Regression-based Parameter Optimization for Binary Output Systems

Jun Cao
Department of Electronic Engineering
Tsinghua University, Beijing, P.R.China, 100084
cao-j12@mails.tsinghua.edu.cn

Huimin Ma
Department of Electronic Engineering
Tsinghua University, Beijing, P.R.China, 100084
mhmpub@tsinghua.edu.cn

***Abstract*—Binary Output Systems (BOSs) generate Bernoulli distributed outputs with the given parameter. Such systems are quite common in various fields, and the system performance is usually measured by success rate or correct rate. Traditional parameter optimization methods utilize system performance approximations calculated by averaging the binary outputs. The binary outputs are used only once in the approximation process, and little about the internal relationship between different binary outputs is considered. In this article, we propose a novel method named Iterative Binary Regression (IBR) for parameter optimization of BOSs. IBR tackles the binary outputs directly and utilizes every binary output repeatedly in the regression process. This feature makes IBR particularly effective when the amount of available binary outputs is small. Considering the distribution of the binary outputs, we propose regression methods based on Least Squared Estimation (LSE), Weighted Least Squared Estimation (WLSE) and Maximum Likelihood Estimation (MLE) for IBR. Numerical comparison with Simultaneous Perturbation Stochastic Approximation (SPSA) and Blind Random Search on hypothesized and real BOSs is provided to show the effectiveness of IBR.**

Keywords – parameter optimization; binary output system; iterative binary regression; utilization rate

I. Introduction

Parameter optimization or parameter tuning is always an important part for practical systems. It is a common problem in fields like machine learning, image processing, control, simulation and others. Traditional parameter optimization methods do not interact directly with the raw outputs of the system. These methods approximate the performance of the system with the data and some performance measure method (usually the average method), and treat the performance as a function with the parameter. Such methods include classical random search algorithms [1], gradient-based search algorithms [2], and the up-to-date biologically-inspired algorithms [3][4]. They are perfectly applicable if the approximation process costs a small amount of resource (e.g., time, money) and the amount of available approximations is large enough. For BOSs, because of the large variance of the binary output, the performance approximation of the system should be the average of a large amount of system outputs with the same parameter [5]. When a single system execution costs much, traditional parameter optimization algorithms meet a great challenge. Researchers in image classification field usually build up large image data sets [6] so as to obtain an accurate correct rate of classification and to compare different algorithms convictively and conveniently [7]. It usually takes minutes or even hours to obtain the correct rate, thus traditional parameter optimization methods are not proper for such BOSs.

In this article, we propose a novel method named IBR for the parameter optimization problem of BOSs. IBR handles the binary outputs directly and improves the utilization rate of these data. Thus, IBR can be particularly applied to problems where the amount of the available binary outputs is very limited.

II. Iterative Binary Regression

The background of our research is finding the best countermeasure for jamming in an infrared simulation system [8] (Fig. 1). This system is a complicated software system that simulates the dynamic process during the approach of an infrared tracking device and a ship. The ship is able to throw out jamming according to a given countermeasure setting so as to escape. A binary output is generated after a system execution which costs about 20s. The performance of a given countermeasure setting is the escape rate of the ship, and it is approximated by averaging the corresponding binary outputs. This problem is a typical parameter optimization problem for a BOS.

Fig. 1. The infrared simulation system.

Parameter optimization can be abstracted to the structure shown in Fig. 2. System S is what actually serves in practice. It comprises an input X, an output Y and a parameter θ. The performance measure part uses S together with some specific method (mostly the average method) to measure the performance of the current θ. The performance function $PF(\theta)$ is used to represent the current performance. X may be a determinate variable or not; S may contain randomness or not. The property of Y depends on S and X, but for most practical systems, Y is a stochastic variable. For BOSs, Y is

a Bernoulli distributed variable, $PF(\theta)$ is the expectation of Y with θ (Eq. 1), and the parameter optimization problem can be represented as Eq. 2.

