Stochastic Approximation for Expensive One-Bit Feedback Systems

Xiaoqin Zhang, Huimin Ma*, and Jinghuan Wen

Abstract: One-bit feedback systems generate binary data as their output and the system performance is usually measured by the success rate with a fixed parameter combination. Traditional methods need many executions for parameter optimization. Hence, it is impractical to utilize these methods in Expensive One-Bit Feedback Systems (EOBFSs), where a single system execution is costly in terms of time or money. In this paper, we propose a novel algorithm, named Iterative Regression and Optimization (IRO), for parameter optimization and its corresponding scheme based on the Maximum Likelihood Estimation (MLE) method and Particle Swarm Optimization (PSO) method, named MLEPSO-IRO, for parameter optimization in EOBFSs. The IRO algorithm is an iterative algorithm, with each iteration comprising two parts: regression and optimization. Considering the structure of IRO and the Bernoulli distribution property of the output of EOBFSs, MLE and a modified PSO are selected to implement the regression and optimization sections, respectively, in MLEPSO-IRO. We also provide a theoretical analysis for the convergence of MLEPSO-IRO and provide numerical experiments on hypothesized EOBFSs and one real EOBFS in comparison to traditional methods. The results indicate that MLEPSO-IRO can provide a much better result with only a small amount of system executions.

Key words: stochastic approximation; parameter optimization; one-bit feedback system; regression; Maximum Likelihood Estimation (MLE)

1 Introduction

Optimization for problems where the objective function is sufficiently complex and corrupted with noise is always a challenge for researchers. Such problems are quite common. Many practical systems are so complicated that even a simplified mathematical model is too complex. Data obtained in practice are from the real environment or generated by a stochastic system, thus they are usually corrupted with noise. Optimization for such problems is called stochastic optimization. Stochastic Approximation (SA) is the basis of traditional stochastic algorithms. Finite-Difference SA (FDSA) and Simultaneous Perturbation SA (SPSA) do optimization with gradient approximations evaluated by the noisy objective function. Besides, Swarm Intelligence (SI) algorithms like Particle Swarm Optimization (PSO) can also apply to stochastic optimization because of their strong search ability. Both of SA and SI utilize the objective function directly. However, in the world of parameter optimization, analyst can only interact. The objective function depends on the performance measure method and the outputs of the system. The raw data are the outputs of the system, and traditional methods neglect the distribution information in the raw data. The distribution information is what we will exploit in the Iterative Regression and Optimization (IRO) algorithm.

Parameter optimization is a common denominator of a variety of problems in fields such as machine learning, image processing, control, simulation,
and manufacturing. Parameter optimization can be abstracted to the structure shown in Fig. 1. System $S$ is what we actually interactive with. It comprises an input $X$, an output $Y$, and a parameter combination $\theta$. The parameter combination is a bulk combination of parameters. The performance measure uses $S$ together with some specific method (mostly the simple averaging method) to measure the performance of the current $\theta$. The performance function $PF(\theta)$ or the objective function is used to represent the current performance. Input $X$ may be a determinate variable or not; $S$ may be an absolutely determinate or contain some randomness. The property of $Y$ depends on $S$ and $X$, but for most practical systems, $Y$ is a stochastic variable.

When $X$ and $S$ are analyzable and simple enough, $PF(\theta)$ has a determinate mathematical representation. Moreover, parameter optimization for $S$ is converted to a problem of maximizing a determinate function. In this case, the analyst does not interact with $S$, but optimizes $PF(\theta)$ directly. Algorithms based on the direct search\[^3\], gradient (e.g., Newton-Raphson), or SI\[^4\] can be effectively applied. Conversely, there are situations where $S$ is extremely complicated, or the distribution of $X$ has no closed-form representation, or $PF(\theta)$ is nonconvex. In this case, $PF(\theta)$ can only be approximated with $\{Y_1(X_1; \theta), Y_2(X_2; \theta), ..., Y_N(X_N; \theta)\}$ with a finite sample set $\{X_1, X_2, ..., X_N\}$ (usually called a test set) from the ensemble of $X$. The approximation is denoted as $\hat{PF}(\theta)$, where $\theta$ is a value. $S$ must be executed $N$ times to obtain $\hat{PF}(\theta)$. When the test set is adequate, $PF(\theta)$ can be treated as $PF(\theta)$ corrupted with ignorable noise, and gradient-free algorithms can still provide an acceptable result. Researchers of image processing are faced up with unanalyzable real world images. Image data sets like ImageNet\[^5\] comprise thousands of images as a substitute for the real world, and $\hat{PF}(\theta)$ applied to these data sets is used to directly compare the different parameter combinations and different algorithms.

