $\alpha$ to a useful value beforehand, a proper value for $\alpha$ is apt to change during the run of the EDA. Hence, good results are hard to obtain with this use of the normal kernels pdf.

### Normal mixture pdf

*Definition*

If we take $w$ normal pdfs instead of only a single one or as many as $|\boldsymbol{\mathcal{S}}|$, we have a trade–off between the cluster–insensitive normal pdf and the cluster–oversensitive normal kernels pdf. The normal mixture pdf for random variables $Y_{\boldsymbol{i}}$ is parameterized by $w$ triples that each consist of a mixture coefficient $\beta_{\boldsymbol{i}}^j$ a vector of $|\boldsymbol{i}|$ means and a symmetric covariance matrix of dimension $|\boldsymbol{i}| \times |\boldsymbol{i}|$:

$$P^{\mathcal{N}_M}_{\left((\beta_{\boldsymbol{i}}^0,\boldsymbol{\mu}_{\boldsymbol{i}}^0,\boldsymbol{\Sigma}^{0,\boldsymbol{i}}),\ldots,(\beta_{\boldsymbol{i}}^{w-1},\boldsymbol{\mu}_{\boldsymbol{i}}^{w-1},\boldsymbol{\Sigma}^{w-1,\boldsymbol{i}}),\right)}(Y_{\boldsymbol{i}})(\boldsymbol{y}) = \sum_{j=0}^{w-1} \beta_{\boldsymbol{i}}^j P^{\mathcal{N}}_{\left(\boldsymbol{\mu}_{\boldsymbol{i}}^j,\boldsymbol{\Sigma}^{j,\boldsymbol{i}}\right)}(Y_{\boldsymbol{i}})(\boldsymbol{y}) \quad (5)$$

*Parameter estimation*

A maximum likelihood estimation cannot be obtained analytically for $w > 1$. Therefore, as an alternative approach, the EM algorithm (Dempster, Laird and Rubin, 1977) can be used. The EM algorithm is a general iterative approach to computing a maximum–likelihood estimate. For the normal–mixture pdf, an EM–algorithm first initializes each mixture coefficient, all means and all covariance matrices to certain values and then updates all of these values iteratively until the algorithm converges or until a maximum number of iterations has been reached. We refrain from presenting a derivation of the update equations and refer the interested reader to the literature (Dempster et al., 1977; Bilmes, 1997).

An alternative manner to estimating a normal mixture pdf is to use clustering (Hartigan, 1975). First, the samples are clustered using a clustering method and subsequently a normal pdf is estimated in each cluster. The drawback of this approach is that from a probabilistic viewpoint, the resulting probability distribution estimation is almost certainly not a maximum likelihood estimate. However, the use of clustering does allow for the modeling of non–linear dependencies between the variables.

*Properties*

The main advantage of the normal mixture pdf is that it provides a trade–off between the normal pdf and the normal kernels pdf. The normal mixture pdf is able to model non–linear dependencies a lot more accurate than when a single normal pdf is used (see Figure 4 for an example). Although the normal mixture pdf doesn't have a flexibility as great as the normal kernels pdf has, the normal mixture pdf is much easier to handle. The reason for this is that the only parameter to set is the number of clusters to construct, which is a lot more transparent than the value of $\alpha$ to set in the case of the normal kernels probability distribution.

A maximum likelihood approach is available to estimate the normal mixture pdf parameters. However, especially as the number of variables and the number of mixture components increases, the result using the EM algorithm becomes unreliable. Similarly, the usefulness of clustering also decreases as the number of variables increases. Hence, it makes sense to factorize the probability distribution so that the probability distribution over all random variables is a composition of normal mixture pdf estimates for smaller sets of variables. Furthermore, the EM algorithm is a time–consuming approach. The clustering approach on the other hand can be quite fast, depending on the clustering algorithm that is used.

*EDAs*

Although the required expressions for computing Bayesian factorizations of the normal mixture pdf have been derived (Bosman, 2003), the resulting for-

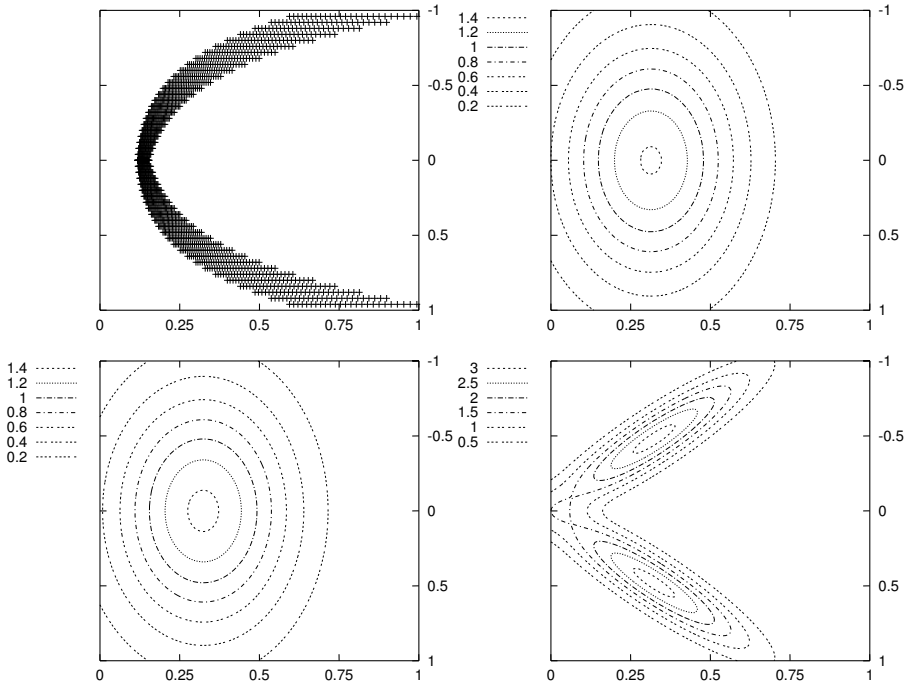


**Fig. 4.** *Top left*: A sample vector that contains non–linear dependencies. *Top right*: density contours of a product of two one–dimensional normal pdfs (maximum–likelihood estimate). *Bottom left*: density contours of a multivariate joint two–dimensional normal pdf (maximum–likelihood estimate). *Bottom right*: density contours of two multivariate joint two–dimensional normal pdfs that have been fit (maximum–likelihood estimate) after the sample vector was clustered.

