Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, eds.

ROBUST SIMULATION OPTIMIZATION WITH STRATIFICATION

Pranav Jain
Sara Shashaani

Industrial and Systems Engineering
North Carolina State University
915 Partners Way
Raleigh, NC 27695, USA

Eunshin Byon

Industrial and Operations Engineering
University of Michigan
333 Hanes Hall
Ann Arbor, MI 48109, USA

ABSTRACT

Stratification has been widely used as a variance reduction technique when estimating a simulation output, whereby the input variates are generated following a stratified sampling rule from previously determined strata. This study shows that an adaptive sampling class of simulation optimization solvers called `ASTRO-DF` could become more robust with stratification, `S-ASTRO-DF`. For a simulation optimization algorithm, we discuss how to monitor the robustness in terms of bias and variance of the outcome and introduce several metrics to compute and compare the robustness of solvers. We find that while stratified sampling improves the algorithm's performance, its robustness is sensitive to the stratification structure. In particular, as the number of strata increases, the stratified sampling-based algorithms may become less effective.

1 INTRODUCTION

Simulation optimization (SO) algorithms are highly stochastic and heavily depend on the quality of the objective function estimator they use. We view the robustness of an SO algorithm as its outcome's consistent quality when run several times to solve a stochastic simulation. An earlier study (Nemirovski et al. 2009) interprets the robustness of the stochastic approximation (SA) method as its applicability to a wide range of problems (not just strongly convex ones) and shows that SA can be, in that sense, more robust than the sample average approximation (SAA) (Kim et al. 2015).

It is more common to study the robustness of an estimator instead of an SO algorithm, and even then, it is difficult to characterize robustness explicitly. Sanchez and Sanchez (2020) express robustness in terms of higher accuracy (lower variance) and better precision (smaller bias). They quantify robustness using a loss function to assess the risk of having bad estimates. This risk function has connections with the asymptotic efficiency of SAA estimators (Glynn and Whitt 1992), which compares the variability of the estimator and the computational cost required to estimate it. Inspired by these connections, we seek to compare the robustness of SO algorithms via minimizing the expected value of a similar loss function on the algorithm outcome. We analyze the robustness of a class of SO algorithms and validate why stratified sampling, which impacts the robustness of estimation, is effective for the robustness of SO.

1.1 Stratified Sampling

Stratified sampling is a well-known variance reduction technique wherein the input domain is divided into multiple disjoint sub-spaces. Stratified sampling reduces the variance by using a weighted average of conditional variances, which by the law of total variance is at most as large as the unconditional variance, i.e., $\mathbb{E}[\text{Var}(A|B)] \leq \text{Var}(A)$, for two random variables A and B. Like a stratified sampling estimator, a conditional Monte Carlo (MC) estimator also conditions a random variable on another; however, they are

fundamentally different. The variance saving for stratified sampling estimator is $\text{Var}(\mathbb{E}[A|B])$ and that for conditional MC estimator is $\mathbb{E}[\text{Var}(A|B)]$ (Ross 2013).

In a stochastic simulation context, let X be a random variable in $\mathbb{R}^d$ and let $g(\cdot)$ be a function such that $g : \mathbb{R}^d \to \mathbb{R}$. Consider m disjoint subsets/strata, denoted by $\mathcal{S}_i$, $i = 1, 2, \cdots, m$, covering the entire input domain, i.e., $\bigcup_{i=1}^m \mathcal{S}_i = \mathbb{R}^d$. Let $p_i = \mathbb{P}\{X \in \mathcal{S}_i\}$ be known, leading to $\mathbb{E}[g(X)] = \sum_{i=1}^m p_i \mathbb{E}[g(X^i)]$, where X^i follows P^i, the probability distribution of data within stratum i. In stratified sampling, n_i i.i.d. copies of X^i are drawn from P^i to estimate $\mu_i := \mathbb{E}[g(X^i)]$ with $\hat{\mu}_i = \sum_{j=1}^{n_i} g(X_j^i)/n_i$, for all $i = 1, 2, \cdots, m$. The stratified sampling estimator $\sum_{i=1}^n p_i \hat{\mu}_i$ is an unbiased estimator of $\mathbb{E}[g(X)]$. Its variance is estimated by $\sum_{i=1}^m p_i^2 \hat{\sigma}_i^2$, where $\hat{\sigma}_i^2$ is the *sample average variance* of n_i simulations of $g(X^i)$, i.e., $\hat{\sigma}_i^2 = (n_i)^{-2} \sum_{j=1}^{n_i} (g(X_j^i) - \hat{\mu}_i)^2$.

Obtaining an effective stratified sampling estimator requires answering two questions:

(i) How to split the input domain, i.e., $\{\mathcal{S}_i\}$?
(ii) How to determine the sample size of each stratum, i.e., $\{n_i\}$?

The first question demands a splitting structure such that within each stratum there are similar observable characteristic, such as the variability of the objective function. Consequently, we can model each stratum by a separate independent distribution. Additionally, stratified sampling is more effective when the variance across strata is large. Mulvey (1983) proposed a method for stratification via optimal cluster analysis. Though this method can give precise stratification, it is computationally expensive. Another splitting approach is using classification and regression trees (CART), which divides the data into subsets, aiming for a heterogeneous variance in each subset by minimizing the sum of squared errors (Breiman et al. 1984; Liu et al. 2022). More recently, Farias et al. (2020) proposed a splitting technique based on similarity functions for classification, which demands accurate modeling of the available dataset's true distribution.

