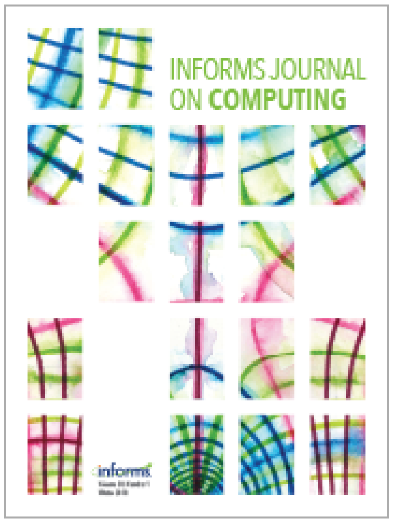
This article was downloaded by: [143.215.33.36] On: 01 August 2018, At: 11:52
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA



INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Gradient-Based Adaptive Stochastic Search for Simulation Optimization Over Continuous Space

Enlu Zhou, Shalabh Bhatnagar

To cite this article:
Enlu Zhou, Shalabh Bhatnagar (2018) Gradient-Based Adaptive Stochastic Search for Simulation Optimization Over Continuous Space. INFORMS Journal on Computing 30(1):154-167. https://doi.org/10.1287/ijoc.2017.0771

Full terms and conditions of use: http://pubsonline.informs.org/page/terms-and-conditions

This article may be used only for the purposes of research, teaching, and/or private study. Commercial use or systematic downloading (by robots or other automatic processes) is prohibited without explicit Publisher approval, unless otherwise noted. For more information, contact permissions@informs.org.

The Publisher does not warrant or guarantee the article's accuracy, completeness, merchantability, fitness for a particular purpose, or non-infringement. Descriptions of, or references to, products or publications, or inclusion of an advertisement in this article, neither constitutes nor implies a guarantee, endorsement, or support of claims made of that product, publication, or service.

Copyright © 2018, INFORMS

Please scroll down for article—it is on subsequent pages

INFORMS is the largest professional society in the world for professionals in the fields of operations research, management science, and analytics.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

http://pubsonline.informs.org/journal/ijoc/

INFORMS JOURNAL ON COMPUTING
Vol. 30, No. 1, Winter 2018, pp. 154–167
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

Gradient-Based Adaptive Stochastic Search for Simulation Optimization Over Continuous Space

Enlu Zhou,[a] Shalabh Bhatnagar[b]

[a] H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332;
[b] Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India

Contact: enlu.zhou@isye.gatech.edu, http://orcid.org/0000-0001-5399-6508 (EZ); shalabh@csa.iisc.ernet.in (SB)

Received: March 3, 2014
Revised: November 17, 2014; December 2, 2015; November 29, 2016; June 18, 2017
Accepted: June 19, 2017
Published Online: January 23, 2018

https://doi.org/10.1287/ijoc.2017.0771

Copyright: © 2018 INFORMS

Abstract. We extend the idea of model-based algorithms for deterministic optimization to simulation optimization over continuous space. Model-based algorithms iteratively generate a population of candidate solutions from a sampling distribution and use the performance of the candidate solutions to update the sampling distribution. By viewing the original simulation optimization problem as another optimization problem over the parameter space of the sampling distribution, we propose to use a direct gradient search on the parameter space to update the sampling distribution. To improve the computational efficiency, we further develop a two-timescale updating scheme that updates the parameter on a slow timescale and estimates the quantities involved in the parameter updating on the fast timescale. We analyze the convergence properties of our algorithms through techniques from stochastic approximation, and demonstrate the good empirical performance by comparing with two state-of-the-art model-based simulation optimization methods.

History: Accepted by Marvin Nakayama, Area Editor for Simulation.
Funding: The authors gratefully acknowledge the support by the National Science Foundation [Grants CMMI-1413790 and CAREER CMMI-1453934], Air Force Office of Scientific Research [Grant FA-9550-14-1-0059], Xerox Corporation, USA, Department of Science and Technology, Government of India, and the Robert Bosch Centre for Cyber-Physical Systems, Indian Institute of Science (IISc).
Supplemental Material: The online supplement is available at https://doi.org/10.1287/ijoc.2017.0771.

Keywords: simulation optimization • model-based optimization • two-timescale stochastic approximation

1. Introduction

Many engineering systems require optimization over continuous decision variables to improve system performance. Due to the scale and complexity of those systems, the system performance is often evaluated through simulation or experimentation that has noise, so only samples of system performance are observed. Such problems are often referred to as simulation optimization problems in the literature. As characterized by Fu et al. (2008), there are four main classes of approaches to simulation optimization over continuous space: (i) sample average approximation, e.g., Rubinstein and Shapiro (1993), de Mello et al. (1999), Kleywegt et al. (2002); (ii) stochastic gradient methods or stochastic approximation, e.g., Kiefer and Wolfowitz (1952), Spall (1992), Kushner and Yin (2004); (iii) sequential response surface methodology, e.g., Barton and Meckesheimer (2006), Huang et al. (2006), Chang et al. (2013); and (iv) deterministic metaheuristics, a broad category of methods that generalize deterministic metaheuristics to the simulation optimization setting, e.g., Ólafsson (2006), Shi and Ólafsson (2000), Andradóttir (2006), Scott et al. (2011). As noted in the survey papers by Fu (2002a, b), when it comes to solving practical problems, the optimization algorithms actually implemented in simulation software are evolutionary algorithms and other metaheuristics from deterministic nonlinear optimization, such as genetic algorithms (Goldberg 1989), tabu search (Glover 1990), and random search (Zabinsky 2003, Andradóttir 2006).

Among deterministic metaheuristics, a recent class of methods under the name of "model-based methods" have been proven to work very well for deterministic nonlinear optimization problems (Zlochin et al. 2004, Hu et al. 2012). Model-based methods typically assume a sampling distribution (i.e., a probabilistic model), often within a parameterized family of distributions, over the solution space, and iteratively carry out the two interrelated steps: (1) draw candidate solutions from the sampling distribution and (2) use the evaluations of these candidate solutions to update the sampling distribution. The hope is that at every iteration, the sampling distribution is biased toward the more promising regions of the solution space, and will eventually concentrate on one or more of the optimal solutions. Examples of model-based algorithms include ant colony optimization (Dorigo and Gambardella 1997), annealing adaptive search (AAS) (Romeijn and Smith 1994), probability collectives (PCs) (Wolpert 2004), the estimation of distribution algorithms (EDAs) (Larranaga

RIGHTSLINK

et al. 1999, Muhlenbein and Paaß 1996), the cross-entropy (CE) method (Rubinstein 2001), model reference adaptive search (MRAS) (Hu et al. 2007), the interacting-particle algorithm (Molvalioglu et al. 2009, 2010), and gradient-based adaptive stochastic search (GASS) (Zhou and Hu 2014). These algorithms retain the primary strengths of population-based approaches such as genetic algorithms, while providing more flexibility and robustness in exploring the entire solution space. There are a few attempts at extending model-based methods to simulation optimization, mostly with a focus of how to allocate simulation budget to candidate solutions in each iteration of the algorithm: Chepuri and de Mello (2005) proposed a simple heuristic sampling scheme to determine the number of simulation replications in each iteration for the CE method; Hu et al. (2008) studied sufficient conditions on the simulation allocation rule in MRAS to guarantee convergence of the algorithm and proposed the method of stochastic model reference adaptive search (SMRAS); more recently, He et al. (2010) developed the algorithm under the name of cross-entropy with optimal computing budget allocation (CEOCBA), which nicely incorporates the idea of optimal computing budget allocation (OCBA) (Chen et al. 2000, Chen and Lee 2010) into each iteration of the CE method.

Motivated by model-based methods but different from the aforementioned approaches, we propose a framework of incorporating gradient search into model-based optimization: by viewing the original problem as another optimization problem over the parameter space of the sampling distribution, we use a direct gradient search to update the parameter while drawing new solutions from the sampling distribution. This idea has been exploited for deterministic nondifferentiable optimization in our earlier work (Zhou and Hu 2012, 2014). In this paper, we extend this idea to the setting of simulation optimization, where the performance of a candidate solution can only be estimated through simulation or experimentation. As a result, the gradient and Hessian terms involved in the gradient search need to be estimated jointly by candidate solutions and their performance estimates. With the aim of reducing the sample size to further improve the efficiency, we develop a two-timescale scheme that updates the parameter on a slow timescale and estimates the gradient and Hessian terms using the samples on a fast timescale. The resultant algorithm is essentially a two-timescale stochastic approximation scheme, which allows us to use theory and tools from the multi-timescale stochastic approximation literature (Borkar 1997, 2008; Bhatnagar and Borkar 1998; Bhatnagar et al. 2013) to analyze the convergence of the algorithm.

The rest of the paper is organized as follows. Section 2 introduces and develops the main idea behind the proposed algorithm. Section 3 presents our proposed algorithm and the two-timescale variant. Section 4 presents the main convergence result. The entire convergence analysis of the algorithms is presented in the accompanying online supplement. Section 5 illustrates the performance of our algorithms by comparing with the CEOCBA method on several benchmark problems and with SMRAS on a simulation optimization problem. Finally, we conclude the paper in Section 6.

2. Main Idea and Algorithm Development

We consider the following simulation optimization problem:

$$\max_{x\in\mathscr{X}} H(x) \triangleq E[h(x,\xi_x)], \tag{1}$$

where $\mathscr{X} \subseteq \mathbb{R}^{d_x}$ is a compact set, ξ_x represents the randomness in the system, and the subscript x signifies the distribution of ξ_x may depend on x, and the expectation is taken with respect to the distribution of ξ_x. Assume that H is bounded on $\mathscr{X}$, i.e., $\exists H_{lb} > -\infty$, $H_{ub} < \infty$ s.t. $H_{lb} \le H(x) \le H_{ub}$. Due to the complexity of the system, the analytical form of H is often not available. For a given x, its feasibility can be determined easily, but its performance $H(x)$ can only be estimated through stochastic simulation or experimentation, which returns a sample of $h(x,\xi_x)$. We also assume $H(x)$ is Lipschitz continuous. Continuity of the objective function is needed because one underlying assumption behind our algorithms is that the solutions in a neighborhood have similar performance.

2.1. Main Idea

In this section, we will explain our main idea on a high level and focus more on the intuition. We will leave the technical details to the next section on the formal development of the algorithms.

