Linkages Detection in Histogram-Based Estimation of Distribution Algorithm

Nan Ding[1] and Shude Zhou[2]

[1] Department of Electronic Engineering, Tsinghua University,
Beijing, 100084
China
[2] China Academy of Electronics and Information Technology,
Beijing, 100041
China

Abstract. In this chapter, we review two methods that deal with the linkage detection in the Histogram-based Estimation of Distribution Algorithms; one is based on probabilistic graphical models, and the other is based on space transformation. The two methods deal with the linkage in the optimization problem with different accuracy and with different computational complexity. Probabilistic graphical model is generally more accurate but always associated to high-cost, while transformation is the opposite. In the following, we will mainly discuss the way to reduce the complexity of the method based on probabilistic graphical model and the way to obtain the transformation which captures the dominant linkage of the problem.

1 Introduction

Estimation of distribution algorithms (EDAs) are a class of evolutionary algorithms that use probabilistic model of promising solutions found so far to obtain new candidate solutions of optimized problems. One of the charming characteristics of this evolutionary paradigm is that the joint probability density can explicitly represent correlation among variables [1,16]. It is verified by several researchers [2] that multivariate-dependency EDAs have the potential ability to optimize hard problems with strong nonlinearity. However, it is also noted that how to efficiently learn the complex probabilistic model is a bottleneck problem. Therefore, obtaining a good balance between the complication of probabilistic models and the efficiency of learning method is a key factor for designing new EDAs.

In continuous domain, the predominant probabilistic model applied by EDAs is based on Gaussian probability distribution. Continuous EDAs based on multivariate Gaussian distribution have polynomial computational complexity. However, the inherent shortcoming of Gaussian-based EDA is that the unimodal model is too rough and thus is likely to mislead the search to a local optimum when solving complex optimization problems. Although clustering techniques such as Gaussian mixture model, Gaussian kernel model are considered to conquer this shortcoming in the literature, their complicated probabilistic distributions make it more difficult to estimate the linkage information, and thus the computational complexity increases remarkably.

Histogram model is another alternative probabilistic model in the continuous EDA. It is also called a kind of discretization of the continuous problem. In comparison with

Y.-p. Chen, M.-H. Lim (Eds.): Linkage in Evolutionary Computation, SCI 157, pp. 25–40, 2008.
springerlink.com

unimodal Gaussian model, histogram probabilistic model is able to represent multiple local optima by bins of different heights. Histogram model has already been used in previous work [3-11,13,14]. For example, marginal histogram models are applied in the FWH [7] by S. Tsutsui et al. and the histogram-based EDA (HEDA) [8,9] by N. Ding et al. B. Yuan et al. [11] also proposed the HEDA as an extension of the PBIL [12]. Q. Zhang et al. introduce EDA/L [14] in which several local search strategies are employed in a marginal histogram model.

In those above algorithms, the complete probability is approximated by the product of the marginal probability of each variable, that is to say, the linkages of the variables are discarded. However, as we know, when optimizing problems with bounded epistasis, the linkage information should be given prior consideration in the process of the evolutionary algorithms. The IDEA [4,5] based on histogram model (IDEA-H) by P.A.N. Bosman et al. used the multivariate histogram model to consider the variable linkage, but they also remarked that the complexity of the IDEA-H grows exponentially when expressing joint probability of multiple random variables.

The aim of the chapter is to conquer the linkage problem of HEDA from two aspects: one is based on probabilistic graphical models (PGM), where the multivariate histogram model is considered; and the other is based on space transformation, where the marginal distribution is built in the transformed space. PGM takes the Markov properties into account, where the variable is independent of each other when given its neighbors in the graphical models, thus it avoids estimating the complete probability. Space transformation works under the assumption that a decent transformation from the original space may cancel out some of the dominant linkages among the variables. The two methods deal with the variables linkages of the optimization problem with different accuracy and with different computational complexity. Probabilistic graphical model is generally more accurate but always associated to high-cost, while transformation is the opposite.

We also want to acknowledge that P. Pošík in [18] had a general introduction to the real-valued evolutionary algorithm on the use of probabilistic model and coordinate transform, which is really helpful to our work. The chapter here would mostly focus on the Histogram-based EDA and would contain the specific concerns about the histogram model.

This chapter is organized as follows. Section 2 briefly reviews the HEDA, especially the marginal HEDA. In Section 3, HEDA based on probabilistic graphical models is introduced and especially we discuss how to reduce its computational complexity. Section 4 is about HEDA based on space transformation.

2 Histogram-Based Estimation of Distribution Algorithm and Its Marginal Case

The histogram-based estimation of distribution algorithm (HEDA) has a main framework as follow:

1. Initialize the histogram model
2. Generate population $P(t)$ by sampling on the histogram model.
3. Evaluate and rank the population $P(t)$.

4. Update the histogram model according to the selected individuals $P'(t)$.
5. Return to step 2 if not terminated.

The marginal histogram model has a general form of

$$P(Z_0,...,Z_{l-1}) = \prod_{i=0}^{l-1} P(Z_i) \tag{1}$$

In Eq.(1), each $P(Z_i)$ $(i = 0,...,l-1)$ denotes a 1-variate histogram model.