mulas are rather involved and not efficient for practical use. Concordantly, no EDA has been reported in the literature as yet that uses such a factorization.

Gallagher, Frean and Downs (1999) proposed an EDA that uses the adaptive mixture pdf by Priebe (1994) for each variable separately (i.e. using the univariately factorized probability distribution). Although their approach outperformed other algorithms, the set of test problems consisted only of two relatively simple two–dimensional problems.

Bosman (2003) experimented with an EDA that uses the EM–algorithm to estimate a normal mixture pdf. Due to the complicated nature of Bayesian factorizations when combined with the normal mixture pdf, only the univariate factorization and the unfactorized probability distribution were tested. In low–dimensional spaces the added flexibility of the normal mixture pdf compared to the single normal distribution resulted in a clear improvement when solving the Rosenbrock function using an unfactorized pdf. The growth of the complexity of the non–linear dependency in the Rosenbrock function however quickly decreases the advantage of the higher flexibility offered by the use of

the normal mixture pdf. Although good results were also obtained on other problems in low–dimensional spaces, the additional computational overhead was found to be considerable.

Although the conditional pdf required for a Bayesian factorization is rather complex, the approach for constructing a marginal–product factorization as is done in the ECGA (see the chapter on ECGA) and the resulting factorized normal mixture pdf are rather straightforward. The resulting probability distribution describes for mutually exclusive subsets of variables a normal mixture pdf than can be constructed in one of the aforementioned manners. Ahn, Ramakrishna and Goldberg (2004) developed an EDA that uses exactly this approach where each normal mixture pdf is estimated using the clustering approach. Factorizing the probability distribution in this manner can allow for a more efficient modeling of independent subproblems. Although improvements were obtained over earlier real–valued EDA approaches, the resulting EDA was still not found to be efficient at solving the Rosenbrock function, especially as the dimensionality of the problem is increased.

### Histogram pdf

*Definition*

Although conceptually the histogram pdf is arguably the most simple way to represent a continuous real–valued probability distribution, we refrain from giving a precise mathematical definition as it is unnecessarily complicated. The histogram pdf splits up the range of each variable into several parts. A probability is associated with each hyperbox (commonly called a *bin*) that represents the probability of a sample to lie anywhere inside the combination of ranges associated with that bin. Note that the number of splits per variable or the relative size of all subranges do not need to be the same although in the most traditional sense, such a fixed–width version of the histogram pdf is the most commonly used.

*Parameter estimation*

A maximum–likelihood estimation is obtained in the same manner as for the integer (or binary) case, i.e. by computing the proportion of samples that fall into a certain bin.

*Properties*

A more detailed estimate of the true underlying probability distribution can be obtained if more bins are used. However, as the number of bins increases, the efficiency of modeling dependencies with the histogram pdf rapidly decreases as many bins will be empty. Moreover, since often the number of bins grows exponentially when expressing the joint probability of multiple random

variables (for instance when using the fixed–width approach), histograms are actually quite inefficient in expressing dependencies.

Furthermore, as the number of bins is increased, the danger of overfitting and lack of generalization increases as well. If more bins are used, the number of empty bins will increase. Drawing more samples from the estimated probability distribution will thus not produce any more samples in these areas, even though these areas might very well contain the global optimum.

*EDAs*

The algorithm by Servet, Trave-Massuyes and Stern (1997) is an adaptation of the original PBIL algorithm. In the algorithm, a range is stored for each variable. For each variable then, a histogram pdf with two bins is maintained, where the first bin corresponds with the first half of the domain and the second bin corresponds with the second half. The probability vector from the original PBIL algorithm now specifies for each variable the probability with which a new value for that variable is generated in the second half of the domain currently stored for that variable. A domain is resized to contain exactly one of the two halves of that domain if the histogram starts to converge to that half of that domain.

Bosman and Thierens (2000a) first investigated the use of an EDA that is based on a univariately factorized histogram distribution. In their work they considered the fixed–width variant. Tsutsui, Pelikan and Goldberg (2001) considered both the fixed–width as well as the fixed–height histogram variants in their work but also only in combination with the univariate factorization. The conclusions of both investigations are that the resulting EDAs are quite well capable of solving problems that do not exhibit dependencies between their variables. If dependencies do occur, the EDAs are not capable of exploiting these dependencies efficiently, which severely limits its practical use.

To overcome the exponential growth of the total number of bins as the joint probability distribution of multiple variables is taken into account, a specialized repeated–splitting technique as used in classification and regression trees can be used (Breiman, Friedman, Olshen and Stone, 1984). The available data is split in a single dimension. To this end all possible axis–parallel splits are investigated and the one that is the most beneficial in splitting up the data is selected. The process is then recursively repeated in the two subsets until no split significantly improves the quality of the overall probability distribution anymore. The resulting bins are not identically sized and are specifically tailored to fit the data, thereby reducing the number of empty bins to 0. This approach was used in an EDA by Pošík (2004). Although the EDA is capable of modeling unfactorized joint probabilities, the approach was found not to be able to minimize Rosenbrock's function. The main reason for this is likely the inability to generalize the probability distribution outside of the current range of selected solutions. Moreover, the approach was found not to be very robust against multimodality.

