Data Sampling Using Bayesian Analysis and its Applications in Simulated Annealing*

Benjamin W. Wah and Minglun Qian
Department of Electrical and Computer Engineering
and the Coordinated Science Laboratory
University of Illinois, Urbana-Champaign
Urbana, IL 61801, USA
E-mail: {wah, m-qian}@manip.crhc.uiuc.edu

Abstract

In this paper, we propose a new probabilistic sampling procedure and its application in simulated annealing (SA). The new procedure uses Bayesian analysis to evaluate samples made already and draws the next sample based on a density function constructed through Bayesian analysis. After integrating our procedure in SA, we apply it to solve a set of optimization benchmarks. Our results show that our proposed procedure, when used in SA, is very effective in generating high-quality samples that are more reliable and robust in leading to global solutions.

1 Introduction

It's well known that simulated annealing (SA) [3, 4] converges asymptotically to a global minimum for any nonlinear, unconstrained nonconvex problem. However, to achieve global convergence, SA normally requires a large number of function evaluations. In order to improve the efficiency of SA, one way is to reduce the number of function evaluations by generating high-quality samples that have high probability of hitting a global minimum.

In this paper, we propose a new sampling procedure in SA by constructing a statistical distribution of the objective function from samples generated, modifying the parameters of the distribution through Bayesian analysis, and drawing the next sample using the distribution estimated.

*Research supported by National Science Foundation Grant CCR 96-32316.

Proc. of the Fifth International Conference on Computer Science and Informatics.

2 Bayesian Analysis-Based Sampling

In *Bayesian analysis-based sampling* (in short, *Bayesian sampling*) of an unconstrained function, we construct a statistical model to describe the probability distribution of the objective function at any unsampled point, and use the distribution to generate the next sample with an improved value with high probability. Specifically, when point x_i is sampled with an objective value μ_i, we assume that at point $x \neq x_i$, the distribution of the objective function taking value f is described by the Cauchy distribution:

$$d(f) = \frac{\sigma(r)}{\pi(\sigma^2(r) + (f - \mu_i)^2)}, \tag{1}$$

where $r = ||x - x_i||_2 > 0$, $\sigma(0) = 0$, and $\sigma(\infty) = \infty$. When more than one samples are available, we assume that the influence of sample x_i on any unsampled point x is governed by weight w_i:

$$g(f, x) = \sum_i w_i \frac{\sigma(||x - x_i||_2)}{\pi(\sigma^2(||x - x_i||_2) + (f - \mu_i)^2)}. \tag{2}$$

Depending on the weights chosen, $g(f, x)$ may need to be normalized. The choice and adjustment of weights by Bayesian analysis is discussed later.

When w_i associated with x_i is large, the distribution of the objective value around x can be inferred from x_i with high probability. In our sampling procedure, we choose a simple form for $\sigma(r)$:

$$\sigma(r) = kr, \tag{3}$$

where k is a factor that reflects the roughness of the terrain. A small k indicates a flat search terrain, whereas a large k implies a rugged terrain. To illustrate the modeling of an objective function, consider a simple 1-dimensional modified Rastrigin function. For $-5.12 \leq$

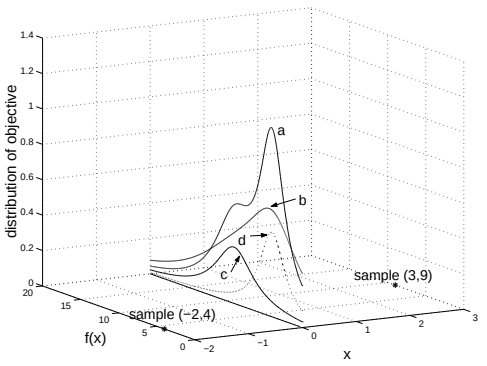

Figure 1. The estimated density of the objective function at $x = 2$ when there are two samples (indicated by $*$) (see text for further explanation).

$x_i \leq 5.12$, the v-dimensional Rastrigin function takes the following form:

$$f(x) = 10v + \sum_{i=1}^{v}(x_i^2 - 10cos(2\pi x_i)). \qquad (4)$$

After drawing two samples $x_1 = -2$ with $\mu_1 = 4$ and $x_2 = 3$ with $\mu_2 = 9$ and assuming weights $w_1 = 0.6$ and $w_2 = 0.4$, Figure 1 plots $g(f, x = 0.0)$ with $k = 1$ and $k = 2$, respectively. Curve c plots (1) when there is only one sample at $(3, 9)$ with $k = 1$, and curve d is for the case when there is only one sample at $(-2, 4)$ with $k = 1$. The combination of these two samples changes the distribution of the objective value at $x = 0$ according to (2): curve a for $k = 1$, and curve b for $k = 2$. These curves show that when k is large, the terrain is rugged, and the distribution function tends to be flat.

