Cross Entropy and Adaptive Variance Scaling in Continuous EDA

Cai Yunpeng
caiyp78@gmail.com

Sun Xiaomin
sxm123@tsinghua.edu.cn

Xu Hua
xuhua@tsinghua.edu.cn

Jia Peifa
dcsjpf@tsinghua.edu.cn

State Key Lab of Intelligent Technology and Systems,
Tsinghua University, Beijing, P.R.China

ABSTRACT

This paper deals with the adaptive variance scaling issue in continuous Estimation of Distribution Algorithms. A phenomenon is discovered that current adaptive variance scaling method in EDA suffers from imprecise structure learning. A new type of adaptation method is proposed to overcome this defect. The method tries to measure the difference between the obtained population and the prediction of the probabilistic model, then calculate the scaling factor by minimizing the cross entropy between these two distributions. This approach calculates the scaling factor immediately rather than adapts it incrementally. Experiments show that this approach extended the class of problems that can be solved, and improve the search efficiency in some cases. Moreover, the proposed approach features in that each decomposed subspace can be assigned an individual scaling factor, which helps to solve problems with special dimension property.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*global optimization*

General Terms

Theory, Algorithms

Keywords

Evolutionary Computation, Estimation of Distribution Algorithms, Adaptive Variance Scaling, Cross Entropy

1. INTRODUCTION

In recent years there has been growing interest to extend Estimation of Distribution Algorithms [16, 13, 22, 15] into continuous optimization domain [24, 27, 12, 21, 4, 18, 28, 1]. Continuous optimization problems are essentially different to combinatory ones in the sense that neither the state space nor the dependency between variables is considered enumerable. It has been discovered that the scaling of model parameters, i.e., parameter fitting, is more complicated in continuous EDAs than it was in discrete cases. Most continuous EDA adopts Gaussian probablistic density function (*pdf*) as probabilistic models. It had been proved by both theoretical studies and experiments that [8, 10, 29, 19] fitting the center and variances/covariance of the model merely according to the distribution of current population might cause premature convergence. Adaptive variance scaling (AVS) has been introduced to continuous EDAs to cope with this situation, and many successful results have been reported [19, 3, 29, 9].

Current approaches of adaptive variance scaling in EDA follows the so-called 1/5-success-rule [23], which is a key concept borrowed from evolutionary strategies (ES). However, Unlike in ES where variance adaptation determines the parameters alone, in continuous EDA, standard model-building procedure provides a basic scaling of parameters, and variance adaption serves merely as a secondary procedure to improve the scaling. Mixing the two procedures causes some problems. The 1/5-success-rule punishes unsuccessful evolution by shrinking the variance, however, in EDA poor evolution progress might be caused by improper shaping of models rather than bad scaling, shrinking the scaling of variance is sometimes misleading and degrades the evolution performance, even causes premature convergence, which will be illustrated in our experiments.

In this paper we propose the cross-entropy adaptive variance scaling (CE-AVS) as a new type of variance scaling method specifically for EDA, which is essentially different from previous methods, so that the above questions can be tackled. The idea is that since currently researchers have interpreted variance adaptation as compensation (e.g. [9]) to the inadequacy of the probabilistic model in EDA, we goes on to measure the bias of the model used in previous generation, and apply compensation to the next generation. By comparing the model distribution and the real distribution of the new population after competition, the inadequacy of the model is measured. A scaling factor can be derived by minimizing the cross entropy of these two distributions, and the factor is used for scaling the variance in the next generation, so that variance adaptation is achieved. Experiments

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

show that this approach extended the class of problems that can be solved, and achieve better efficiency in searching. The experiments also show the feature of CE-AVS that each decomposed subspace can be assigned an individual scaling factor, which helps to solve problems with special dimension property, e.g., the SharpRidge function and the ParaRidge function [9].

The remainder of the paper is organized as follows. Section 2 briefly reviews the adaptive variance scaling methods and their application in EDA, the problems encountered when mixing adaptive variance scaling with EDA are illustrated. In section 3 the cross-entropy adaptive variance scaling is proposed. Experimental results are given in section 4 before concluding in section 5.

2. ADAPTIVE VARIANCE SCALING

In this section we first give a review on the history of adaptive variance scaling and its application in EDA. We then present some examples showing the defects of current AVS strategy, so that the purpose of this paper is made clear.

2.1 Adaptive Variance Scaling in Evolutionary Strategy

Adaptive variance scaling is the local search of a better value for the variance or covariance matrix of a *pdf* to improve the performance of mutation in evolution algorithms. The concept comes along with the proposal of evolutionary strategy(ES). The first class of AVS strategy is the so-called 1/5-success-rule proposed by Rechenberg [23]. The strategy tunes the variance of a distribution incrementally by observing the performance of current parameters. A given variance is chosen and several mutations are performed with the variance, then the success rate of the mutations is accounted. If the success rate reaches a expected threshold, the variance is increased to encourage more exploration, otherwise the variance is decreased. In this way the variance is kept oscillating near the optimal value given that the threshold is properly chosen:

$$\begin{aligned} if\ N_{succ}/N_{muta} > p_{thres} \quad & then \quad \sigma^{g+1} = \sigma^g \cdot a \\ & else \quad \sigma^{g+1} = \sigma^g / a \end{aligned} \tag{1}$$