### 4.2 Mixture–of–factorizations probability distributions

A mixture–probability distribution is a weighted sum of probability distributions, each of which is a function of all random variables. The weights in the sum are all positive and sum up to one to ensure that the summation is itself a probability distribution. Although we have already encountered mixture probability distributions when we considered factorized probability distributions in Section 4.1 (i.e. the normal mixture pdf), in this section we consider the mixture as the main structure of the probability distribution. In Section 4.1 the main structure was the factorization. The general form of the mixture probability distribution over all random variables $\boldsymbol{Y}$ is:

$$P(\boldsymbol{Y}) = \sum_{i=0}^{k-1} \alpha^i P^i(\boldsymbol{Y}) \tag{6}$$

where $k$ is the number of mixture components, $\alpha^i \geq 0$ holds for all $i \in \{0, 1, \ldots, k-1\}$ and $\sum_{i=0}^{k-1} \alpha^i = 1$.

The mixture probability distribution can be seen as a logical extension of the factorization approach. It can be viewed upon as a way of using multiple factorizations by taking each probability distribution in the mixture to be a factorization. The use of mixture probability distributions intuitively appears to be an especially useful approach in the presence of multiple local optima as each component in the mixture can be used to focus on a single local optimum and thereby distribute the factorization search bias of the EDA over multiple regions of interest. To a lesser extent one of the virtues of a mixture probability distribution is that by using a combination of (simpler) probability distributions, more involved probability distributions can be constructed that for instance model non–linear dependencies better. However, this is already the case when using a factorization of a mixture pdf as already demonstrated in Figure 4 and hence is less of a virtue contributed by the mixture probability distribution on its own.

#### By means of clustering

Using clustering to subdivide the set of solutions for which to estimate a probability distribution followed by the subsequent estimation of probability distributions for each cluster separately was already discussed when we reviewed the normal mixture pdf in Section 4.1. Before its use by Ahn et al. (2004) in the construction of the factorized normal mixture probability distribution, the use of clustering to construct mixture probability distributions in EDAs was initially investigated in the binary domain by Pelikan and Goldberg (2000) using the $k$–means clustering algorithm (Hartigan, 1975). This approach was concurrently investigated in the context of real–valued EDAs by Bosman and Thierens (2001a) (see also (Bosman, 2003)). In their work,

a different, slightly faster clustering algorithm was used, called the leader algorithm (Hartigan, 1975). The probability distributions in the mixture are factorized normal pdfs. Hence, the resulting probability distribution is actually a normal mixture probability distribution. The results generally indicate an increase in optimization performance especially on the Rosenbrock function (see Figure 3 for an illustration). However, because no means has been proposed to detect the number of clusters that is actually required, the approach requires significantly more evaluations on problems where a mixture probability distribution is not required. Unfortunately, the improvement of using the mixture of factorized normal pdfs over the use of a single normal pdf on the Rosenbrock function decreases rapidly as the dimensionality of the problem increases. The reason for this is that the number of clusters required to model the non–linear dependencies between the problem variables properly, increases also with an increase in the problem dimensionality. As this in turn requires a larger population size to ensure enough solutions in each cluster to estimate a factorized normal pdf, the increase in search efficiency in terms of the number of required evaluations to solve the problem, decreases.

The difference with the factorized normal mixture pdf used by Ahn et al. (2004) is that because in their probability distribution the top–structure is a factorization, if the number of variables between which non–linear dependencies exist is limited and the groups of such non–linearly dependent variables are additively decomposable in the problem to be optimized, the factorization can match this additive decomposability structure whereas the mixture pdf can match the (smaller) non–linear dependencies. This additive decomposability and the smaller non–linear dependency can clearly not be exploited effectively by the top–level mixture probability distribution. Indeed, on an additively decomposable variant of the Rosenbrock function, the approach by Ahn et al. (2004) outperforms the approach by Bosman and Thierens (2001a). However, it should be clear that the incentive behind the mixture probability distribution is inherently different, namely to distribute the search bias over the search space and thus to be able to cope with multiple regions of interest in the search space simultaneously. The mixture probability distribution approach is hence more general as it also allows for instance to use the factorized normal mixture pdf in the EDA by Ahn et al. (2004) in each cluster, effectively combining the positive features of the EDA by Ahn et al. (2004) with the incentive behind the approach by Bosman and Thierens (2001a).

### 4.3 Other probability distributions

Although the use of the parametric pdfs in combination with factorizations and the use of clustering techniques to obtain mixture probability distributions as discussed in the previous subsections are representative of the most common techniques, other, often more involved, techniques exist.

**Top–level discretization and bottom–level continuous modeling**

A different use of the normal kernels pdf was proposed by Ocenasek and Schwarz (2002). In their EDA, a discrete decision tree (Friedman and Goldszmidt, 1996) is built for each variable by discretizing all other real–valued random variables using axis–parallel splits. The variable that is found to influence the current variable of interest the most is used to split up the data range. The procedure is recursively repeated after splitting the data. The rationale behind this approach is that once no more splitting is found to be beneficial, the leaf nodes in the decision tree correspond to subranges in the data set that can be properly estimated using univariately factorized pdfs. The pdf used by Ocenasek and Schwarz (2002) is the normal kernels pdf with $\alpha = 1$. Since now the variances are specifically tailored to the subsets constructed by the decision trees, this particular use of the normal kernels pdf is much more robust. Although the resulting EDA was found to obtain much better results on the Rosenbrock function, the overall performance of the algorithm was not found to be significantly better than previously developed EDAs. Moreover, the optimum of the Rosenbrock could still not be reached satisfactorily (Kern, Müller, Hansen, Büche, Ocenasek and Koumoutsakos, 2004).