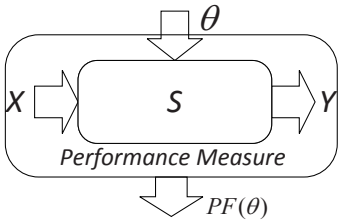

Fig. 2. Abstraction of parameter optimization.

$$PF(\theta) = E(Y(X;\theta)) \tag{1}$$

$$\theta_m = \arg\max_{\boldsymbol{\omega}\in\Omega_{\boldsymbol{\omega}}} E(Y(X;\theta)) \tag{2}$$

Traditional parameter optimization methods utilize the approximation of $PF(\theta)$ (denoted as $\hat{PF}(\theta)$ where θ is some value) to find the best parameter, as shown in Eq. 3. For BOSs, $\hat{PF}(\theta)$ is the average of N binary outputs with θ (Eq. 4), hence $\hat{PF}(\theta)$ has a standard deviation up to $0.5/\sqrt{N}$ (Eq. 5). Eq. 3, 4 show that the binary outputs are used only once to approximate the corresponding performance. When the available amount of data is small, $\hat{PF}(\theta)$ could be seen as $PF(\theta)$ corrupted with a significant noise (Fig. 3). In this case, methods based on $\hat{PF}(\theta)$ are ineffective and misleading.

$$\theta_m = \arg\max\{\hat{PF}(\theta_1), \hat{PF}(\theta_2), ...\} \tag{3}$$

$$\hat{PF}(\theta) = \frac{\sum_{i=1}^{N} Y_i(\theta)}{N} \tag{4}$$

$$\sigma(\hat{PF}(\theta)) = \sqrt{\frac{PF(\theta)(1-PF(\theta))}{N}} \in [0, \frac{1}{2\sqrt{N}}] \tag{5}$$

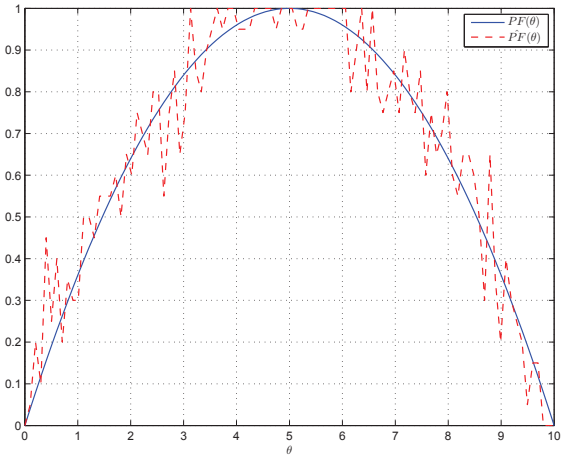

Fig. 3. An example of a noisy $\hat{PF}(\theta)$ when N is 20.

The valuable computational resource should be used to explore the whole parameter space, not to obtain an accurate approximation. Iterative Binary Regression (IRB) differs from traditional methods on this aspect. Given N system executions, generating

$$\{(\theta_1, Y_1(\theta_1)), (\theta_2, Y_2(\theta_2)), ..., (\theta_N, Y_N(\theta_N))\}$$

is better than generating

$$\{(\theta, Y_1(\theta)), (\theta, Y_2(\theta)), ..., (\theta, Y_N(\theta))\}$$

on the aspect of exploration ability. We call this process *Expanding*. With all of the data from *Expanding* process, a proper regression method R is able to grab the global feature of the underlying $PF(\theta)$ (Fig. 4, Eq. 6). In the regression process, all of the data are used repeatedly to obtain a better approximation of $PF(\theta)$. The approximation of $PF(\theta)$ is denoted as $\hat{PF}(\theta;\boldsymbol{\omega})$ where θ is a variable. The global optimum of $\hat{PF}(\theta;\boldsymbol{\omega})$ provides an approximation of the global optimum of $PF(\theta)$. The optimization of $\hat{PF}(\theta;\boldsymbol{\omega})$ is a traditional optimization problem with many effective algorithms.