However, $S$ may be expensive in terms of time or money for a single execution. We call such a system an Expensive Output System (EOS). In this case, the number of system executions has an upper limit when applicability of the parameter optimization algorithm is taken into consideration. As a consequence, $\hat{PF}(\theta)$ has significant variance, or in other words, $PF(\theta)$ is corrupted with significant noise. Parameter optimization for EOSs can be converted to optimization for noisy functions. Stochastic search algorithms\[^1\] can apply to noisy and unknown objective functions. Direct search algorithms\[^3,6\] are based on direct use of $\hat{PF}(\theta)$. They are better at handling determinant, or slightly noisy objective functions. When coping with significant noise, direct search algorithms rely on some averaging method to reduce the impact of noise, or set an acceptance threshold in the stop criterion to increase the probabilistic evidence of the result. Both approaches greatly increase the number of system executions. Prototypical algorithms like the Robbins-Monro algorithm\[^7\] and the Kiefer-Wolfowitz algorithm\[^8\] are effective for problems with convex expectation functions. Both FDSA and SPSA\[^1,9\] modify the candidate through a stochastic gradient calculated with the noisy $\hat{PF}(\theta)$. They actually perform a type of averaging across iterations instead of at some candidate, thus saving some system executions. However, FDSA and SPSA still need too many system executions to provide an acceptable result for significantly noisy problems. Besides, FDSA and SPSA are based on the gradient descent method and are prone to converge to a local optimum. Moreover, the precious data generated by $S$ are used only once to obtain the approximation of the gradient, which is an inefficient usage of $S$. To solve nonconvex stochastic optimization problems, Ghadimi et al.\[^10\] proposed a randomized stochastic projected gradient algorithm, but this is also inefficient. Hence, parameter optimization for EOSs is a great challenge for traditional methods.

The problem addressed in this paper is the parameter optimization of Expensive One-Bit Feedback Systems (EOBFSs). The EOBFSs are an important and common subset of EOSs. Practical EOSs that use the correct rate or the success rate as the performance measure are EOBFSs. For example, a complex recognition system which takes a long period to recognize an object is an

---

**Fig. 1** Abstraction of parameter optimization.
EOBFS, and a real-time simulation system which takes a long time to give the result of hit or miss is also an EOBFS. Noise in PF(θ) of an EOBFS is more serious than that of an EOS because of the Bernoulli distributed data. For solving the parameter optimization of EOSs, we propose an effective and novel algorithm, IRO, and its corresponding scheme for EOBFSs, namely MLEPSO-IRO.

The rest of the paper is organized as follows. Section 2 gives an explicit formulation of the parameter optimization of EOBFSs; Section 3 describes the proposed algorithm, namely IRO and MLEPSO-IRO; Section 4 provides numerical experiments of MLEPSO-IRO. and its corresponding scheme for EOBFSs, namely IRO, and Section 5 presents the conclusion for the research.

2 Problem Formulation

An application of this research, which is also the background for this paper, is finding the best countermeasure for jamming in the infrared simulation system named ThuSIR[11,12] (Fig. 2). ThuSIR is a complex software system that simulates the dynamic process during the approach of an infrared tracking device and a ship. The ship can throw out jamming smoke according to a given countermeasure setting so as to escape. Binary feedback is generated after a system execution, with zero representing a failed jamming countermeasure and one representing a successful one. A single system execution of ThuSIR takes 20 s, so ThuSIR is a typical EOBFS, and the problem of finding the best countermeasure is a typical parameter optimization problem of an EOBFS (Eq. (1)). The performance of a given countermeasure setting of θ for ThuSIR is the success rate of the corresponding jamming countermeasure; hence, the performance function PF(θ) is the expectation of Y, as shown in (Eq. (2)).

\[
\begin{align*}
\theta_m &= \arg \max_{\theta \in \Omega_\theta} PF(\theta) \\
PF(\theta) &= E(Y(X; \theta))
\end{align*}
\]

Fig. 2 ThuSIR.

Typically, Y and X in the parameter optimization of EOBFSs are usually difficult or impossible to analyze, so Eq. (2) could not be directly applied. The traditional approach is to execute the system S for N times with input parameter θ fixed and uses an averaging method to approximate the expectation of Y (Eqs. (3) and (4)). The standard variance of the approximation PF(θ) is \(\sqrt{PF(\theta)(1-\text{PF}(\theta))/N} \leq 0.5/\sqrt{N}\), which is relatively large for Y (0 or 1) and decreases slowly with increasing N. Large N is unbearable for EOBFSs. This is the essential reason why traditional algorithms cannot be directly applied to parameter optimization of the EOBFSs.

\[
\begin{align*}
\hat{PF}(\theta) &= \frac{\sum_{i=1}^{N} Y_i(\theta)}{N} \\
\theta_m &= \arg \max\{\hat{PF}(\theta_1), \hat{PF}(\theta_2), \ldots\}
\end{align*}
\]

where \(\theta_m\) is a random variable. When comparing different algorithms for parameter optimization of EOBFSs, analysts are concerned with three aspects of the results:

(1) What is the probability of convergence to a given acceptance region?
(2) What is the average performance of all possible results?
(3) How many total system executions does the algorithm need?

