STATISTICAL LEARNING IN OPTIMIZATION: GAUSSIAN MODELING FOR POPULATION SEARCH

Shotaro Akaho
Email:akaho@etl.go.jp

Information Science Division, Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba 305-8568, Japan

ABSTRACT

Population search algorithms for optimization problems such as Genetic algorithm is an effective way to find an optimal value, especially when we have little information about the objective function. Baluja has proposed effective algorithms modeling the distribution of elites explicitly by some statistical model. We propose such an algorithm based on Gaussian modeling of elites, and analyze the convergence property of the algorithm by defining the objective function as a stochastic model. We point out that the algorithms based on the explicit modeling of the elites' distribution tend to converge to unpreferable local optima, and we modify the algorithm to conquer the defect.

KEYWORDS: probabilistic and statistical methods, optimization, population search

1. INTRODUCTION

When we want to optimize some complex function with a lot of local optima, random search (Monte Carlo) algorithms are effective to avoid converging to a non-global optimum. In order to generate good candidate solutions in random search algorithm, we need to extract the structure of good solutions (elites), especially if we don't have enough knowledge on the function.

Stochastic gradient descent algorithms such as Gibbs sampler are a class of popular random search algorithms, but those only generate candidate solutions neighboring a single current solution and it does not model the structure of elites.

On the other hand, Genetic algorithms[1] implicitly model the structure of elites through the selection and crossover operator. The structure called 'schema' is fragile because the crossover operator often breaks the schema.

Recently, Baluja et al[2, 3, 4] have proposed the algorithm which explicitly model the statistical structure of elites. The algorithm maintains a parametric distribution of elites, and repeats a Monte Carlo optimization step based on the distribution as well as a learning step to estimate the parameter of the distribution. They use several kinds of statistical model such as special form of Bayesian belief networks, and the algorithms shows superior performance to other random algorithms.

In this paper, we propose a similar algorithm based on Gaussian modeling mainly for the sake of simplicity of analysis, while Bayesian belief network is difficult to analyze because it includes combinatorial structure. We model the objective (fitness) function as a statistical distribution to analyze the convergence property of the algorithm. By the analysis, we found that the algorithms based on a explicit modeling of elites tend to converge to the δ function even if elites are broadly distributed, which causes a convergence to a non-global local optimum solution. To deal with that problem, we introduce an extension process of the variance.

2. BASIC ALGORITHM

The basic algorithm is similar to Baluja's algorithm except that the domain is Euclidean space and the statistical model is Gaussian distribution in our algorithm. Suppose $\mathcal{N}[\boldsymbol{\mu}, V]$ represent the Gaussian distribution with mean $\boldsymbol{\mu}$ and the variance V.

Algorithm (before improved):

1. Initialize $\boldsymbol{\mu}_0, V_0$; Let $t := 0$.
2. Generate population with respect to $\mathcal{N}[\boldsymbol{\mu}_t, V_t]$.
3. Select the $100 \times \theta$ % individuals from the population according to their fitness values, where θ is a prefixed parameter.
4. Let $\boldsymbol{\mu}_{t+1}, V_{t+1}$ be the mean and the variance of the selected individuals.
5. Let $t := t + 1$; Go to step 2.

By this algorithm, V_t converges to 0 matrix as shown later, which can make the search space too narrow. We provide an improved algorithm in section 6.

3. STATISTICAL MODEL FOR FITNESS FUNCTION

We model the fitness function as a statistical distribution. Some fitness functions in the real world are stochastic themselves. Even if the fitness function is deterministic, we can sometimes regard the complicated microstructure of the function as stochastic noise. Moreover, we can consider a set of functions instead of special

forms of functions by statistical modeling and the analysis becomes also easier. In the following, the convergence of the algorithm is analyzed in the sense of the average over ensembles of populations and fitness functions. This corresponds to the average behavior of infinite population. See [5] on a similar analysis on Genetic algorithms.