The core of marginal histogram-based estimation of distribution algorithm is to estimate the marginal distributions of $P(Z_i)$ $(i = 0,...,l-1)$ and then to generate new individuals by sampling on $P(Z_i)$ for each variable. The FWH [7] and the sur-shr-HEDA [9] both belong to this class of HEDA. We now have a brief review of these two algorithms.

In the FWH, the height of each bin in each variable Z_i is proportional to the count of the selected individuals in it. To sample each variable of the new individual, firstly a bin is sampled according to the heights of the bins of the variable, and then the real value of that variable of the new individual is uniformly sampled in the domain of the bin. Since the height of the bin can be normalized to be equal to the probability of the bin, in the later description we will no longer differentiate the two phrases "the height of the bin" and "the probability of the bin".

In the sur-shr-HEDA, two specific strategies for HEDA, the surrounding effect and the shrinking strategy were developed to conquer the two drawbacks of the HEDA. The initial population should be large enough to sample the variables with a lot of bins; otherwise, many bins will never get a chance to be sampled. The solution accuracy is greatly influenced by the width of bins and highly accurate solutions can only be achieved by setting enough number of bins.

With surrounding effect, if an individual in a certain bin No.*i* is selected, not only the No.*i* bin gets an improvement on its height, but the surrounded bins, i.e. the No.(*i*+1) and No.(*i*-1) bins, also get the minor improvements (always the height of its surrounded bin times the *surrounding factor*) on their heights respectively. Using the surrounding effect, those bins with heights of zero have the opportunity to be sampled. Furthermore, it has been shown in our previous work that the HEDA with surrounding effect can find the best bins near the current sampled bins with hill-climbing during the search process. That ensures the algorithm to find the optimal bin with small population, even when the number of bins is large.

For the shrinking strategy, if the height of the highest bin of a variable is over the threshold value, the domain of that variable will shrink to the domain of that highest bin, and the new domain will be divided into bins as the initial step of the algorithm. Since the searching space gradually shrinks, using the shrinking strategy will make the solution accurate enough.

Experimental results have already shown that the HEDA combining both the surrounding effect and the shrinking strategy performs excellently in continuous optimization, especially, in those problems with multiple local optima. [9]

Marginal histogram-based estimation of distribution algorithm is frequently applied in practice for its simplicity and efficiency. However, the obvious drawback of the marginal probability estimation is that it loses the ability to detect any variable

linkages of the problem. This drawback is serious in the problems that variables are strongly dependent with each other.

In the following, we supply two solutions for this problem. The first is by applying probabilistic graphical model; the second is by applying a space transformation.

3 HEDA Based on Probabilistic Graphical Models

Applying probabilistic graphical models (PGM) in HEDA was first put forward by P.A.N. Bosman in his IDEA-H [4] (If we regard HEDA as a special case of discrete EDAs, the history of applying probabilistic graphical model is even longer, for example, see [3].) In the analysis of the IEDA-H, Bosman regarded that the computational complexity of the IDEA-H grows exponentially with the maximum number of the variables that a variable conditionally depends on, which becomes the bottleneck of the IDEA-H. Recently, an accelerated algorithm, which is called the dHEDA, with polynomial complexity dictated by the size of the population, was proposed in [10]. Since the computational complexity is the main concern for HEDA based on PGM, we will discuss the dHEDA in details.

3.1 General Framework

In general, the HEDA based on PGM share the same framework as any other HEDAs, except that it iteratively updates and samples from $P(Z_0,...,Z_{l-1})$ under the assumption that:

$$P(Z_0,...,Z_{l-1}) = P(Z_{j_{l-1}}) \bullet \prod_{i=0}^{l-2} P(Z_{j_i} \mid Z_{\pi(j_i)_0},...,Z_{\pi(j_i)_{|\pi(j_i)|-1}}) \tag{2}$$

In Eq.(2), $\{Z_0,...,Z_{l-1}\} = \{Z_{j_0},...Z_{j_{l-1}}\}$, but they are in different order; while $\{Z_{\pi(j_i)_0},...,Z_{\pi(j_i)_{|\pi(j_i)|-1}}\} \subseteq \{Z_{j_{i+1}},...Z_{j_{l-1}}\}$, $|\pi(j_i)| \le k$. Z_i $(i = 1,...,l)$ is the discrete random variable which represents the bin indices that take values from $\{1,\ldots,n_b\}$.

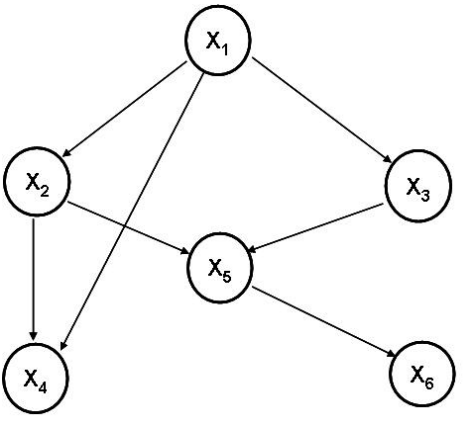

Fig. 1. $P(x_1,...,x_6) = P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_1)P(x_4 \mid x_1,x_2)P(x_5 \mid x_2,x_3)P(x_6 \mid x_5)$

That is to say, the probabilistic graphical model simplifies the complete factorized probability function by assuming the Markov properties of the variables, see Fig.1. as an example. In the expansion of the complete probability, each variable is regarded to be conditionally independent of any other variables given its parent variables (the nodes with an edge pointing to the current node in the graph, for example, x_1 and x_2 are both parent variables of x_4). Interested readers may consult to [19] for more knowledge about PGM.