For EOBFSs, 0 ≤ PF(θ) ≤ 1 is always true, so we define the Acceptance Region (AR), in a simplified way (Eq. (5)). Hence, the probability of the result in the given acceptance region, ARP, is provided by Eq. (6).

\[
\begin{align*}
\text{AR}(\epsilon) &= \{\theta | PF(\theta) \geq \epsilon, \theta \in \Omega_\theta\} \\
\text{ARP}(\epsilon) &= P(\theta_m \in \text{AR}(\epsilon))
\end{align*}
\]

The average performance is defined using Eq. (7). The average performance indicates the average performance of all possible results given by the parameter optimization algorithm. Hence, a trivial parameter optimization algorithm that selects \(\theta\) in \(\Omega_\theta\) under a uniform distribution as the result, namely \(\theta_m\) (the blind random search), has the average ARPS, as stated by Eq. (8). Analysts also care the ratio of the average performance and PF(\(\theta_{\max}\)) (Eq. (9)), which shows the average performance that is irrelevant to the maximum of PF(\(\theta\)). APRRS is the APR of blind search, which is the baseline of APR.

\[
\begin{align*}
\text{AP} &= E_{\theta_m}(PF(\theta_m)) = \int_{\Omega_\theta} PF(\theta)p_{\theta_m}(\theta)d\theta \\
\text{APRS} &= \int_{\Omega_\theta} \frac{PF(\theta)}{\text{size}(\Omega)} d\theta
\end{align*}
\]
Among all acceptance regions for Eqs. (5) and (6), there is a special region where $e = \text{APRS}$. APRS measures the probability of the result that is at least better than the average performance of the blind random search.

Aspects (1) and (2) are the most important criterions for stochastic algorithms when they are addressing general optimization problems. When it comes to parameter optimization of EOBFSs, aspect (3) is of the same, or even more importance. If an algorithm needs too many system executions, it will not be applicable for EOBFSs in practice. These three aspects will be used for comparing different algorithms in the numerical experiments.

### 3 IRO and MLEPSO-IRO

Traditional stochastic search algorithms exploit $\hat{\text{PF}}(\theta)$ to find the optimum (Eq. (4)). In order to obtain $\hat{\text{PF}}(\theta)$, $N$ system executions are needed with the same $\theta$. This is not effective for EOBFSs, where the total number of available system executions is very limited compared to the parameter optimization of general systems. The valuable data should be used repeatedly, rather than only once in the process of obtaining $\text{PF}(\theta)$. Also, the valuable computational resource should be used to explore the whole parameter space, not just to obtain an accurate $\text{PF}(\theta)$ with some $\theta$.

These two traditional shortcomings are overcome in IRO. First, given $N$ system executions, generating $\{(\theta_1, Y_1(\theta_1)), (\theta_2, Y_2(\theta_2)), \ldots, (\theta_N, Y_N(\theta_N))\}$ is better than generating $\{(\theta, Y_1(\theta)), (\theta, Y_2(\theta)), \ldots, (\theta, Y_N(\theta))\}$ on the aspect of parameter space exploration ability. The statement will be proved by experiment in Section 4, and we call this phenomenon Expanding. Second, with all of the data of the Expanding process, a proper regression method is able to grab the global feature of the underlying $\text{PF}(\theta)$ (Figs. 3 and 4). In the regression process, all the data are used repeatedly. Besides, the detailed distribution information of the data can be utilized to obtain a better approximation of $\text{PF}(\theta)$ (see Section 3.1). The approximation of $\text{PF}(\theta)$ using sample data $\{(\theta_1, Y_1(\theta_1)), (\theta_2, Y_2(\theta_2)), \ldots, (\theta_N, Y_N(\theta_N))\}$ is denoted as $\hat{\text{PF}}(\theta; \omega)$ where $\omega$ is the parameter in the function model. The global optimum of $\text{PF}(\theta; \omega)$ provides an approximation of the global optimum of $\text{PF}(\theta)$. The optimization of $\hat{\text{PF}}(\theta; \omega)$ is a traditional optimization problem with many effective algorithms (Eq. (10)). The regression process and the optimization process comprise the heart of IRO for parameter optimization of EOBFSs.

$$\theta_m = \arg \max_{\theta \in \Omega_2} \hat{\text{PF}}(\theta; \omega)$$

The details of IRO are as follows (Algorithm 1).