Suppose the probability that each $\boldsymbol{x}$ becomes an elite is proportional to $\mathcal{N}(\boldsymbol{\mu}_*, V_*)$,

$$p(\text{elite} \mid \boldsymbol{x}) = q\phi(\boldsymbol{x}; \boldsymbol{\mu}_*, V_*), \tag{1}$$

and the probability that $\boldsymbol{x}$ becomes a non-elite is given by $1 - p(\text{elite} \mid \boldsymbol{x})$, where $\phi(\boldsymbol{x}, \boldsymbol{\mu}, V)$ is a probability density function of Gaussian, and q, $\boldsymbol{\mu}_*, V_*$ are unknown parameters. The coefficient q must be less than $1/\phi(\mathbf{0}; \mathbf{0}, V_*)$ in order to keep the probabilities positive.

The fitness value $F(\boldsymbol{x})$ is taken from some unknown distribution $g_{\rm e}(F)$ if $\boldsymbol{x}$ is an elite and $g_{\rm ne}(F)$ otherwise. $g_{\rm e}(F)$ and $g_{\rm ne}(F)$ are independent of $\boldsymbol{x}$ itself. We assume that elites have always larger fitness values than those of non-elites,

$$\min\{F \mid g_{\rm e}(F) > 0\} > \max\{F \mid g_{\rm ne}(F) > 0\}. \tag{2}$$

In the following analysis, we don't need explicit forms of $g_{\rm e}(F)$ and $g_{\rm ne}(F)$.

Let us summarize how the fitness is determined stochastically: For each query $\boldsymbol{x}$,

1. $\boldsymbol{x}$ is determined to be elite or not in probability $p(\text{elite} \mid \boldsymbol{x})$ and $(1 - p(\text{elite} \mid \boldsymbol{x}))$,
2. the fitness of $\boldsymbol{x}$ is determined with respect to $g_{\rm e}(F)$ when $\boldsymbol{x}$ is an elite, or $g_{\rm ne}(F)$ when $\boldsymbol{x}$ is a non-elite.

For the sake of simplicity, we use the one dimensional notation x, μ and σ^2 instead of $\boldsymbol{x}, \boldsymbol{\mu}$ and V below except for section 6, since the result is essentially the same.

Let a_t be the frequency of elites included in the population generated in the step 2 of the basic algorithm, which is given by

$$\begin{aligned} a_t &= \int q\phi(x; \mu_*, \sigma_*^2)\, \phi(x; \mu_t, \sigma_t^2)\, \mathrm{d}x, \\ &= q\phi(\tilde{\mu}_t; 0, \sigma_*^2 + \sigma_t^2), \end{aligned} \tag{3}$$

where

$$\tilde{\mu}_t \equiv \mu_t - \mu_*. \tag{4}$$

First we consider the case when a_t is larger than θ in section 4, and then generalize it in section 5.

4. EASY CASE ($a_t \geq \theta$)

If $a_t \geq \theta$, all selected individuals are elites. Therefore, the mean and the variance of selected samples at the $(t+1)$-th step are given by

$$\tilde{\mu}_{t+1} = B_t \tilde{\mu}_t, \tag{5}$$

$$\frac{1}{\sigma_{t+1}^2} = \frac{1}{\sigma_t^2} + \frac{1}{\sigma_*^2}, \tag{6}$$

where

$$B_t \equiv \frac{\sigma_*^2}{\sigma_*^2 + \sigma_t^2}. \tag{7}$$

The equation above implies that σ_t^2 converges to 0 not σ_*^2 and the distribution is getting close to δ function. It also makes the convergence of μ_t slow. Such phenomena occur also in Baluja's algorithm and the algorithm sometimes can become weak against deceptive problems.

In order to overcome it, let us transform σ_{t+1}^2 to $\hat{\sigma}_{t+1}^2 = 2\sigma_{t+1}^2$ and use it at the successive iteration. Then $\hat{\sigma}_t^2$ converges to σ_*^2 and μ_t converges to μ_* exponentially. This technique is generalized in the following section.

5. GENERAL CASE

In this section, we generalize the result of the previous section to all range of a_t.