As in many model-based methods, we introduce a parameterized family of densities $\{f(x;\theta)\}$ as the sampling distribution, where θ is the parameter that will be updated over iterations. We will illustrate our idea by first considering the function

$$L(\theta) \triangleq \int H(x)f(x;\theta)\,dx.$$

It is easy to see that

$$L(\theta) \le H(x^*),$$

and the equality is achieved if and only if all the probability mass of $f(x;\theta)$ concentrates on a subset of the set of global optima. If such a θ exists, it will recover the optimal solution and the optimal function value. For example, assuming the optimal solution x^* is unique and $f(x;\theta)$ is a Gaussian distribution, then x^* will be recovered by a degenerate distribution that concentrates only at x^*, which corresponds to the mean of the Gaussian being equal to x^* and the variance being zero.

In contrast to $H(x)$, which does not necessarily have differentiability, the function $L(\theta)$ is continuous and

RIGHTSLINK

differentiable in θ under some mild conditions on $f(x;\theta)$, and hence direct gradient methods may be used to update θ by taking advantage of this structural property. Moreover, the updated sampling distribution $f(x;\theta)$ guides the stochastic search on the solution space of $H(x)$, and hence maintains a global exploration of the solution space. Viewed from the perspective of balancing exploration and exploitation, the direct gradient search on θ tries to exploit the promising region on the θ-space while the sampling distribution keeps exploration on the x-space. This leads to a natural idea of incorporating direct gradient search within the framework of model-based optimization, with the hope to combine the relative fast convergence of gradient search with the global exploration of model-based optimization. More specifically, such an incorporation can be done by iteratively carrying out the following two steps:

1. Generate candidate solutions from the sampling distribution.
2. Based on the evaluations of the candidate solutions, update the parameter of the sampling distribution via direct gradient search.

For the second step above, computing the gradient and (approximate) Hessian is easy and only requires evaluation or estimation of the objective function $H(x)$. Assuming that the derivative and integral can be interchanged (which will be formally justified later), we can derive an expression for the gradient of $L(\theta)$ as follows:

$$\begin{aligned}\nabla_\theta L(\theta) &= \nabla_\theta \int H(x) f(x;\theta)\,dx\\ &= \int H(x)\frac{\nabla_\theta f(x;\theta)}{f(x;\theta)} f(x;\theta)\,dx\\ &= E_{f_\theta}[H(x)\nabla_\theta \ln f(x;\theta)].\end{aligned}$$

Throughout the paper, we will use the notation E_{f_θ} to denote the expectation taken with respect to the distribution $f(\cdot;\theta)$, which is parameterized by θ. We note that any unbiased estimator $\hat{H}(x)$ of $H(x)$ leads to unbiased estimation for $E_{f_\theta}[H(x)\nabla_\theta \ln f(x;\theta)]$, since

$$\begin{aligned}E[\hat{H}(x)\nabla_\theta \ln f(x;\theta)] &= E_{f_\theta}[E[\hat{H}(x)]\nabla_\theta \ln f(x;\theta)]\\ &= E_{f_\theta}[H(x)\nabla_\theta \ln f(x;\theta)],\end{aligned}$$

where the inner expectation is taken with respect to the randomness in $\hat{H}(x)$. Hence, to estimate the gradient $\nabla_\theta L(\theta)$, we can first draw a sample x from $f(\cdot;\theta)$ and then plug into $\hat{H}(x)\nabla_\theta \ln f(x;\theta)$.

2.2. Formal Development

For the development of a simple and yet flexible algorithm, we introduce a shape function S_θ: $\mathbb{R} \to \mathbb{R}^+$, where the subscript θ signifies the possible dependence of the shape function on the parameter θ. The function S_θ satisfies the following condition:

For every fixed θ, $S_\theta(y)$ is strictly increasing in y, and bounded from above and below for any $y \in (-\infty, +\infty)$. Moreover, for every fixed y, $S_\theta(y)$ is continuous and nondecreasing in θ.

The shape function makes the objective function value positive, and the strictly increasing property preserves the order of solutions and, in particular, the optimal solution. Other conditions on the shape function are regularity conditions to facilitate the derivation and analysis later. The shape function also imposes a weighting scheme on the samples based on sample function evaluations, and hence guides the sampling distribution toward the more promising regions where the candidate solutions have better performance; this point will be discussed in detail in Section 3. Then, for an arbitrary but fixed θ', we define

$$L(\theta;\theta') = \int S_{\theta'}(H(x)) f(x;\theta)\,dx.$$

By the condition on the shape function, it is easy to see that for a fixed θ',

$$L(\theta;\theta') \le S_{\theta'}(H(x^*)), \tag{2}$$

and the equality is achieved if and only if all the probability mass of $f(x;\theta)$ is concentrated on a subset of the set of global optima. We further define

$$l(\theta;\theta') \triangleq \ln L(\theta;\theta').$$

Since $\ln(\cdot)$ is a strictly increasing function, $l(\theta;\theta')$ has the same set of optimal solutions as $L(\theta;\theta')$.

Following our main idea outlined before, we propose a stochastic search algorithm that carries out the following two steps at each iteration: let θ_k be the parameter obtained at the kth iteration,

1. Generate candidate solutions from $f(x;\theta_k)$.
2. Update the parameter to θ_{k+1} using a Newton-like iteration for $\sup_{\theta\in\Theta} l(\theta;\theta_k)$.

The second step requires one to compute the gradient and Hessian of $l(\theta;\theta_k)$, whose analytical expressions are provided in Proposition 1 below when the sampling distribution is chosen to be an exponential family of densities (see Proposition 2 in Zhou and Hu 2014 for the same result and result for the case of general distributions; we provide the proof here for completeness). The definition of exponential families (cf. Barndorff-Nielsen 1978) is stated as follows.

Definition 1. A family $\{f(x;\theta)\colon \theta \in \Theta \subset \mathbb{R}^{d_\theta}\}$ is an *exponential family of densities* if its probability density function (pdf) satisfies

$$\begin{aligned}f(x;\theta) &= \exp\{\theta^T T(x) - \phi(\theta)\},\\ \phi(\theta) &= \ln\left\{\int \exp(\theta^T T(x))\,dx\right\},\end{aligned} \tag{3}$$

where $T(x) = [T_1(x), T_2(x), \ldots, T_d(x)]^T$ is the vector of sufficient statistics, $\theta = [\theta_1, \theta_2, \ldots, \theta_d]^T$ is the vector of natural parameters, and $\Theta = \{\theta \in \mathbb{R}^{d_\theta}\colon |\phi(\theta)| < \infty\}$ is the parameter space that is an open set with a nonempty interior.

RIGHTSLINK

Proposition 1. *Assume that $f(x;\theta)$ is twice differentiable in θ and that $\nabla_\theta f(x;\theta)$ and $\nabla^2_\theta f(x;\theta)$ are bounded on $\mathcal{X}$ for any θ. Furthermore, if $f(x;\theta)$ is in an exponential family of densities, then*

$$\nabla_\theta l(\theta;\theta')|_{\theta=\theta'} = E_{g_{\theta'}}[T(X)] - E_{f_{\theta'}}[T(X)],$$
$$\nabla^2_\theta l(\theta;\theta')|_{\theta=\theta'} = \mathrm{Cov}_{g_{\theta'}}[T(X)] - \mathrm{Cov}_{f_{\theta'}}[T(X)],$$

where $E_{f_{\theta'}}$ and $\mathrm{Cov}_{f_{\theta'}}$ denote the expectation and covariance with respect to $f(\cdot;\theta')$, and $E_{g_{\theta'}}$ and $\mathrm{Cov}_{g_{\theta'}}$ denote the expectation and covariance taken with respect to the pdf $g(\cdot;\theta')$, which is defined by

$$g(x;\theta) = \frac{S_\theta(H(x))f(x;\theta)}{\int S_\theta(H(y))f(y;\theta)\,dy}. \tag{4}$$

Proof. Consider the gradient of $l(\theta;\theta')$ with respect to θ,

$$\begin{aligned}\nabla_\theta l(\theta;\theta')|_{\theta=\theta'} &= \frac{\nabla_\theta L(\theta;\theta')}{L(\theta;\theta')}\bigg|_{\theta=\theta'}\\ &= \frac{\int S_{\theta'}(H(x))f(x;\theta)\nabla_\theta \ln f(x;\theta)\,dx}{L(\theta;\theta')}\bigg|_{\theta=\theta'}\\ &= E_{g_{\theta'}}[\nabla_\theta \ln f(X;\theta')].\end{aligned} \tag{5}$$

Differentiating (5) with respect to θ, we obtain the Hessian

$$\begin{aligned}&\nabla^2_\theta l(\theta;\theta')|_{\theta=\theta'}\\ &= \frac{\int S_{\theta'}(H(x))f(x;\theta)\nabla^2_\theta \ln f(x;\theta)\,dx}{L(\theta;\theta')}\\ &\quad+\frac{\int S_{\theta'}(H(x))\nabla_\theta \ln f(x;\theta)(\nabla_\theta f(x;\theta))^T\,dx}{L(\theta;\theta')}\\ &\quad-\frac{(\int S_{\theta'}(H(x))f(x;\theta)\nabla_\theta \ln f(x;\theta)\,dx)(\nabla_\theta L(\theta;\theta'))^T}{L(\theta;\theta')^2}\bigg|_{\theta=\theta'}.\end{aligned}$$

Using $\nabla_\theta f(x;\theta) = f(x;\theta)\nabla_\theta \ln f(x;\theta)$ in the second term on the right-hand side, the above expression can be rewritten as

$$\begin{aligned}&\nabla^2_\theta l(\theta;\theta')|_{\theta=\theta'}\\ &\quad= E_{g_{\theta'}}[\nabla^2_\theta \ln f(X;\theta')]\\ &\qquad+E_{g_{\theta'}}[\nabla_{\theta'} \ln f(X;\theta')(\nabla_{\theta'} \ln f(X;\theta'))^T]\\ &\qquad-E_{g_{\theta'}}[\nabla_\theta \ln f(X;\theta')]E_{g_{\theta'}}[\nabla_\theta \ln f(X;\theta')]^T\\ &\quad= E_{g_{\theta'}}[\nabla^2_\theta \ln f(X;\theta')]+\mathrm{Cov}_{g_{\theta'}}[\nabla_\theta \ln f(X;\theta')].\end{aligned} \tag{6}$$

Furthermore, if $f(x;\theta)=\exp\{\theta^T T(x)-\phi(\theta)\}$,

$$\nabla_\theta \ln f(x;\theta) = T(x) - E_{f_\theta}[T(X)]. \tag{7}$$

Plugging (7) into (5) yields

$$\nabla_\theta l(\theta;\theta')|_{\theta=\theta'} = E_{g_{\theta'}}[T(X)] - E_{f_{\theta'}}[T(X)].$$