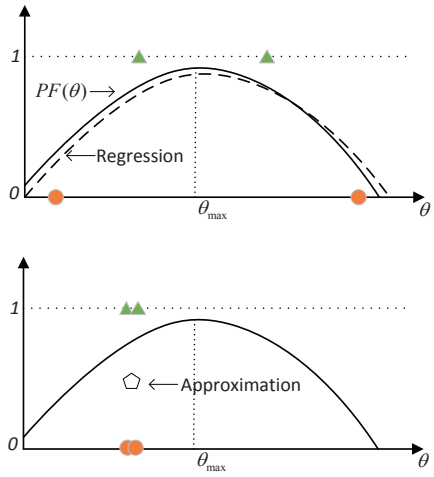

Fig. 4. Intuitive diagram of the effect of *Expanding* (top) and tradition (bottom).

$$\hat{PF}(\theta;\boldsymbol{\omega}) = R((\theta_1, Y_1(\theta_1)), (\theta_2, Y_2(\theta_2)), ..., (\theta_N, Y_N(\theta_N))) \tag{6}$$

Details of IBR are shown in Algorithm 1. The existence of $\mathcal{D}$ and the regression process make IBR completely different from traditional methods. All of the data in $\mathcal{D}$ are of the same importance, and all of them participate in finding the optimum throughout the algorithm.

III. Regression Method in IBR

IBR presents an instructive algorithm architecture. The effectiveness of the final applicable algorithm varies with the choice of the regression method. Mostly, little about the regularity of the objective BOS is known, thus it is advisable to apply linear regression model (Eq. 7). The choice of basis function depends on the problem. One dimensional parameter space is firstly considered.

The binary output of a BOS can be split into $PF(\theta)$ and noise. Eq. 8 - 11 show that the binary output is corrupted with variant variance Bernoulli noise, rather than equal variance

Algorithm 1: IBR

$\mathcal{D} = \emptyset$;
for $i = 1..max\_iteration$ **do**
  for $j = 1..N$ **do**
    Generate θ_{ij}, $\theta_{ij} \sim U(\Omega_\theta)$;
    Generate $Y_{ij}(\theta_{ij}) = Y(X; \theta_{ij})$ by executing the system;
  end
  Add $\{(\theta_{i1}, Y_{i1}(\theta_{i1})), (\theta_{i2}, Y_{i2}(\theta_{i2})), ..., (\theta_{iN}, Y_{iN}(\theta_{iN}))\}$ to $\mathcal{D}$;
  Execute regression R with $\mathcal{D}$ and get $\hat{PF}(\theta; \boldsymbol{\omega})_i$;
  if $\|\hat{PF}(\theta; \boldsymbol{\omega})_i - \hat{PF}(\theta; \boldsymbol{\omega})_{i-1}\| \leq \epsilon$ **then**
    break;
  end
end
return the optimum of $\hat{PF}(\theta; \boldsymbol{\omega})_i$ as θ_m;

Gaussian noise which is a common hypothesis of general regression problems.

$$y = \hat{PF}(\theta; \boldsymbol{\omega}) = \omega_0 + \sum_{i=1}^{M} \omega_i \phi_i(\theta) = \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta) \tag{7}$$

$$Y(\theta) = PF(\theta) + \Big(Y(\theta) - PF(\theta)\Big) = PF(\theta) + N(\theta) \tag{8}$$

$$f_{N(\theta)}(n) = \begin{cases} PF(\theta) & \text{if } n = 1 - PF(\theta) \\ 1 - PF(\theta) & \text{if } n = -PF(\theta) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$EN(\theta) = 0 \tag{10}$$

$$var(N(\theta)) = PF(\theta)\Big(1 - PF(\theta)\Big) \tag{11}$$

A. LSE-IBR

Usually, researchers tend to apply Least Squares Estimation (LSE) to minimizes the squared error between observed data and the expected values calculated by the regression model (Eq. 12). With LSE adopted in IBR, we get Least Squares Estimation IBR (LSE-IBR). LSE corresponds to MLE when the noise is Gaussian with equal variance.