The parameters that are important in our model are k and w_i. During sample generation, these parameters are dynamically adjusted to better reflect a search terrain using Bayesian analysis, which infers the posterior probability of an event by using information on the prior probability of the event and observations made so far. Suppose there are m independent events, with event M_i having prior probability $P(M_i)$. Given event A whose occurrence is related to the occurrence of M_i by conditional probability $P(A|M_i)$, then the posterior probability of M_i when A occurs should be adjusted according to the following Bayesian rule:

$$P(M_i|A) = \frac{P(A|M_i)P(M_i)}{\sum_{u=1}^{m} P(A|M_u)P(M_u)}. \qquad (5)$$

We first apply (5) to adjust roughness factor k. Assuming that k takes a set of discrete values, say $\kappa_1, \cdots, \kappa_m$, and event K_j means that k takes value κ_j. Let event V happens when a new sample x' has objective value μ'. Further, let x be a sample generated already with objective value μ. Then according to (1), the prior probability $P(V|K_i)$ is:

$$P(V|K_j) = \frac{\sigma(r)}{\pi(\sigma^2(r) + (\mu - \mu')^2)}, \qquad (6)$$

where $r = ||x - x'||_2$. After defining these events, (5) can be used to calculate the posterior probability $P(K_j|V)$, which will then be used as the new prior probability $P(K_j)$ when the next sample is drawn. When there are multiple samples generated, (6) can be applied multiple times, and the mean of the k_j's is taken to be the roughness factor k in (1) and (2):

$$k = \sum_{j=1}^{m} P(K_j)\kappa_j. \qquad (7)$$

Similarly, we use (5) to adjust w_i after a new sample is generated. Here, event W_i assumes that the newly generated sample is influenced only by the i^{th} sample. The prior probability of W_i is:

$$P(W_i) = w_i. \qquad (8)$$

$P(V|W_i)$ is defined to be the probability that the new sample takes a value of μ' when the new sample is only influenced by the i^{th} sample, and has the same expression as (6). Using these definitions, we can apply (5) to find the posterior probability of $P(W_i|V)$. The posterior probability will then be used in the next round of sampling as the new prior probability for $P(W_i)$ and be used to adjust w_i.

To generate a new sample with an improved objective value from distribution (2), we set a level at which we wish the new improved objective value F to be at:

$$F = min(\mu_i) - \epsilon, \qquad i = 1, \cdots, n, \qquad (9)$$

where n is the number of generated samples used in our statistic model, and ϵ is a small value. Note that a small ϵ is desirable because F much smaller than the best generated sample is not accurately predicted by (2). Based on (2) and a composition method [1], we can now generate the next sample.

The composition method first generates a random integer Z between 0 and n with probability $P(Z = i) = s_i$, and then generates a random variate r with density t_Z. After generating r, it generates a point uniformly distributed on the sphere surface with x_Z as center and r as radius.

According to (2), the probability distribution that the objective function takes $f = F$ over the search space (excluding points sampled already) can be expressed as:

$$g_F(x) = c\sum_{i=1}^{n} w_i \frac{kr_i}{\pi(k^2 r_i^2 + (F - \mu_i)^2)} = \sum_{i=1}^{n} s_i t_i(r_i), \quad (10)$$

Table 1. Experimental results on evaluating 23 benchmark functions by EA [6], SQP [5], SIMANN [2] and SABS. "Avg. best solution" represents the average best solutions in the last generation of EP over 50 runs. SQP was run 100 times if the objective function was differentiable, and SA and SABS were each run 10 times. CPU times were measured by averaging the results of 10 runs in each case. All runs were done on Pentium 3/450 computers with Solaris. Note that for f_7, a global solution is considered found if it is 0.2 or less.