Then the core of the HEDA based on PGM is to estimate each item in the right hand side of Eq.(2). In each generation, there are mainly two jobs to estimate the PGM:

1. The probability density structure (PDS) $\left\{Z_{j_i}, Z_{\pi(j_i)_0}, ..., Z_{\pi(j_i)_{\pi(j_i)|-1}}\right\}$ have to be found.
2. The probability density function (PDF) $\left\{P(Z_{j_i} \mid Z_{\pi(j_i)_0}, ..., Z_{\pi(j_i)_{\pi(j_i)|-1}})\right\}$ and $P(Z_{j_{l-1}})$ have to be calculated.

In the following discussion of this section, more details of the above two jobs in the dHEDA are presented.

3.2 PDS Search

In each generation, the PDS $\left\{Z_{j_i}, Z_{\pi(j_i)_0}, ..., Z_{\pi(j_i)_{\pi(j_i)|-1}}\right\}$ $(i = 0, ..., l-1)$, is learnt from the selected individuals. To minimize the difference between Eq.(2) and the complete factorized probabilistic function, like the IDEA-H[4], the dHEDA minimize the following expression when using the Kullback-Leibler (K-L) divergence as a metric:

$$J(Z_0, ..., Z_{l-1}) = H(Z_{j_{l-1}}) + \sum_{i=0}^{l-2} H(Z_{j_i} \mid Z_{\pi(j_i)_0}, ..., Z_{\pi(j_i)_{\pi(j_i)|-1}}) \tag{3}$$

where $H(\bullet)$ denotes the entropy. For variable i with k parents, the *conditional entropy* is calculated by:

$$H(Z_i \mid Z_{\pi(i)_0}, ..., Z_{\pi(i)_{k-1}}) = H(Z_i, Z_{\pi(i)_0}, ..., Z_{\pi(i)_{k-1}}) - H(Z_{\pi(i)_0}, ..., Z_{\pi(i)_{k-1}}) \tag{4}$$

If the variables are of n_b bins, the joint entropy of $\{Z_0, ..., Z_{n-1}\}$ is calculated by:

$$H(Z_0, ..., Z_{n-1}) = -\sum_{z_0=0}^{n_b-1} ... \sum_{z_{n-1}=0}^{n_b-1} P_{Z_0, ..., Z_{n-1}}(z_0, ..., z_{n-1}) \ln(P_{Z_0, ..., Z_{n-1}}(z_0, ..., z_{n-1})) \tag{5}$$

where $P(Z_0 = z_0, ..., Z_{n-1} = z_{n-1})$ is equal to the summation of the probability factors of those selected individuals that fall into the *super-bin* $(Z_0 = z_0, ..., Z_{n-1} = z_{n-1})$ in the dHEDA. Note that we introduce the new term *super-bin* to avoid confusion with the concept of *bin* in marginal case. The concept of *super-bin* is no different from that of *bin* except that *super-bin* denotes the bin in more than 1 dimension. We also use $(z_0, ..., z_{n-1})$ as a short hand of $(Z_0 = z_0, ..., Z_{n-1} = z_{n-1})$ in the following.

In order to find the promising PDS, greedy methods can be employed to minimize $J(Z_0,...,Z_{l-1})$. It has been verified that without considering the calculation cost of Eq.(3), we can minimize $J(Z_0,...,Z_{l-1})$ using greedy methods with polynomial computational complexity. For example, Bosman et al. proposed 3 methods for 3 different general structures (chain structure, tree structure and Bayesian Network structure). These graphical models are successful applied in Gaussian-based EDA [4] and exhibit acceptable polynomial computational complexity.

3.3 Computation Issue on the K-L Divergence

The computation issue on the K-L divergence $J(Z_0,...,Z_{l-1})$ is important because it is the metric to lead the PDS search. It is clear from Eq.(3)(4) that the core of calculating $J(Z_0,...,Z_{l-1})$ is to calculate the joint entropy. In other words, if the entropy calculation is efficient, it will be also efficient to obtain $J(Z_0,...,Z_{l-1})$. However, the obstacle in histogram-based EDA is that the computational complexity of direct calculation of $J(Z_0,...,Z_{l-1})$ using Eq.(5) is unaffordable, because it would take exponential complexity on the maximum number of the parents nodes over all the nodes.

However, it is noticed that the probability of the super-bin $P(Z_0 = z_0,...,Z_{n-1} = z_{n-1})$ is non-zero if and only if there are selected individuals in the current generation falling into this super-bin. Since there are only the number of selected individuals n_{best} for PGM estimation, given any $\left\{Z_i, Z_{\pi(j_i)_0},...,Z_{\pi(j_i)_{|\pi(j_i)|-1}}\right\}$, there exist only at most n_{best} super-bins with non-zero probability.