The second class of AVS strategy is proposed by Schwefel as Mutative Strategy Parameter Control (MSC) [26]. MSC tunes the mutate variance through competition. An individual variance is assigned to each solution. In each generation, the variance of the offspring is generated by randomly modifying the variance of their parents, and then each offspring solution is generated by mutating the parent solution with the corresponding offspring variance. The best offspring is chosen and its variance is kept to the next generation:

$$\begin{aligned} \sigma_k^{g+1} &= \sigma^g e^{\xi_k}, & \xi_k &= rand\{-1/3, 1/3\} \\ \boldsymbol{x}_k^{g+1} &= \boldsymbol{x}^g + \sigma_k^{g+1}\boldsymbol{z}_k, & \boldsymbol{z}_k &\sim N(0, \boldsymbol{I}) \end{aligned} \tag{2}$$

MSC is beneficial in making use of a larger population. Moreover, the method can be extended to adapt arbitrary normal distributions rather than isotropic ones [26].

The De-randomized Strategy Parameter Control(DSC) by Ostermeier et.al. [20] extends MSC by adopting accounting technique instead of random mutating. DSC separates the variance into two parts: the "global step" represented by a scaling factor, and the "individual step" represented by a covariance matrix. In each generation, the scaling factor is modified according to the mean distance between the successful mutated offspring and current center:

$$\begin{aligned} \sigma^{g+1} &= \sigma^g exp(\frac{||\boldsymbol{p}|| - E||N(0,\boldsymbol{I}||}{d \cdot E||N(0,\boldsymbol{I}||}) \\ \boldsymbol{x}_k^{g+1} &= \boldsymbol{x}^g + \sigma_k^{g+1}\boldsymbol{z}_k \end{aligned} \tag{3}$$

where $||\boldsymbol{p}||$ is the so-called "cumulative path length" relative to the mean distance of selected offsprings around the center. The method is implemented in CMA-ES [11] and achieved great success, but calculating $||\boldsymbol{p}||$ is a complicated and essential step in CMA-ES, thus the approach is not easy to be adopted in other algorithms.

2.2 Adaptive Variance Scaling for EDA

Traditional estimation of distribution algorithms do not involve AVS process, since it is assumed that the distribution of current population provides a proper scaling of the probabilistic model. However, experiment results have shown that fitting the Gaussian model based on the diversity of current population will sometimes cause the variance to shrink too fast [19, 29, 10], so that premature convergence will occur. Theoretical analysis shows that this phenomenon occurs when 1) the initial center of the Gaussian model is too far away from the optimum [8, 10]; 2) the function landscape is statistically deceptive in global scale [29]. It has been discovered that artificially expanding the variance is beneficial in coping with this problem [29]. Based on these results, adaptive variance scaling for EDA is proposed.

Currently, most AVS methods used in EDA are based on the 1/5-success-rule. In EDA it is not efficient to create multiple competing populations, thus MSC is not preferred. Porting DSC to EDA is rather a complicated task, which has not been investigated to the best of our knowledge. amBOA [19] and AVS-IDEA [3] are two typical continuous EDA with variance scaling. amBOA uses the adaptive rule described in Eq. 1, while AVS-IDEA modified the adaptive rule that the variance expands when the best solution is improved in the last generation and shrinks otherwise.

2.3 Defects of Current AVS Methods in EDA

The usage of AVS is rather different in EDA compared to those in ES. In ES the self-adaptation strategy determines the variance of covariance matrix directly. While in EDA, the standard model-building procedure provides a basic scaling of parameters, i.e., the basic covariance matrix Γ, and AVS provide a scaling factor c^{AVS} [9] to adjust the matrix. The combined matrix, $c^{AVS}\Gamma$, is used for sampling instead of Γ. The 1/5-success-rule observes the performance effected by the overall variance, while puts rewards solely on c^{AVS}, regardless of any possible change to Γ. This scaling style causes some negative effects. Grahl et.al. [9] pointed out that AVS slows down the search process when the problems can actually be solved without scaling, since high success rate in mutation do not always indicate that the overall variance should be increased. Correlation-Triggered Adaptive Variance Scaling (CT-AVS) is proposed to avoid the abuse of AVS.

Table 1: Relation of Structure Learning and AVS in Sphere Function

Prop	No AVS		AVS	
	α	β	α	β
1/1	3115	1.18	2292	1.17
1/8	4332	1.41	2716	1.39
1/16	4410	1.47	2966	1.49
0	4999	1.56	3103	1.71

In this paper we disclose another defect of the above scaling style. Current AVS methods not only expand the scaling factor improperly in some cases, but also shrink it improperly in some other cases. When the covariance matrix Γ is in bad shape, the success rate of the mutation will also be very low. In this case, shrinking the scaling factor will not help to resolve the improper shape, on the contrary, it will exaggerate the effect and even cause premature convergence if no precaution is taken.