Differentiating (7) with respect to θ, we obtain

$$\nabla^2_\theta \ln f(x;\theta) = -\mathrm{Cov}_{f_\theta}[T(X)]. \tag{8}$$

Plugging (7) and (8) into (6) yields

$$\nabla^2_\theta l(\theta;\theta')|_{\theta=\theta'} = \mathrm{Cov}_{g_{\theta'}}[T(X)] - \mathrm{Cov}_{f_{\theta'}}[T(X)]. \quad \square$$

With the analytical expressions of the gradient and Hessian, we are ready to use a gradient-based approach to update the parameter θ. To ensure the parameter updating is along the ascent direction of $l(\theta;\theta')$, we approximate the Hessian by the slightly perturbed second term in $\nabla^2_\theta l(\theta';\theta')$, i.e., a negative definite term $-(\mathrm{Cov}_{f_{\theta'}}[T(X)]+\epsilon I)$, where I is the identity matrix and ϵ is a small positive constant. The reason for this approximation is that $\mathrm{Cov}_{f_\theta}[T(X)] = E[-\nabla^2_\theta \ln f(X;\theta)]$ is the Fisher information matrix, whose inverse provides a lower bound on the variance of an unbiased estimator of the parameter, leading to the fact that $\mathrm{Cov}_{f_\theta}[T(X)]^{-1}$ is the minimum variance stepsize for stochastic approximation (as we will show later, our algorithm can be interpreted as a stochastic approximation procedure on θ). In summary, we update the parameter as follows:

$$\begin{aligned}\theta_{k+1} &= \Pi_\Theta\{\theta_k+\alpha_k(\mathrm{Cov}_{f_{\theta_k}}[T(X)]+\epsilon I)^{-1}\nabla_\theta l(\theta_k;\theta_k)\}\\ &= \Pi_\Theta\{\theta_k+\alpha_k(\mathrm{Cov}_{f_{\theta_k}}[T(X)]+\epsilon I)^{-1}\\ &\qquad\cdot(E_{g_{\theta_k}}[T(X)]-E_{f_{\theta_k}}[T(X)])\},\end{aligned} \tag{9}$$

where α_k is a positive stepsize, and Π_Θ denotes the projection operator that projects an iterate back onto the parameter space Θ.

To have an implementable algorithm, we still need to evaluate or estimate the expectation and covariance terms in (9). The expectation term $E_{f_{\theta_k}}[T(X)]$ can be evaluated analytically in most cases. For example, if $f(\cdot;\theta_k)$ belongs to the family of Gaussian distributions, then $E_{f_{\theta_k}}[T(X)]$ reduces to the mean and second moment of the Gaussian distribution. The covariance term $\mathrm{Cov}_{f_{\theta_k}}[T(X)]$ may not be directly available or could be too complicated to compute analytically, but it can be approximated by the sample covariance using the candidate solutions drawn from $f(\cdot;\theta_k)$. Recalling (4), the remaining term $E_{g_{\theta_k}}[T(X)]$ can be rewritten as

$$E_{g_{\theta_k}}[T(X)] = E_{f_{\theta_k}}\left[T(X)\frac{S_{\theta_k}(H(X))}{E_{f_{\theta_k}}[S_{\theta_k}(H(X))]}\right].$$

Hence we can draw i.i.d. samples $\{x_k^i, i=1,\dots,N_k\}$ from $f(x;\theta)$ and compute normalized weights of the samples according to

$$w_k^i \propto S_{\theta_k}(H(x_k^i)), \quad i=1,\dots,N_k; \quad \sum_{i=1}^{N_k} w_k^i = 1,$$

and approximate $E_{g_{\theta_k}}[T(X)]$ by $\sum_{i=1}^{N_k} w_k^i T(x_k^i)$. Since the exact function value is not available in simulation optimization in the algorithm, we will use a performance estimate $\hat{H}(x_k^i)$ (instead of $H(x_k^i)$) to compute the weight as above.

RIGHTSLINK

3. Algorithms: Gradient-Based Adaptive Stochastic Search for Simulation Optimization (GASSO) and Two-Timescale Gradient-Based Adaptive Stochastic Search for Simulation Optimization (GASSO-2T)

With the prior derivations, we propose the following algorithm.

Algorithm 1 (GASSO)

1. *Initialization*: choose an exponential family of densities $\{f(\cdot;\theta), \theta \in \Theta\}$, and specify a small positive constant ϵ, initial parameter θ_0, sample size sequence $\{N_k\}$ that satisfies $N_k \to \infty$ $(k \to \infty)$, simulation budget sequence $\{M_k\}$ that satisfies $M_k \to \infty$ $(k \to \infty)$, and stepsize sequence $\{\alpha_k\}$ that satisfies $\sum_{k=0}^{\infty} \alpha_k = \infty$, $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$. Set $k=0$.

2. *Sampling*: draw samples $x_k^i \overset{\text{iid}}{\sim} f(x;\theta_k)$, $i=1,2,\ldots,N_k$. For each x_k^i, evaluate the performance by simulation to generate $h(x_k^i,\xi_k^{i,j})$, $j=1,\ldots,M_k$, and compute the performance estimate $\hat{H}(x_k^i)=(1/M_k)\sum_{j=1}^{M_k} h(x_k^i,\xi_k^{i,j})$.

3. *Estimation*: compute the normalized weights $\hat{w}_k^i$ according to

$$\hat{w}_k^i = \frac{S_{\theta_k}(\hat{H}(x_k^i))}{\sum_{j=1}^{N_k} S_{\theta_k}(\hat{H}(x_k^j))},$$

and then estimate $E_{g_{\theta_k}}[T(X)]$ and $\mathrm{Cov}_{f_{\theta_k}}[T(X)]$ as follows:

$$\hat{E}_{g_{\theta_k}}[T(X)] = \sum_{i=1}^{N_k} \hat{w}_k^i T(x_k^i),$$

$$\widehat{\mathrm{Cov}}_{f_{\theta_k}}[T(X)] = \frac{1}{N_k-1}\sum_{i=1}^{N_k} T(x_k^i)T(x_k^i)^T - \frac{1}{N_k^2-N_k}\left(\sum_{i=1}^{N_k} T(x_k^i)\right)\left(\sum_{i=1}^{N_k} T(x_k^i)\right)^T.$$

4. *Updating*: update the parameter θ according to

$$\theta_{k+1} = \Pi_{\tilde{\Theta}}\left\{\theta_k + \alpha_k\left(\widehat{\mathrm{Cov}}_{f_{\theta_k}}[T(X)] + \epsilon I\right)^{-1} \cdot \left(\hat{E}_{g_{\theta_k}}[T(X)] - E_{f_{\theta_k}}[T(X)]\right)\right\},$$

where $\tilde{\Theta} \subseteq \Theta$ is a nonempty compact and convex set that closely approximates Θ, and $\Pi_{\tilde{\Theta}}$ denotes the projection operator that projects an iterate back onto the set $\tilde{\Theta}$ by choosing the closest point in $\tilde{\Theta}$.

In the initialization step (step 1), the conditions on the sample size sequence and stepsize sequence are imposed to guarantee the convergence, which will be shown in the next section. For continuous problems, some common choices of sampling distribution include Gaussian family, truncated Gaussian family, Gamma family, and Dirichlet family. The specific choice is largely driven by the constraint set (or in other words, the solution space $\mathscr{X}$) of the problem. For example, if the solution space is unbounded (although we assume $\mathscr{X}$ is compact for the purpose of convergence analysis, the algorithm can be applied to problems with unbounded solution space), then Gaussian is a good choice; if the solution space is a (scaled) simplex (with or without its interior), then Dirichlet is a good choice, since a sample from a Dirichlet distribution of appropriate dimension lies in the solution space; if the solution space has an irregular shape, we may use a truncated Gaussian family and need to use accept-reject sampling to generate candidate solutions that lie within the solution space.

The sampling step (step 2) draws candidate solutions from the current sampling distribution and runs simulation to obtain their performance estimates. The estimation step (step 3) computes the gradient and Hessian estimates using the samples. One common choice of the shape function S_θ is the level/indicator function used in the CE method and MRAS: $I\{H(x) \geq \gamma_\theta\}$, where γ_θ is the $(1-\rho)$-quantile $\gamma_\theta \triangleq \sup_r\{r: P_{f_\theta}\{x \in \mathscr{X}: H(x) \geq r\} \geq \rho\}$, and P_{f_θ} denotes the probability with respect to $f(\cdot;\theta)$. Another similar choice is $(H(x)-H_l)I \cdot \{H(x) \geq \gamma_\theta\}$, where H_l is a lower bound on $H(x)$ so that $S_\theta(H(x))$ is always nonnegative. These shape functions essentially prune the level sets below γ_θ and give equal or more weight to the elite solutions in the level set above γ_θ. By varying ρ, we can adjust the percentile of elite samples that are selected to update the next sampling distribution: the smaller ρ, the fewer elite samples selected, and hence more emphasis is put on exploiting the neighborhood of the current best solutions. These shape functions are found to work well in terms of balancing the trade-off between solution accuracy and convergence speed when the parameter ρ is relatively small (e.g., $\rho \in [0.1, 0.2]$); intuitively, it is because the inferior candidate solutions usually have no implication on the promising regions that contain good solutions. Note the indicator function does not satisfy the regularity condition on S_θ (such as continuity) needed for analysis, so instead, we use a continuous approximation

$$S_\theta(H(x)) = \frac{1}{1+e^{-S_0(H(x)-(1-\delta)\gamma_\theta)}}, \tag{10}$$

where S_0 is a large positive constant and δ is a constant in $(0,1)$. This continuous approximation can be made arbitrarily close to the indicator function $I\{H(x) \geq \gamma_\theta\}$ by choosing sufficiently large S_0 and small δ. When the approximation is close enough, we observe that there is no difference in numerical results between using the exact indicator function or the continuous approximation. Other possible choices of the shape function include $S_\theta(H(x)) = \exp(H(x))$, and $S_\theta(H(x)) = \exp(H(x)I\{H(x) \geq \gamma_\theta\})$, which work well for some problems.

The projection operator in the updating step (step 4) is used to ensure the iterate stays inside the parameter

RIGHTSLINK
Copyright Clearance Center

space Θ. To make sure the projection yields a unique point inside Θ, we take a compact and convex approximation $\tilde{\Theta}$ of Θ. Moreover, to make the projection operation computationally easy, for most of the sampling distributions mentioned above, $\tilde{\Theta}$ can be a hyperrectangle having the form $\tilde{\Theta} = \{\theta \in \Theta : a_i \le \theta_i \le b_i\}$ for constants $a_i < b_i$, $i = 1, \dots, d$, and these constants can be set so as to ensure $\tilde{\Theta}$ is a close approximation to Θ; other more general choices of $\tilde{\Theta}$ may also be used (see, e.g., Kushner and Yin 2004, Section 4.3).