The second question depends on $\hat{\sigma}_i^2$ and p_i. An inaccurate estimate of these two values can reduce the effectiveness of stratified sampling and possibly lead it to malfunction and produce worse estimates of the objective function. To determine the variance of the stratum, we need to sample some fixed number of points initially and then choose the optimal sample size. Chaddha et al. (1971) suggested graphical procedures to determine the optimal allocation of the total sample size. Huddleston et al. (1970) determined the optimal sample allocation of strata via convex programming. Bretthauer et al. (1999) used branch and bound methods to choose the optimal sample size of each stratum. Another common method is adaptive optimal allocation that minimizes the variance within each stratum (Etoré and Jourdain 2010; Kawai 2010). Recently, Glynn and Zheng (2021) proposed an optimal simulation budget allocation criteria for strata following the strong approximation theory and delta method.

All of these studies focus on the optimal sample size at a single point, however, optimization involves a sequence of points. While the estimator's accuracy is important, the other concern for optimization is the search efficiency. A naive approach entails using a fixed n_i for each stratum irrespective of the point in the search trajectory, which may not be efficient, although, Zhao and Zhang (2014) analytically showed improved convergence rate with a fixed sample size for a stratified SA. Adaptive sample size choices have proven more successful in SO (Shashaani et al. 2018; Bollapragada et al. 2018; Curtis and Scheinberg 2020). To the best of our knowledge, Espath et al. (2021), Liu et al. (2022), and Jain et al. (2021) are the only ones studying adaptive sample sizes for a stratified SO.

Additionally, the traditional stratified sampling considers that the stratification structure is known *a priori*. But in many data-driven cases, this may not be true. In particular, stratification structure can vary based on the progress made during optimization to provide optimal variance reduction and improvement in the search. Pettersson and Krumscheid (2021) proposed an adaptive stratification algorithm for optimization that greedily divides the input domain using hyperrectangles or simplices to maximize the reduction of estimated variance. This algorithm can cause an additional variance, which is not accounted for when looking at stratified sampling at a fixed point. During optimization, this additional variance may or may not make the stratified sampling estimator worse than the crude MC estimator.

1.2 Robustness of a Simulation Optimization Algorithm

Recall the unconstrained SO problem: $\min_\theta f(\theta)$, where $f : \mathbb{R}^d \to \mathbb{R}$ is unknown with noisy observations accessible from a stochastic simulation. Moreover, let $\theta^* = \arg\min_\theta f(\theta)$ exist. Suppose there is an SO algorithm at our disposal that generates a sequence of solutions iteratively using the estimated function value at the previously visited solutions. Given the random input $X|Z$ sampled from $P_{X|Z}$, which is the input model (data) for the realization Z of the SO algorithm, and the decision θ, each simulation run generates an output $F(\theta, X|Z)$ that is a random variable with $\mathbb{E}_{X|Z}[F(\theta, X|Z)] = f(\theta|Z)$. This is to emphasize that the input model of each realization of the algorithms is conditional on that realization. In a stochastic simulation setting, input data of a single run, henceforth *macroreplication*, of the SO algorithm is generated from random substreams that are allocated to the random stream used for that macroreplication (Eckman et al. 2022). In a data-driven setting, Z generates samples of the available data used for that macroreplication of the SO algorithm. We maintain Z in the notations as it is needed in the SO outcomes' analysis. For a realized macroreplication of the SO algorithm, i.e., $Z = z$, the problem becomes

$$\text{minimize}_\theta \; f(\theta|z) = \int F(\theta, X|z) dP_{X|z}, \tag{1}$$

generating a sequence of solutions $\{\theta_k(z),\ k = 1, 2, \cdots\}$. Hence, each macroreplication of the SO algorithm produces a distinct sequence of solutions. If the algorithm has convergence properties with high probability, then one can roughly expect $\mathbb{E}_Z[\theta_k(Z)] \to \theta^*$ as $k \to \infty$.

In this paper, we measure the SO algorithm's risk with the loss function $\ell_k^f(Z) := (f_k(Z) - f^*)^2$, where $f_k(Z) := f(\theta_k(Z))$ and $f^* = f(\theta^*)$ is the minimum objective function value we wish to attain. The loss function can also be defined with respect to the solution, i.e., $\ell_k^\theta(Z) := \|\theta_k(Z) - \theta^*\|^2$. The loss function defines the *algorithmic risk*, as $r_k^\theta := \mathbb{E}_Z[\ell_k^\theta(Z)]$, which is what we use to infer as the inverse of *robustness* in an SO algorithm, i.e., small r_k^θ implies high robustness. The original use of the loss function is for evaluating the efficiency of an estimator at a *fixed* point (Sanchez and Sanchez 2020), which also links to the asymptotic efficiency (Glynn and Whitt 1992) of an estimator. For an SO algorithm, the loss function has a value (risk) at different intermediate budget points. One can alternatively compare two SO algorithms' robustness in finite time given an SO budget (the allowed number of simulation runs, denoted by τ) by computing the area under the *risk curves*, i.e., $\int_0^\tau r^\theta(t)dt$, where $r^\theta(t)$ is a continuous counterpart of r_k^θ over time with t representing the SO budget. We will refer to the total number of simulation runs by the end of iteration k as $W_k(Z)$ in one macroreplication and let $K(Z) = \min\{k : W_k(Z) \leq \tau\}$ be the last iteration before termination.