1. Initialize the data set $\mathcal{D}$ to be $\emptyset$.
2. Generate $\{(\theta_1, Y_1(\theta_1)), (\theta_2, Y_2(\theta_2)), \ldots, (\theta_N, Y_N(\theta_N))\}$ with $\theta_i$ chosen independently from the uniform distribution over $\Omega_2$, and merge them with the existing $\mathcal{D}$.
3. Solve the regression model with $\mathcal{D}$ and get $\hat{\text{PF}}(\theta; \omega)$.
4. Optimize $\hat{\text{PF}}(\theta; \omega)$ and get the current best parameter combination $\theta_m$.
5. Check the stop criterion. If $\theta_m$ has converged, end the algorithm, otherwise, go back to Step 2.

Step 2 indicates that the generated data $(\theta_i, Y_1(\theta_i))$ will be used repeatedly. This improves the utilization rate of the valuable data and fits with the particular...
property of EOBFS. The regression part and the optimization part in IRO are not totally independent. They could interact with each other in a proper way to improve the effectiveness of the algorithm (see Section 3.1). Figure 5 provides an intuitive view of how IRO works.

IRO presents an instructive algorithm architecture, but how to integrate the regression part with the appropriate optimization algorithm remains a challenge due to the complexity of the parameter optimization of EOBFSs. Considering the special binary feature of EOBFSs, we propose a scheme named MLEPSO-IRO, where Maximum Likelihood Estimation (MLE) and PSO implement the regression part and the optimization part in IRO, respectively.

### 3.1 Regression in MLEPSO-IRO

In general, a regression method contains two essential parts. The first part lies in the selection of a function model. The linear regression model is easy to handle and usually has a closed-form solution (Eq. (11)). For a general EOBFS, nothing about the underlying regularity of PF(θ) is known except some binary data. Hence, it is advisable to choose the linear regression model to approximate PF(θ). The choice of basis function depends on the problem. For example, if PF(θ) is polynomial, the basis function should be the power function. the one-dimensional parameter space is first considered.

\[
\hat{PF}(\theta; \omega) = a_0 + \sum_{i=1}^{M} \omega_i \phi_i(\theta) = \omega^T \phi(\theta) \tag{11}
\]

The second part is related to the method of optimizing ω in the function model. Usually, the least squares method is adopted. For general regression problems, it is assumed that data is corrupted with *i.i.d.* Gaussian noise. With this assumption, the least-squares method has a probabilistic interpretation of maximizing the likelihood. Nevertheless, when it comes to parameter optimization of EOBFSs, data are corrupted with Bernoulli noise (Eqs. (12)–(14)). Besides, the variance of the noise at θ is dependent on PF(θ) (Eq. (15)). Thus, the optimization method for the function model should be reconsidered.

\[
Y(\theta) = PF(\theta) + (Y(\theta) - PF(\theta)) = PF(\theta) + N(\theta) \tag{12}
\]

\[
f_N(\theta)(n) = \begin{cases} 
PF(\theta), & \text{if } n = 1 - PF(\theta); \\
1 - PF(\theta), & \text{if } n = PF(\theta); \\
0, & \text{otherwise} 
\end{cases} \tag{13}
\]

\[
E(N(\theta)) = 0 \tag{14}
\]

\[
\text{var}(N(\theta)) = PF(\theta)(1 - PF(\theta)) \tag{15}
\]

The least-squares method is the result of maximizing the likelihood of the data set when the noise is *i.i.d.* Gaussian noise [13]. Analogous to the derivation of the least-squares method, it is easy to obtain the specific solution for regression on a binary data set. The distribution of \( \mathcal{D} = \{(\theta_{ij}, Y_{ij}(\theta_{ij}))\}_{i,j=1,2,3,...} \) from an EOBFS can be represented as Eq. (16). Hence, the likelihood of \( \mathcal{D} \) is given by Eq. (17).

\[
p(Y(\theta)) = PF(\theta)^{Y(\theta)}(1 - PF(\theta))^{1 - Y(\theta)} \tag{16}
\]

\[
p(\mathcal{D}) = \prod_{i=1}^{N} PF(\theta_i)^{Y_i(\theta_i)}(1 - PF(\theta_i))^{1 - Y_i(\theta_i)} \tag{17}
\]

\[
\ln p(\mathcal{D}) = \sum_{i=1}^{N} Y_i(\theta_i) \ln PF(\theta_i) + \sum_{i=1}^{N} (1 - Y_i(\theta_i)) \ln(1 - PF(\theta_i)) \tag{18}
\]

With \( PF(\theta_i) \) substituted by \( \hat{PF}(\theta; \omega) \) in Eq. (18), the objective function is obtained as Eq. (19). Besides,
PF(θ) is the expectation of a Bernoulli distributed variable Y(θ), hence PF(θ) is certain to lie in [0, 1]. Consequently, a bound constraint on PF(θ; ω) should be appended (Eq. (20)).