3.1. GASSO-2T

The parameter updating step in GASSO can be interpreted as a stochastic approximation scheme for evaluating the zeros of $\Pi_{\tilde{\Theta}}\{\mathrm{Cov}_{f_\theta}[T(X)] + \epsilon I)^{-1}\nabla_\theta l(\theta;\theta)\}$. To reduce the sample size per iteration and further improve the efficiency, we can also use a stochastic approximation scheme to update the Hessian and gradient estimates but on a faster timescale. That is, at kth iteration we carry out the following steps:

1. Draw candidate solutions $x_k^i \overset{\text{iid}}{\sim} f(x;\theta_k)$, $i = 1, \dots, N$, and simulate to obtain $h(x_k^i, \xi_k^i)$, $i = 1, \dots, N$.
2. Update the gradient and Hessian estimates on the *fast* timescale with stepsize β_k.
3. Update the parameter θ_k on the *slow* timescale with stepsize α_k.

The stepsizes satisfy the standard conditions in two-timescale stochastic approximation (Borkar 2008) as shown in the following assumption.

Assumption 1. *The stepsizes satisfy the following conditions*:

$$\sum_k \alpha_k = \sum_k \beta_k = \infty, \tag{11}$$

$$\sum_k (\alpha_k^2 + \beta_k^2) < \infty, \tag{12}$$

$$\lim_{k\to\infty} \frac{\alpha_k}{\beta_k} = 0. \tag{13}$$

By Assumption 1, as k goes to infinity, β_k dominates α_k. Therefore the parameter θ is updated on the slow timescale with stepsize α_k, while the gradient and Hessian estimates are updated on the fast timescale with stepsize β_k. When k is large, the parameter θ_k remains almost unchanged compared with the gradient and Hessian estimates, therefore θ_k is often said to be quasi-static to the fast timescale. This implies that the sampling distribution can be viewed as fixed over many iterations while the gradient and Hessian estimates are updated, and therefore, often a much smaller sample size is needed for every iteration.

With the above idea, we propose the following two-timescale variant of GASSO, named as GASSO-2T.

Algorithm 2 (GASSO-2T)

1. *Initialization*: choose an exponential family of densities $\{f(\cdot;\theta)\}$, and specify a small positive constant ϵ, initial parameter θ_0, sample size N (integer ≥ 1), simulation budget $\{M_k\}$ that satisfies $M_k \to \infty$ ($k \to \infty$), and stepsize sequences $\{\alpha_k\}$ and $\{\beta_k\}$ that satisfy Assumption 1. Set $k = 0$. Set $L_0(1) = 0$, $G_0(1) = 0$, $ET_0(1) = 0$, $T_0(1) = 0$, $Q_0(1) = 0$.

2. *Sampling*: draw samples $x_k^i \overset{\text{iid}}{\sim} f(x;\theta_k)$, $i = 1, 2, \dots, N$. For each x_k^i, evaluate the performance by simulation to generate $h(x_k^i, \xi_k^{i,j})$, $j = 1, \dots, M_k$, and compute the performance estimate $\hat{H}(x_k^i) = (1/M_k)\sum_{j=1}^{M_k} h(x_k^i, \xi_k^{i,j})$.

3. *Estimation*:

(a) First, we estimate $L(\theta_k;\theta_k)$ in the following manner: For $i = 1, \dots, N$, update

$$L_k(i+1) = L_k(i) + \beta_k(S_{\theta_k}(\hat{H}(x_k^i)) - L_k(i)). \tag{14}$$

Set $L_{k+1}(1) := L_k(N+1)$.

(b) Next, we estimate $E_{g_{\theta_k}}[T(X)]$ as follows: For $i = 1, \dots, N$, update

$$G_k(i+1) = G_k(i) + \beta_k\left(\frac{S_{\theta_k}(\hat{H}(x_k^i))}{L_k(N+1)} T(x_k^i) - G_k(i)\right). \tag{15}$$

Set $\hat{E}_{g_{\theta_k}}[T(X)] := G_k(N+1)$. Set $G_{k+1}(1) := G_k(N+1)$.

(c) Finally, we estimate $\mathrm{Cov}_{f_{\theta_k}}[T(X)]$ as follows: For $i = 1, \dots, N$, update

$$P_k(i+1) = P_k(i) + \beta_k(T(x_k^i) - P_k(i)) \tag{16}$$

$$Q_k(i+1) = Q_k(i) + \beta_k(T(x_k^i)T(x_k^i)' - Q_k(i)). \tag{17}$$

Set $\widehat{\mathrm{Cov}}_{f_{\theta_k}}[T(X)] := Q_k(N+1) - P_k(N+1)P_k(N+1)'$. Set $P_{k+1}(1) := P_k(N+1)$ and $Q_{k+1}(1) := Q_k(N+1)$.

4. *Updating*: update the parameter θ according to

$$\theta_{k+1} = \Pi_{\tilde{\Theta}}\Big\{\theta_k + \alpha_k\big(\widehat{\mathrm{Cov}}_{f_{\theta_k}}[T(X)] + \epsilon I\big)^{-1} \cdot \big(\hat{E}_{g_{\theta_k}}[T(X)] - E_{f_{\theta_k}}[T(X)]\big)\Big\}, \tag{18}$$

where $\tilde{\Theta} \subseteq \Theta$ is a nonempty compact and convex set that closely approximates Θ, and $\Pi_{\tilde{\Theta}}$ denotes the projection operator that projects an iterate back onto the set $\tilde{\Theta}$ by choosing the closest point in $\tilde{\Theta}$.

5. *Stopping*: check if some stopping criterion is satisfied. If yes, stop and return the current best sampled solution; else, set $k := k+1$ and go back to step 2.

GASSO-2T differs from GASSO in the estimation step (step 3), where the expectation and covariance terms are estimated through stochastic approximation iterations instead of sample average estimates. For example, in step 3(a) the samples generated at the current iteration are used to update the estimate for $L(\theta_k;\theta_k)$ sequentially, and the final estimate $L_k(N+1)$ of the current iteration is set to be the initial estimate $L_{k+1}(1)$ of the next iteration. In contrast to GASSO which estimates $L(\theta_k;\theta_k)$ by the sample average of samples generated at the current iteration, GASSO-2T uses all the samples that have been generated so far.

RIGHTSLINK

Because GASSO-2T uses all the samples generated in the current and previous iterations for estimation, GASSO-2T only needs a fixed and usually small sample size at each iteration. To make it easier to see, consider the case that only a single sample x_k is drawn in iteration k, i.e., $N=1$, for all iterations. Then, for example, the iterate in step 3(a) can be written as

$$L_{k+1}=L_k+\beta_k(S_{\theta_k}(\hat{H}(x_k))-L_k)=(1-\beta_k)L_k+\beta_k S_{\theta_k}(\hat{H}(x_k)),$$

where we suppress the index i because $N=1$. By induction, we can write out L_{k+1} as a weighted sum of all sample values $S_{\theta_0}(\hat{H}(x_0)),\ldots,S_{\theta_k}(\hat{H}(x_k))$ from the first iteration to the current iteration, with smaller weight on the samples further away in the past and all weights summing up to one. The intuitive reason that we can use this weighted-sum estimate is because θ_k is updated on a slow timescale, and thus there is little change in θ_k from iteration to iteration viewed by the L_k-iterate that is updated on the fast timescale; hence, samples from previous iterations can be used to estimate the expected value under the current distribution $f(\cdot;\theta_k)$, but their weights need to be discounted to take into account that there is still a little difference between previous distributions and the current one. Therefore the effective sample size for estimation of the gradient and Hessian in GASSO-2T is roughly equal to the total sample size of all iterations, and this effective sample size automatically increases to infinity as the iteration number goes to infinity, which makes estimation error go to zero in the limit. Theoretically, the sample size per iteration can be $N=1$ in GASSO-2T, but, in practice, we usually use a sample size from 50 to 100 for numerical stability. In contrast, since GASSO only uses the samples generated at the current iteration to estimate the gradient and Hessian, it requires the sample size per iteration to increase to infinity (i.e., $N_k\to\infty$, $k\to\infty$), to ensure the estimation error goes to zero so that the convergence of the algorithm can be guaranteed. This intuition will be made formal in the convergence analysis.

4. Main Convergence Results

In this section, we first present Assumptions 2 and 3. Due to the lack of space, the detailed convergence proof is presented in the accompanying online supplement. Nonetheless, we state here our main convergence results, viz., Theorem 1, Lemma 1, and Corollary 1, on the convergence of the iterates obtained from GASSO-2T. The proofs of Lemma 1 and Corollary 1 are also presented here as they are independent of other results. Our analysis draws upon techniques from stochastic approximation literature (Kushner and Yin 2004, Borkar 2008). The convergence proof for GASSO is straightforward and is briefly mentioned in Remark 2 of the online supplement.

We do not analyze the rate of convergence of our algorithm. In general, it is hard to obtain rate of convergence results for two-timescale stochastic approximation, see, however, Konda and Tsitsiklis (2004) where rate results for a two-timescale scheme with linear (faster and slower) recursions have been obtained.

We make the following regularity assumptions.