1.3 Our Contribution

For SO, robust algorithms increase reliability. Notably, in parameter calibration or simulation validation, because of (i) the unknown distribution of the data and (ii) the stochasticity associated with sampling the data, robustness becomes elusive and essential. Since the intrinsic noise in the data is unavoidable, we focus on enhancing the robustness of the SO algorithms with adaptive stratified sampling to reduce the sampling error. With theoretical and numerical studies, we will analyze these effects on robustness compared to a benchmark asymptotically and in finite time. We focus on a class of stochastic trust-region-based algorithms called `ASTRO-DF` (Shashaani et al. 2018; Ha et al. 2021), which stands for Adaptive Sampling Trust-Region Optimization for Derivative-Free problems, well-suited for the data-driven studies later. In the following sections, we will lay out the framework for analyzing the stratified `ASTRO-DF`, or `S-ASTRO-DF`, and investigate the finite-time performance in a suite of experiments.

2 STRATIFIED SAMPLING WITHIN SIMULATION OPTIMIZATION

First, we present the notations and definitions for an SO algorithm that uses crude MC or `c-MC` vs. stratified MC or `s-MC`. Then, we discuss `ASTRO-DF` and `S-ASTRO-DF` as a specific example.

2.1 Notations and Definitions

As a convention in the paper, we use lowercase font for real numbers and uppercase font for random variables. SO algorithms with c−MC use SAA to map $Z \to f_k(n|Z) := \hat{\mathbb{E}}_{X|Z}[F(\theta_k(Z), X|Z)]$ at iteration k with SAA using n samples drawn from $P_{X|Z}$ and $\hat{\sigma}^2_k(n|Z) = \sum_{j=1}^{n}(F(\theta_k(Z), X_j|Z) - f_k(n|Z))^2/n^2$ as the estimated sample variance. Conditioning on Z determines solutions visited in a specific macroreplication of the algorithm. The s−MC SO algorithms uses m strata determined *a prioi* and independent of Z, $\boldsymbol{\mathcal{S}}$ as a vector of strata structures, and $\boldsymbol{n} = (n_1, n_2, \cdots, n_m)$ as a vector of sample sizes to map $Z \to \{f_k(n_i|Z, \mathcal{S}_i)\}_{i=1}^m$, where $f_k(n_i|Z, \mathcal{S}_i) := \hat{\mathbb{E}}_{X|Z,\mathcal{S}_i}[F(\theta_k(Z, \boldsymbol{\mathcal{S}}), X|Z, \mathcal{S}_i)]$ is estimated by SAA with n_i samples from $P_{X|Z,\mathcal{S}_i}$. Then define the objective function and variance estimator respectively as

$$f_k(\boldsymbol{n}|Z, \boldsymbol{\mathcal{S}}) := \sum_{i=1}^{m} p_i f_k(n_i|Z, \mathcal{S}_i) \text{ and } \hat{\sigma}^2_k(\boldsymbol{n}|Z, \boldsymbol{\mathcal{S}}) := \sum_{i=1}^{m} p_i^2 \hat{\sigma}^2_k(n_i|Z, \mathcal{S}_i).$$

Hence, c−MC algorithm results in the stochastic process $\{\theta_k(Z), f_k(n|Z), \hat{\sigma}_k(n|Z)\}$ while s−MC algorithm results in the stochastic process $\{\theta_k(Z), f_k(\boldsymbol{n}|Z, \boldsymbol{\mathcal{S}}), \hat{\sigma}_k(\boldsymbol{n}|Z, \boldsymbol{\mathcal{S}})\}$. For a fixed point such as θ_0 (the initial solution), we know $\text{Var}(f_0(n|Z)) \geq \text{Var}(f_0(\boldsymbol{n}|Z, \boldsymbol{\mathcal{S}}))$. But we would like to characterize $\{\theta_k(Z)\}$ in comparison with $\{\theta_k(Z, \boldsymbol{\mathcal{S}})\}$. To eliminate the optimization bias (Mak et al. 1999), one must evaluate the solutions at intermediate budget points with a separate SAA estimator, denoted by $\hat{f}(\cdot)$, that uses (common) random samples independent from those used for the optimization. Hence to compare the robustness of the two algorithms, one would compute the function estimate of their resulting solution sequences $\{\hat{f}(\theta_k(Z))\}$ and $\{\hat{f}(\theta_k(Z, \boldsymbol{\mathcal{S}}))\}$ and compare their corresponding loss functions.

2.2 ASTRO-DF Algorithm and Adjustments

ASTRO-DF is an almost sure globally convergent SO with two features: trust-region optimization which uses a local model within a moving neighborhood to find the next incumbent, and adaptive sample sizes, where sample sizes n and n_i become $N_k(Z)$ and $N_k(Z, \mathcal{S}_i)$, i.e., random and dependent on the point being visited. Trust-region methods approximate the true objective function by generating an easy-to-handle local model at each iteration. Optimizing the local model within the trust-region suggests a candidate for the incumbent solution at the next iteration. Trust-region is typically a closed ball around the current incumbent solution denoted by $\mathcal{B}_k(Z) = \{\theta : \|\theta - \theta_k(Z)\|_2 \leq \Delta_k(Z)\}$, where $\Delta_k(Z)$ is the trust-region radius. With a surrogate model $M_k(\theta|Z)$ generated to approximate the true objective function at iteration k, a candidate solution $\tilde{\theta}_{k+1}(Z)$ is identified as $\arg\min_{\theta \in \mathcal{B}_k(Z)} M_k(\theta|Z)$. If sufficient reduction in the estimated objective function value is achieved at the candidate point, the algorithm accepts it as the new incumbent and expands the trust-region radius. Otherwise, it starts the next iteration at the previous incumbent with a shrunk trust-region to generate a more accurate model. The fact that $\Delta_k(Z) \to 0$ as $k \to \infty$ almost surely and $\|\nabla M_k(\theta|Z) - \nabla f(\theta)\|/\Delta_k(Z) = \mathcal{O}_p(1)$ for all $\theta \in \mathcal{B}_k(Z)$ guarantees the convergence of the algorithm with iteration complexity of $\mathcal{O}_p(\epsilon^{-2})$ for an ϵ-stationary solution (Ha et al. 2021). Adaptive sampling determines $N_k(Z)$ by maintaining a balance between the stochastic error and optimality gap, i.e., less samples when far away and more near θ^*. In ASTRO-DF, the optimality gap is proxied by the square of the trust-region radius:

$$N_k(Z) = \min\left\{n \geq \lambda_k : \sqrt{\lambda_k}\hat{\sigma}_k(n|Z) \leq \Delta_k^2(Z)\right\}, \tag{2}$$

where κ is a positive constant and λ_k is the deterministic sample size lower bound satisfying $\lambda_k = \mathcal{O}(k^{1+\epsilon})$. We denote the function estimate for the candidate solution by $\tilde{f}_{k+1}(\tilde{N}_{k+1}(Z)|Z)$, where $\tilde{N}_{k+1}(Z)$ is its sample size. The stochastic error in ASTRO-DF can further be reduced by integrating stratified sampling with adaptive sampling. Intuitively, stratified sampling when combined with adaptive sampling can generate a more accurate surrogate model, resulting in a better search trajectory. Given a splitting

structure $\boldsymbol{\mathcal{S}}$, we need to allocate budget to each stratum for each visited solution. In S-ASTRO-DF (stratified ASTRO-DF) we use the adaptive sampling criteria (2), yet replace the standard error with $\hat{\sigma}_k(\boldsymbol{N}_k(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}})$ following the definition in Section 2.1. We define the lower bound sequence for stratum i as $\lambda_k(Z,\mathcal{S}_i) = \lceil n_0 + w_k(Z,\mathcal{S}_i) \times \left(\max\{n_0^{\text{all}}, \lambda_k\} - mn_0\right) - 0.5\rceil$, where

$$w_k(Z,\mathcal{S}_i) = \frac{p_i\hat{\sigma}_{k-1}(N_{k-1}(Z,\mathcal{S}_i)|Z,\mathcal{S}_i)}{\sum_{j=1}^{m} p_j\hat{\sigma}_{k-1}(N_{k-1}(Z,\mathcal{S}_j)|Z,\mathcal{S}_j)} \tag{3}$$

is the weight of stratum i during iteration k using the previous incumbent, n_0 is the minimum sample size of each stratum and n_0^{all} is the minimum number of samples from all strata. We then check the adaptive sampling criteria in (2). If the inequality in (2) is not satisfied, we keep on sampling one point at a time. To choose which stratum to add a point to, we use a selective randomized method (Tong 2006; Zhang et al. 2022), with a customized probability for each stratum. For optimal allocation ideally the strata weights $w_k(Z,\mathcal{S}_i)$ should be equal to $q_k(Z,\mathcal{S}_i) := N_k(Z,\mathcal{S}_i)/\sum_{j=1}^{m} N_k(Z,\mathcal{S}_j)$, the proportion of samples that belong to stratum i. Based on this we randomly sample from stratum i with probability:

$$\pi_k(Z,\mathcal{S}_i) = \Pr\{\text{sampling from stratum } i \text{ at iteration } k\} = \frac{w_k(Z,\mathcal{S}_i)\mathbb{I}\{w_k(Z,\mathcal{S}_i) > q_k(Z,\mathcal{S}_i)\}}{\sum_{j=1}^{m} w_k(Z,\mathcal{S}_j)\mathbb{I}\{w_k(Z,\mathcal{S}_j) > q_k(Z,\mathcal{S}_j)\}}. \tag{4}$$

The probability mass function $(\pi_k(Z,\mathcal{S}_i),\ i = 1,2,\cdots,m)$ evolves to guarantee that the strata with sub-optimal allocation are more likely to be selected. Algorithm 1 lists the steps of S-ASTRO-DF. We skip the model construction details of ASTRO-DF; see (Jain et al. 2021) for more.