According to Eq.(5), the value of the joint entropy is only contributed by those non-zero super-bins. Thus, without altering the result, the expression of joint entropy is rewritten as:

$$H(Z_0,...,Z_{n-1}) = -\sum_{i=1}^{N} P_{Z_0,...,Z_{n-1}}(z^i{}_0,...,z^i{}_{n-1}) \ln(P_{Z_0,...,Z_{n-1}}(z^i{}_0,...,z^i{}_{n-1})) \quad (6)$$

In Eq.(6), $N \le n_{best}$ indicates the number of non-zero super-bins, and $P_{Z_0,...,Z_{n-1}}(z_0,...,z_{n-1})$ is the probability (height) of super-bin $(z_0,...,z_{n-1})$ which at least one of the selected individuals falls in. The reason of $N \le n_{best}$ is because there might be more than one individual falling into the same super-bin.

So, the process to calculate the entropy of certain joint variables is as following. Initially, $H(Z_0,...,Z_{n-1}) = 0$. We first pick an unpicked individual; gain its super-bin $(z^i{}_0,...,z^i{}_{n-1})$; check if $(z^i{}_0,...,z^i{}_{n-1})$ exists in the memory if yes we improve the height of super-bin $(z^i{}_0,...,z^i{}_{n-1})$, if no we create a new super-bin $(z^i{}_0,...,z^i{}_{n-1})$ and improve its height. After all individuals are picked; we sum up the probability (heights) of the super-bins in the memory times its logarithms and get the entropy.

Now, let us analyze the computational complexity of the calculation of $H(Z_i, Z_{\pi(i)_0},...,Z_{\pi(i)_{k-1}})$. The step to find those individuals that belong to the same bin, and to sum up their improvements to get the height of that bin takes $O[n_{best}^2 \bullet (k+1)]$.

Thus, the complexity of $O[n_{best}^2 \bullet (k+1)]$ is taken to calculate $H(Z_i, Z_{\pi(i)_0}, ..., Z_{\pi(i)_{k-1}})$. Overall, the complexity of $O[n_{best}^2 \bullet l^2]$ is taken to calculate $J(Z_0, ..., Z_{l-1})$.

Overall, we can conclude that we are able to find the PDS with polynomial time applying the above method to calculate entropies. This is much tractable than the way that directly applies Eq.(5) to calculate entropies.

3.4 PDF Calculation

It is noticed that all the PDFs are the byproducts of entropy computation. Thus, there is no more extra step for PDF calculation if we save their results during the PDS search.

So far, we have introduced the main framework of the HEDA based on probabilistic graphical model, especially the details of entropy computation because of its importance in computational complexity.

There is something more that needs to be remarked: it seems tricky that the probabilities of many super-bins are equal to zero. Does that mean the later-generation individuals will never get a chance to be sampled in those zero-probability bins (which is obviously unreasonable for its lack of general knowledge)? This is not the case. In fact, in the first step, the sampling process chooses the bins only according to the height of the bin, which means those zero-height bins cannot be chosen; but in the second step, the real-value of the individuals can be sampled out of the chosen bins if we applies a variant surrounding effect. Interested readers might refer to [10] for details.

3.5 Experimental Results

We design two experiments: the first is to test the computational efficiency of the accelerated algorithm; the second is to test the performance of the HEDA based on PGM on several benchmark continuous optimization problems.

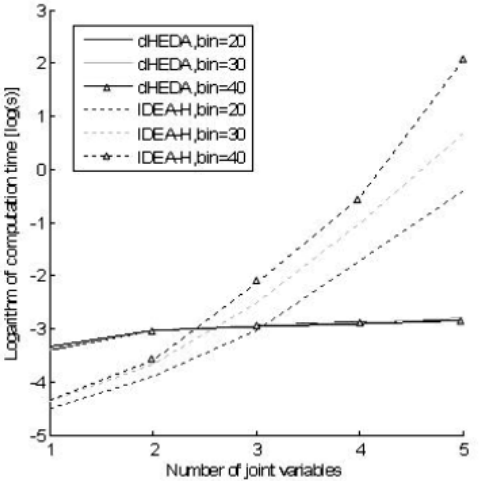

Fig. 2. The computation time to calculate the joint entropy by the dHEDA and the IDEA-H

First, let us examine the computational time of the dHEDA and that of the IDEA-H. The experimental results are shown in fig.2.

The number of the bins is respectively 20, 30, 40; the number of the variables in the joint entropy is from 1 to 5; and the number of the selected individuals is 100. The reason that we only test at most 5 joint variables is because our PC memory excesses when we try to run the test of 6 joint variables by the IDEA-H with 40 bins. 20 times are run before the average time is calculated. The experiment is based on CPU-Intel Pentium 2.8GHz.

In Fig.2, we notice that the time by the IDEA-H grows remarkably with the increase of the number of the joint variables. If we check the time in the case of bin=40 in TABLE-IV, the time of 5 joint variables entropy calculation by the IDEA-H is over 100s. On the other hand, the time by the dHEDA increases slowly and remains as a small amount of time ($\approx 10^{-3}$ s).

The other point is the great difference between them in the computational time with the growing number of the bins: the time by the IDEA-H has a distinct grow, but the time by the dHEDA hardly rises.

These experimental results are concordant to our earlier analysis. Recall that the computational complexity for calculating $H(Z_i, Z_{\pi(i)_0}, ..., Z_{\pi(i)_{k-1}})$ is $O[(n_b)^{k+1}]$ for IDEA-H, but only $O[n_{best}^2 \bullet (k+1)]$ for the dHEDA.