$$\boldsymbol{\omega}_{LSE} = \arg\max_{\boldsymbol{\omega} \in \Omega_{\boldsymbol{\omega}}} \sum_{i=1}^{N} \Big(Y_i(\theta_i) - \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta)\Big)^2 \tag{12}$$

B. WLSE-IBR

LSE does not take advantage of the distribution property of the noise. But for the regression problem in IBR, each binary output is from a Bernoulli distribution that can be modeled with the unknown $PF(\theta)$. Taking this into account, we provide Weighted Least Squared Estimation IBR (WLSE-IBR) and Maximum Likelihood Estimation IBR (MLE-IBR). WLSE is an extension of LSE when the variances of data are known. Observations with large variance are less accurate, so they should play a smaller role in the process of summing squared errors. As is shown in Eq. 11, the variance of the noise is determined by the unknown $PF(\theta)$. Hence, we get the WLSE estimator as Eq. 13. Eq. 13 proves to be convex.

$$\boldsymbol{\omega}_{WLSE} = \arg\max_{\boldsymbol{\omega} \in \Omega_{\boldsymbol{\omega}}} \sum_{i=1}^{N} \frac{\Big(Y_i(\theta_i) - \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta)\Big)^2}{\boldsymbol{\omega}^T \boldsymbol{\phi}(\theta)(1 - \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta))} \tag{13}$$

C. MLE-IBR

MLE is a procedure of finding the value of the unknown parameters in the given probability distribution model that makes the probability of the observed data maximum. In practice, an accurate probability distribution model of data is not easy to obtain, but for the regression problem in IBR, the Bernoulli noise can be easily modeled by the unknown $PF(\theta)$ (Eq. 14, 15). Finally, we get the MLE estimator as Eq. 16. Eq. 16 proves to be convex.

$$p(Y(\theta)) = PF(\theta)^{Y(\theta)}\Big(1 - PF(\theta)\Big)^{1 - Y(\theta)} \tag{14}$$

$$p(\mathcal{D}) = \prod_{i=1}^{N} PF(\theta_i)^{Y_i(\theta_i)}\Big(1 - PF(\theta_i)\Big)^{1 - Y_i(\theta_i)} \tag{15}$$

$$\boldsymbol{\omega}_{MLE} = \arg\max_{\boldsymbol{\omega} \in \Omega_{\boldsymbol{\omega}}} \sum_{Y_i(\theta_i)=1} ln\Big(\boldsymbol{\omega}^T \boldsymbol{\phi}(\theta_i)\Big) + \sum_{Y_i(\theta_i)=0} ln\Big(1 - \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta_i)\Big) \tag{16}$$

D. Constraint

$PF(\theta)$ is the expectation of a Bernoulli distributed variable $Y(\theta)$, hence $PF(\theta)$ is certain to lie in $[0, 1]$. Consequently, a bound constraint on $\hat{PF}(\theta; \boldsymbol{\omega})$ should be added to the mentioned three regression methods (Eq. 17). However, this constraint is hard to handle because of the arbitrariness of θ. In practice, we choose K points uniformly from the parameter space to approximate the arbitrariness of θ as Eq. 18. Eq. 18 also proves to be convex.

$$subject\ to: 0 \leq \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta) \leq 1, \forall \theta \in \Omega_\theta \tag{17}$$

$$subject\ to: 0 \leq \boldsymbol{\omega}^T \boldsymbol{\phi}(\theta_k) \leq 1 \quad \theta_k \sim U(\Omega_\theta), k = 1, 2, ..., K \tag{18}$$

As all of the objective functions in the regression method and the constraint are convex, we can use convex optimization tools (e.g., CVX in Matlab) to solve the regression model. As the dimension of the parameter space increases, the size of $\boldsymbol{\omega}$ grows rapidly. It is actually a power law growth. The phenomenon is called the curse of dimensionality. Hence, IBR is suitable for parameter optimization of BOSs with low dimensional parameter space.