Algorithm 1 S-ASTRO-DF Algorithm

1: **Parameters:** Stratification structure and corresponding probabilities $\{(\mathcal{S}_i, p_i),\ i = 1,2,\cdots,m\}$, initial solution θ_0 and radius Δ_0, number of strata m, maximum budget τ, minimum sample size $n_0 \geq 2$ and $n_0^{\text{all}} \geq n_0 m$, deterministic sequence λ_k, success threshold $\eta > 0$, and post-processing estimator $\hat{f}(\cdot)$.
2: **initialization:** Set $k = 0$ and $W_0(Z,\boldsymbol{\mathcal{S}}) = 0$.
3: **while** $W_k(Z,\boldsymbol{\mathcal{S}}) < \tau$ **do**
4: Construct $M_k(\cdot|Z,\boldsymbol{\mathcal{S}})$ using $f_k(\boldsymbol{N}_k(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}})$ and update $W_k(Z,\boldsymbol{\mathcal{S}})$.
5: Minimize $M_k(\cdot|Z,\boldsymbol{\mathcal{S}})$ within $\mathcal{B}_k(Z,\boldsymbol{\mathcal{S}})$ to obtain a candidate solution $\tilde{\theta}_{k+1}(Z,\boldsymbol{\mathcal{S}})$.
6: Estimate $\tilde{f}_{k+1}(\tilde{\boldsymbol{N}}_{k+1}(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}})$ following the adaptive stratified sampling and update $W_k(Z,\boldsymbol{\mathcal{S}})$.
7: **if** the success ratio $\rho_k(Z,\boldsymbol{\mathcal{S}}) := \frac{f_k(\boldsymbol{N}_k(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}}) - \tilde{f}_{k+1}(\tilde{\boldsymbol{N}}_{k+1}(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}})}{M_k(\theta_k(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}}) - M_k(\tilde{\theta}_{k+1}(Z,\boldsymbol{\mathcal{S}})|Z,\boldsymbol{\mathcal{S}})} > \eta$ **then**
8: Set $\theta_{k+1}(Z,\boldsymbol{\mathcal{S}}) = \tilde{\theta}_{k+1}(Z,\boldsymbol{\mathcal{S}})$ and $\Delta_{k+1}(Z,\boldsymbol{\mathcal{S}}) > \Delta_k(Z,\boldsymbol{\mathcal{S}})$.
9: **else**
10: Set $\theta_{k+1}(Z,\boldsymbol{\mathcal{S}}) = \theta_k(Z,\boldsymbol{\mathcal{S}})$ and $\Delta_{k+1}(Z,\boldsymbol{\mathcal{S}}) < \Delta_k(Z,\boldsymbol{\mathcal{S}})$.
11: **end if**
12: Set $k = k+1$, compute $w_k(Z,\mathcal{S}_i)$ for all $i = 1,2,\cdots,m$ following (3), and go to step 4.
13: **end while**
14: **output:** $\{\theta_k(Z,\boldsymbol{\mathcal{S}}), W_k(Z,\boldsymbol{\mathcal{S}})\}$ for all $k = 1,2,\cdots,K(Z,\boldsymbol{\mathcal{S}})$.

3 ROBUSTNESS ANALYSIS

In this section, we explore the effect of stratified sampling on algorithmic robustness by comparing the algorithmic risk $r_k^{\theta,\text{C-MC}} = \mathbb{E}_Z[\|\theta_k(Z) - \theta^*\|^2]$ for ASTRO-DF and $r_k^{\theta,\text{S-MC}} = \mathbb{E}_Z[\|\theta_k(Z,\boldsymbol{\mathcal{S}}) - \theta^*\|^2]$ for S-ASTRO-DF. The algorithmic risk captures both the accuracy of the solutions and their precision. By comparing the algorithm's outcome with the correct optimum (θ^*), we look at how the final solution varies for different macroreplications, signified by Z. The risk values $r_k^{\theta,\text{C-MC}}$ and $r_k^{\theta,\text{S-MC}}$ measure the quality of

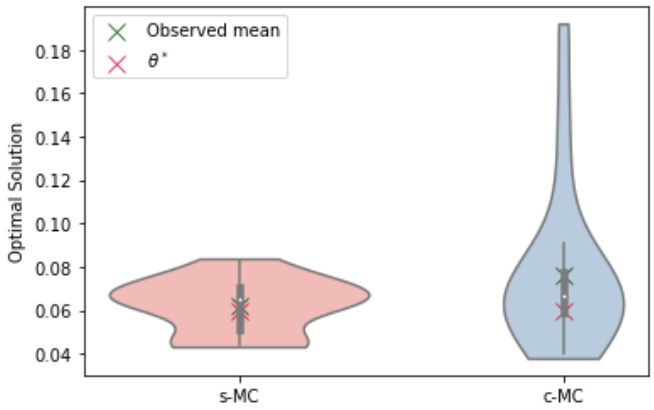


Figure 1: Illustration of how algorithmic risk at termination, i.e., $r^{\theta}(\tau)$ monitors robustness: final solution of `s-MC` is more precise with (lower variance) and more accurate (mean closer to θ^*) than `c-MC`.

the solution. A robust algorithm demonstrates consistency and sharpness near θ^*. Inherent stochasticity in an SO algorithm may or may not amplify algorithmic risk as the randomness within the algorithm could increase the variability of the final solutions across macroreplications. The proposed risk measure thus considers the algorithm's sensitivity to varying input and its intrinsic stochasticity. Figure 1 shows the distribution of the final output of the two algorithms on one problem using common random numbers (CRN) for sharpness of the comparison; details to follow in Section 4. For `s-MC`, the observed mean is close to θ^*, and the solution does not vary significantly across macroreplications. For `c-MC`, the observed mean is much higher than θ^*, and the outcome changes drastically for different macroreplications. The final solution of `s-MC` is more precise and accurate, and hence, more robust.