**Probability distributions based on latent variables**

A few alternative approaches currently used in real–valued EDAs are based on the use of latent, or hidden, variables. These techniques attempt to model the underlying data source by projecting the data onto another domain while attempting to retain the most important features. Often, the dimensionality of the data is then reduced.

An example of such techniques is the well–known principal component analysis (PCA) (Jolliffe, 1986). In PCA, $l' < l$ vectors are chosen such that the variance in those vectors is the largest when projecting the $l$–dimensional data onto these vectors. The latent variables are the newly introduced variables that are used to model the data. Another approach in which latent variables are used, is the Helmholtz machine. A Helmholtz machine is closely related to neural networks and consists of a layer of input variables representing the $l$ dimensions of the data and provides for multiple layers of latent variables. Connections between these layers allow for the learning of a model that describes the data, as well as the generation of new data.

Bishop (1999) indicated how PCA can be used to estimate probability distributions and how to generate new samples from the estimated probability distributions. Using normal pdfs, the PCA–based estimated probability distribution over the selected solutions, is an $l$–dimensional normal probability distribution. This approach has been used for real–valued optimization (Shin and Zhang, 2001; Cho and Zhang, 2001; Shin, Cho, and Zhang, 2001). The authors also used Helmholtz machines in combination with normal pdfs. The results obtained are comparable to those obtained with factorized probability

distributions, but the number of latent variables is fixed beforehand, whereas the approaches using factorized probability distributions are able to learn the structure of the probabilistic model from data.

In the approach by (Cho and Zhang, 2002) a mixture of factor analyzers is used. Standard factor analysis is a latent variable model that is based on a linear mapping between the random variables and the latent variables, resulting in a normal distribution that is modeled over the original random variables. An EM–algorithm is used to find parameter estimates for the latent variables in each mixture component as well as the mixture coefficients themselves. The number of mixture components is fixed beforehand as are again the number of latent variables. The results for numerical optimization indicate better performance for the mixture over the single factor analysis and other non–mixture real–valued probabilistic model–building EAs, but the structures of the latent–variable models were composed by hand.

The recentmost EDA based on latent variable models was introduced by Cho and Zhang (2004). The probability distribution used in their EDA is based on the variational Bayesian independent component analyzers mixture model by Choudrey and Roberts (2003). Although the formal equations involved are out of scope for this chapter, we note that this recent way of estimating probability distributions overcomes many of the problematic issues that traditional models such as mixture of normal distributions estimated with EM algorithms have. In principle, the model is capable of modeling any density function as long as enough components are used in the mixture. Traditionally, there exists a problem with computing the required number of components. In the model by Choudrey and Roberts (2003) this problem is solved using Bayesian inference. The use of this extremely powerful model in an EDA resulted in an improvement over earlier proposed real–valued EDAs. However, although the approach obtained much better results on the Rosenbrock function, it was still not able to converge to the optimum for the 10–dimensional case using a population size of 6400. Although the model is in theory capable of capturing the structure of the search space, the number of solutions that is required to actually construct this probability distribution well enough is extremely large.

## 5 Analysis and redesign of real–valued EDAs

### 5.1 Analysis

Following the traditional motivation from discrete spaces and the lessons learned from discrete GA research (Goldberg, 2002b), one assumes that initially enough information is supplied to the EDA, i.e. all the building blocks are initially in the population. This is typically ensured by making the population large enough. All that the EDA is now required to do is to detect the building blocks and mix them properly. From the previous chapters we

already know that by using maximum–likelihood estimates for discrete spaces in combination with factorization selection techniques, EDAs can be built that very efficiently meet with this requirement and consequently achieve efficient optimization. By using a proper factorization, the important substructures of the optimization problem at hand are identified and specific combinations of bits can be assigned high probabilities of replication when generating new solutions. Hence, the important substructures never get lost during optimization and are combined effectively to reach the optimal solution.

All real–valued EDAs for numerical optimization that we discussed so far employ a probability–distribution–estimation technique that is equally bent on describing the set of selected solutions as well as possible. Whenever possible, maximum–likelihood estimates are used. An approximation of the maximum–likelihood estimate is used otherwise. One could validly argue that this approach thus follows a direct translation of the EDA approach in the discrete domain into the continuous domain. Alternatively, one can argue that this approach conforms completely to the design of the optimal EDA that uses the true probability distribution as discussed in Section 3. Still, real–valued EDAs so far have made for a far smaller success story than have EDAs for the discrete domain, even though the range of actual EDA approaches is wider in the continuous domain. For instance, no EDA discussed so far in this chapter is able to optimize the Rosenbrock function efficiently even though this problem is unimodal and smooth. This has recently led researches to take a closer look at what is happening with EDAs in the continuous domain.