Finally, a regularization item is added to avoid the phenomenon of over-fitting.

The final regression model is as follows.

\[
\omega_{\text{MLE}} = \max \sum_{i=1}^{N} Y(\theta_i) \ln(\omega^T \phi(\theta_i)) + \\
\sum_{i=1}^{N} (1 - Y(\theta_i)) \ln(1 - \omega^T \phi(\theta_i)) + \lambda \|\omega\|^2_2 
\]

subject to \(0 \leq \omega^T \phi(\theta) \leq 1, \forall \theta \in \Omega_\theta\) (19)

As Y(θ) is 0 or 1, we can obtain a clearer formulation as follows.

\[
\omega_{\text{MLE}} = \max \sum_{Y_i(\theta_i) = 1} \ln(\omega^T \phi(\theta_i)) + \\
\sum_{Y_i(\theta_i) = 0} \ln(1 - \omega^T \phi(\theta_i)) + \lambda \|\omega\|^2_2 
\]

subject to \(0 \leq \omega^T \phi(\theta) \leq 1, \forall \theta \in \Omega_\theta\) (20)

The objective function proves to be convex, but the constraint (Eq. (22)) is difficult to handle because of the arbitrariness of \(\theta\). In practice, we choose \(M\) points uniformly from the parameter space to approximate the arbitrariness of \(\theta\) (Eq. (23)). The approximate constraint also proves to be convex. Thus, we can use convex optimization tools (e.g., CVX in Matlab) to solve the regression model.

\[
0 \leq \omega^T \phi(\theta_k) \leq 1,
\]

\[
\theta_k \sim U(\Omega_\theta), \quad k = 1, 2, ..., M 
\] (23)

As the dimension of the parameter space increases, the size of \(\omega\) grows rapidly, with the power law. The phenomenon is called the curse of dimensionality. Hence, MLEPSO-IRO is suitable for parameter optimization of EOBFSs with low-dimensional parameter space.

### 3.2 Optimization in MLEPSO-IRO

The optimization part of IRO is a traditional real-parameter optimization problem. The analytical result of the regression is usually nonlinear and nonconvex with \(\theta\). It is difficult for classical optimization methods to guarantee a global optimum. Modern algorithms based on SI impose no restrictions on the form of the objective function, and provide a result which converges in probability. Among them, PSO[2] (Eqs. (24) and (25)), is easy to implement, converges quickly for explicit objective functions, and is convenient to generalize to higher-dimensional problems. Hence, PSO is chosen to implement the optimization part of IRO in MLEPSO-IRO.

\[
v_i(t + 1) = w v_i(t) + c_1 r_1 (p_i(t) - x_i(t)) + c_2 r_2 (p_g(t) - x_i(t)) \quad (24)
\]

\[
x_i(t + 1) = x_i(t) + v_i(t) \quad (25)
\]

Here,

- \(r_1\) and \(r_2\) are random variables uniformly chosen from [0, 1];
- \(w, c_1, c_2\) are usually constant coefficients in (0, 1);
- \(p_i\) is the best position found by the \(i\)-th particle, also called the personal best;
- \(p_g\) is the best position found by the \(i\)-th particle’s neighborhood, also called the global best.

Equation (24) indicates that the velocity, which determines the amount of change, is determined by three components: momentum (the \(v_i(t)\) part), cognitive (the \(p_i\) part), and social (the \(p_g\) part). This equation makes PSO an effective optimization method. More details can be found in Ref. [2].

PSO is a stochastic algorithm, so it does not always converge to the same point. To weaken the randomness of its result, \(\theta_{m_i}\) in Algorithm 1 of MLEPSO-IRO is used to initialize the social component of PSO in the next iteration. Through this interaction, PSO of iteration \((i + 1)\) in MLEPSO-IRO is more likely to converge to \(\theta_{m_i}\) in the prior iteration. This interaction makes MLEPSO-IRO more likely to converge, especially when PF(θ) is flat, or has several local optima.

### 3.3 Convergence analysis

MLEPSO-IRO contains a regression part based on MLE and an optimization part based on PSO. For PSO, complete theoretical analysis[14] and user-oriented guidelines[15] for its convergence have been presented. Hence, the ability to find the global optimum of the optimization part is guaranteed. In the regression part, the data are corrupted with variant Bernoulli noise, but we will still show that the regression part converges to the underlying PF(θ) with the data set size increasing.