Next, we examine the performance of HEDA based on PGM with three other continuous EDAs: the UMDA$_c$, the IDEA-G [4], and the sur-shr-HEDA [9] on 4 benchmark continuous problems: Sphere Function, Summation Cancellation Function, Schwefel-1 Function, and Schwefel-2 Function.

Table 1. Parameter Settings of the 4 Algorithms

	Parameter Settings	
UMDA$_c$	Maximal evaluation number=3×10^5	
	Population size=375	
	Select-rate=20%	
IDEA-G	Maximal evaluation number=3×10^5	
	Population size=375	
	Select-rate=20%	
sur-shr-HEDA	Maximal evaluation number=3×10^5,	Select rate=20%
	Population size=375,	Mutation rate=5%
	Bins number=99,	Surrounding factor=10%
dHEDA	Maximal evaluation number=3×10^5,	Select rate=20%
	Population size=375,	Mutation rate=5%
	Bins number=99,	Surrounding factor=10%

Among the 4 algorithms, the $UMDA_c$ and the IDEA-G are based on Gaussian distribution, the sur-shr-HEDA and the dHEDA are based on histogram model; the $UMDA_c$ and the sur-shr-HEDA are marginal HEDAs, the IDEA-G and the dHEDA are HEDA based on PGM. Note that we use the dHEDA to substitute the IDEA-H because of the computational concern. The settings of the 4 algorithms are listed in Table 1.

Among the 4 problems, Sphere Function and Summation Cancellation Function are unimodal, while the other 2 are multimodal. The Sphere Function and Schwefel-1 Function are separable, while the other 2 are non-separable. The problems are listed in Table 2.

Table 2. Problem Descriptions

	Representation
Sphere	$F(\vec{y}) = \sum_{i=0}^{l-1} y_i^2$
Summation Cancellation	$F(\vec{y}) = 1/\left(10^{-5} + \sum_{i=0}^{l-1} \lvert X_i \rvert\right)$ where $X_0 = y_0$, $X_i = y_i + X_{i-1}$
Schwefel-1	$F(\vec{y}) = \sum_{i=0}^{l-1} -y_i \bullet \sin(\sqrt{\lvert y_i \rvert})$
Schwefel-2	$F(\vec{y}) = \sum_{i=0}^{l-1} \left[(y_i^2 - y_0)^2 + (y_i - 1)^2 \right]$

	Domain	Dimension	Type	Optimum
Sphere	[-100,100]	l=30	Min	0
Summation Cancellation	[-3,3]	l=10	Max	10^5
Schwefel-1	[-500,500]	l=30	Min	-12569.5
Schwefel-2	[-5,5]	l=20	Min	0

Each of the algorithms is run for 20 times for each problem, and the mean value, the best case, and the standard deviation for the 20 runs are collected in Table 3.

According to the results in Table 3, we can mainly summarize two points as follows:

The first is that the histogram-based EDAs outperform the Gaussian-based EDAs on the above multimodal problems. For example, in the Schwefel-1 Function (which is always regarded as a typical multimodal test problem), both of the histogram-based EDAs are able to achieve the optimum of the problem, while the Gaussian-based EDAs both fail. The excellent performance made by the histogram-based EDAs agrees to our earlier remarks: it is straightforward for the histogram model to estimate the multimodal distribution. Meanwhile, since the Gaussian-based EDAs are unimodal, their models are too rough to estimate the multimodal problems efficiently.

Table 3. Experimental Results

		Best Case	Mean Value	Standard Dev.
Sphere	$UMDA_c$	2.816e-84	1.276e-83	1.064e-83
	IDEA-G	3.423e-161	2.562e-160	2.992e-160
	sur-shr-HEDA	5.872e-15	6.753e-15	7.698e-16
	dHEDA	5.104e-19	7.496e-19	1.304e-19
Summation Cancellation	$UMDA_c$	8.635	5.798	2.403
	IDEA-G	66199	12805	20704
	sur-shr-HEDA	99998	93973	22016
	dHEDA	10^5	10^5	2.031e-2
Schwefel-1	$UMDA_c$	-11622.8	-10637.5	6.289e2
	IDEA-G	-5277.4	-4719.6	2.976e2
	sur-shr-HEDA	-12569.5	-12569.4	1.458e-1
	dHEDA	-12569.5	-12569.5	1.415e-7
Schwefel-2	$UMDA_c$	8.131e-5	2.576e-2	1.984e-2
	IDEA-G	4.324e-5	1.543e-2	1.652e-1
	sur-shr-HEDA	3.793e-19	3.145e-4	1.403e-2
	dHEDA	1.987e-21	3.161e-8	1.426e-7

The second is the contrast between the algorithms based on PDS without linkage and the ones based on PDS with linkage. It appears in the experiment that the dHEDAs outperforms the sur-shr-HEDA in both of the non-separable problems. The IDEA-G outperforms the $UMDA_c$ in the unimodal non-separable problems, but the two algorithms gain the comparable results on those multimodal non-separable problems. In the Summation Cancellation Function, for example, the IDEA-G's mean value is above 10^4 while the $UMDA_c$ is below 10^1 ; and the dHEDA succeeds all of the times but the sur-shr-HEDA performs unstably with large deviations. This fact clearly illustrates the meaning of the learning of the linkage information in the IDEA-G and the dHEDA.