The apparent failure of real–valued EDAs to solve the Rosenbrock function was first noticed by Bosman and Thierens (2001b) (see also (Bosman, 2003)). The explanation given for this failure was that because no assumptions are made on the source of the data from which to estimate the probability distribution, real–valued EDAs disregard gradient information. In other words, even part of an interesting region is discovered and the selected solutions are located on a slope towards the optimum, the EDA can converge *on this slope* since it only effectively searches inside the area covered by the selected solutions. Through maximum–likelihood estimates there is no means of generalizing the estimated density outside this area. The premature convergence on the Rosenbrock function was discussed further more recently by Bosman and Grahl (2005) as well as by Yuan and Gallagher (2005). Premature convergence was also observed by Kern et al. (2004) on the even simpler tilted–plane function where the optimum is located at a boundary of the search space. For a real–valued EDA using the normal pdf with maximum–likelihood estimates it was proved by Grahl, Minner and Rothlauf (2005a) (see also (Grahl, Minner and Rothlauf, 2005b)) that the mean of this EDA can indeed only move a limited distance through the search space. This distance is dependent on the selection intensity and the initial variance (i.e. spread of the initial population). Unless the optimum lies within this bounded distance, the variance will go to 0 too quickly causing the EDA to converge prematurely.

One could now argue that this is an unfair analysis as it violates the assumption that initially enough data is available, i.e. that the optimum inside the region covered by the initial population. If this is not the case, one should consider the analogy of solving the one–max problem with a binary EDA when for some variables not a single 1 is present in the population. It is however important to note that in the continuous domain that such essential information (i.e. the structure of the problem to be optimized, typically the building blocks in the discrete case) easily gets lost during optimization. In general, the reason for loss of essential information is that the problem structure is not matched by the search bias of the optimizer. For instance, if a binary EDA is not capable of detecting the individual deceptive trap functions, the additively decomposable trap function cannot be optimized efficiently because the building blocks will have a too small probability of surviving variation.

In the continuous domain however, capturing the essential information in the probability distribution is much more difficult (Bosman and Grahl, 2005). Problem structure in the continuous domain is exemplified by the contour lines of the function to be optimized. Hence, the true distribution of the selected individuals can be of virtually any shape. This means that we need both the universal approximation property of our probability distribution as well as an infinite sample size to ensure that the problem structure is effectively exploited by the EDA. However, such universal approximation is computationally intractable. In practice, a continuous EDA will have to rely on tractable probability distributions such as the normal pdf. This means that the contours of the probability distribution may no longer match with the contour lines of the function to be optimized. Consequently, the notion of dependency as observed in the estimated probability distribution does not have to match with the actual dependency of the variables according to the optimization problem at hand. Hence, linkage information in the continuous domain cannot by definition be extracted from the estimated probability distribution. What is even more important to realize is that by using relatively simple pdfs the property of having (local) optima inside the area covered by the selected solutions may easily be lost during optimization. This will put the EDA back in the position from where it can provably not find the optimum using maximum–likelihood estimates, even when the selected solutions are located on a slope.

### 5.2 Redesign

To overcome the problem identified above, three approaches can generally speaking be taken. We will discuss these three approaches in turn.

#### More involved probability distributions

The capacity of the class of probability distribution used in the EDA can be increased. As pointed out in Section 4.3, the approach by Cho and Zhang (2004) is probably the best example of this approach. But even though the model

used here is extremely flexible, the population size that is generally required to get the estimated probability distribution right and not lose the encapsulation of optima during optimization is enormous. Moreover, the computational costs of estimating such involved probability distributions is considerable.

### Explicit gradient exploitation

To allow the EDA to search outside of its range the explicit use of gradient information can be enforced by hybridizing the EDA with local search operators. Although such a hybrid approach can indeed be effective (Bosman and Thierens, 2001b), such hybridization is possible for all EAs. Hence, we seek a more specific solution to the problem at hand. Moreover, if desired, the resulting improved EDA can then still be hybridized.

### Variance scaling

The third approach is strikingly simple in its nature, but it is specifically designed for use in real–valued EDAs and has already been shown to be a very effective principle (Ocenasek, Kern, Hansen, Müller and Koumoutsakos, 2004; Bosman and Grahl, 2005; Yuan and Gallagher, 2005). The basic idea is to set the variance of the probability distribution during the search not (only) according to the maximum–likelihood approach, but (also) according to other criteria such as the success rate.

In the approach by Ocenasek et al. (2004) the decision–trees combined with the normal kernels pdf is used (Ocenasek and Schwarz, 2002). In addition however, a scaling factor is maintained. When drawing new solutions from the estimated probability distribution, the variance of a normal kernel is scaled by multiplication with this factor. The size of the scaling factor depends on the success rate of the restricted tournament selection operator (i.e. the number of times an offspring solution is better). An adaptive scheme is used that changes the scaling factor to ensure that this success rate stays in–line with the 1/5 success rule for evolution strategies (ES) (Bäck and Schwefel, 1993).

Yuan and Gallagher (2005) showed that by scaling the variance of the estimated normal probability distribution by a factor of 1.5, an EDA based on this variance–scaled normal probability distribution is capable of finding the optimum of the Rosenbrock function. Although in itself this result is striking, a fixed scaling factor will in general not be optimal. In the approach by Bosman and Grahl (2005) a simple, but effective, adaptive–variance–scaling scheme is proposed for use in an EDA that uses the factorized normal pdf. Similar to the approach by Ocenasek et al. (2004) a scaling factor is maintained and upon drawing new solutions from the probability distribution, the covariance matrix is multiplied by this scaling factor. Updating this scaling factor is done differently however. If the best fitness value improves in one generation, the current size of the variance allows for progress. Hence, a further enlargement of the variance may allow for further improvement in the next generation and

the scaling factor is increased. If on the other hand the best fitness does not improve, the range of exploration may be too large to be effective and the scaling factor is (slowly) decreased. In addition to variance scaling Bosman and Grahl (2005) also show how a test can be constructed specifically for the normal distribution that allows to distinguish between situations where variance scaling is required and where variance scaling is not required. The test is based on (ranked) correlation between density and fitness.