Assumption 2. (i) *The set $\mathscr{X}$ is compact.*

(ii) *For a fixed $\theta\in\tilde{\Theta}$, the function $S_\theta(H(\cdot))$ is Lipschitz continuous with Lipschitz constant $K_\theta<\infty$.*

(iii) *The set of functions $\{S_\theta(\cdot),\theta\in\tilde{\Theta}\}$ is uniformly bounded.*

(iv) *For any fixed $\theta'\in\tilde{\Theta}$, the function $L(\theta;\theta')$ is twice continuously differentiable for $\theta\in\tilde{\Theta}$.*

Assumption 3. (i) *The map $x\mapsto T(x)$ is bounded on $\mathscr{X}$.*

(ii) *For any fixed x, the pdf $f(x;\theta)$ is continuous in $\theta\in\Theta$.*

Note that Assumption 2(i) is imposed for the convergence analysis. In practice, we may apply the algorithm to more general problems where $\mathscr{X}$ is not compact. We use the shorthand notation $\nabla_\theta l(\theta';\theta')$ for $\nabla_\theta l(\theta;\theta')|_{\theta=\theta'}$. The ordinary differential equation (ODE) corresponding to the θ-update in step 4 of Algorithm 2 is as follows:

$$\dot{\theta}(t)=\tilde{\Pi}_{\tilde{\Theta}}\{V(\theta(t))^{-1}\nabla_\theta l(\theta(t);\theta(t))\},\tag{19}$$

where the operator $\tilde{\Pi}_{\tilde{\Theta}}(\cdot)$ is defined as follows: for any continuous function $v:\mathbb{R}^d\to\mathbb{R}^d$,

$$\tilde{\Pi}_{\tilde{\Theta}}(v(y))=\lim_{\eta\to 0}\left(\frac{\Pi_{\tilde{\Theta}}(y+\eta v(y))-\Pi_{\tilde{\Theta}}(y)}{\eta}\right).$$

Let Z denote the set of equilibria of (19) where

$$Z\triangleq\{\theta\in\tilde{\Theta}\mid\tilde{\Pi}_{\tilde{\Theta}}\{V(\theta)^{-1}\nabla_\theta l(\theta;\theta)\}=0\}.$$

Theorem 1. *Under Assumptions 1–3, $\theta_k\to Z$ w.p.1 as $k\to\infty$. Note that $\theta_k\to Z$ means $\lim_{k\to\infty}\mathrm{dist}(\theta_k,Z)=0$, where $\mathrm{dist}(\theta,Z)=\min_{\theta'\in Z}|\theta-\theta'|$.*

Lemma 1. *The stationary points $\{\theta:\nabla_\theta l(\theta;\theta)=0\}$ lie in the set Z.*

Proof. Suppose $W(\theta)\triangleq -l(\theta;\theta)$. Then, $W(\theta)$ works as a Lyapunov function for the ODE (19). This can be seen in the following manner: Note that

$$\begin{aligned}\frac{dW(\theta)}{dt}&=(\nabla_\theta W(\theta))'\frac{d\theta}{dt}\\&=-\nabla_\theta l(\theta;\theta)'\tilde{\Pi}_{\tilde{\Theta}}\{V(\theta)^{-1}\nabla_\theta l(\theta;\theta)\}.\end{aligned}$$

If $\theta\in\tilde{\Theta}^0$ (the interior of $\tilde{\Theta}$), then for $\eta>0$ small enough,

$$F_{\theta,\eta}\triangleq\theta+\eta V(\theta)^{-1}\nabla_\theta l(\theta;\theta)\in\tilde{\Theta}^0$$

as well; hence, by definition of $\tilde{\Pi}_{\tilde{\Theta}}(\cdot)$, we get

$$\tilde{\Pi}_{\tilde{\Theta}}(V(\theta)^{-1}\nabla_\theta l(\theta;\theta))=V(\theta)^{-1}\nabla_\theta l(\theta;\theta).$$

This would also be true in the case of $\theta \in \partial\tilde{\Theta}$ (the boundary of $\tilde{\Theta}$) when $F_{\theta,\eta} \in \tilde{\Theta}^0$ for small $\eta > 0$. Since $V(\theta)^{-1}$ is positive definite and symmetric,

$$-\nabla_\theta l(\theta;\theta)' V(\theta)^{-1} \cdot \nabla_\theta l(\theta;\theta) \begin{cases} < 0, & \text{for } \theta \text{ s.t. } \nabla_\theta l(\theta;\theta) \neq 0; \\ = 0, & \text{o.w.} \end{cases}$$

Thus the points where $\nabla_\theta l(\theta;\theta) = 0$ lie in the set Z. □

Remark 1. In principle, because of the projection, there may exist spurious fixed points that lie in Z (see Kushner and Yin 2004), for which

$$-\nabla_\theta l(\theta;\theta)' \tilde{\Pi}_{\tilde{\Theta}}\{V(\theta)^{-1}\nabla_\theta l(\theta;\theta)\} = 0$$

even when $\nabla_\theta l(\theta;\theta) \neq 0$. Such fixed points would, however, lie on the boundary ($\partial\tilde{\Theta}$) of $\tilde{\Theta}$. In case the equilibria of the unprojected ODE, i.e., (19), without the $\tilde{\Pi}_{\tilde{\Theta}}$ operator, lie outside $\tilde{\Theta}$, then the equilibria of the projected ODE would lie in $\partial\tilde{\Theta}$ itself. Such equilibria would also be close to the ones of the unprojected ODE since $\tilde{\Theta}$ closely approximates Θ.

As argued in Remark 2 of the online supplement, Theorem 1 and Lemma 1 also hold in the case of GASSO. Assuming that the set Z comprises only of isolated equilibria, these results together imply that GASSO and GASSO-2T converge to a stationary point that solves $\nabla_\theta l(\theta;\theta) = 0$. Let's assume this stationary point is a local optimal solution (not a saddle point). Next, we show the relationship between such a point and a local optimal solution to the original problem (1).

Corollary 1. *If θ^* is a local optimal solution to the problem $\sup_{\theta\in\Theta} l(\theta,\theta)$ and $f(\cdot;\theta^*)$ is a degenerate distribution, then the support of $f(\cdot;\theta^*)$ denoted by x^* is a local optimum of problem* (1).

Proof. This can be shown by contradiction. If it is not true, then there exists a degenerate distribution $f(\cdot;\theta')$ such that its support x' is in the neighborhood of x^* and $H(x') > H(x^*)$. Note that θ^* and θ' are in $\partial\Theta$ (the boundary of Θ) since they are both degenerate, and hence θ' is in the neighborhood (confined to $\partial\Theta$) of θ^*. Denote by $\{\theta'_n \in \Theta, n = 1,2,\ldots\}$ a sequence in Θ such that $\theta'_n \to \theta'$ as $n \to \infty$, and similarly, $\{\theta^*_n \in \Theta, n = 1,2,\ldots\}$ a sequence in Θ such that $\theta^*_n \to \theta^*$ as $n \to \infty$. Therefore $\lim_{\theta'_n\to\theta'} l(\theta'_n;\theta'_n) = \ln S_{\theta'}(H(x')) > \ln S_{\theta^*}(H(x^*)) = \lim_{\theta^*_n\to\theta^*} l(\theta^*_n;\theta^*_n)$, where the inequality follows from the strictly increasing property of the logarithm function and the forms of $S_\theta(\cdot)$ of all the choices considered in this paper. It contradicts with the assumption that θ^* is a locally optimal solution to the problem $\sup_{\theta\in\Theta} l(\theta;\theta)$. Therefore x^* has to be a local optimal solution of the original problem. □

Remark 2. Corollary 1 implies that GASSO and GASSO-2T converge to a locally optimal solution of the original problem if the limiting sampling distribution is a degenerate distribution. Note that the parameter of a nondegenerate distribution may also be a stationary point, but such a θ^* is an unstable limiting point in practice. The reason is that a sample estimate of the gradient $\nabla_\theta l(\theta;\theta)$ will not be exactly equal to zero, due to the variability caused by sampling from a nondegenerate distribution. Hence the θ-iterate, driven by the gradient-based algorithm, will not rest on such a point. This intuition has also been empirically verified in our numerical experiments, where we have always observed the covariance matrix of the sampling distribution converge to a zero matrix.

5. Numerical Experiments

In this section, we demonstrate the performance of GASSO and GASSO-2T on several benchmark problems and a simulation optimization problem, and conclude with our recommendations on algorithm parameter choices for practitioners.

5.1. Benchmark Problems

We test the algorithms on six benchmark continuous optimization problems with additive noise ξ_x. We consider three cases for the noise ξ_x: (i) stationary noise, where ξ_x is normally distributed with mean 0 and variance 100; (ii) increasing noise, where ξ_x is normally distributed with mean zero and variance $\|x\|_2^2$; (iii) decreasing noise, where ξ_x is normally distributed with mean zero and variance $100/(\|x\|_2^2 + 1)$. The test functions are all 10 dimensional, and their expressions are listed below. Specifically, Powell (H_1) is a badly scaled function; Trigonometric (H_2) and Rastrigin (H_3) are multimodal functions with a large number of local optima, and the number of local optima increases exponentially with the problem dimension; Pintér (H_4) and Levy (H_5) are multimodal and badly scaled problems; and the weighted sphere function (H_6) is a multidimensional concave function.

(1) Powell singular function ($n = 10$)

$$h_1(x,\xi_x) = -1 - \sum_{i=2}^{n-2}[(x_{i-1} + 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4] + \xi_x,$$

where $x^* = (0,\ldots,0)^T$, $H_1(x^*) = E[h_1(x^*,\xi_x)] = -1$.

(2) Trigonometric function ($n = 10$)

$$h_2(x,\xi_x) = -1 - \sum_{i=1}^{n}[8\sin^2(7(x_i - 0.9)^2) + 6\sin^2(14(x_i - 0.9)^2) + (x_i - 0.9)^2] + \xi_x,$$

where $x^* = (0.9,\ldots,0.9)^T$, $H_2(x^*) = E[h_2(x^*,\xi_x)] = -1$.

RIGHTSLINK

(3) Rastrigin function ($n = 10$)

$$h_3(x, \xi_x) = -\sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)) - 10n - 1 + \xi_x,$$

where $x^* = (0, \ldots, 0)^T$, $H_3(x^*) = E[h_3(x^*, \xi_x)] = -1$.

(4) Pintér's function ($n = 10$)

$$\begin{aligned} &h_4(x, \xi_x) \\ &= -\Bigg[\sum_{i=1}^{n} i x_i^2 + \sum_{i=1}^{n} 20i \sin^2(x_{i-1}\sin x_i - x_i + \sin x_{i+1}) \\ &\quad + \sum_{i=1}^{n} i \log_{10}(1 + i(x_{i-1}^2 - 2x_i + 3x_{i+1} - \cos x_i + 1)^2)\Bigg] \\ &\quad - 1 + \xi_x, \end{aligned}$$

where $x^* = (0, \ldots, 0)^T$, $H_4(x^*) = E[h_4(x^*, \xi_x)] = -1$.

(5) Levy function ($n = 10$)

$$\begin{aligned} &h_5(x, \xi_x) \\ &= -1 - \sin^2(\pi y_1) - \sum_{i=1}^{n-1}[(y_i - 1)^2(1 + 10\sin^2(\pi y_i + 1))] \\ &\quad - (y_n - 1)^2(1 + 10\sin^2(2\pi y_n)) + \xi_x, \end{aligned}$$

where $y_i = 1 + x_i/4$, $x^* = (0, \ldots, 0)^T$, $H_5(x^*) = E[h_5(x^*, \xi_x)] = -1$.