IV. Numerical Experiments

Numerical experiments for stochastic algorithms or stochastic problems cannot guarantee to give the same numerical result even for the same problem. In order to compare different algorithms, various acceptance regions and the corresponding acceptance probabilities are provided in the numerical experiments. As $PF(\theta) \in [0,1]$ is always true, we define the acceptance region in a simpler way as Eq. 19, thus the corresponding probability is as Eq. 20. One ϵ corresponds to one acceptance probability, thus we get an acceptance curve to describe the effectiveness of the parameter optimization algorithm on the current problem. We also define the average performance as Eq. 21 to indicate the average performance of all possible results given by the parameter optimization algorithm. The average performance enables us to compare different algorithms numerically.

$$AR(\epsilon) = \{\theta | PF(\theta) \geq \epsilon, \theta \in \Omega_\theta\} \tag{19}$$

$$ARP(\epsilon) = P(\theta_m \in AR(\epsilon)) \tag{20}$$

$$AP = E_{\theta_m}(PF(\theta_m)) = \int_{\Omega_\theta} PF(\theta) p_{\theta_m}(\theta) d\theta \tag{21}$$

We experimented LSE-IBR, WLSE-IBR, MLE-IBR, SPSA and Blind Random Search [1] on three different $PF(\theta)$ functions. These functions include a unimodal function (Fig. 5 top, Eq. 22, Quadratic), a multimodal function (Fig. 5 middle, Eq. 23, Guassian2) and a complex function from the infrared simulation system (Fig. 1, 5 bottom, OldTrack). In the experiments, we used at most 100 binary outputs, which is far less than the amount that traditional methods need. Biologically-inspired algorithms are based on swarm intelligence, thus need much more performance approximations than other traditional methods, and are sure to be ineffective with only 100 binary outputs. We chose SPSA to represent traditional methods. The implementation of SPSA provided in [2] was adopted, which is proved to be practical and effective for unimodal $PF(\theta)$ if the noise in $\hat{PF}(\theta)$ is low enough. In the experiments, the basic function was power function, M was 5, N was 20, $max\_iteration$ was 5 and the ϵ in the stop criteria was 0.005. CVX in matlab was used. The acceptance curves for the three problems are shown in Fig. 6.

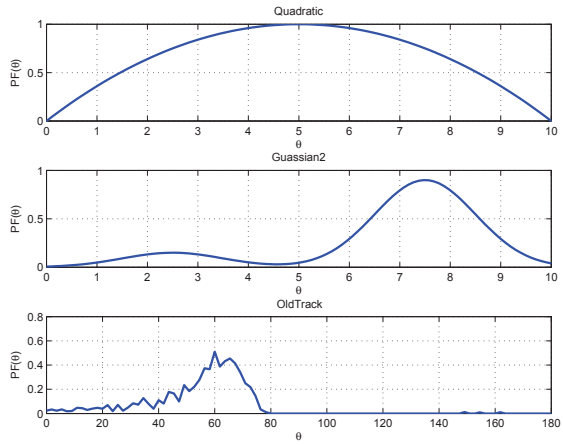

Fig. 5. Three different $PF(\theta)$ functions.

$$PF(\theta) = -\frac{1}{25}\theta^2 + \frac{2}{5}\theta \tag{22}$$

$$PF(\theta) = \frac{3}{10}e^{-\frac{(\theta-\frac{5}{2})^2}{2}} + \frac{9}{10}e^{-\frac{(\theta-\frac{15}{2})^2}{2}} \tag{23}$$