According to the asymptotic property of the MLE[11], if \(\log p(D; \omega)\) has derivatives and the Fisher information (Eq. (26)) is nonzero (known as...
regularity conditions), \( \omega_{\text{MLE}} \) asymptotically satisfies a
distribution with mean value \( \omega^* \) and variance \( I^{-1}(\omega^*) \).

\[
I(\omega^*) = -E \left[ \frac{\partial^2 \ln p(D; \omega^*)}{\partial \omega^* \partial \omega^*} \right] 
\tag{26}
\]

Here, \( \omega^* \) is the real value of the unknown parameter, and \( I(\omega^*) \) is the corresponding Fisher information. \( I(\omega^*) \) and \( I^{-1}(\omega^*) \) are both positive definite.

This theorem indicates that \( \omega_{\text{MLE}} \) is asymptotically unbiased and asymptotically attains the CRLB. In other words, \( \omega_{\text{MLE}} \) is asymptotically optimal. However, the asymptotic property of the MLE provides little direct implication on the convergence property of \( \omega_{\text{MLE}} \). We should prove that the items on the diagonal of \( I^{-1}(\omega^*) \) converge to 0.

According to Eq. (18), Eq. (26) can be transformed into Eq. (27). Hence, for the \( j \)-th item of \( \omega_{\text{MLE}} \), its asymptotical variance is given by Eq. (28). As a basis function, there must be some \( \theta_j^* \) that makes \( \phi_j(\theta_j^*) \) nonzero. It is reasonable to hypothesize that the base functions are continuous, so there must be a positive \( \delta \), that makes \( |\phi_j(\theta_j^*)| \geq |\phi_j(\theta_j^* \pm \delta)|, \forall \theta \in [\theta_j^* - \delta, \theta_j^* + \delta] \). As \( 0 \leq \text{PF}(\theta_j)(1 - \text{PF}(\theta_j)) \leq 1/4 \), and as we define \( C_{j\beta} \) to be the number of data generated with \( \theta \in [\theta_j^* - \delta, \theta_j^* + \delta] \), we get Eq. (28).

\[
I(\omega^*) = \sum_{i=1}^{N} \frac{\phi(\theta_i)\phi^T(\theta_i)}{\omega^T\phi(\theta_i)(1 - \omega^T\phi(\theta_i))} = \sum_{i=1}^{N} \frac{\phi(\theta_i)\phi^T(\theta_i)}{\text{PF}(\theta_i)(1 - \text{PF}(\theta_i))} 
\tag{27}
\]

\[
[I(\omega^*)]_{jj} = \sum_{i=1}^{N} \frac{\phi_j(\theta_i)^2}{\text{PF}(\theta_i)(1 - \text{PF}(\theta_i))} \geq \phi_j(\theta_j^*)^2 \sum_{\theta \in [\theta_j^* - \delta, \theta_j^* + \delta]} 1 = \phi_j(\theta_j^*)^2 C_{j\beta} \tag{28}
\]

\( C_{j\beta} \) is a random variable. As \( \theta \) is chosen uniformly from \( \Omega_\theta \), \( C_{j\beta} \) is actually distributed according to a binomial distribution \( B(N, p) \), where \( p = 2\delta/L \) and \( L \) is the size of \( \Omega_\theta \). Hence, when \( N \to +\infty \), \( C_{j\beta} \) asymptotically satisfies the normal distribution \( N(Np, Np(1-p)) \). For \( \forall K > 0 \), we have the result of Eq. (29), in which \( \Phi \) is the Gauss error function. Hence, we have Eq. (30). Considering Eq. (28), we arrive at \( [I(\omega^*)]_{jj} \to +\infty \). Hence, \( \omega_{\text{MLE}} \) converges to \( \omega^* \) in probability (Eq. (31)).

\[
\lim_{N \to +\infty} P(C_{j\beta} \leq K) \approx \lim_{N \to +\infty} \Phi \left( \frac{K - Np}{\sqrt{Np(1-p)}} \right) = 0 \tag{29}
\]

\[
C_{j\beta} \xrightarrow{\text{pr}} +\infty \tag{30}
\]

\[
\omega_{\text{MLE}} \xrightarrow{\text{pr}} \omega^* \tag{31}
\]

4 Numerical Experiment

We experimented with MLEPSO-IRO, SPSA, and the blind random search on three different PF(\( \theta \)) functions, including a unimodal function (Fig. 6a, Eq. (32)-Quadratic), a multimodal function (Fig. 6b, Eq. (33)-Gaussian2), and a complex function from the real EOBFS, named ThuSIR (Fig. 6c-OldTrack).