(6) Weighted sphere function ($n = 10$)

$$h_6(x, \xi_x) = -1 - \sum_{i=1}^{n} i x_i^2 + \xi_x,$$

where $x^* = (0, \ldots, 0)^T$, $H_6(x^*) = E[h_6(x^*, \xi_x)] = -1$.

We compare our algorithms to the CEOCBA method proposed by He et al. (2010). CEOCBA is a state-of-the-art stochastic optimization algorithm. It incorporates the successful ranking and selection procedure OCBA (Chen et al. 2000) into each iteration of the well-known CE method, by smartly allocating the simulation budget to candidate solutions at every iteration. He et al. (2010) have shown that CEOCBA is a great improvement on the standard CE method with equal allocation. Theoretically, the OCBA approach is only applicable to the case of additive Gaussian noise in the objective function, although heuristic extension to other additive noises is also possible. In contrast, our method does not require such additive and/or Gaussian assumption on the simulation noise, and is valid for general forms of objective functions.

In all three algorithms, we use the independent multivariate Normal distribution $\mathcal{N}(\mu_k, \Sigma_k)$ as the parameterized distribution $f(\cdot; \theta_k)$, where the covariance matrix is a diagonal matrix. We also test the instance of GASSO using the correlated multivariate Normal with a full covariance as the sampling distribution. The two GASSO implementations are denoted as GASSO (independent Normal) and GASSO (multivariate Normal), respectively, in the following tables and figures. The initial mean μ_0 is chosen randomly according to uniform distributions on $[-30, 30]^n$, and the initial covariance matrix is set to be $\Sigma_0 = 1{,}000 I_{n\times n}$, where n is the dimension of the problem, and $I_{n\times n}$ is the identity matrix with size n. We observe in the experiment that the performance of the algorithm is insensitive to the initial candidate solutions, if the initial variance is large enough.

Since CEOCBA uses the elite samples at each iteration to update the sampling distribution, a comparable choice in our algorithms is to set the shape function as the indicator function $S_{\theta_k}(h(x, \xi)) = I\{h(x, \xi) \geq \gamma_{\theta_k}\}$. However, the indicator function does not satisfy our assumption on the shape function in the convergence analysis, therefore we use a continuous approximation of the form (10) with $S_0 = 10^5$ and $\delta = 10^{-3}$, which makes $S_{\theta_k}(h(x, \xi))$ a very close approximation to $I\{h(x, \xi) \geq \gamma_{\theta_k}\}$. In all three algorithms, the quantile parameter ρ is set to be 0.1, and the $(1-\rho)$-quantile γ_{θ_k} is estimated by the $(1-\rho)$ sample quantile of the function values corresponding to all the candidate solutions generated at the kth iteration. In our experiment, we observe that the results using the previous two shape functions are indistinguishable (which is not surprising at all because of the close approximation). In practice, to simplify implementation, we can simply use the indicator function.

The sample size for each iteration is set to be $N = 1{,}000$ for GASSO and CEOCBA; the sample size is $N = 100$ for GASSO-2T, since GASSO-2T can use a small sample size due to the two-timescale updating scheme. For each candidate solution, we evaluate $M = 10$ times to estimate its performance in GASSO and GASSO-2T; for a fair comparison, the simulation budget T per iteration in CEOCBA is set to be $T = NM = 10{,}000$. Please note that although the asymptotic convergence requires the sample size N_k and simulation budget M_k go to infinity as the iteration k goes to infinity, it is sufficient to use constant N and M in practice when the total iteration number is finite. Other parameters in GASSO and GASSO-2T are set as follows: the small constant $\epsilon = 10^{-10}$, and the stepsizes $\alpha_k = 50/(k + 1{,}500)^{0.6}$ and $\beta_k = 1/(k + 1{,}500)^{0.55}$, which satisfy Assumption 1. Other parameters in CEOCBA are set as follows: the initial number of function evaluations for each candidate solution $n_0 = 5$, the budget increment $\Delta = 100$, and we also apply the smoothing parameter updating procedure in DeBoer et al. (2005) on the parameter updating to prevent premature convergence, where the smoothing parameter is chosen to be $\nu = 0.5$ as suggested in He et al. (2010). For all algorithms, we have implemented versions that use common random numbers (CRN) in evaluating the performance across different candidate solutions of the same iteration; these versions

RIGHTSLINK

Table 1. Stationary Noise—Average Performance of GASSO (Independent Normal), GASSO (Multivariate Normal), GASSO-2T, and CEOCBA

		GASSO (indep. Nor.)		GASSO (multi. Nor.)		GASSO-2T		CEOCBA	
Variable	H^*	$\bar{H}^*$	*std_err*	$\bar{H}^*$	*std_err*	$\bar{H}^*$	*std_err*	$\bar{H}^*$	*std_err*
Powell H_1	−1	−1.000	0.000	−1.001	0.000	−1.001	0.000	−5.144	3.271
Trigonometric H_2	−1	−1.000	0.000	−1.000	0.000	−1.000	0.000	−1.000	0.000
Rastrigin H_3	−1	−1.000	0.000	−1.000	0.000	−1.040	0.028	−1.000	0.000
Pintér H_4	−1	−1.002	0.044	−1.000	0.000	−1.565	0.017	−1.809	0.015
Levy H_5	−1	−1.012	0.000	−1.002	0.000	−1.000	0.000	−1.002	0.000
Sphere H_6	−1	−1.000	0.000	−1.000	0.000	−1.000	0.000	−1.000	0.000

have demonstrated noticeable improvement over versions without CRN. The results presented next are versions with CRN.

We run each of these three algorithms 50 times independently. The average performance of the algorithms are presented in Table 1 and Figure 1 in this paper and Tables 2–3 and Figures 2–3 in the online supplement. In particular, Table 1/Figure 1, Table 2/Figure 2 (online supplement), and Table 3/Figure 3 (online supplement) show the results for stationary noise, increasing noise, and decreasing noise, respectively. In the tables, we denote by H^* the true optimal value of the test function, and $\bar{H}^*$ the average of the function values $H(\mu_K)$ returned by the 50 runs of an algorithm, where K is the number of iterations of an algorithm, and *std_err* the standard error of the optimal function values. In the experiments, we found the computation time of function evaluations dominates the time of other steps, therefore we compare the performance of the algorithms with respect to the total number of function evaluations in the figures.

For all the test functions, the average optimal function values returned by GASSO (independent Normal) and GASSO (multivariate Normal) are very close, but GASSO (multivariate Normal) usually converges slightly faster, probably because it uses a larger family of sampling distributions. GASSO-2T converges faster than all other algorithms and often reduces the total number of function evaluations needed for convergence by about 3–4 times, which confirms the benefit of using the two-timescale scheme. Overall, CEOCBA and GASSO have similar performance in terms of solution accuracy and convergence speed, but GASSO is more accurate in Powell and Pintér's functions. It is worth mentioning that although GASSO currently allocates the simulation budget equally to the candidate solutions, its performance is very comparable to CEOCBA and is sometimes more accurate. The OCBA scheme may also be incorporated into GASSO and GASSO-2T to further improve their empirical performance.

Table 2. Product Promotion Example—Average Performance of GASSO, GASSO-2T, and SMRAS

	GASSO		GASSO-2T		SMRAS	
Variable	$\bar{H}^*$	*std_err*	$\bar{H}^*$	*std_err*	$\bar{H}^*$	*std_err*
$Q = 10$, $S = 5$	305.77	0.28	305.95	0.23	301.39	0.89
$Q = 10$, $S = 10$	352.62	0.78	352.88	0.61	344.75	1.33

5.2. A Simulation Optimization Example

We also test our algorithms on a simulation optimization problem of product promotion management, which is adapted from Jacko (2016). Retail stores or supermarkets need to decide how to allocate products of Q different categories to S promotion shelves, where these products are more likely to attract customers and bring in more revenues to the retailers. Each promotion shelf has a limited space B_s. There is a random customer demand D_q for each product category q. Each sold product of category q yields a profit r_q, and each unsold product remains in the promotion space and incurs a holding cost h_q per day. The goal is to determine $x_{q,s}$, the allocation of product category q $(=1,\ldots,Q)$ on the promotion shelf s $(=1,\ldots,S)$, to maximize the total profit over a promotion period of T days. This optimization problem can be written as follows:

$$\max_{\{x_{s,q}\}} \sum_{t=1}^{T}\sum_{q=1}^{Q} E\big[r_q \min(D_q, I_q) - h_q \max(I_q(t) - D_q(t), 0)\big] \tag{20}$$

$$\text{s.t. } I_q(1) = \sum_{s=1}^{S} x_{s,q}; \quad I_q(t+1) = \max(I_q(t) - D_q(t), 0), \quad t = 1,\ldots,T, \tag{21}$$