4 HEDA Based on Space Transformation

Although it has already been introduced the accelerated method in probabilistic graphical models, the complexity burden is comparably large in the HEDA. One much simpler method is to deal with the linkages in marginal case.

The method is based on the space transformation, especially the linear transformation. P. Pošík in [17,18] proposed several successful evolutionary algorithms on the use of space transformation. We will introduce the transformation according to the covariance-matrix of the samples in this section and apply the method in the HEDA. We call that algorithm to be the Marginal Estimation of Distribution Algorithm in

Characteristic Space of Covariance-Matrix (CM-MEDA). Intrinsically, it is equal to preprocessing the samples by PCA in EDA. Similar work can also been found in [15].

4.1 Covariance-Matrix of the Samples

We first review several basic and well-known properties of Covariance-Matrix. The Covariance-Matrix of a set of samples is defined as follows:

Given a set of *m* samples $\vec{x_i}$, where $\vec{x_i} = (x_{i1}, x_{i2}, ..., x_{il})^T$

The Covariance-Matrix is defined as

$$C = \frac{1}{m-1}\sum_{i=1}^{m}(\vec{x_i} - \vec{\bar{x}_i})(\vec{x_i}^T - \vec{\bar{x}_i}^T) \tag{7}$$

And it has the following important properties:

1. The Covariance-Matrix is real and symmetric, that is:
$$c_{ij} \in R \ (\forall i, j = 1, ..., l) \text{ and } C = C^T$$
2. The Covariance-Matrix is diagonalizable, that is:
$$\exists Q, \ s.t. \ C = Q[\lambda]Q^{-1}$$
$$\text{where } [\lambda] = diag(\lambda_1, \lambda_2, ..., \lambda_l)$$
3. All the eigenvectors with different eigenvalues are orthogonal with each other.
4. There exists a matrix $P = (\vec{p_1}, \vec{p_2}, ..., \vec{p_l})$, which satisfies $C = P[\lambda]P^{-1}$ and all $\vec{p_i}$ are orthogonal with each other, that is $P^T = P^{-1}$.

Finally, we formulate some important representations. The space defined in *P* is the characteristic space of the Covariance-Matrix *C*, and each p_i is the basis of the space. Clearly, according to Property 2 and Property 4, we have $C = P^T[\lambda]P$, where $[\lambda] = diag(\lambda_1, \lambda_2, ..., \lambda_l)$.

4.2 General Framework

The CM-MEDA is an algorithm which estimates the marginal distribution of the problem on a transformed space: the characteristic space of the Covariance-Matrix of the selected samples.

The algorithm begins with an initialized population that is generated randomly. In each generation, it evaluates the fitness of each sample and selects several promising samples. The above two step has no difference from any other evolutionary algorithms.

Then the CM-MEDA calculates the Covariance-Matrix *C* on the selected samples and then calculates the matrix *P*, which defines the characteristic space of the Covariance- Matrix *C*.

After *P* is obtained, the selected samples are transformed to the new space by multiplying P^T, so we can obtain the value on each variable in the new coordinate space. Assuming a selected sample of *x*, this step is finished by $\vec{x'} = P^T \cdot \vec{x}$.

The marginal distributions are estimated in the new space. After the marginal distributions of the variables in the new space have been estimated, new $\vec{\vec{x}}'$ are sampled according to the estimated distribution in each variable.

At last, we transform the samples back into the initial coordinate space, according to $\vec{\vec{x}} = (P^{-1})^T \cdot \vec{\vec{x}}' = P \cdot \vec{\vec{x}}'$.

Note that when we transform the samples back, some samples might out of constraints. If that happens, we discard those "illegal" samples and resample the new ones. After we have built the new population, we evaluate the fitness and go to the next generation.

In short, the framework of CM-MEDA is as followed:

1. Initialize the population
2. Select the samples with high fitness
3. Calculate the Covariance-Matrix C of the selected samples and build the matrix P
4. Transform the selected samples onto the characteristic space of the Covariance-Matrix by $\vec{x}' = P^T \bullet \vec{x}$
5. Estimate the marginal models on the new space according to the distribution of the transformed selected samples
6. Make new samples $\vec{\vec{x}}'$ according to the marginal models in the transformed space
7. Transform the new samples from the transformed space back to the original space by $\vec{\vec{x}} = P \cdot \vec{\vec{x}}'$ and check if all the new samples are legal. Resample the illegal samples.
8. Return to step 2 if not terminated.

4.3 CM-MEDA in Histogram Case

In the histogram-based CM-MEDA, the marginal distribution in the transformed space is based on histogram model. There is only one specific concern besides the general framework of the CM-MEDA in the histogram case, that is: How to decide the domain (i.e. upper bound and lower bound) of the histogram model in the transformed space.

We solve this problem in a simple way: the domain of each variable is decided according to the range of the selected samples in each variable. Note that, under this way the algorithm can naturally shrink the domain of the variables. Besides, to avoid loss of generality, we make the domain of the variable be a little larger than the range of the variables. In our algorithm, there is one more bin to the left of the leftmost sample and one more bin to the rightmost sample. The leftmost sample is in the center of the bin it belongs to, and so is the rightmost sample. See Fig.3 as an illustration.