If $a_t < \theta$, non-elites are included in selected individuals, and the distribution of selected individuals is given by

$$p(x) = \frac{1}{\theta}[\psi_t(x) + \frac{\theta - a_t}{1 - a_t}\{\phi_t(x) - \psi_t(x)\}], \tag{8}$$

where $\phi_t(x) \equiv \phi(x; \mu_t, \sigma_t^2)$, and $\psi_t(x) \equiv q\phi_t(x)\phi(x; \mu_*, \sigma_*^2)$. We can derive the mean value of a random variable ξ as

$$E_{t+1}[\xi] = (1 - C_t)E_{\rm elite}[\xi] + C_t E_t[\xi], \tag{9}$$

where $C_t \equiv (\theta - a_t)/\{\theta(1 - a_t)\}$, and $E_{\rm elite}$ is a mean operator for elites which is given in the previous section. By putting $C_t = 0$, eq. (9) includes the previous section; hence we redefine C_t as

$$C_t \equiv \frac{[\theta - a_t]_+}{\theta(1 - a_t)}, \tag{10}$$

to apply to any value of a_t, where $[s]_+$ equals to s if $s > 0$ and 0 otherwise.

We obtain μ_{t+1} and σ_{t+1}^2 by straightforward calculations,

$$\tilde{\mu}_{t+1} = \{(1 - C_t)B_t + C_t\}\tilde{\mu}_t, \tag{11}$$

$$\begin{aligned} \sigma_{t+1}^2 &= \{1 - (1 - C_t)\frac{\sigma_t^2}{\sigma_*^2 + \sigma_t^2}\}\sigma_t^2 \\ &\quad + (1 - C_t)C_t(1 - B_t)^2\tilde{\mu}_t^2, \end{aligned} \tag{12}$$

where B_t is defined by (7). In this general case, σ_t^2 also converges to 0 if μ_t converges to μ_*.

Since μ_t converges to μ_* while $0 < C_t < 1$, we concentrate to the estimation of σ_*^2. Since eq. (12) includes an

unknown parameter C_t, we have to estimate C_t as well. Let $\hat{\sigma}_t^2$ and $\hat{C}_t$ be the estimation values of σ_*^2 and C_* at time t respectively, where C_* is a value of C_t defined at $\mu_t = \mu_*$ and $\sigma_t^2 = \sigma_*^2$.

If $\hat{\sigma}_t^2 = \sigma_*^2$ and $\mu_t = \mu_*$, we have

$$\sigma_{t+1}^2 = \frac{1 + C_*}{2}\sigma_*^2, \tag{13}$$

from (12), therefore if $\hat{C}_t$ is a good approximation of C_*, σ_*^2 can be estimated by

$$\hat{\sigma}_{t+1}^2 = \frac{2}{1 + \hat{C}_t}\sigma_{t+1}^2. \tag{14}$$

If $\hat{\sigma}_{t+1}^2 > \hat{\sigma}_t^2$, $\hat{C}_t$ is considered to be smaller than C_* in order that $\hat{\sigma}_t^2$ converges, and vice versa. Therefore, we can construct an update formula for $\hat{C}_t$ by

$$\hat{C}_{t+1} = \hat{C}_t + \varepsilon \left\{ \frac{\hat{\sigma}_{t+1}^2}{\hat{\sigma}_t^2} - 1 \right\}, \tag{15}$$

where ε is a small constant. We can prove that equations (14) and (15) are locally stable around $(\sigma_t^2, C_t) = (\sigma_*^2, C_*)$.

6. IMPROVED ALGORITHM

The result obtained by the previous section can be generalized to a higher dimensional case in a similar way. The improved algorithm is given as follows:

Algorithm (improved):

1. Initialize $\boldsymbol{\mu}_0, \hat{V}_0, \hat{C}_0$; Let $t := 0$
2. Generate population with respect to $\mathcal{N}[\boldsymbol{\mu}_t, \hat{V}_t]$.
3. Select the $100 \times \theta$ % individuals from the population according to their fitness values.
4. Let $\boldsymbol{\mu}_{t+1}, V_{t+1}$ be the mean and the variance of the selected individuals.
5. Let $\hat{V}_{t+1} = \dfrac{2}{1 + \hat{C}_t} V_{t+1}$.
6. Let $\hat{C}_{t+1} = u\left[\hat{C}_t + \varepsilon \left\{ \dfrac{\mathrm{trace}[\hat{V}_{t+1}]}{\mathrm{trace}[\hat{V}_t]} - 1 \right\}\right]$, where ε is a small constant and

$$u[s] \equiv \begin{cases} 1 & \text{if } s > 1 \\ s & \text{if } 0 \le s \le 1 \\ 0 & \text{if } s < 0 \end{cases} \tag{16}$$

7. Let $t := t + 1$; Go to step 2.

7. SIMPLE EXPERIMENT

We show a simple computer simulation to validate our improved algorithm.