3.1 Assumptions

In our analysis, we assume the following:

(1) The SAA estimation error for a fixed point, i.e., given the history, has mean zero and uniformly bounded variance almost surely, i.e., there exist $\sigma < \infty$ and $\sigma_i < \infty \ \forall i = 1, 2, \cdots, n$ such that:

$$\mathbb{P}\{Z : \mathbb{E}[f_k(n|Z)|\mathcal{F}_k(Z)] = f_k(Z)\} = \mathbb{P}\{Z : \mathbb{E}[f_k(n|Z, \mathcal{S}_i)|\mathcal{F}_k(Z, \boldsymbol{\mathcal{S}})] = f_k(Z, \mathcal{S}_i),\ \forall i\} = 1,$$
$$\mathbb{P}\{Z : \text{Var}(F(\theta_k(Z), X|Z)|\mathcal{F}_k(Z)) \leq \sigma^2\} = \mathbb{P}\{Z : \text{Var}(F(\theta_k(Z, \boldsymbol{\mathcal{S}}), X|Z, \mathcal{S}_i)|\mathcal{F}_k(Z, \boldsymbol{\mathcal{S}})) \leq \sigma_i^2,\ \forall i\} = 1,$$

where $\mathcal{F}_k(Z)$ and $\mathcal{F}_k(Z, \boldsymbol{\mathcal{S}})$ are the corresponding SO filtrations up to iteration k,

(2) Both `ASTRO-DF` and `S-ASTRO-DF` converge in expectation:

$$\lim_{k\to\infty} \mathbb{E}_Z[\theta_k(Z)] = \lim_{k\to\infty} \mathbb{E}_Z[\theta_k(Z, \boldsymbol{\mathcal{S}})] = \theta^*.$$

(3) There exist the probability measures $\mathcal{Z}$ and $\mathcal{Z}^{\boldsymbol{\mathcal{S}}}$ such that $\theta_k(Z) \Rightarrow \mathcal{Z}$ and $\theta_k(Z, \boldsymbol{\mathcal{S}}) \Rightarrow \mathcal{Z}^{\boldsymbol{\mathcal{S}}}$ as $k \to \infty$, where $\Rightarrow$ signifies weak convergence.

3.2 Main Result

In this section we state and sketch the proof of the analytical result for `S-ASTRO-DF` in comparison with `ASTRO-DF`. We will use the notations $G_k(Z) = \nabla M_k(\theta_k(Z)|Z)$ and $G_k(Z, \boldsymbol{\mathcal{S}}) = \nabla M_k(\theta_k(Z, \boldsymbol{\mathcal{S}})|Z, \boldsymbol{\mathcal{S}})$ for the model gradient and $H_k(Z) = \nabla^2 M_k(\theta_k(Z)|Z)$ and $H_k(Z, \boldsymbol{\mathcal{S}}) = \nabla^2 M_k(\theta_k(Z, \boldsymbol{\mathcal{S}})|Z, \boldsymbol{\mathcal{S}})$ for the model Hessian in `ASTRO-DF` and `S-ASTRO-DF`, respectively.

Theorem 1 If the assumptions listed above hold, then $r^{\theta,\text{S-MC}}(\tau) \leq r^{\theta,\text{C-MC}}(\tau)$, i.e., the algorithmic risk of `S-ASTRO-DF` at termination with the total budget of $\tau < \infty$ is smaller than that of `ASTRO-DF`.

We only provide the proof sketch here. Let Z be given, representing one run of ASTRO-DF. We note that at termination, the number of iterations for S-ASTRO-DF is $K(Z, \boldsymbol{\mathcal{S}})$ while that for ASTRO-DF is $K(Z)$. We can write $\|\theta_k(Z) - \theta_{k-1}(Z)\|^2 \leq \Delta_{k-1}^2(Z)$ and hence:

$$\begin{aligned}\mathbb{E}_Z[\|\theta_k(Z) - \theta^*\|^2] &= \mathbb{E}_Z\left[\sum_{k'=1}^{k} \|\theta_{k'}(Z) - \theta_{k'-1}(Z)\|^2\right] + \|\theta_0 - \theta^*\|^2 \\ &\leq \mathbb{E}_Z\left[\sum_{k'=1}^{k} \Delta_{k'-1}^2(Z)\right] + \|\theta_0 - \theta^*\|^2.\end{aligned} \tag{5}$$

Since $\Delta_k^2(Z) \leq \Delta_0 \gamma^k$ (γ is the expansion coefficient of the ASTRO-DF algorithm), for a random iteration $K(Z)$, the first term on right hand side of (5) can be written as $c_1 + c_2 \mathbb{E}_Z[(\gamma^2)^{K(Z)}]$, where $c_1 = \Delta_0^2(1-\gamma)^{-1}$ and $c_2 = \Delta_0^2 \gamma^{-1}(1-\gamma)^{-1}$. It then remains to be shown that the moment-generating function of $K(Z)$ is larger than that of $K(Z, \boldsymbol{\mathcal{S}})$ implying that the average number of iterations for S-ASTRO-DF is less than the average number of iterations for ASTRO-DF. For intuition, the algorithm with a smaller number of iterations is more likely to have less successful iterations indicating that the algorithm gets to a better point faster, hence showing better convergence. We show that S-ASTRO-DF will have fewer iterations before termination by showing that

$$\Pr\left\{\Delta_{K(Z,\boldsymbol{\mathcal{S}})}(Z, \boldsymbol{\mathcal{S}}) < \Delta_{K(Z)}(Z)\right\} > 1 - \delta,$$

for some $\delta > 0$, then we can conclude that $\Delta_k(Z) = \mathcal{O}(\|\nabla M_k(Z)\|)$ will imply that the model gradient norm with stratified version of the algorithm is likely to be smaller than that of the crude version.

4 CASE STUDY: PARAMETER CALIBRATION IN WIND POWER SYSTEMS

This section numerically compares the two estimators in a case study that involves parameter calibration in wind power systems. Computer models are often used to simulate a physical process because of their cost-effectiveness compared to full-scale physical experiments. Computer models make certain assumptions linked with parameters to replicate the physical process. These parameters cannot generally be observed or measured in the real world. This calls for parameter calibration, wherein we use the observational data to determine the parameter values that minimize model discrepancy (Kennedy and O'Hagan 2001).