The evidence can be found by repeating the experiments in [9]. It is reported in [9] that for functions that can be solved without scaling, AVS is slower, even in the sense of minimal-evaluations (or minimal convergence population size) cases. We discover that the phenomenon will not be so obvious if the algorithm could learn the problem structure perfectly, Taking the case of the Sphere Function:

$$f(x) = \sum_{i=1}^{l} x_i^2 \quad x_i^0 \in [-10, 5] \tag{4}$$

The minimal number of evaluations required by IDEA is smaller that those of IDEA-AVS. However, in our experiments we found that for UMDA, adopting AVS actually speed up the optimization on the same function in minimal-evaluations case. To test the relationship between structure learning and variance scaling, we designed a mixed, EGNA-based [12] algorithm to enable different preciseness of model structure. The mixed strategy comprises two structure-learning styles. One adopts the univariate model, which precisely describes the space decomposition of the function. The other adopts a very simple learning strategy which merely detects pair-wise correlation between variables and adds an arc for significant correlations in the Gaussian network. We modified the proportion of the two styles in evolution and test how it affects the efficiency of no-AVS and AVS algorithms. A scalability test is performed concerning the mean number of evaluations versus the problem dimension. The detailed experiment configuring follows the one later explained in section **??** except that tournament replacement is adopted instead of truncate replacement. The minimal-evaluations test result is depicted in Table. 1, where the average number of evaluation N and dimensionality l follows:

$$N = \alpha(l/10)^{\beta} \tag{5}$$

and the column "Prop" shows the proportion of applying the univariate model in evolution. For example, 1/8 means that univariate model is adopted once in 8 generations and pair-wise model is adopted in other cases.

From the result we see that with precise structure learning the algorithm with AVS has better scaling property. When the quality of structure learning decreases, the algorithm with AVS degrade more sharply and it is approached or even outperformed by the one without AVS.

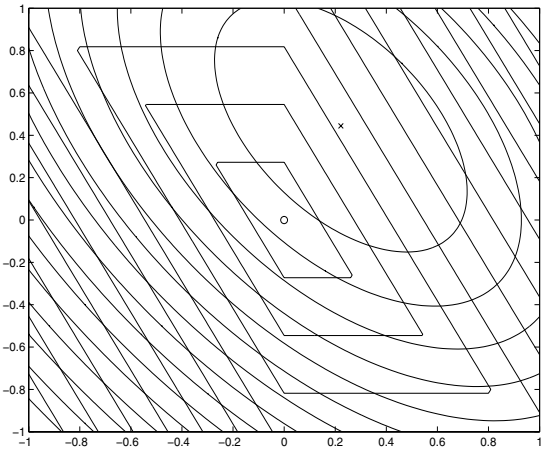

Figure 1: The shape of Gaussian model in the shifted SumCan function, the long axis is almost orthogonal to the direction of the global optimum(the circle)

It should be noted that if no structure learning is adopted, i.e., using the EMNA [14] algorithm, the AVS version will also outperform the non-AVS version. Because a badly-built but fixed structure affects both algorithms, while the fluctuation in model structure affects more on the AVS process.

This phenomenon not only degrades the evolution speed, in some situation it will even cause premature convergence. Taking the case of the shifted SumCan Funtion:

$$f(x) = \frac{1}{10^{-}5 + \sum_{i=1}^{l}|y_i|} \quad y_i = \sum_{k=1}^{i}(x_k - s)$$
$$x_i^0 \in [-0.16, 0.16] \tag{6}$$

If we choose a significant large s, so that the optimum is far from the initial region, EDAs will have to climb up the slope with the aid of adaptive variance scaling. The contour of the function is parallelogram, so that during the process samples are more likely to scatter along the sides of the obtuse edge, orthogonal to the direction of the gradient, rather than to distribute along the climbing-path, as is depicted in Figure 1. The shape of the Gaussian model become more and more compressed along the gradient direction and the rate of successful mutation decreases. The adaptation rule is then triggered and the overall variance is shrunken, however, this does not help to increase the success rate, on the contrary, new samples are generated in a more and more restricted region and the algorithm is stuck in local optimum. Artificially placing a lower bound for the scaling factor helps to alleviate the trap, however, the bound is not universal and it may degrade the performance in solving other problems.

3. CROSS ENTROPY ADAPTIVE VARIANCE SCALING

In this section we propose the cross-entropy adaptive variance scaling as a new type of AVS method to overcome the

defect of existing approaches. The method takes into account not only the success mutation rate, but also the space distribution of current population, so that the effect of structure learning is taken into account.

3.1 Basic Idea

It has been stated [9] that the role of AVS in EDA is to compensate the lack of adequacy or competence in probabilistic modeling. If the model is not competent in describing the function landscape, more random search should be preferred and a larger scaling factor should be assigned. Traditional AVS do not strictly follow this rule, instead, it strives to achieve optimal progress by balancing the mutation step length and the mutation success rate. DSC [20] tracks the motion of the model center and assign a large scaling factor if the cumulated effect of previous motions are significant, but does not involve the model structure in scaling. CE-AVS tries to go further in taking into account both the shifting of the center and the shape of the model distribution.

In EDA, the only way to measure the inadequacy of the model by observation is to compare it with the real distribution of elite solutions. For EDAs with the replacement step [18, 4], this can be partly achieved by comparing the spatial distribution of the survivors after competition and the expectation of the model. For EDAs without replacement, the task is rather complicated and we leave it unsolved in this paper.