			EA	SQP:DONLP2		SA: SIMANN		SABS		Ratio (SABS/SA)	
Problem ID	Dim. n	Global Solution	Avg. best solution	Best solution	% with best sol.	Best solution	% with best sol.	Best solution	% with best sol.	CPU time	Number of samples
f_1	30	0	**0**	**0**	**100**	**0**	**100**	**0**	**100**	1.14	0.359
f_2	30	0	2.29×10^{-2}	N/A	N/A	**0**	**100**	**0**	**100**	0.273	0.074
f_3	30	0	4.83×10^{-3}	**0**	**100**	**0**	**100**	**0**	**100**	0.417	0.139
f_4	30	0	0.3	N/A	N/A	**0**	**100**	**0**	**100**	0.324	0.075
f_5	30	0	5.06	**0**	66	**0**	**100**	**0**	**100**	0.871	0.139
f_6	30	0	**0**	N/A	N/A	**0**	**100**	**0**	**100**	0.770	0.313
f_7	30	0	0.3	N/A	N/A	0.309	20	**0.176**	**100**	0.814	0.243
f_8	30	-12569.5	-12554.5	-9094.7	0.00	-11740.7	0.00	$\mathbf{-12569.5}$	**100**	0.184	0.044
f_9	30	0	0.046	90.541	0.00	7.084	0.00	**0.000**	**100**	0.225	0.057
f_{10}	30	0	0.00483	17.987	0.00	**0.00000**	**100**	**0.00000**	**100**	0.241	0.099
f_{11}	30	0	0.0249	**0.00**	**100**	**0.0000**	**100**	**0.0000**	**100**	0.800	0.377
f_{12}	30	0	**0**	N/A	N/A	**0**	**100**	**0**	**100**	0.168	0.102
f_{13}	30	0	1.6×10^{-4}	N/A	N/A	**0**	90.0	**0**	**100**	0.487	0.292
f_{14}	2	0.9980	1.22	**0.9980**	8.0	**0.9980**	**100**	**0.9980**	**100**	0.259	0.242
f_{15}	4	0.0003075	4.7×10^{-4}	**0.0003075**	35.0	**0.0003075**	**100**	**0.0003075**	**100**	0.499	0.155
f_{16}	2	-1.0316	-1.03	$\mathbf{-1.0316}$	60.0	$\mathbf{-1.0316}$	**100**	$\mathbf{-1.0316}$	**100**	1.817	0.260
f_{17}	2	0.397887	0.398	**0.397887**	**100**	**0.397887**	**100**	**0.397887**	**100**	1.203	0.194
f_{18}	2	3	**3.0**	**3.000**	70	**3.000**	**100**	**3.000**	**100**	0.637	0.097
f_{19}	4	-3.86278	-3.86	$\mathbf{-3.86278}$	63	$\mathbf{-3.86278}$	**100**	$\mathbf{-3.86278}$	**100**	1.175	0.356
f_{20}	6	-3.32	-3.28	$\mathbf{-3.32}$	69	$\mathbf{-3.32}$	**100**	$\mathbf{-3.32}$	**100**	0.475	0.167
f_{21}	4	-10.153	-6.86	$\mathbf{-10.153}$	32	$\mathbf{-10.153}$	**100**	$\mathbf{-10.153}$	**100**	0.908	0.100
f_{22}	4	-10.403	-8.27	$\mathbf{-10.403}$	43	$\mathbf{-10.403}$	**100**	$\mathbf{-10.403}$	**100**	0.935	0.139
f_{23}	4	-10.536	-9.10	$\mathbf{-10.536}$	47	$\mathbf{-10.536}$	**100**	$\mathbf{-10.536}$	**100**	1.620	0.231

where $r_i = ||x - x_i||_2$, c is a normalization factor, and s_i and t_i are, respectively:

$$\begin{aligned} s_i &= \frac{w_i log_e \frac{k^2 T^2 + (F-\mu_i)^2}{(F-\mu_i)^2}}{\sum_{j=1}^{n} w_j log_e \frac{k^2 T^2 + (F-\mu_j)^2}{(F-\mu_j)^2}} \\ t_i(r_i) &= \frac{2k^2 r_i}{log_e \frac{k^2 T^2 + (F-\mu_i)^2}{(F-\mu_i)^2} (k^2 r_i^2 + (F-\mu_i)^2)}. \end{aligned} \tag{11}$$

Here, T is a bound that limits the distance between a sample to be generated and any of the samples generated already. The use of T is necessary as we have little confidence about the objective value of points far away from samples generated already. To generate a sample according to (10), we can use the composition method since s_i and t_i are normalized probability vector and density function, respectively.

3 Results on Using Bayesian Sampling in Simulated Annealing

In this section, we show our results in applying Bayesian sampling in SA and compare the performance of our modified SA, called SABS, with DONLP2, a sequential quadratic programming (SQP) package [5] for solving nonlinear programming problems, and traditional SA implemented by SIMANN [2]. SABS was built by replacing pure random sampling in SIMANN by our Bayesian sampling procedure.