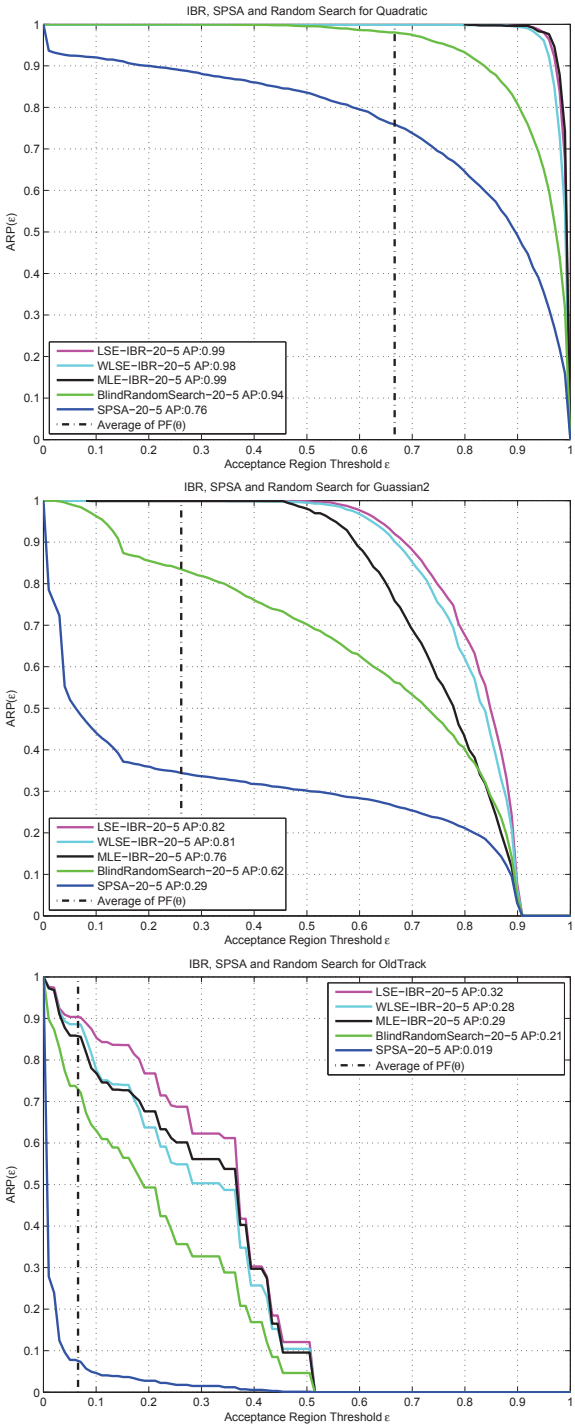

Fig. 6. Comparison between IBR, SPSA and Blind Random Search. The curve *LSE-IBR-20-5* means that LSE-IBR is used with N equals 20 and $max\_iteration$ equals 5. The average of $PF(\theta)$ means the average performance of $PF(\theta)$ itself. *BlindRandomSearch-20-5* means 5 approximations are used.

In the three cases, IBR is much better than SPSA and Blind Random Search. IBR has the best performance on unimodal, multimodal or complex $PF(\theta)$, thus it is reasonable for IBR to have good performance on other problems. SPSA presents the worst result, because it is really a great challenge for SPSA to tackle these problems where the amount of performance approximations is so limited and the noise is so significant.

In Fig. 6, LSE-IBR provides the best result consistently among the three IBR algorithms. This is a surprise because WLSE-IBR and MLE-IBR exploit the distribution information of the binary outputs rather than just minimize the squared error. For MLE-IBR, the reason may be the inaccurate solution of $\boldsymbol{\omega}_{MLE}$ as the objective function contains log. Warnings of inaccurate solution were also shown in the optimization process of CVX.

Fig. 8 and 9 provide a successful and failed example respectively. In Fig. 8, $\hat{PF}(\theta;\boldsymbol{\omega})$ approaches $PF(\theta)$ successfully because the distribution of $\mathcal{D}$ matches the underlying $PF(\theta)$ well. This is a case of high likelihood, in comparison with the failed case in Fig. 9 where the generated data are even not very symmetrical.

OldTrack comes from the infrared simulation system (Fig. 1). The system costs about 20 seconds per execution, and we spent almost two days executing the system 10,000 times to get the $PF(\theta)$ in one common case (Fig. 1). In this case, we set the threat angle to be 90 deg, made the direction of the wind uniformly chosen from $[-180, 180]$ deg, attached a typical tracking method, set other environment parameters to be their own typical value, and fixed all other parameters of the countermeasure except the horizontal angle of the infrared jamming (denoted as θ). With IBR, we need at most 100 system executions to find the best θ, which costs about half an hour. As IBR needs N independent binary outputs per iteration, we could execute N or a suitable number of system instances parallelly to shorten the total execution time to minutes. The result in Fig. 6 shows that LSE-IBR can give a horizontal angle in the region shown in Fig. 7 with a probability of 0.90.