4.4 Relation to Principal Component Analysis

Principal component analysis (PCA) is a method that seeks for a projection which best represents the samples in a least-squares sense [20]. It intrinsically provides a promising way of finding the projections which mostly capture the variance of the

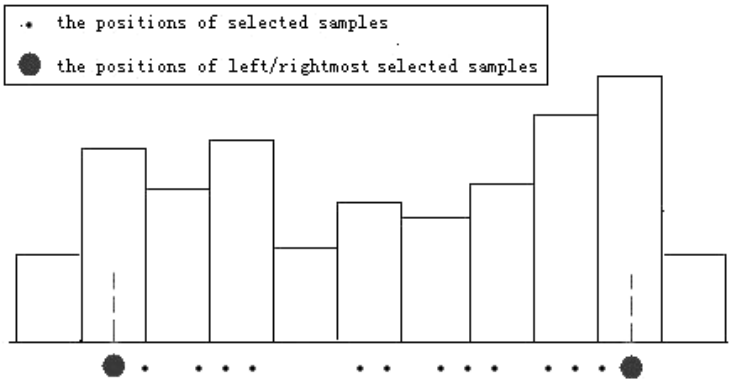

Fig. 3. The position of the selected samples and the domain of the bins. Note that the leftmost and the rightmost selected samples are in the center of the bins that they belong to. And there is one more bin to the left of the leftmost sample and one more bin to the right of the rightmost sample.

samples. For example, the first component (the principal component) is the projection that the samples have the maximum variance; the second component is the projection with the maximum variance in the subspace orthogonal to the first component, and etc.

Since the components given in PCA are the same as the eigenvectors of Covariance-Matrix in the CM-MEDA, we can alternatively think that the core of the CM-MEDA is to find the projections which maximize the variance in the space iteratively, because each eigenvector is the projection which has the maximum variance of the samples in the subspace orthogonal to the eigenvectors with higher eigenvalues. The projection which has the maximum variance of samples is the one which mostly needs estimation in evolutionary optimization, because its ultimate aim is to find one optimal point not a scatter of the samples.

Thus, we infer that the CM-MEDA can improve the convergence of the EDA by making distribution estimation on the set of axes which maximize the variance of the samples from the viewpoint of PCA. Similar work may also refer to [15,16,17].

According to the framework above, we can find that the CM-MEDA has only three more steps than the MEDA: calculation of covariance matrix, calculation of eigenvectors, and transformation. The calculation of covariance matrix takes $O(n_{best} \bullet l^2)$, where n_{best} is the number of selected samples, and l is the dimension of the samples. The eigenvectors can be efficiently calculated in $O(l^3)$. And the transformation takes $O(l^2)$. Therefore, it is clear that the CM-MEDA is much faster than the HEDA based on PGM, even the dHEDA. (Recall that it takes $O({n_{best}}^2 \bullet l^2)$ to calculate K-L divergence in the dHEDA.)

However, we also note that the CM-MEDA can only accurately capture the linear linkages; while the non-linear linkages are always approximated intrinsically. In contrast, the HEDA based on PGM is able to capture more complex relations among variables if less restrictions on its PDS.

4.5 Experimental Results

We provide a brief experiment to compare the CM-MEDA and MEDA in histogram case on 4 continuous optimization problems.

The general settings of the two algorithms are same: the population size is 1000; 20% of the individuals are selected from the population to estimate the model; the number of the bins in each bin is arbitrarily set to be 100; 40 iterations are run each time. The MEDA use the mechanism of the sur-shr-HEDA to update the histogram model. The surrounding factor is set to be 10%, and the mutation rate is set to be 5%.

The 4 test functions: Fun-1, Fun-2, Schwefel, Rosenbrock are all non-separable and share a similar formulation. The reason that we choose several functions in this

Table 4. Problem Descriptions

	Representation
Fun-1	$Fun\_1(x_1,...x_n)=\sum_{i=1}^{n-1}100\bullet(x_{i+1}-1.3\bullet x_i)^2+(1-x_1)^2$
Fun-2	$Fun\_2(x_1,...x_n)=\sum_{i=1}^{n-1}[(x_{i+1}-x_i^2)^2+(1-x_i)^2]$
Schwefel	$Schwefel(x_1,...x_n)=\sum_{i=1}^{n}[(x_1-x_i^2)^2+(1-x_i)^2]$
Rosenbrock	$Rosenbrock(x_1,...x_n)=\sum_{i=1}^{n-1}[100\bullet(x_{i+1}-x_i^2)^2+(1-x_i)^2]$

	Domain	Dimension	Type	Optimum
Fun-1	[-5,5]	10	Min	0
Fun-2	[-5,5]	10	Min	0
Schwefel	[-5,5]	10	Min	0
Rosenbrock	[-5,5]	10	Min	0

Table 5. Experimental Results

		CM-MEDA	MEDA
Fun-1	Best	0.1775	1.7321
	Mean	0.2760	3.4524
	Std.	0.0801	0.7483
Fun-2	Best	3.879e-11	0.0319
	Mean	0.0015	0.0571
	Std.	0.0057	0.0112
Schwefel	Best	1.320e-10	0.0378
	Mean	0.0152	0.0682
	Std.	0.0359	0.0184
Rosenbrock	Best	6.5271	4.2003
	Mean	7.2160	8.0493
	Std.	0.3181	1.4204

formulation is because this formulation of functions is usually hard for Evolutionary Algorithms to solve because of their strong linkages between variables. The problem descriptions are listed in Table 4.