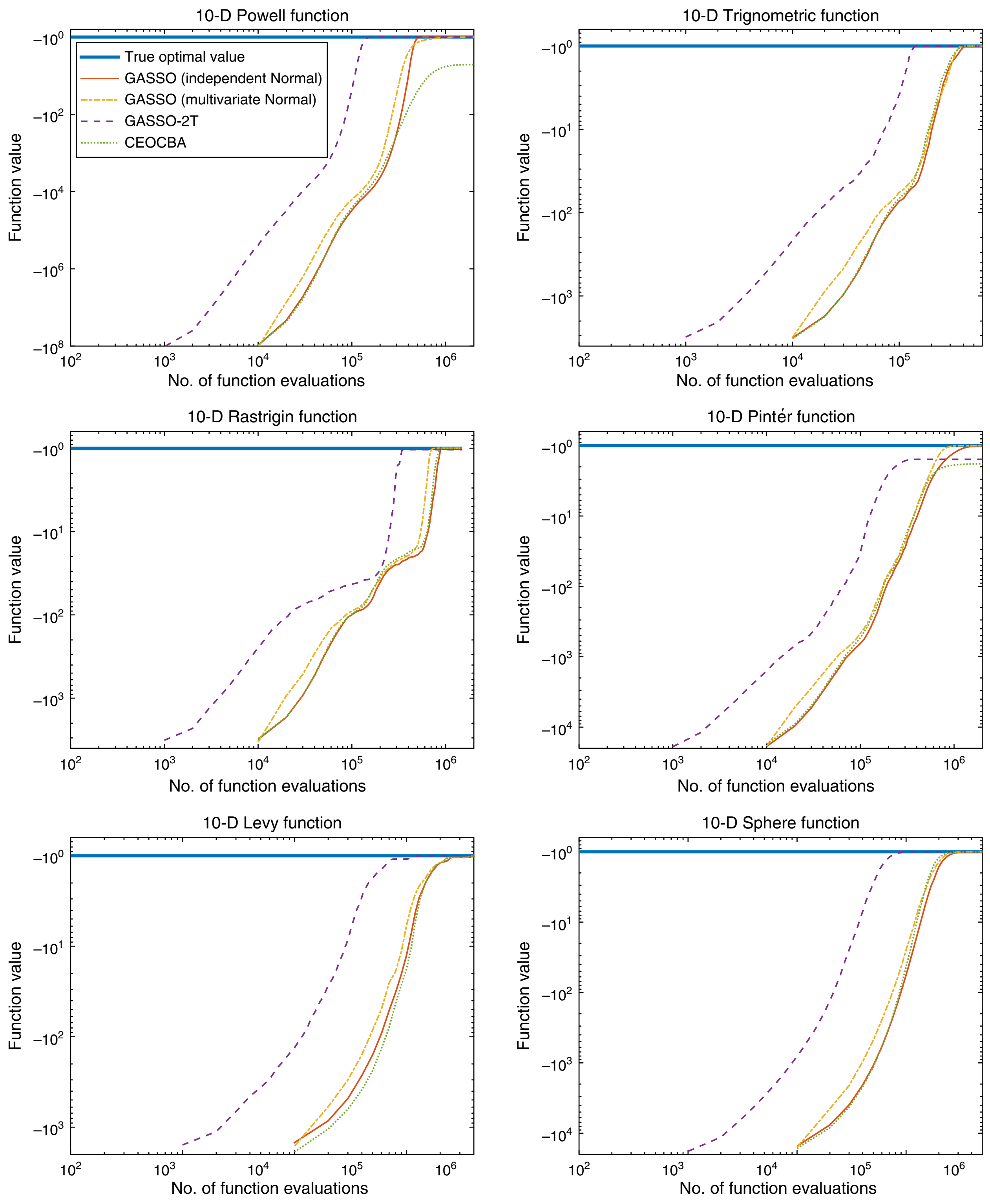
$$\sum_{q=1}^{Q} x_{s,q} \le B_s, \quad s = 1,\ldots,S, \tag{22}$$

$$x_{s,q} \ge 0, \quad s = 1,\ldots,S,\ q = 1,\ldots,Q. \tag{23}$$

In the constraint (21), $I_q(t)$ represents the inventory level of product category q at time t. The constraints (22) and (23) specify the budget constraint and nonnegativity constraint on the allocation, respectively. We ignore the integrality constraint on the allocation, assuming any solution can be rounded to a nearest integer solution with similar performance. Notice that the dimension of the problem is $Q \times S$. The objective function (20) can only be evaluated through simulation, via

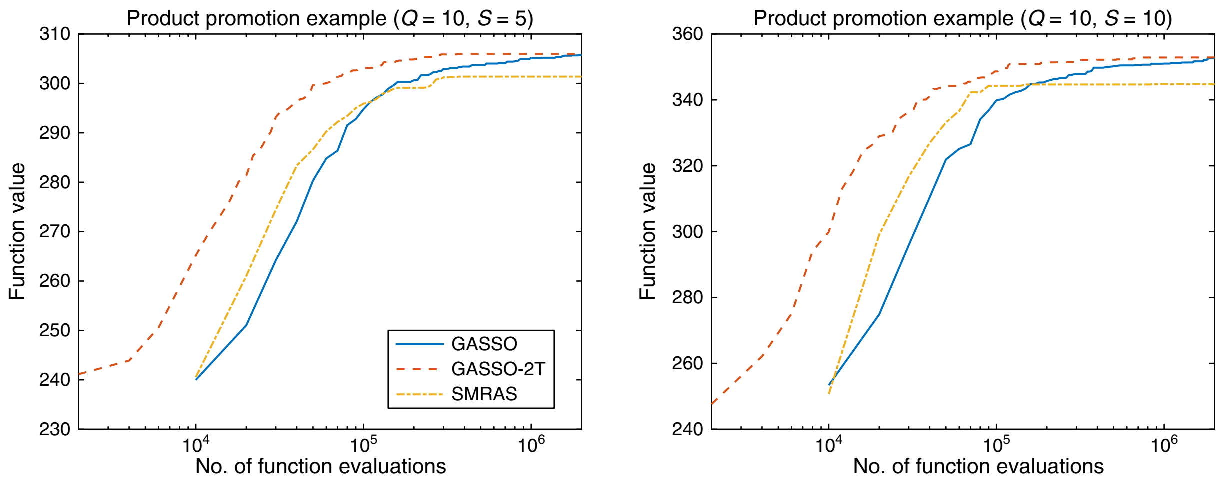
RIGHTSLINK

Figure 1. (Color online) Stationary Noise—Average Performance of GASSO (Independent Normal), GASSO (Multivariate Normal), GASSO-2T, and CEOCBA



simulating the evolution of $I_q(t)$ according to (21). In our numerical experiment, we use the following problem setting: $T = 7$ (i.e., time horizon is one week), $Q = 10$, $S = 5$, or 10, $B_s = 5$ for all s, D_q follows a uniform distribution on $[0, 20]$; r_q is randomly generated from a uniform distribution on $[0, 20]$ and the specific instance presented here is $[15.2, 15.4, 6.4, 7.2, 17.1, 18.2, 13.8, 19.6, 15.8, 15.0]$; h_q is generated from a uniform distribution on $[0, 10]$, and the specific instance presented here is $[2.9, 0\,7.6, 2.5, 3.9, 6.8, 1.5, 6.1, 7.1, 1.9, 4.4]$.

Figure 2. (Color online) Product Promotion Example—Average Performance of GASSO, GASSO-2T, and SMRAS



The constraints (22) and (23) imply that for each s, the solution vector $x_s = [x_{s,1}, \dots, x_{s,Q}]$ lives inside a $(Q+1)$-dimensional simplex (multiplied by B_s). Therefore, in the implementation of GASSO and GASSO-2T, we use the Dirichlet distribution (which is an exponential family of distributions) as the parameterized sampling distribution, because the support of a Dirichlet distribution is a simplex, which means that any samples generated from an appropriate Dirichlet would satisfy the constraints (22) and (23). More specifically, the pdf of the sampling distribution is $f(y;\theta) = \prod_{s=1}^{S} f(y_s;\theta_s)$, where for each s

$$f(y_s;\theta_s) = \frac{\prod_{i=1}^{Q+1} y_{s,i}^{\theta_{s,i}-1}}{B(\theta)}, \quad y_{s,i} \ge 0 \,\forall\, i, \quad \sum_{i=1}^{Q+1} y_{s,i} = 1,$$

where $B(\theta)$ is a multinomial Beta function. By taking the transformation $x_{s,i} = B_s \times y_{s,i}, i = 1, \dots, Q$, we note that the $x_{s,i}$'s satisfy the constraints (22) and (23). We set the initial parameter θ as an all-one vector, the stepsize $\alpha_k = 50/(k+1{,}000)^{0.6}$ and $\beta_k = 1/(k+1{,}000)^{0.501}$, and the sample size $N = 200$ per iteration for GASSO-2T; all the other parameter settings are the same as in the previous section.

For this problem, although we can still apply CEOCBA numerically, it is not a fair comparison because CEOCBA is derived based on the assumption of an additive Gaussian noise in the objective function, which obviously is far from satisfied by the objective function in this example. Instead, we compare performance with the SMRAS proposed by Hu et al. (2008), which is an extension of the very competitive model-based algorithm MRAS to the stochastic optimization setting. For fair comparison with GASSO and GASSO-2T, we use the same sample size $N = 1{,}000$ for each iteration and $M = 10$ for the number of evaluations of each candidate solution, and use the Dirichlet distribution above as the sampling distribution as well. All the other parameters are set as recommended in Hu et al. (2008).

Table 2 and Figure 2 show the average performance of the three algorithms based on 10 independent runs, for the problems with 50 ($Q = 10, S = 5$) and 100 ($Q = 10, S = 10$) dimensions, respectively. The results show that GASSO and GASSO-2T converge to a better solution than SMRAS, SMRAS converges faster than GASSO but slower than GASSO-2T, and GASSO-2T again demonstrates a considerable computational saving compared to GASSO. Lastly, we should mention many resource allocation problems take a similar form as this problem here with the same budget and nonnegativity constraints on the resource allocation, for example, users' power allocation to transmit signal through different channels, see Hayashi and Luo (2009). If GASSO or GASSO-2T is to be applied on these problems, Dirichlet distribution would be a natural choice for the sampling distribution because its support coincides with the constraint set.

5.3. Recommendation on Parameter Choice

Based on theoretical intuition and our numerical experience, we give some general recommendations on the parameter settings in the algorithms.

Theoretically, the sample size per iteration needs to increase to infinity as the number of iterations goes to infinity in GASSO, and it can be constant for all iterations in GASSO-2T. In practice, since all algorithms have to stop after a finite number of iterations, we often set the sample size to be large enough. If the sample size is too small, there is too much noise in the estimate and the algorithm may not even converge; as we increase the sample size, the convergence usually improves but the improvement starts to diminish after a certain point. We recommend $N \in [500, 1{,}000]$ for GASSO and $N \in [50, 100]$ for GASSO-2T for problems of dimension under 100, and, in general, a larger sample size is needed as the problem dimension increases. To find the sweet spot, a practitioner can start with a relatively small sample size and gradually increase the

RIGHTSLINK

sample size until no further improvement on convergence can be observed. How to adaptively and automatically set the sample size in these algorithms is one of our future research directions, along the lines of prior work, see, Byrd et al. (2012), Hashemi et al. (2014).

The stepsize is probably the most difficult parameter to set in these algorithms, just as this is a common problem with many stochastic approximation algorithms. Usually, a smaller stepsize sequence provides a more accurate solution but takes more computational time, and on the contrary, a larger stepsize may cause the algorithm to converge too fast and prematurely to a bad local solution. Therefore we need to balance the initial stepsize and the decay rate of the sequence. We use a stepsize of the form $\alpha_k = \alpha_0/(C+k)^{\alpha}$, therefore roughly speaking, the ratio α_0/C determines the initial stepsize and the denominator $(C+k)^{\alpha}$ determines the decay rate. For GASSO, we recommend a large $C \in [1{,}000, 20{,}000]$ and a small $\alpha \in (0.5, 0.65)$ for a slow decay rate, and then choose α_0 as large as possible to increase the convergence speed while still maintaining convergence to a good solution (once α_0 becomes too large, the algorithm starts to become nonconvergent). For GASSO-2T, we need to additionally set the stepsize β_k, and we recommend to use the same form as α_k, i.e., $\beta_k = \alpha_0/(C+k)^{\beta}$, where the exponent β should be smaller than α and between $[0.501, 0.55]$.