This case study is for wind power systems. The wake effect parameter significantly affects the power generated in multi-turbine wind farms. Wake represents the deficit in wind speed at a downstream turbine because of one or more turbines in the upstream direction. Engineering wake models often characterize wake effects because of their computational simplicity. Jensen wake model is one such model, which for a given set of input conditions, gives the effective wind speed at each turbine in the wind farm (Jensen 1983). Power generated at a turbine is then estimated using the effective wind speed in conjunction with the power curve (Liu et al. 2022). The power curve is a characteristic of a wind turbine that relates the effective wind speed to the power generated. The effective wind speed calculations in the Jensen wake model are sensitive to a parameter called the wake decay coefficient. Though standard values of 0.075 for onshore wind farms and 0.04 for offshore wind farms are suggested for the wake decay coefficient, some recent work has shown that these values are not necessarily optimal (You et al. 2017). We thus aim to calibrate the wake decay coefficient for the given data set collected from an offshore wind farm by minimizing the estimation error of the computer model. For more details see (Jain et al. 2021).

4.1 Implementation and Discussion

We first compare S-ASTRO-DF with varying numbers of strata against ASTRO-DF. To explore the effect of the additional uncertainty of stratification in the algorithm, we also compare static stratification,

S-ASTRO-DF$(s,\cdot)$, for all macroreplications against dynamic stratification, S-ASTRO-DF$(d,\cdot)$ which changes at each macroreplication. Thus, S-ASTRO-DF$(d,\cdot)$ algorithms have additional inherent stochasticity, whereas S-ASTRO-DF$(s,\cdot)$ do not. The second argument in both is the number of strata. We compare these algorithms by varying different input conditions: the initial solution ($\theta_0 \in \{0.02, 0.15\}$), the initial trust-region radius ($\Delta_0 = \{0.08, 0.04\}$), and the shrinkage factor ($\gamma_2 \in \{0.64, 0.8\}$). The shrinkage factor shrinks the trust-region radius upon rejection of the candidate solution. We report the performance of the algorithms for different input conditions via six outputs: the final value, the algorithmic risk, and the overall risk or the area under the risk curve, all for the objective function value and the solution.

We run 20 independent macroreplications for each algorithm with CRN to estimate and compare the robustness. Each macroreplication starts at the same initial point (θ_0) and has the total budget of $\tau = 10,000$. For each experiment, Z is the random number generator seed that is used to divide the data randomly into a training set of size 70% for optimization and a test set of size 30% for post-processing. X on the other hand is a set of points sampled from these training and test sets. The intermediate solutions reported during optimization at various budget points are re-evaluated using the entire test set. In S-ASTRO-DF$(s,\cdot)$ algorithms, the stratification structure is determined a priori using the entire data, whereas S-ASTRO-DF$(d,\cdot)$ determines the stratification structure using the training set at the start of each macroreplication. We have data of size 11,656 collected from 30+ turbines for the wind case study, which is larger than the total budget considered in this set of experiments. The data consists of the weather conditions (wind speed, wind direction, turbulence intensity) and power generated at all the turbines for the corresponding weather conditions and is recorded at the met mast when the met mast is not under wake, and the wind direction ranges between 165° and 315°. We use CART on turbulence intensity as the stratification variable for splitting the data into multiple strata. The optimal value of the wake decay coefficient (θ^*) is approximately 0.06 (identified from a separate experiment with larger budget), which we use to measure the algorithmic risk and therefore, the robustness using just a portion of the data.

4.2 Results and Discussion

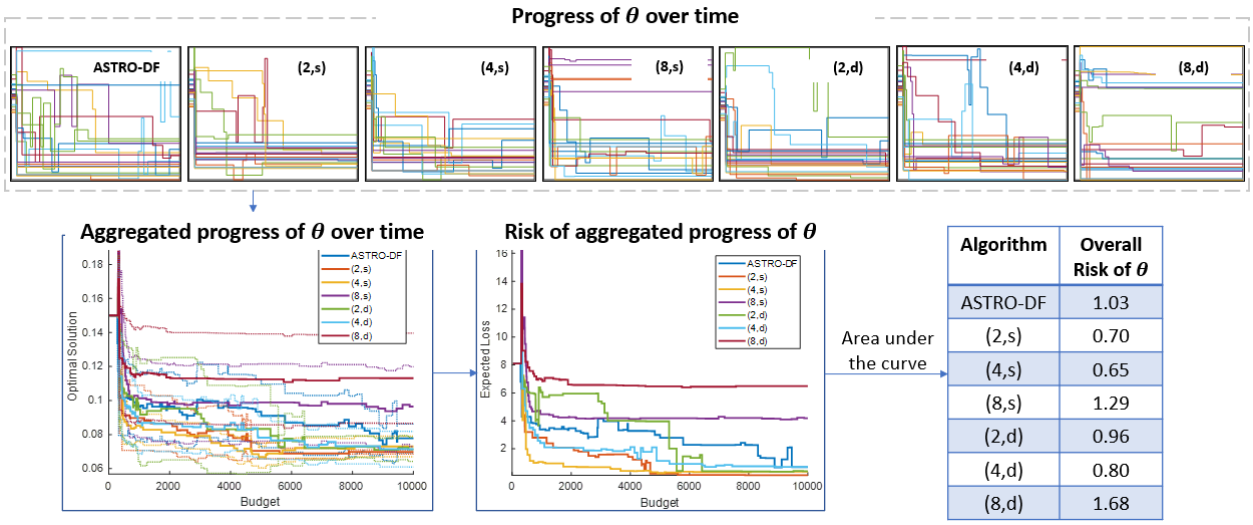
Since the initial point is identical for all the algorithms and we use CRN, the loss trajectories start at the same point and evolve according to how well each algorithm performs. Figure 2 shows the evolution of the new algorithmic risk value across time, following the concept of progress curves (Eckman et al. 2022) and using the MATLAB version of SimOpt library (Eckman et al. 2020). We can gather the progress of each algorithm over time for each macroreplication (plots at the top row). These progress plots can be summarized in an aggregate progress curve to easily overlay all algorithms in one plot (bottom left plot). We compare the algorithmic risk over time (bottom middle plot), which also further summarizes the progress of each solver to one curve. We can summarize one curve even further into a single number by computing the area under the risk trajectories (bottom right table).