The domain is one dimensional space $\mathcal{R}$. The fitness function $F(x)$ is firstly defined on discrete points $x_k \equiv k/1000$ where k is an integer. $F(x)$ for other x is defined as $F(x_k)$, where x_k is the closest value to x. $F(x_k)$ is randomly generated as described in section 4, and the parameters are $q = 0.3$, $\mu_* = 0.0$ and $\sigma_*^2 = 2.0^2$. The distributions of the fitness g_{e} and g_{ne} are uniform distribution on $[1, 2)$ and $[0, 1)$ respectively. However the result does not depend on g_{e} and g_{ne} if eq. (2) is fullfilled.

The initial values of parameters for the algorithm are determined as $\mu_0 = 3.0$, $\hat{\sigma}_0^2 = 2.0^2$ and $\hat{C}_0 = 1.0$. The threshold of selection $\theta = 0.2$ and the learning coefficient $\varepsilon = 0.5$.

We tried 30 experiments for different random seeds, and compared the performance of the improved algorithm, the basic algorithm, and the random search based on uniform distribution on $[-15, 15]$.

Fig.1 shows the average of the function, $I[t - t_c]$, where t_c is the time when the optimal solution is found, and $I[s] = 1$ if $0 \le s$, and $I[s] = 0$ if $s < 0$.

The improved algorithm shows an apparently better performance than the other algorithms. The convergence property is shown in figs. 2 and 3. The standard deviation in the basic algorithm converges 0 very quickly, and the convergence of the mean parameter almost stops, while the mean parameter is moving around 0 in the improved algorithm.

The change of $\hat{C}_t$ is shown in fig.4. Since the value does not converge until 100 steps, the standard deviation in the improved algorithm also decreases, but the value is distributed around 0.7 and 0.8, which avoids to converge σ_t^2 to 0 quickly.

8. CONCLUSION AND FUTURE WORKS

We have proposed a new algorithm based on Gaussian modeling of elites. The algorithm converges to the true model by introducing an extension process of the variance parameter. The algorithm is locally stable, but the proof of global stability is a future problem.

In this paper, the domain is assumed to be Euclidean space, but a similar algorithm can be derived in binary domain. The analysis of the binary domain will be reported in the forthcoming paper.

Our analysis is currently on the average behavior. In practice, the dynamics of search deviates from the average because the population size is finite and the fitness function is fixed. It is also necessary to verify that the

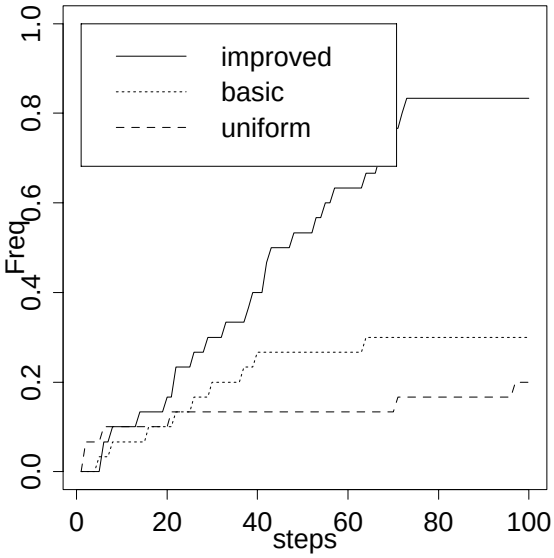

Figure 1: Frequency of optimal solution found. $I(t-t_c)$ are averaged over 30 experiments.