The results of the above mentioned EDAs that employ the scaling of the variance are significantly better than those of the earlier EDAs that only employ maximum–likelihood estimates. The algorithms are for instance capable of finding the optimum of the Rosenbrock function efficiently. The results of the algorithm by Bosman and Grahl (2005) on the Rosenbrock function are shown in Figure 3. The key to their success is that the EDAs without variance scaling are very efficient if the structure of the problem is not too complicated, such as is the case for the sphere function. If the structure gets more involved, there is hardly any means to generalize the probability distribution over the entire search space. With the variance–scaling approach the EDA is equipped with the possibility to shift its focus by means of variance adaptation and to go and find local structures that it can efficiently exploit without the additional variance scaling.

The results obtained with variance–scaling bring the EDAs in strong competition with one of the leaders in (evolutionary) continuous optimization, the CMA–ES (Hansen and Ostermeier, 2001; Hansen, Müller and Koumoutsakos, 2003; Kern et al., 2004). Strikingly, the EDA approach using the normal pdf is at the top level similar to the CMA–ES. Both algorithms compute values for the parameters of the normal pdf and subsequently draw new solutions from this normal pdf. Hence, the CMA–ES can validly be argued to actually be an EDA. The difference between the approaches lies in how the parameters are set. In the pioneering EDA approaches, maximum–likelihood estimates are used on the basis of the set of selected solutions. It has been observed however that this more often than not doesn't lead to efficient optimization. The CMA–ES on the other hand has been found to be a very efficient optimizer. The CMA–ES actually uses the same estimate for the mean (i.e. the center of mass (also called sample average) of the selected solutions). The differences lies in the way the covariance matrix is built. Strikingly, it is exactly in the changing of the covariance matrix (by scaling) that the EDA approach has recently been found to have room for improvement. In the CMA–ES, the covariance matrix is built using information about how the mean of the population has shifted the previous generation. This information is combined with the (by multiplication slowly fading) memory or cumulation of shifts in earlier generations to obtain what is called the *evolution path*. In this approach there is no need to explicitly check for success in generating new better solutions as the shift towards better regions automatically influences the evolution path, which is subsequently used to determine the covariance matrix.

As a result of using the mean shift, the CMA–ES is clearly capable of exploiting gradient properties of a problem, which is exactly what was hypothesized to be lacking in maximum–likelihood EDAs (Bosman and Thierens, 2001b; Bosman, 2003). A final striking correspondence between the CMA–ES and the EDAs that use variance–scaling is that the covariance matrix in the CMA–ES is actually factorized into the multiplication of a covariance matrix and what is called a global stepsize. The global stepsize is also computed on the basis of the evolution path. This factorization is identical to the variance–scaling approach in EDAs, the only difference being in terminology and the way in which the scaling factor (or global stepsize) is computed.

It is interesting to see that although the background and motivation for the (real–valued) EDA approach and the (CMA–)ES approach are different, the developments in these areas appear to be converging onto a similar approach. Future research will have to point out which of the approaches is the overall best way to go and whether the benefits of both approaches can be integrated to arrive at even better evolutionary optimizers.

## 6 Summary and conclusions

In this chapter we have discussed the design of real–valued EDAs for numerical optimization. We have indicated what types of problem difficulty we typically expect and what the optimal EDA for numerical optimization looks like. Subsequently we provided an overview of existing EDA approaches and indicated that already there exists a vast array of approaches as many different probability distributions have been studied.

Although the capacity of the class of probability distribution used in the EDA has grown significantly, real–valued EDAs that attempt to estimate the probability distribution as well as possible (i.e. typically using maximum–likelihood estimates) from the set of selected solutions seem not to work well on problems such as the Rosenbrock problem. Although this problem is unimodal and has nice smooth gradient properties, EDAs tend to converge prematurely while still on the slope towards the optimum.

The main problem is that in continuous problems, the structure of a problem is characterized by the contours of the function to be optimized. These contours can take any shape. Hence fitting the problem structure in the continuous case requires the intractable universal approximation property and huge amounts of data, rending the EDA approach inefficient. It is therefore much more convenient to rely on the use of simpler probability distributions. However, by doing so, important regions of the search space may get lost during optimization because they are assigned extremely low densities due to the mismatch between the density contours of the estimated probability distribution and the density contours of the fitness function. To still be able to use simpler probability distributions alternative means of overcoming problems of premature convergence should be found. To this end it is much more

convenient to view problem structure as an arrangement of slopes and peaks in the search space. These simpler substructures are much easier to take into account and to build inductive search biases for. Actually, the current range of EDAs is already capable of searching such simpler substructures efficiently. Hence, what should in addition be provided is the means to shift between these substructures and ultimately focus on the most interesting one. To this end variance–scaling approaches have recently been proposed and have been shown to lead to much improved EDAs.

This significant improvement signals that there is a new open road to explore for the design of real–valued EDAs for numerical optimization; and this one has a fast lane.

## References

Ahn, C. W., Ramakrishna, R. S. and Goldberg, D. E. (2004). Real-coded Bayesian optimization algorithm: Bringing the strength of BOA into the continuous world, *in* K. Deb et al. (eds), *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–2004*, Springer–Verlag, Berlin, pp. 840–851.

Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*, John Wiley & Sons Inc., New York, New York.

Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization, *Evolutionary Computation* **1**(1): 1–23.

Baluja, S. and Caruana, R. (1995). Removing the genetics from the standard genetic algorithm, *in* A. Prieditis and S. Russell (eds), *Proceedings of the Twelfth International Conference on Machine Learning*, Morgan Kauffman, Madison, Wisconsin, pp. 38–46.

Bilmes, J. (1997). A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models, Technical Report ICSI TR-97-021, Department of Electrical Engineering, University of Berkeley, Berkeley, California.