Originally EDA expects that the model should cover the contour of the function landscape with some preciseness and evolution merely means the refining of the model. In this sense, the contour of the offspring population after competition should be a contracted version of the model contour nesting inside it. However, in continuous EDA this is rarely the case. The difference between the model and the offspring is caused by the following factors:

- Transition. In continuous EDA the model is at most a local approximation to the function contour. With the evolution progresses the location of the population will change (perhaps approaching the optimal region), and the function contour changes with it.

- Model Defects. Before an optimal region is discovered, the model will not bias to it, but with random sampling some elite solution might appear within it and after selection or replacement these solution are emphasized. EDAs without AVS do react to this situation but the modification to the model is sometimes insufficient.

- Randomness. Statistical errors always exist in parameter fitting during model building. This effect is distinct from the above one in the sense that it will not lead to better fitness in average.

The former two items can attribute to the inadequacy of the model, while the latter one cannot. However, it is hard to distinguish these three factors merely by observation, so in this paper we just tries to reduce effect of the third factor by choosing a less randomized parameter fitting strategy.

After knowing the difference of the two distributions, we attempt to scale the variance of the model so that it can cover all important areas indicated by the offspring population, if they had not been previously covered. We do this with the aid of cross entropy.

Cross entropy, also known as relative entropy or the Kullback-Leibler divergence (KLD) [6], serves as a universal measure to compare two distributions:

$$D(\boldsymbol{p}(\boldsymbol{x})||\boldsymbol{q}(\boldsymbol{x})) = \int_{\Omega} \boldsymbol{p}(\boldsymbol{x}) \log \frac{\boldsymbol{p}(\boldsymbol{x})}{\boldsymbol{q}(\mathbf{x})} \tag{7}$$

The metric has been introduced to EDA to solve both structure learning [17] and parameter fitting [2, 7, 5] problems. And we suggest applying it to variance scaling.

3.2 Formulation of CE-AVS

Currently, most continuous EDAs adopt Gaussian models or Gaussian kernel models. Representing the multivariate Gaussian distribution with the following notation:

$$\boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\Gamma}, \boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Gamma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Gamma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}, \quad \boldsymbol{x} \in R^n \tag{8}$$

Where $\boldsymbol{x}$ is the optimization variables, $\boldsymbol{\mu}$ is the model center and $\boldsymbol{\Gamma}$ is the covariance matrix.

In some generation of a certain EDA algorithm, we get a parent population of solutions P. The algorithm then apply some short of selections to P, and create a Gaussian model $N_M(\boldsymbol{\mu_M}, \boldsymbol{\Gamma_M}, \boldsymbol{x})$ from it. Then, the algorithm perform random sampling according to N_M and obtain an offspring population O. In the replacement step, O competes with P and they are merged to a new population P', which comes into the next evolutionary step. No we can estimate P' by a normal distribution $N_{P'}(\boldsymbol{\mu_{P'}}, \boldsymbol{\Gamma_{P'}}, \boldsymbol{x})$. The cross entropy between N_M and N_P is calculated by:

$$\begin{aligned} &D(N(\boldsymbol{\mu_{P'}}, \boldsymbol{\Gamma_{P'}}, \boldsymbol{x})||N(\boldsymbol{\mu_M}, \boldsymbol{\Gamma_M}, \boldsymbol{x})) \\ &= \frac{1}{2} ln \frac{|\boldsymbol{\Gamma_M}|}{|\boldsymbol{\Gamma_{P'}}|} + \frac{1}{2} tr(\boldsymbol{\Gamma_{P'}}(\boldsymbol{\Gamma_M}^{-1} - \boldsymbol{\Gamma_{P'}}^{-1})) \\ &+ \frac{1}{2} tr(\boldsymbol{\Gamma_M}^{-1}(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})^T) \end{aligned} \tag{9}$$

The above equation can be found in [25]. The idea of cross-entropy AVS is that if we expand or shrink the scaling of the model N_M, the divergence between them will be reduced. We can find a scaling factor that minimizes the above divergence, which means that the adjusted model will make full use of the reinforcement information from the new population. The scaling factor can then be adopted for sampling in the next generation.

Now we add a scaling factor a to the model, the extended model is denoted by $N_M(\boldsymbol{\mu_M}, a^2\boldsymbol{\Gamma_M}, \boldsymbol{x})$. We tries to vary

a to minimize the cross entropy:

$$\begin{aligned} & D(N(\boldsymbol{\mu_{P'}}, \boldsymbol{\Gamma_{P'}}, \boldsymbol{x}) || N(\boldsymbol{\mu_M}, a^2 \boldsymbol{\Gamma_M}, \boldsymbol{x})) \\ & = \frac{1}{2} ln \frac{a^{2l} |\boldsymbol{\Gamma_M}|}{|\boldsymbol{\Gamma_{P'}}|} + \frac{1}{2a^2} tr(\boldsymbol{\Gamma_{P'}} (\boldsymbol{\Gamma_M}^{-1} - a^2 \boldsymbol{\Gamma_{P'}}^{-1})) \\ & + \frac{1}{2a^2} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})^T) \end{aligned} \quad (10)$$

where l is the problem dimension.