Each of the algorithms is run for 20 times in each problem, and the mean value, the best case, and the standard deviation for the 20 runs are collected in Table 5.

The results in Table 5 verify the superiority of the CM-MEDA in histogram case in those non-separable cases. For example, when solving Schwefel function, the best solution obtained by the CM-MEDA in 20 runs is 1.320e-10, while the best solution of the MEDA is 0.0378; the mean value of the CM-MEDA is 0.0015, while the mean value of the MEDA is 0.0682.

5 Summary and Further Work

In this chapter, we have reviewed the two methods in HEDA which are able to detect linkage among variables in the optimization problems. Using probabilistic graphical model generally represents the relations more accurately but also is of high-cost; while using space transformation is quite fast, but it only roughly represents the major possible relationships.

There is, however, still much further work in both topics.

The first, for the HEDA, the number of bins in certain problems is always decided arbitrarily in most papers. An alternative method might apply the prior on the bins, such as Dirichlet Process, and then use Bayesian analysis in estimating the model.

The second, for the HEDA based on PGM, it is still interesting to have deeper research on dealing with the case of small number of samples with great number of bins in a joint probability.

The third, for the HEDA based on space transformation, to find a more efficient transformation, or to define the kernel space might allow the algorithm to capture more complex and non-linear linkages.

The fourth, there is little research on comparing the two method above directly. Besides, the attempts to combine the above two methods might have even improved performance, since the space transformation might reduce the number of parents of each nodes in the graphical model.

References

[1] Larranaga, P., Lozano, J.A.: Estimation of Distribution Algorithms: A NewTool for Evolutionary Computation. Kluwer Academic Publishers, Dordrecht (2002)

[2] Pelikan, M.: Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms. Springer, Heidelberg (2005)

[3] Pelikan, M., Goldberg, D.E., Tsutsui, S.: Getting the best of both worlds: Discrete and continuous genetic and evolutionary algorithms in concert. Information Science 156, 147–171 (2003)

[4] Bosman, P.A.N., Thierens, D.: An algorithmic framework for Density Estimation based Evolutionary Algorithms. Utrecht University technical report UU-CS-1999-46 (1999)

[5] Bosman, P.A.N., Thierens, D.: Numerical optimization with real- valued Estimation of Distribution Algorithm. Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications, 91–120 (2006)
[6] Lu, Q., Yao, X.: Clustering and learning Gaussian distribution for continuous optimization. IEEE trans. on Systems, Man and Cybernetics-Part C 35(2), 195–204 (2005)
[7] Tsutsui, S., Pelikan, M., Goldberg, D.E.: Evolutionary algorithm using marginal histogram models in continuous domain. In: 2001 Genetic and Evolutionary Computation Conference Workshop, San Francisco, CA, pp. 230–233 (2001)
[8] Ding, N., Zhou, S., Sun, Z.: Optimizing continuous problems using Estimation of Distribution Algorithms based on histogram model. In: The 6th International Conference of Simulated Evolution and Learning, Hefei, China, pp. 545–562 (2006)
[9] Ding, N., Zhou, S., Sun, Z.: Histogram-based Estimation of Distribution Algorithm: a competent method for continuous optimization. J. Computer Science and Technology 23(1), 35–42 (2008)
[10] Ding, N., Xu, J., Zhou, S., Sun, Z.: Reducing Computational Complexity of Estimating Multivariate Histogram-based Probabilistic Model. In: 2007 IEEE Congress on Evolutionary Computation, pp. 111–118 (2007)
[11] Yuan, B., Gallagher, M.: Playing in continuous spaces: Some analysis and extension of population-based incremental learning. In: IEEE Congress on Evolutionary Computation, Canberra, Australia, pp. 443–450 (2003)
[12] Baluja, S.: Population-based incremental learning. Carnegie Mellon University, Technical Report. CMU-CS-94-163 (1994)
[13] Cantu-Paz, E.: Supervised and Unsupervised Discretization Methods for Evolutionary Algorithm. In: The Genetic and Evolutionary Computation Workshop, San Francisco, CA, pp. 213–216 (2001)
[14] Zhang, Q., Sun, J., Tsang, E., Ford, J.: Hybrid Estimation of Distribution Algorithm for Global Optimization. Engineering Computations 21(1), 91–107 (2004)
[15] Zhang, Q., Allinson, N.M., Yin, H.: Population Optimization Algorithm Based on ICA. In: the First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks (2000)
[16] Zhang, Q., Mühlenbein, H.: On the convergence of a class of estimation of distribution algorithms. IEEE Trans. Evolutionary Computation 8(2), 127–136 (2004)
[17] Pošík, P.: On the Utility of Linear Transformations for Population-Based Optimization Algorithms. In: the 16th World Congress of the International Federation of Automatic Control (2005)
[18] Pošík, P.: On the Use of Probabilistic Models and Coordinate Transforms in Real-Valued Evolutionary Algorithms. PhD Dissertation, Czech Technical University (2007)
[19] Lauritzen, S.L.: Graphical Models. Clarendon Press, Oxford (1996)
[20] Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley Interscience, Chichester (2000)