\[
\text{PF}(\theta) = -\frac{1}{25} \theta^2 + \frac{2}{5} \theta \tag{32}
\]

\[
\text{PF}(\theta) = \frac{3}{10} \times e^{-\frac{(\theta - 5)^2}{2}} + \frac{9}{10} \times e^{-\frac{(\theta - 15)^2}{2}} \tag{33}
\]

We also conducted a series of 100 Expanding experiments, and in each experiment 128 groups of sampling data are generated, with \( n \) different values of \( \theta \), and \( n = 11, 12, 13, ..., 128 \), then we executed regression described in Section 3.1. The average result of the 100 experiments provides an experimental verification of the Expanding process (Fig. 4). In this experiment, PF(\( \theta \)) is defined as Gaussian2 function (Eq. (33)).

We did not compare MLEPSO-IRO with SI algorithms because they need much more data than SPSA. The implementation of SPSA provided in Ref. [1] was adopted, which proved to be practical and effective for unimodal PF(\( \theta \)) if the noise is kept low. We chose the power function as the basic function in the regression part and the maximum order \( M \) was five.
The probabilities for different acceptance regions are presented as a curve named the ARP curve (Figs. 7–9).

The ARP curves of the three problems are shown in Figs. 7–9. In the three cases, MLEPSO-IRO is much better than SPSA and the blind random search. MLEPSO-IRO has the best performance on unimodal, multimodal, and complex PF(θ) functions; hence, it is reasonable for MLEPSO-IRO to have good performance on other problems.

In Fig. 7 where PF(θ) is a quadratic function, MLEPSO-IRO is significantly better than SPSA. For SPSA, the effect of the algorithm improves when the amount of data used for obtaining PF(θ) increases. MLEPSO-IRO, however, is better at utilizing the valuable data, and thus gives a better result.

In Figs. 8 and 9, SPSA is much worse because SPSA is more likely to obtain the real gradient and then moves the candidate to the real optimum. Also, the number of iterations is slightly influenced (Fig. 10b). However, for MLEPSO-IRO, things are a little different. The regression in MLEPSO-IRO provides a better PF(θ; ω) when the amount of data used in the regression increases; thus, when the amount of data per iteration increases, the total number of iterations decreases (Fig. 10a). MLEPSO-IRO utilizes generated data repeatedly. The accuracy of PF(θ; ω) is concerned only with the total amount of data, not the amount of data per iteration. This difference also implies that MLEPSO-IRO is better at utilizing the valuable data, and thus gives a better result.

This is because when the variance of PF(θ) decreases, SPSA is more likely to obtain the real gradient and then moves the candidate to the real optimum. Also, the number of iterations is slightly influenced (Fig. 10b). However, for MLEPSO-IRO, things are a little different. The regression in MLEPSO-IRO provides a better PF(θ; ω) when the amount of data used in the regression increases; thus, when the amount of data per iteration increases, the total number of iterations decreases (Fig. 10a). MLEPSO-IRO utilizes generated data repeatedly. The accuracy of PF(θ; ω) is concerned only with the total amount of data, not the amount of data per iteration. This difference also implies that MLEPSO-IRO is better at utilizing the valuable data, and thus gives a better result.

In Figs. 8 and 9, SPSA is much worse because
of the complexity of the $\text{PF}(\theta)$. OldTrack is flat in a wide region, and has many local maxima. However, MLEPSO-IRO remains the best algorithm and provides a good result in these two cases.

Figures 11 and 12 provide a successful and a failed example, respectively. In Fig. 11, the approximation $\text{PF}(\theta; \omega)$ approaches $\text{PF}(\theta)$ successfully because the distribution of the generated data matches well the underlying $\text{PF}(\theta)$. This is a case of high likelihood, compared with the failed case in Fig. 12, where the generated data are not even symmetrical.

![Fig. 11 A successful example of MLEPSO-IRO.](image1)

![Fig. 12 A failed example of MLEPSO-IRO.](image2)
OldTrack comes from ThuSIR. In practice, there’s no need or no possibility of knowing the underlying PF($\theta$). But if we want to assess the performance of some parameter optimization algorithm, we must know all the information of the function to be optimized. ThuSIR costs about 20 s per execution, and we spent almost two days executing ThuSIR 10,000 times to obtain PF($\theta$) in one common case. In one common case, we set the threat angle to be $90^\circ$, made the direction of the wind uniformly chosen from $[-180^\circ, 180^\circ]$, attached a typical tracking method, set other environment parameters to be their own typical values, and fixed all other parameters of the countermeasure except the horizontal angle $\theta$ of the infrared jamming. Hence, the remaining problem is to find the best horizontal angle $\theta$ with which the ship survives with the largest probability. The answer is the $\theta_{\text{max}}$ of OldTrack. With MLEPSO-IRO, which iterates at most 5 times and generates 20 data per iteration, we need at most 100 system executions. Hence, we need only about half an hour. As IRO generates $N$ independent data per iteration, we could execute $N$ or a suitable number of system instances in parallel to shorten the total execution time to minutes (Fig. 13). The result in Fig. 9 shows that MLEPSO-IRO can provide a horizontal angle better than the average with a probability of 0.86 (ARP(APRS)). In other words, MLEPSO-IRO will provide a value in the region AR(APRS) (Fig. 14) with a probability of 0.86.