Taking the derivative of Eq. 10 produces:

$$\begin{aligned} & \frac{\partial}{\partial a} D(N(\boldsymbol{\mu_{P'}}, \boldsymbol{\Gamma_{P'}}, \boldsymbol{x}) || N'(\boldsymbol{\mu_M}, a^2 \boldsymbol{\Gamma_M}, \boldsymbol{x}) \\ & = \frac{\partial}{\partial a} [l \cdot lna + \frac{1}{2} ln \frac{|\boldsymbol{\Gamma_M}|}{|\boldsymbol{\Gamma_{P'}}|} + \frac{1}{2a^2} tr(\boldsymbol{\Gamma_{P'}} \boldsymbol{\Gamma_M}^{-1}) - \frac{1}{2} \\ & + \frac{1}{2a^2} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\mu}_M - \boldsymbol{\mu_{P'}})(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})^T] \\ & = \frac{l}{a} - \frac{1}{a^3} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\Gamma_{P'}} + (\boldsymbol{\mu}_M - \boldsymbol{\mu_{P'}})(\boldsymbol{\mu}_M - \boldsymbol{\mu_{P'}})^T) \end{aligned} \quad (11)$$

Forcing the above equation to zero, we derive:

$$a = \sqrt{\frac{1}{l} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\Gamma_{P'}} + (\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})^T)} \quad (12)$$

Eq. 12 is the basic equation for cross-entropy adaptive variance scaling.

3.3 Two-Stage Compositive Scaling

Eq. 12 does not function well in practice. The reason is that it takes into account mainly the effect of model defects but not model transition, thus the compensation is not sufficient. To overcome this problem, we suggested to apply a two-stage composite scaling which compensate both effect separately.

The first stage of compensation is the same as the basic step Eq. 12 described above, which deals with the biases of the model:

$$a_1 = \sqrt{\frac{1}{l} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\Gamma_{P'}} + (\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})(\boldsymbol{\mu}_M - \boldsymbol{\mu}_{P'})^T)}$$

If the model is not biased, it will happen that $\boldsymbol{\Gamma_{P'}} \simeq w \boldsymbol{\Gamma_M}$, where w is a scalar smaller than but close to 1, and $\boldsymbol{\mu_M} \simeq \boldsymbol{\mu_{P'}}$, a_1 will be nearly but smaller than 1 so that the algorithm encourages exploiting the regions around the model center.

The second stage measures the movement of the model center across generations, so that transition effect is discovered. Denoting the Gaussian model built from P' by $N'_M(\boldsymbol{\mu'_M}, \boldsymbol{\Gamma'_M}, \boldsymbol{x})$. If no transition takes place between the two generations, N_M show be the same as $\boldsymbol{\mu'_M}$ in average. Following similar process we get:

$$a_2 = \sqrt{\frac{1}{l} tr(\boldsymbol{\Gamma_M}^{-1} (\boldsymbol{\mu}_M - \boldsymbol{\mu}'_M)(\boldsymbol{\mu}_M - \boldsymbol{\mu}'_M)^T)}$$

In the above equation we ignore the transition indicated by the difference of $\boldsymbol{\Gamma_M}$ and $\boldsymbol{\Gamma'_M}$, due to the complexity in calculation. If the old model had indicated the optimal region, a_2 will be zero.

We set the overall scaling factor as:

$$a = a_1 + a_2 \quad (13)$$

The full procedure of cross-entropy can be now given by:

1. Initialize a population P, set $a = 1$;
2. Create model $N_M(\boldsymbol{\mu_M}, \boldsymbol{\Gamma_M})$;
3. Sample with $N_M(\boldsymbol{\mu_M}, a^2 \boldsymbol{\Gamma_M})$ and obtain an offspring population O;
4. Perform replacement $P + O \rightarrow P'$;
5. Model P' with $N_{P'}(\boldsymbol{\mu_{P'}}, \boldsymbol{\Gamma_{P'}})$;
6. Calculate a_1 by N_M and $N_{P'}$;
7. Create new model $N'_M(\boldsymbol{\mu'_M}, \boldsymbol{\Gamma'_M})$;
8. Calculate a_2 by N_M and N'_M;
9. Update a, $P \leftarrow P'$, $N_M \leftarrow N'_M$, go to step 3;

3.4 Implemental Details and Features of CE-AVS

In continuous EDA, in order to sample a Gaussian distribution, a Cholesky decomposition is usually performed to the covariance matrix $\boldsymbol{\Gamma} = LL^T$. If structure learning methods (such as Gaussian Networks) is involved, the decomposition has been carried out implicitly, perhaps with changing of order for variances. $tr(\boldsymbol{\Gamma_M}^{-1} \boldsymbol{\Gamma_{P'}})$ can be computed by solving two triangular linear equations $LY = \boldsymbol{\Gamma'_P}$ and $L^T X = Y$, then compute $tr(X)$. For a problem dimension m and population size m, the computational complexity for calculating the covariance matrix is $O(mn^2)$ and for matrix multiplying is $O(n^3)$, while structure learning in EDA costs at least $O(n^3)$, usually $O(n^4)$ or more, so that the additional computation burden is acceptable.

If the function landscape can be partitioned into subspaces, thus the matrix L can be partitioned, the above calculation can be performed in each subspace and summed up to a total scaling factor, or different factors can be applied to each subspace. This is sometimes (but not always) beneficial, which will be demonstrated in one of our experiments.