Table I provides more numerical comparison of the algorithms.

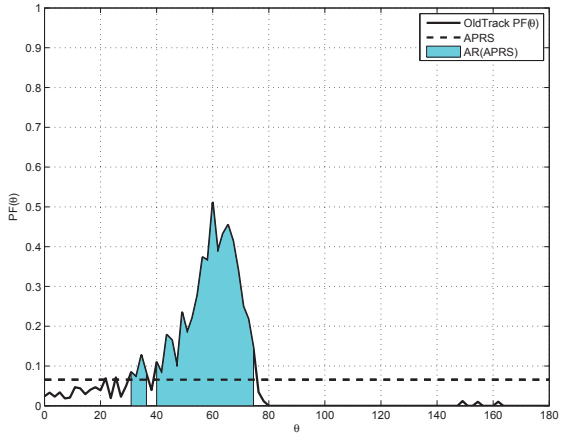

Fig. 7. AR(Average of $PF(\theta)$) of OldTrack.

V. Conclusion

In this article, we proposed a novel algorithm named IBR for parameter optimization. Traditional algorithms use system performance approximations that are evaluated with a large amount of system executions. IBR needs data to be generated at different places in the parameter space, and obtains the global feature of the performance function with some proper regression method. This improves the exploration ability of the algorithm and makes it possible to repeatedly utilize the valuable data. Hence IBR can apply to systems that are expensive in terms of time or money. LSE-IBR, WLSE-IBR and MLE-IBR are three applicable schemes of IBR. WLSE-IBR and MLE-IBR take advantage of the Bernoulli distribution information of the data. We compared LSE-IBR, WLSE-IBR, MLE-IBR, SPSA and Blind Random Search, and the result showed that LSE-IBR, WLSE-IBR, MLE-IBR are far better than traditional methods. We also experimented on a real BOS which costs 20s for a single execution, and the result proved that IBR saves much time and is able to make a great contribution in practice.

References

[1] J. C. Spall, "Stochastic optimization," in *Handbook of computational statistics*. Springer, 2012, pp. 173–201.

[2] J. C. SPALL, "Simultaneous perturbation stochastic approximation," *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*, pp. 176–207, 2003.

[3] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[4] I. Aydin, M. Karakose, and E. Akin, "A multi-objective artificial immune algorithm for parameter optimization in support vector machine," *Applied Soft Computing*, vol. 11, no. 1, pp. 120–129, 2011.

[5] F. Coenen and P. Leng, "Obtaining best parameter values for accurate classification," in *Data Mining, Fifth IEEE International Conference on*. IEEE, 2005, pp. 4–pp.

[6] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2010.

[7] X. Chen and H. Ma, "Learning a compact latent representation of the bag-of-parts model," in *IEEE ICIP*, 2014.

[8] M. H. Hou Yu, "Analysis of an infrared interference antagonizing window tracking algorithm," *IET Conference Proceedings*, pp. 496–499(3), January 2012.

TABLE I. COMPARISON OF ALGORITHMS

Algorithm	Problem	ARP(Average of $PF(\theta)$))	AP	Algorithm	Problem	ARP(Average of $PF(\theta)$))	AP
Blind Random Search	Quadratic	0.98	0.94	Blind Random Search	Guassian2	0.84	0.62
SPSA	Quadratic	0.76	0.76	SPSA	Guassian2	0.34	0.29
LSE-IBR	Quadratic	**1.00**	**0.99**	LSE-IBR	Guassian2	**1.00**	**0.82**
WLSE-IBR	Quadratic	**1.00**	0.98	WLSE-IBR	Guassian2	**1.00**	0.81
MLE-IBR	Quadratic	**1.00**	**0.99**	MLE-IBR	Guassian2	**1.00**	0.76
Blind Random Search	OldTrack	0.74	0.21				
SPSA	OldTrack	0.08	0.02				
LSE-IBR	OldTrack	**0.90**	**0.32**				
WLSE-IBR	OldTrack	0.89	0.28				
MLE-IBR	OldTrack	0.86	0.29				