Bishop, C. M. (1999). Latent variable models, *in* M. I. Jordan (ed.), *Learning in Graphical Models*, The MIT Press, Cambridge, Massachusetts.

Bosman, P. A. N. (2003). *Design and Application of Iterated Density–Estimation Evolutionary Algorithms*, PhD thesis, Utrecht University, Utrecht, the Netherlands.

Bosman, P. A. N. and Grahl, J. (2005). Matching inductive search bias and problem structure in continuous estimation of distribution algorithms, Technical report, Mannheim Business School, Department of Logistics, http://www.bwl.uni-mannheim.de/Minner/hp/_files/forschung/reports/tr-2005-03.pdf.

Bosman, P. A. N. and Thierens, D. (2000a). Continuous iterated density estimation evolutionary algorithms within the IDEA framework, *in* M. Pelikan et al. (eds), *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference — GECCO–2000*, Morgan Kaufmann, San Francisco, California, pp. 197–200.

Bosman, P. A. N. and Thierens, D. (2000b). Expanding from discrete to continuous estimation of distribution algorithms: The IDEA, *in* M. Schoenauer

et al. (eds), *Parallel Problem Solving from Nature – PPSN VI*, Springer–Verlag, Berlin, pp. 767–776.

Bosman, P. A. N. and Thierens, D. (2001a). Advancing continuous IDEAs with mixture distributions and factorization selection metrics, *in* M. Pelikan and K. Sastry (eds), *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference — GECCO–2001*, Morgan Kaufmann, San Francisco, California, pp. 208–212.

Bosman, P. A. N. and Thierens, D. (2001b). Exploiting gradient information in continuous iterated density estimation evolutionary algorithms, *in* B. Kröse et al. (eds), *Proceedings of the Thirteenth Belgium–Netherlands Artificial Intelligence Conference BNAIC–2001*, pp. 69–76.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and Regression Trees*, Wadsworth, Pacific Grove, California.

Buntine, W. (1994). Operations for learning with graphical models, *Journal of Artificial Intelligence Research* **2**: 159–225.

Cho, D.-Y. and Zhang, B.-T. (2001). Continuous estimation of distribution algorithms with probabilistic principal component analysis, *Proceedings of the 2001 Congress on Evolutionary Computation — CEC–2001*, IEEE Press, Piscataway, New Jersey, pp. 521–526.

Cho, D.-Y. and Zhang, B.-T. (2002). Evolutionary optimization by distribution estimation with mixtures of factor analyzers, *Proceedings of the 2002 Congress on Evolutionary Computation — CEC–2002*, IEEE Press, Piscataway, New Jersey, pp. 1396–1401.

Cho, D.-Y. and Zhang, B.-T. (2004). Evolutionary continuous optimization by distribution estimation with variational Bayesian independent component analyzers mixture model, *in* X. Yao et al. (eds), *Parallel Problem Solving from Nature – PPSN VIII*, Springer–Verlag, Berlin, pp. 212–221.

Choudrey, R. A. and Roberts, S. J. (2003). Variational mixture of Bayesian independent component analyzers, *Neural Computation* **15**(1): 213–252.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm, *Journal of the Royal Statistic Society* **Series B 39**: 1–38.

Edmonds, J. (1967). Optimum branchings, *Journal of Research of the National Bureau of Standards* **71B**: 233–240. Reprinted in Math. of the Decision Sciences, *Amer. Math. Soc. Lectures in Appl. Math.*, 11:335–345, 1968.

Edwards, D. (1995). *Introduction to Graphical Modelling*, Springer–Verlag, Berlin.

Etxeberria, R. and Larrañaga, P. (1999). Global optimization using Bayesian networks, *in* A. A. O. Rodriguez et al. (eds), *Proceedings of the Second Symposium on Artificial Intelligence CIMAF–1999*, Institute of Cybernetics, Mathematics and Physics, pp. 332–339.

Friedman, N. and Goldszmidt, M. (1996). Learning Bayesian networks with local structure, *in* E. Horvits and F. Jensen (eds), *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence UAI–1996*, Morgan Kaufmann, San Francisco, California, pp. 252–262.

Gallagher, M., Frean, M. and Downs, T. (1999). Real–valued evolutionary optimization using a flexible probability density estimator, *in* W. Banzhaf et al. (eds), *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–1999*, Morgan Kaufmann, San Francisco, California, pp. 840–846.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learing*, Addison Wesley, Reading, Massachusetts.

Goldberg, D. E. (2002a). *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*, Vol. 7 of *Series on Genetic Algorithms and Evolutionary Computation*, Kluwer Academic Publishers.

Goldberg, D. E. (2002b). *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*, Vol. 7 of *Series on Genetic Algorithms and Evolutionary Computation*, Kluwer Academic Publishers.

Grahl, J., Minner, S. and Rothlauf, F. (2005a). An analysis of iterated density estimation and sampling in the UMDAc algorithm, *Late–Breaking Papers of the Genetic and Evolutionary Computation Conference — GECCO–2005*.

Grahl, J., Minner, S. and Rothlauf, F. (2005b). Behaviour of UMDAc with truncation selection on monotonous functions, *Proceedings of the 2005 Congress on Evolutionary Computation — CEC–2005*, IEEE Press, Piscataway, New Jersey.

Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies, *Evolutionary Computation* **9**(2): 159–195.

Hansen, N., Müller, S. D. and Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA–ES), *Evolutionary Computation* **11**(1): 1–18.

Hartigan, J. (1975). *Clustering Algorithms*, John Wiley & Sons, Inc., New York, New York.