Unlike previous AVS methods described in section 2, CE-AVS do not rely on the historical value of the scaling factor, so that it is more beneficial in complex situations. For multimodal algorithms, in each generation, current model for each cluster can be attached to the closest cluster of the last generation, and calculate the scaling factor as in the unimodal case. In this way an individual factor can be assigned to each cluster to explore different function landscape simultaneously. So, CE-AVS brings much flexibility in variance adaptation.

Table 2: Scalability of AVS and CEAVS with and without Exact Structure Information (Truncate Replacement)

Function	UMDA-AVS			UMDA-CEAVS			SEGNA-AVS			SEGNA-CEAVS		
	α	β	$l=80$	α	β	$l=80$	α	β	$l=80$	α	β	$l=80$
Sphere	902	1.32	32778	846	1.27	28384	3940	1.73	141300	3210	1.63	99100
Ellipse	3983	1.10	39880	2656	1.28	36854	4744	1.75	204825	3971	1.70	139720
Cigar	3907	1.17	45581	2897	1.28	40795	4707	1.71	162000	5160	1.75	235845
Tablet	2921	1.21	36645	2186	1.26	29801	6002	1.51	152470	5162	1.42	128490
Cigar Tablet	3759	1.21	47277	2829	1.28	40529	5393	1.78	231615	4552	1.68	158400
Two Axes	3165	1.27	43632	2617	1.29	38042	4952	1.68	171290	3165	1.78	128180
Different Powers	3816	1.30	58462	3285	1.29	48297	9798	1.57	275115	5653	1.66	187730
Para Ridge	1236	0.77	6260	1450	0.70	6374	6406	1.27	90055	5653	1.43	107715
Sharp Ridge	812	0.88	4910	696	0.73	3280	3402	1.49	84015	1837	1.69	67425
Rosenbrock							27100	2.71	7490600	23557	2.68	6150900

4. EXPERIMENTS AND RESULTS

In this section we perform experiments to validate the property of CE-AVS discussed above. The validation is carried out by comparing CE-AVS and traditional AVS on some benchmark functions, using the same baseline EDA algorithms. We made some rough comparison on two versions of traditional AVS described in [19] and [9], and found that the latter performs better on the benchmark problems in this section, so we chose it for comparison with CE-AVS.

The experiments in this section share the configuration as follows: The population size is chosen to be a geometric series from 2 to 1600 with the ratio of $\sqrt{2}$ (which we found to be the minimal gap that produces statistical-significant differences), and we seek for the population size that successes 20 consequent runs and obtains fastest convergence (which might not always be the minimally required population size). Then we refine them with 100 independent consecutive runs and verify that at least 95 runs are successful, and report their average result.

Two structure learning schemes are used in the experiments. The first scheme encodes the problem structure in the model (which is full dependence for the Rosenbrock function and univariate model for other functions in the first two experiments). The second scheme adopts the simple learning strategy described in section 2.3, which is referred to as Simple-EGNA (SEGNA) in this section. Truncate replacement [13] and Continuous Boltzmann Selection [5] are adopted for all algorithms to speed up the search efficiency.

The first experiment compared the performance of both strategies under different quality of structure learning. The experiment uses the benchmark function in [9] and perform a scalability analysis. The dimensionality of the test functions is chosen to be $l \in \{5, 10, 20, 40, 60, 80\}$. The stop condition is set to 10^{-10}. The results are depicted in Table. 2. The display format in the table follows Eq. 5, and additional column $l = 80$ is added showing the mean number of evaluations that an algorithm required to reach the target precision for 80-dimension problems. It can be concluded from the results that CE-AVS outperforms traditional AVS in almost all functions except that in the Parabolic Bridge funtion the difference is insignificant. Although on some data AVS has lower scaling coefficient β, but the constant item α is significantly larger so that eventually it demands more evaluations in all dimensions tested. It should be noted that when no exact structure information is provided the advantage of CE-AVS is more significant, which shows its robustness.

Table 3: Test Results on Shifted SumCan Function

s	EMNA	EMNA-AVS	EMNA-CEAVS
0	43020	45810	30930
0.01	172830	$[4e-8]$	59507
0.04	185197	$[2e-7]$	58257
0.16	280510	$[8e-7]$	60872
0.64	$[1e-6]$	$[3e-6]$	64037
4	$[1e+5]$	$[1.9e-5]$	209000
64	$[1e+5]$	$[3.8e-4]$	206673

The second experiment shows that CE-AVS can solve some problems that traditional AVS will be trapped, where the trap is not even a local optimum. The 10-dimensional Shifted SumCan Function in Eq. 6 is used for testing. In this experiment we apply AVS and CE-AVS on EMNA algorithm, so that there will not be concerns about the quality of structure learning (with structure learning the number of evaluations for successful runs is reduced by the trapped ones are not changed). The shifting distance s is varied and its effect to each algorithm is observed. The result is depicted in Table. 3. In the table, the square brackets show the mean terminal error of unsuccessful runs and other numbers show the average required number of fitness evaluation in 20 runs to reach 10^{-10} precision within the optimum. From the result we see that traditional AVS performs even worse that EMNA without variance scaling, due to the improper shrinking of the variance. The trap can be avoided only by forcing the scaling factor to be larger than 0.7 in this problem. While CE-AVS do not suffer from low success mutation rate.