The performance of the algorithms is not very sensitive to other parameter settings. We recommend the following: shape function set to be $I\{H(x) > \gamma_\rho\}$, or $(H(x) - H_l)I\{H(x) > \gamma_\rho\}$ (the former choice is more robust while the latter is often faster), quantile parameter $\rho \in [0.1, 0.2]$, and initial parameter of the sampling distribution set to be close to a uniform distribution over the solution space.

6. Conclusion

In this paper, we developed a gradient-based adaptive stochastic search approach to simulation optimization over continuous solution space. Our proposed algorithm iteratively draws candidate solutions from a parameterized sampling distribution and updates the parameter of the sampling distribution using a direct gradient search over the parameter space. To reduce the number of candidate solutions that need to be generated per iteration, we further incorporated into the algorithm a two-timescale updating scheme, which updates the parameter on the slow timescale and the quantities involved in the parameter updating on the fast timescale. We showed asymptotic convergence of the proposed algorithms. Numerical results on several benchmark problems and a simulation optimization problem showed our algorithms have superior empirical performance, and the two-timescale algorithm improves computational efficiency by several fold. We also provided a detailed recommendation for practitioners on the parameter setting in the algorithms.

Acknowledgments

The authors thank Xi Chen, graduate student in the Department of Industrial and Enterprise Systems Engineering at University of Illinois Urbana–Champaign, for assisting with the numerical experiments in the preliminary version (Zhou et al. 2014) of this work. The authors also thank the Area Editor Marvin Nakayama, the associate editor, and two anonymous referees for their careful reading of the paper and very constructive comments that led to a substantially improved paper.

References

Andradóttir S (2006) An overview of simulation optimization with random search. Henderson SG, Nelson BL, eds. *Handbooks in Operations Research and Management Science: Simulation*, Vol. 13 (Elsevier, Amsterdam), 617–632.

Barndorff-Nielsen OE (1978) *Information and Exponential Families in Statistical Theory* (Wiley, New York).

Barton RR, Meckesheimer M (2006) Metamodel-based simulation optimization. Henderson SG, Nelson BL, eds. *Handbooks in Operations Research and Management Science: Simulation*, Vol. 13 (Elsevier, Amsterdam), 535–574.

Bhatnagar S, Borkar VS (1998) A two-time-scale stochastic approximation scheme for simulation based parametric optimization. *Probab. Engrg. Inform. Sci.* 12(4):519–531.

Bhatnagar S, Prasad HL, Prashanth LA (2013) *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*. Lecture Notes in Control and Information Sciences Series, Vol. 434 (Springer, London).

Borkar VS (1997) Stochastic approximation with two time scales. *Systems Control Lett.* 29(5):291–294.

Borkar VS (2008) *Stochastic Approximation: A Dynamical Systems Viewpoint* (Cambridge University Press, Cambridge, UK).

Byrd RH, Chin GM, Nocedal J, Wu Y (2012) Sample size selection in optimization methods for machine learning. *Math. Programming* 134(1):127–155.

Chang K-H, Hong LJ, Wan H (2013) Stochastic trust-region response-surface method (STRONG)—A new response-surface framework for simulation optimization. *INFORMS J. Comput.* 25(2): 230–243.

Chen C-H, Lee LH (2010) *Stochastic Simulation Optimization: An Optimal Computing Budget Allocation*. System Engineering and Operations Research (World Scientific Publishing, Singapore).

Chen C-H, Lin J, Yucesan E, Chick SE (2000) Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynam. Systems: Theory Appl.* 10(3):251–270.

Chepuri K, de Mello TH (2005) Solving the vehicle routing problem with stochastic demands using the cross entropy method. *Ann. Oper. Res.* 134(1):153–181.

DeBoer PT, Kroese DP, Mannor S, Rubinstein RY (2005) A tutorial on the cross-entropy method. *Ann. Oper. Res.* 134(1):19–67.

de Mello TH, Shapiro A, Spearman ML (1999) Finding optimal material release times using simulation-based optimization. *Management Sci.* 45(1):86–102.

Dorigo M, Gambardella LM (1997) Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evolutionary Comput.* 1(3):53–66.

Fu MC (2002a) Optimization for simulation: Theory vs. practice. *INFORMS J. Comput.* 14(3):192–215.

Fu MC (2002b) Simulation optimization in the future: Evolution or revolution? *INFORMS J. Comput.* 14(3):226–227.

Fu MC, Chun C-H, Shi L (2008) Some topics for simulation optimization. Mason SJ, Hill RR, Möch L, Rose O, Jefferson T, Fowler JW, eds. *Proc. 2008 Winter Simulation Conf., Miami, FL*, 27–38.

Glover F (1990) Tabu search: A tutorial. *Interfaces* 20(4):74–94.

Goldberg DE (1989) *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley Longman Publishing Co., Inc., Boston).

Hashemi F, Ghosh S, Pasupathy R (2014) On adaptive sample rules for stochastic recursions. Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA, eds. *Proc. 2014 Winter Simulation Conf.* (IEEE, Piscataway, NJ), 3959–3970.

Hayashi S, Luo Z (2009) Spectrum management for interference-limited multiuser communication systems. *IEEE Trans. Inform. Theory* 55(3):1153–1175.

He D, Lee LH, Chen C-H, Fu MC, Wasserkrug S (2010) Simulation optimization using the cross-entropy method with optimal computing budget allocation. *ACM Trans. Modeling Comput. Simulation* 20(1):4:1–4:22.

Hu J, Fu MC, Marcus SI (2007) A model reference adaptive search method for global optimization. *Oper. Res.* 55(3):549–568.

Hu J, Fu MC, Marcus SI (2008) A model reference adaptive search method for stochastic global optimization. *Comm. Inform. Systems* 8(3):245–276.

Hu J, Wang Y, Zhou E, Fu MC, Marcus SI (2012) A survey of some model-based methods for global optimization. Hernández-Hernández D, Minjárez-Sosa A, eds. *Optimization, Control, and Applications of Stochastic Systems: In Honor of Onésimo Hernández-Lerma* (Birkhäuser, Basel, Switzerland), 157–179.

Huang D, Allen TT, Notz WI, Zeng N (2006) Global optimization of stochastic black-box systems via sequential kriging metamodels. *J. Global Optim.* 34(3):441–466.

Jacko P (2016) Resource capacity allocation to stochastic dynamic competitors: Knapsack problem for perishable items and index-knapsack heuristic. *Ann. Oper. Res.* 241(1–2):83–107.

Kiefer J, Wolfowitz J (1952) Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.* 23(3):462–466.

Kleywegt A, Shapiro A, de Mello TH (2002) The sample average approximation method for stochastic discrete optimization. *SIAM J. Optim.* 12(2):479–502.

Konda VR, Tsitsiklis JN (2004) Convergence rate of linear two-time-scale stochastic approximation. *Ann. Appl. Probab.* 14(2):796–819.

Kushner HJ, Yin GG (2004) *Stochastic Approximation Algorithms and Applications*, 2nd ed. (Springer-Verlag, New York).

Larranaga P, Etxeberria R, Lozano JA, Pena JM (1999) Optimization by learning and simulation of Bayesian and Gaussian networks. Technical Report EHU-KZAA-IK-4/99, Department of Computer Science and Artificial Intelligence, University of the Basque Country, Bilbao, Spain.

Molvalioglu O, Zabinsky ZB, Kohn W (2009) The interacting-particle algorithm with dynamic heating and cooling. *J. Global Optim.* 43(2–3):329–356.

Molvalioglu O, Zabinsky ZB, Kohn W (2010) Meta-control of an interacting-paricle algorithm. *Nonlinear Anal.: Hybrid Systems* 4(4):659–671.

Muhlenbein H, Paaß G (1996) From recombination of genes to the estimation of distributions: I. Binary parameters. Voigt HM, Ebeling W, Rechenberg I, Schwefel HP, eds. *Parallel Problem Solving from Nature-PPSN IV* (Springer-Verlag, Berlin), 178–187.

Ólafsson S (2006) Metaheuristics. Henderson SG, Nelson BL, eds. *Handbooks in Operations Research and Management Science: Simulation*, Vol. 13 (Elsevier, Amsterdam), 633–654.

Romeijn HE, Smith RL (1994) Simulated annealing for constrained global optimization. *J. Global Optim.* 5(2):101–126.

Rubinstein RY (2001) Combinatorial optimization, ants and rare events. Uryasev S, Pardalos PM, eds. *Stochastic Optimization: Algorithms and Applications* (Kluwer Academic Publishers, Dordrecht, Netherlands), 304–358.

Rubinstein RY, Shapiro A (1993) *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method* (John Wiley & Sons, New York).

Scott W, Frazier PI, Powell WB (2011) The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression. *SIAM J. Optim.* 21(3): 996–1026.

Shi L, Ólafsson S (2000) Nested partitions method for stochastic optimization. *Methodol. Comput. Appl. Probab.* 2(3):271–291.

Spall JC (1992) Multivariate stochastic approximation using simultaneous perturbation gradient approximation. *IEEE Trans. Automatic Control* 37(3):332–341.

Wolpert DH (2004) Finding bounded rational equilibria part I: Iterative focusing. Vincent T, ed. *Proc. Eleventh Internat. Sympos. Dynam. Games Appl., Tucson, AZ*.

Zabinsky ZB (2003) *Stochastic Adaptive Search for Global Optimization*. Nonconvex Optimization and Its Applications, Vol. 72 (Springer, New York).

Zhou E, Hu J (2012) Combining gradient-based optimization with stochastic search. Rose O, Uhrmacher AM, eds. *Proc. 2012 Winter Simulation Conf., Berlin*.

Zhou E, Hu J (2014) Gradient-based adaptive stochastic search for non-differentiable optimization. *IEEE Trans. Automatic Control* 59(7):1818–1832.

Zhou E, Bhatnagar S, Chen X (2014) Simulation optimization via gradient-based stochastic search. Buckley SJ, Miller JA, eds. *Proc. 2014 Winter Simulation Conf.* (IEEE, Piscataway, NJ), 3869–3879.

Zlochin M, Birattari M, Meuleau N, Dorigo M (2004) Model-based search for combinatorial optimization: A critical survey. *Ann. Oper. Res.* 131(1–4):373–395.

RIGHTSLINK
Copyright Clearance Center