Holland, J. H. (1975). *Adaptation in Natural and Artifical Systems*, University of Michigan Press, Ann Arbor, Michigan.

Jolliffe, I. T. (1986). *Principal Component Analysis*, Springer–Verlag, Berlin.

Kern, S., Müller, S. D., Hansen, N., Büche, D., Ocenasek, J. and Koumoutsakos, P. (2004). Learning probability distributions in continuous evolutionary algorithms — a comparative review, *Natural Computing* **3**(1): 77–112.

Larrañaga, P., Etxeberria, R., Lozano, J. A. and Peña, J. M. (2000). Optimization in continuous domains by learning and simulation of Gaussian networks, *in* M. Pelikan et al. (eds), *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference — GECCO–2000*, Morgan Kaufmann, San Francisco, California, pp. 201–204.

Lauritzen, S. L. (1996). *Graphical Models*, Clarendon Press, Oxford.

Mühlenbein, H. and Höns, R. (2005). The estimation of distributions and the minimum relative entropy principle, *Evolutionary Computation* **13**(1): 1–27.

Ocenasek, J. and Schwarz, J. (2002). Estimation of distribution algorithm for mixed continuous-discrete optimization problems, *2nd Euro-International Symposium on Computational Intelligence*, pp. 227–232.

Ocenasek, J., Kern, S., Hansen, N., Müller, S. and Koumoutsakos, P. (2004). A mixed Bayesian optimization algorithm with variance adaptation, *in* X. Yao et al. (eds), *Parallel Problem Solving from Nature – PPSN VIII*, Springer–Verlag, Berlin, pp. 352–361.

Pelikan, M. and Goldberg, D. E. (2000). Genetic algorithms, clustering, and the breaking of symmetry, *in* M. Schoenauer et al. (eds), *Parallel Problem Solving from Nature – PPSN VI*, Springer–Verlag, Berlin, pp. 385–394.

Pelikan, M. and Goldberg, D. E. (2001). Escaping hierarchical traps with competent genetic algorithms, *in* L. Spector et al. (eds), *Proceedings of the GECCO–2001 Genetic and Evolutionary Computation Conference*, Morgan Kaufmann, San Francisco, California, pp. 511–518.

Pelikan, M. and Goldberg, D. E. (2003). Hierarchical BOA solves ising spin glasses and maxsat, *in* E. Cantú-Paz et al. (eds), *Proceedings of the GECCO–2003 Genetic and Evolutionary Computation Conference*, Springer–Verlag, Berlin, pp. 1271–1282.

Pelikan, M., Goldberg, D. E. and Cantú-Paz, E. (1999). BOA: The Bayesian optimization algorithm, *in* W. Banzhaf et al. (eds), *Proceedings of the GECCO–1999 Genetic and Evolutionary Computation Conference*, Morgan Kauffman, San Francisco, California, pp. 525–532.

Pošík, P. (2004). Distribution tree–building real–valued evolutionary algorithm, *in* X. Yao et al. (eds), *Parallel Problem Solving from Nature – PPSN VIII*, Springer–Verlag, Berlin, pp. 372–381.

Priebe, C. E. (1994). Adaptive mixtures, *Journal of the American Statistical Association* **89**(427): 796–806.

Rudlof, S. and Köppen, M. (1996). Stochastic hill climbing with learning by vectors of normal distributions, *in* T. Furuhashi (ed.), *Proceedings of the First Online Workshop on Soft Computing (WSC1)*, Nagoya University, Nagoya, Japan, pp. 60–70.

Russel, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*, Prentice Hall, Englewood Cliffs, New Jersey.

Sebag, M. and Ducoulombier, A. (1998). Extending population–based incremental learning to continuous search spaces, *in* A. E. Eiben et al. (eds), *Parallel Problem Solving from Nature – PPSN V*, Springer–Verlag, Berlin, pp. 418–427.

Servet, I., Trave-Massuyes, L. and Stern, D. (1997). Telephone network traffic overloading diagnosis and evolutionary computation technique, *in* J. K. Hao et al. (eds), *Proceedings of Artificial Evolution '97*, Springer–Verlag, Berlin, pp. 137–144.

Shin, S.-Y. and Zhang, B.-T. (2001). Bayesian evolutionary algorithms for continuous function optimization, *Proceedings of the 2001 Congress on Evolutionary Computation — CEC–2001*, IEEE Press, Piscataway, New Jersey, pp. 508–515.

Shin, S.-Y., Cho, D.-Y., and Zhang, B.-T. (2001). Function optimization with latent variable models, *in* A. Ochoa et al. (eds), *Proceedings of the Third International Symposium on Adaptive Systems ISAS–2001 — Evolutionary Computation and Probabilistic Graphical Models*, Institute of Cybernetics, Mathematics and Physics, pp. 145–152.

Tatsuoka, M. M. (1971). *Multivariate Analysis: Techniques for Educational and Psychological Research*, John Wiley & Sons Inc., New York, New York.

Thierens, D. and Goldberg, D. (1993). Mixing in genetic algorithms, *in* S. Forrest (ed.), *Proceedings of the fifth conference on Genetic Algorithms*, Morgan Kaufmann, pp. 38–45.

Tsutsui, S., Pelikan, M. and Goldberg, D. E. (2001). Evolutionary algorithm using marginal histogram in continuous domain, *in* M. Pelikan and K. Sastry (eds), *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference — GECCO–2001*, Morgan Kaufmann, San Francisco, California, pp. 230–233.

Yuan, B. and Gallagher, M. (2005). On the importance of diversity maintenance in estimation of distribution algorithms, *in* H.-G. Beyer et al. (eds), *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–2005*, ACM Press, New York, New York, pp. 719–726.