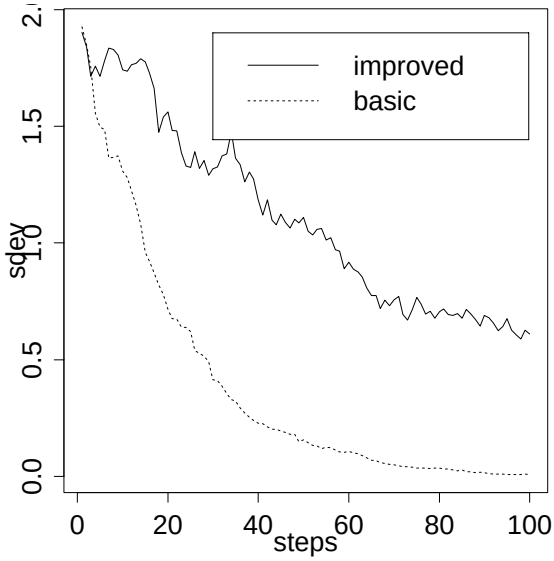

Figure 3: The convergence of the standard deviation parameter averaged over 30 experiments.

stochastic model of the fitness function is a good approximation of the real world problems.

References

[1] Goldberg, D.E. : *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley, 1989.

[2] De Bonet, J., Isbell, C. and Viola, P.: MIMIC: finding optima by estimating probability densities, *Advances in Neural Information Processing Systems*, 8, 1996.

[3] Baluja, S. : Genetic algorithms and explicit search statistics, *Advances in Neural Information Processing Systems*, 8, 1996.

[4] Baluja, S. and Davies, S.: Using optimal dependency-trees for combinatorial optimization: learning the structure of the search tree, *Advances in Neural Information Processing Systems*, 9, 1997.

[5] Vose, M.D. and Liepins, G.E.: Punctuated equilibria in genetic search, Complex Systems, 5, 31–44, 1991.

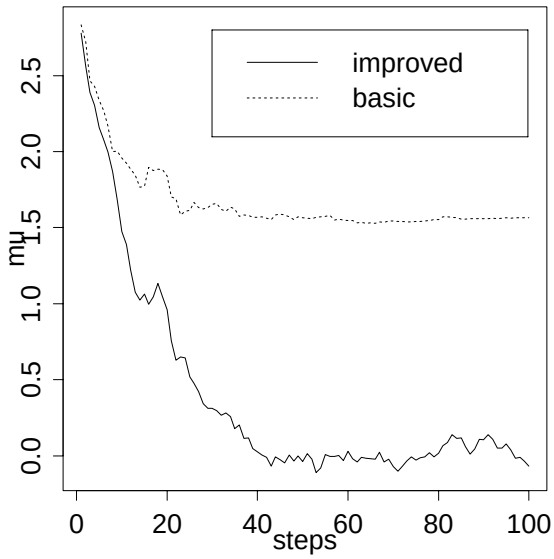

Figure 2: The convergence of the mean parameter averaged over 30 experiments.

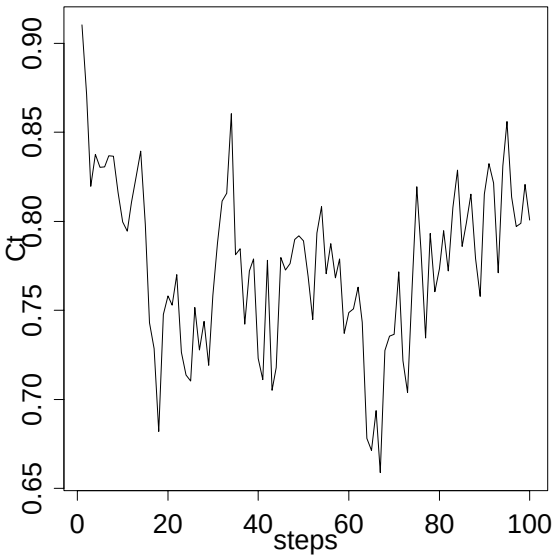

Figure 4: The convergence of the value $\hat{C}_t$ averaged over 30 experiments.