The third experiment shows the effect of adopting multiple scaling factor. Taking the Parabolic Function and the Sharp Bridge Function in the first experiment, because the first dimension in both function is dominating and the covariance matrix is ill-posed, the performance of all the algorithms are influenced by it. In this experiment we explicitly separates the first variable in the calculation of the covariance matrix, so that the performance of both AVS and CE-AVS are improved, and CE-AVS clearly outperforms AVS, as is shown in Fig. 2. We go on to calculate one separate scaling factor

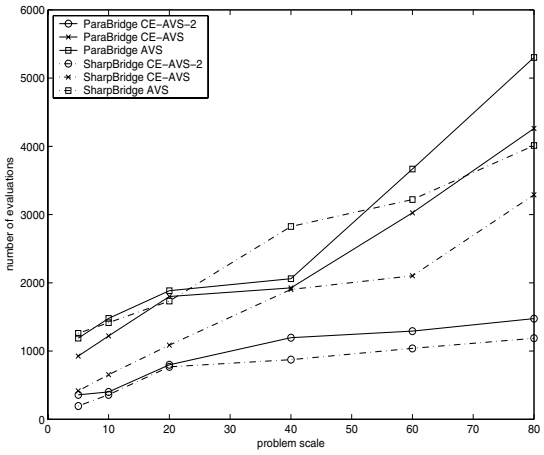

Figure 2: The effect of using multiple scaling factor in Parabolic Bridge and Sharp Bridge Function

for the first variable and another scaling factor for all other variables, we see an boost on the performance and the required number of evaluations is reduced significantly, as is shown in the "-CE-AVS-2" item in Fig. 2. Which persuades that adopting multiple scaling factors for each subspace can be beneficial.

Finally we illustrate the impact of the randomness in parameter fitting to the performance of CE-AVS. We repeat experiment 1 but tournament replacement is adopted instead of truncate replacement, which is more random. The result is shown in Tab. 4. In this case, CE-AVS is advantageous only when the model structure is inexact, while the performance of AVS is improved. This shows the importance of filtering out the randomness in model fitting.

5. CONCLUSION AND FUTURE WORKS

In this paper the cross entropy adaptive variance scaling is proposed as a new type of AVS method specifically for continuous EDA. Unlike traditional AVS approaches that tune the scaling factor incrementally, the method measures the divergence between the probabilistic model and the resulted offspring and calculates a scaling factor immediately. Experiments shows that the proposed method extends the class of problems that can solved, and achieve faster convergence speed in a set of test functions when the randomness of the model fitting is not so significant. With CE-AVS, multiple scaling factors can be assigned simultaneously to different subspaces, or different clusters in multi-modal EDA algorithms, which can boost the optimization performance in special problems.

CE-AVS is not a competitive approach to CT-AVS. Both approaches strive to detect and alleviate the malfunction of variance scaling from different scope. Integration both methods will be beneficial in creating a sophisticated AVS strategy.

It had been pointed out that although CE-AVS is less vulnerable to the defect of structure learning, current model can not rule out the effect of random noises in model fitting and it will slow down the convergence under some model building schemes. A more sophisticated model should be developed to eliminate this effect. On the other hand although the advantage of adopting multiple scaling factors has been illustrated, applicable methods of decomposing the space according to different scaling property is not developed. More over, the benefit of applying the approach on multimodal EDA algorithms is not yet tested. Solving these problems will extend the practicality of CE-AVS, and might bring more insight about the nature of variance adaptation in EDA.

6. ACKNOWLEGEMENT

This work is supported by the National Nature Science Foundation of China (Grant No: 60405011, 60575057).

7. REFERENCES

[1] C. Ahn, R. Ramakrishna, and D. Goldberg. Real-coded bayersian optimization algorithm:bringing the strength of boa into the continuous world. In *Proceedings of the GECCO-2004 Conference on Genetic and Evolutionary Computation*, pages 840–851. Springer Verlag, 2004.

[2] A. Berny. Statistical Machine Learning and Combinatorial Optimization. In *Theoretical Aspects of Evolutionary Computation*, pages 287–306. Springer Verlag, 2001.

[3] P. Bosman and J. Grahl. Matching inductive search bias and problem structure in continuous estimation of distribution algorithms. Technical Report 03/2005, Dept. of Logistics, Mannheim Business School, 2005.

[4] P. Bosman and D. Thierens. Expanding from Discrete to Continuous Estimation of Distribution Algorithms: The IDEA. In *Parallel Problem Solving from Nature - PPSN VI*, pages 767–776, Paris, France, Sep. 2000. Springer-Verlag.

[5] Y. Cai, X. Sun, and P. Jia. Probabilistic Modeling for Continuous EDA with Boltzmann Selection and Kullback-Leibeler Divergence. In *Proceedings of the 8th Conference on Genetic and Evolutionary Computation*, pages 389–396. ACM Press, 2006.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, 1991.

[7] M. Gallagher and M. Frean. Population-Based Continuous Optimization, Probabilistic Modelling and Mean Shift. *Evolutionary Computation*, 13(1):29–42, 2005.

[8] C. González, J. Lozano, and P. Larrañaga. Mathematical Modelling of UMDAc Algorithm with Tournament Selection: Behaviour on Linear and Quadratic Functions. *International Journal of Approximate Reasoning*, 31(2):313–340, 2002.