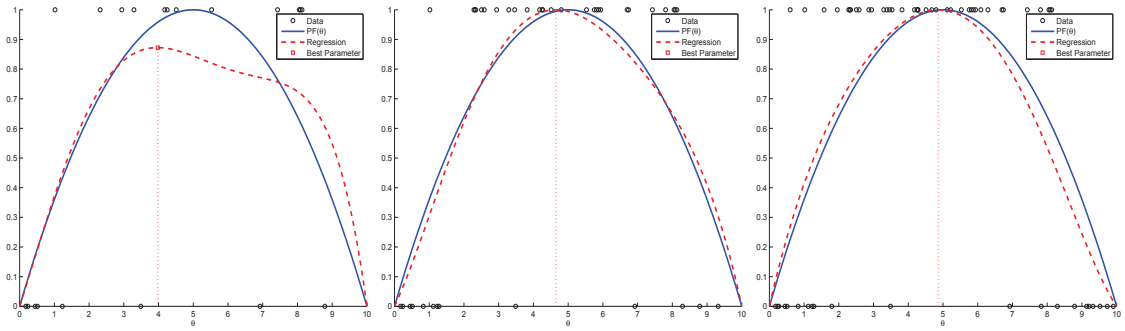

Fig. 8. A successful example.

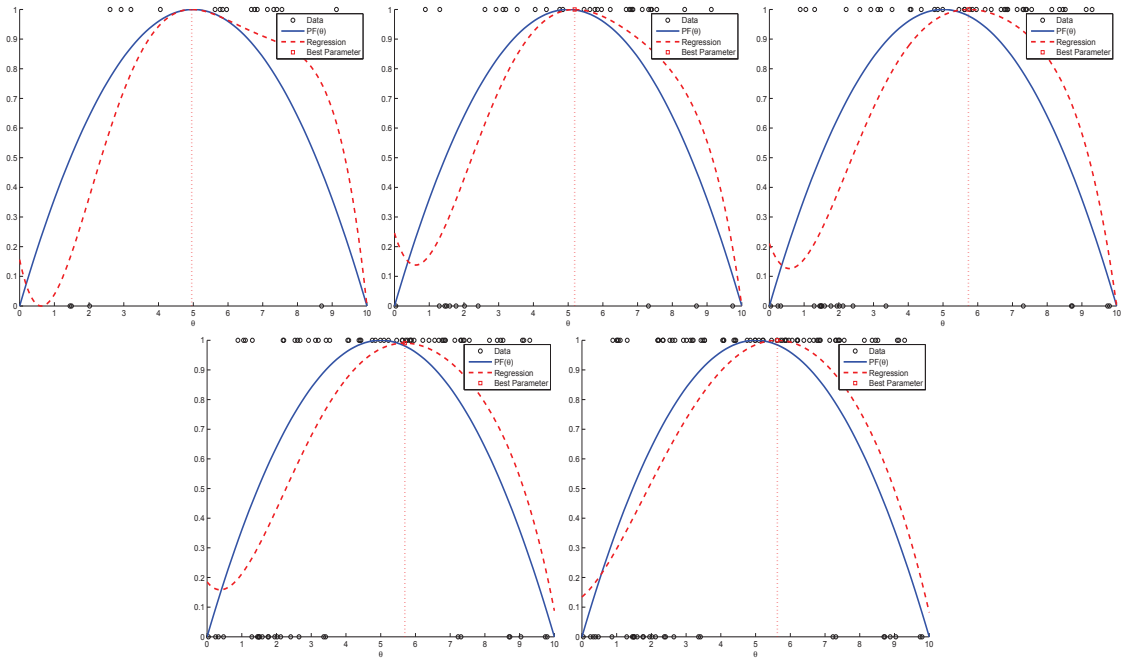

Fig. 9. A failed example.