Table 1 shows the results of applying these algorithms on 23 optimization benchmarks collected in [6]: f_1 to f_7 are unimodal functions, f_8 to f_{13} are multimodal functions with many local minima, and f_{14} to f_{23} have a few local minima. For comparison, we also list the results evaluated by evolutionary programming (EP) [6].

Our results show that SABS performs the best in terms of both solution quality and probability of finding global optima. It performs much better than EP in solving multimodal functions. It performs better in solving complex multimodal functions than SQP whose solution quality may depend heavily on its starting points. SABS generates less samples than traditional SA, especially in solving multimodal functions with a large number of local minima. SABS also takes less time in most cases to find global solutions. Functions for which SABS takes longer time are those very simple unimodal functions, such as f_1, and simple low-dimensional functions with a

Table 2. Experimental results comparing SA and SABS in evaluating modified Rastrigin functions of various dimensions. See Table 1 for explanation of "Succ. ratio" and "CPU time". Note that the result of SA on the 60-dimensional function is listed here only for computing the overhead of Bayesian analysis.

Number of dimensions	SABS					SA				
	Best sol.	Avg. sol.	Succ. ratio	No. of samples	CPU time (sec.)	Best sol.	Avg. sol.	Succ. ratio	No. of samples	CPU time (sec.)
10	0.0	0.0	10/10	3.055×10^5	15.40	0.0	0.6	4/10	10^7	79.34
20	0.0	0.0	10/10	8.240×10^5	54.3	3.0	5.3	0/10	4×10^7	583.1
30	0.0	0.0	10/10	5.652×10^6	472.4	7.1	10.0	0/10	10^8	2102.9
40	0.0	0.0	10/10	8.144×10^6	860.2	7.1	21.2	0/10	2×10^8	5746.7
50	0.0	0.0	10/10	1.278×10^7	1521.1	14.9	22.1	0/5	10^9	35296.3
60	0.0	0.0	10/10	1.973×10^7	2691.1	166.9	182.9	0/5	5×10^7	2551.84

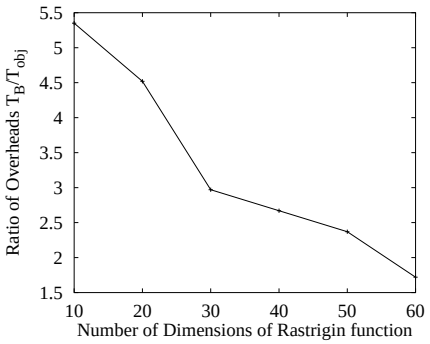

Figure 2. SABS overhead versus problem size

few local minima, such as f_{16}, f_{17}, f_{19} and f_{23}.

The reason why SABS works well is attributed to Bayesian sampling that helps provide high-quality samples, leading a search quickly to low-energy states and convergence. By modeling the surface of an objective function, Bayesian sampling also helps a search perceive a good region quickly instead of a blind search. Consequently, SABS is much more robust in achieving global convergence.

Bayesian analysis generally incurs additional computational overhead. Define the normalized overhead to be the CPU time for Bayesian analysis T_B normalized by the time for objective-function evaluation T_{obj}. Assuming that the CPU time for SABS is attributed to objective-function evaluations and Bayesian analysis only and that the CPU time for SA is due to function evaluations alone, we can compute the normalized Bayesian overhead $\frac{T_B}{T_{obj}}$ from the data in Table 2 that lists the results of applying SABS and SA to evaluate 10- to 60-dimensional modified Rastrigin functions.

Figure 2 plots the normalized overhead in evaluating modified Rastrigin functions of various dimensions. The results illustrate an important observation that more complex objective functions (in Rastrigin functions with higher dimensions) lead to smaller normalized overheads in Bayesian analysis.

References

[1] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York-Berlin-Heidelberg-Tokyo, 1986.

[2] W. L. Goffe, G. D. Ferrier, and J. Rogers. Global optimization of statistical functions with simulated annealing. *Journal of Econometrics*, 60:65–99, 1994.

[3] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[4] C. Koulamas, S. Antony, and R. Jaen. A survey of simulated annealing applications to operations research problems. *Omega*, 22:41–56, 1994.

[5] P. Spellucci. An SQP method for general nonlinear programs using only equality constrained subproblems. *Mathematical Programming*, 82:413–448, 1998.

[6] Xin Yao. Evolutionary programming made faster. *IEEE Tran. on Evolutionary Computation*, 3(2):82–102, July 1999.