[9] J. Grahl, P. Bosman, and F. Rothlauf. The Correlation-Triggered Adaptive Variance Scaling IDEA. In *Proceedings of the 8th Conference on Genetic and Evolutionary Computation*, pages 397–404. ACM Press, 2006.

Table 4: Scalability of AVS and CEAVS with and without Exact Structure Information (Tournament Replacement)

Function	UMDA-AVS			UMDA-CEAVS			SEGNA-AVS			SEGNA-CEAVS		
	α	β	$l = 80$	α	β	$l = 80$	α	β	$l = 80$	α	β	$l = 80$
Sphere	2292	1.17	26885	2400	1.26	32443	3103	1.72	123695	3659	1.50	81982
Ellipse	3049	1.16	34137	2980	1.28	42680	4199	1.70	140632	4109	1.63	111900
Cigar	3623	1.12	37418	3558	1.26	48403	4792	154582	1.69	4638	1.65	140197
Tablet	2754	1.10	25869	2582	1.25	34709	3636	1.59	95460	3526	1.57	85177
Cigar Tablet	3285	1.17	38091	3359	1.28	47372	4705	1.70	157200	4800	1.62	136845
Two Axes	2997	1.19	34547	3130	1.27	44222	3794	1.68	121342	3880	1.64	107220
Different Powers	4395	1.14	44424	4014	1.27	58773	6242	1.60	181640	6178	1.60	169792
Para Bridge	1011	0.93	7371	1066	0.91	7684	3568	1.49	77302	6091	1.59	164062
Sharp Bridge	734	0.96	5885	670	0.90	4776	2066	1.60	57420	1982	1.73	69870

[10] J. Grahl, S. Minner, and F. Rothlauf. Behaviour of UMDAc with Truncation Selection on Monotonous Functions. In *The 2005 IEEE Congress on Evolutionary Computation*, pages 2553–2559, Edinburgh, Scotland, 2005. IEEE.

[11] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolutionary strategiess. *Evolutionary Computation*, 9(2):159–195, 2001.

[12] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Optimization in Continuous Domains by Learning and Simulation of Gaussian Networks. In *Proceedings of the GECCO-2000 Workshop in Optimization by Building and Using Probabilistic Models*, pages 201–204, San Francisco, CA, USA, 2001. Morgan Kaufmann.

[13] P. Larrañaga and J. Lozano. *Estimation of Distribution Algorithms, A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.

[14] P. Larrañaga, J. A. Lozano, and E. Bengoetxea. Estimation of Distribution Algorithms based on Multivariate Normal and Gaussian Networks. Technical Report EHU-KZAA-IK-1-01, University of the Basque Country, 2001.

[15] J. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea. *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*. Springer, 2006.

[16] H. Mühlenbein. The Equation for Response to Selection and Its Use for Prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[17] H. Mühlenbein and R. Höns. The Estimation of Distributions and the Minimum Relative Entropy Principle. *Evolutionary Computation*, 13(1):1–27, 2005.

[18] J. Oceanasek and J. Schwarz. Estimation of Distribution Algorithm for Mixed Continuous-Discrete Optimization Problems. In *2nd Euro-International Symposium on Computational Intelligence*, pages 227–232, Kosice, Slowakia, 2002. IOS Press.

[19] J. Ocenasek, S. Kern, N. Hansen, S. Müller, and P. Koumoutsakos. A Mixed Bayesian Optimization Algorithm with Variance Adaptation. In *Parallel Problem Solving from Nature - PPSN VIII*, pages 352–361. Springer Verlag, 2004.

[20] A. Ostermeier, A. Gawelczyk, and N. Hansen. A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation*, 2(4):369–380, 1994.

[21] M. Pelikan, D. Goldberg, and S. Tsutsui. Getting the Best of Both Worlds: Discrete and Continuous Genetic and Evolutionary Algorithms in Concert. *Information Sciences*, 156(3-4):147–171, 2003.

[22] M. Pelikan, S. K., and C.-P. E. *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Springer, 2006.

[23] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, 1973.

[24] S. Rudolf and M. Köppen. Stochastic Hill Climbing by Vectors of Normal Distributions. In *Proceedings of the First Online Workshop on Soft Computing*, Nagoya, Japan, 1996.

[25] K. S. *Information theory and statistics*. John Wiley and Sons, 1959.

[26] H.-P. Schwefel. *Numerical Optimization of Computer Models*. Wiley, 1981.

[27] M. Sebag and A. Ducoulombier. Extending Population-Based Incremental Learning to Continuous Search Spaces. In *Parallel Problem Solving from Nature - PPSN V.*, pages 418–427, Amsterdam, Netherlands, Sep. 1998. Springer-Verlag.

[28] S. Shin and B.-T. Zhang. Bayesian Evolutionary Algorithms for Continuous Function Optimization. In *Proc. 2001 Congress on Evolutionary Computation*, pages 508–515. IEEE, 2001.

[29] B. Yuan and M. Gallagher. On the Importance of Diversity Maintenance in Estimation of Distribution Algorithms. In *Proceedings of the 7th Conference on Genetic and Evolutionary Computation*, pages 719–726. ACM Press, 2005.