Table 1 provides more numerical comparison of the algorithms. From the table, we can see MLEPSO-IRO is better than the other two algorithms.

| Problem  | Algorithm       | ARP(APRS) | AP  | APR |
|----------|-----------------|-----------|-----|-----|
| Quadratic| Blind random search | 0.98      | 0.94| 0.94|
|          | SPSA            | 0.76      | 0.76| 0.76|
|          | MLEPSO-IRO      | **1.00**  | **0.99** | **0.99** |
| Guassian2| Blind random search | 0.84      | 0.62| 0.69|
|          | SPSA            | 0.34      | 0.29| 0.33|
|          | MLEPSO-IRO      | **1.00**  | **0.72** | **0.80** |
| OldTrack | Blind random search | 0.74      | 0.21| 0.41|
|          | SPSA            | 0.07      | 0.02| 0.04|
|          | MLEPSO-IRO      | **0.86**  | **0.30** | **0.59** |

5 Conclusion

In this paper, we proposed a novel algorithm, named IRO, for parameter optimization and a corresponding scheme, named MLEPSO-IRO, to solve the parameter optimization problem of EOBFSs. IRO contains a regression part and a subsequent optimization part. Unlike traditional algorithms that evaluate the system performance with a large amount of system executions, MLEPSO-IRO generated data at different places in the parameter space to obtain the global feature of the performance function. This improves the exploration ability of the algorithm and makes it possible to repeatedly utilize the valuable data. MLEPSO-IRO can be applied to systems that are expensive in terms of time or money, and to those that are difficult for traditional algorithms to handle. MLEPSO-IRO also takes the Bernoulli distribution information of the noise in the data with MLE. These are all managed in the regression part of MLEPSO-IRO. The subsequent optimization part of MLEPSO-IRO is a real-parameter optimization problem and is conquered by a slightly modified version of PSO. We compared MLEPSO-IRO with SPSA and blind random search, and demonstrated that MLEPSO-IRO is far better than the other two algorithms. We also experimented on a real EOBFS named ThuSIR, and proved that MLEPSO-IRO can
References

[1] J. C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. John Wiley & Sons, 2005.

[2] J. Kennedy, Particle swarm optimization, in *Encyclopedia of Machine Learning*. Springer, 2010, pp. 760–766.

[3] T. G. Kolda, R. M. Lewis, and V. Torczon, Optimization by direct search: New perspectives on some classical and modern methods, *SIAM Review*, vol. 45, no. 3, pp. 385–482, 2003.

[4] I. Aydin, M. Karakose, and E. Akin, A multi-objective artificial immune algorithm for parameter optimization in support vector machine, *Information Sciences*, vol. 192, pp. 120–142, 2002.

[5] J. Deng, W. Dong, R. Socher, B. Spasojevic, L. Li, K. Li, and L. Fei-Fei, ImageNet: A large-scale hierarchical image database, in *IEEE CVPR*, 2009, pp. 248–255.

[6] J. C. Spall, Stochastic optimization, in *Handbook of Computational Statistics*. Springer, 2012, pp. 173–201.

[7] H. Robbins and S. Monro, A stochastic approximation method, *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[8] J. Kiefer and J. Wolfowitz, Stochastic estimation of the maximum of a regression function, *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

Xiaoqin Zhang received the BEng degree from Tsinghua University, China in 2015, where he is currently pursuing the PhD degree. His research interests include multi-agent decision making, optimization, and 3D simulation.

Jinghuan Wen received the BEng degree from Tsinghua University, China in 2014, where he is currently pursuing the PhD degree. His research interests include 3D simulation, computer graphics and optimization.

Huimin Ma received the MS and PhD degrees in mechanical electronic engineering from Beijing Institute of Technology, China in 1998 and 2001, respectively. She is an associate professor with the Department of Electronic Engineering of Tsinghua University, and the director of 3D Image Simulation Lab. She worked as a visiting scholar in University of Pittsburgh in 2011. She is also the executive director and the vice secretary general of China Society of Image and Graphics. Her research and teaching interests include 3D object recognition and tracking, system modeling and simulation, and psychological base of image cognition.