Table 1 summarizes the results of the algorithms for each experiment. The third and fourth columns report the mean and 95% confidence interval (CI) of the final objective function value and the final solution at termination (after the total budget τ is exhausted), respectively, i.e., $\theta(\tau)$ and $\hat{f}(\theta(\tau))$. The fifth and the seventh column summarize the algorithmic risk at termination in terms of the objective function value, $r^f(\tau)$, and the solution value, $r^\theta(\tau)$, respectively. Further, instead of looking at the algorithmic risk only at termination, it is important to see how it changes during optimization for each experiment. The sixth and eighth columns summarize the overall algorithmic risk, estimated as the area under the loss curve.

We observe that for $\theta_0 = 0.15$, which is an initial solution in a hard – flat – region, the algorithms with two strata perform the best. The first four experiments show that as we increase the number of strata, the algorithm's performance tends to degrade. In fact, the algorithms with eight strata perform worse than the original ASTRO-DF in all four experiments. We speculate that more strata for flat regions leads to an unnecessarily large sample size, which causes overfitting and more bias, resulting in less robust solutions. When we change the initial point to $\theta_0 = 0.02$, which is in an easy – steep – region, the algorithms with

Table 1: Summary of the results for all the experiments evaluated over 20 independent macroreplications with * marking statistically significant difference with 90% significance level when compared to ASTRO-DF and bold font signifying the best values in each column (within an experiment with multiple algorithms).

Experiments		Algorithms	$\hat{f}(\theta(\tau))$ Mean-CI ($\times 10^4$)	$\theta(\tau)$ Mean-CI ($\times 10^{-2}$)	$r^f(\tau)$ Final ($\times 10^9$)	$\int_{t=0}^{\tau} r^f(t)dt$ Overall ($\times 10^{13}$)	$r^\theta(\tau)$ Final ($\times 10^{-4}$)	$\int_{t=0}^{\tau} r^\theta(t)dt$ Overall ($\times 10^2$)
$\theta_0 = 0.15$	Exp 1: $\Delta_0 = 0.08$ $\gamma_2 = 0.64$	ASTRO-DF	3.52 ± 0.09	7.8 ± 0.8	1.25	1.33	6.86	0.28
		(2,s)	**3.46 ± 0.04***	**6.9 ± 0.1***	**1.20**	**1.25**	**1.02**	0.12
		(4,s)	3.50 ± 0.07	7.3 ± 0.6	1.23	**1.25**	3.58	**0.08**
		(8,s)	3.65 ± 0.16	9.6 ± 2.3	1.35	1.37	41.82	0.45
		(2,d)	3.48 ± 0.08	7.1 ± 0.7	1.22	1.28	3.88	0.31
		(4,d)	3.48 ± 0.10	7.1 ± 1.0	1.22	1.27	7.02	0.17
		(8,d)	3.76 ± 0.17	11.3 ± 2.6	1.43	1.43	64.80	0.67
	Exp 2: $\Delta_0 = 0.08$ $\gamma_2 = 0.8$	ASTRO-DF	3.54 ± 0.10	7.8 ± 1.0	1.26	1.28	8.29	0.17
		(2,s)	3.48 ± 0.06*	7.1 ± 0.4*	1.21	**1.23**	2.08	**0.06**
		(4,s)	3.47 ± 0.05*	6.7 ± 0.5*	1.20	1.25	1.94	0.07
		(8,s)	3.56 ± 0.12	8.3 ± 1.8	1.28	1.30	23.11	0.28
		(2,d)	**3.44 ± 0.05***	**6.2 ± 0.7***	**1.19**	1.24	2.65	0.13
		(4,d)	3.46 ± 0.05*	6.9 ± 0.4*	1.20	**1.23**	**1.61**	0.08
		(8,d)	3.57 ± 1.46	8.7 ± 2.4	1.28	1.33	38.23	0.44
	Exp 3: $\Delta_0 = 0.04$ $\gamma_2 = 0.8$	ASTRO-DF	3.48 ± 0.07	7.4 ± 0.7	1.21	1.27	3.65	0.15
		(2,s)	**3.42 ± 0.05***	6.4 ± 0.4*	**1.17**	**1.24**	**1.16**	0.10
		(4,s)	3.47 ± 0.07	6.9 ± 0.6	1.21	1.25	3.06	**0.09**
		(8,s)	3.57 ± 0.12	8.3 ± 1.6	1.28	1.33	18.29	0.27
		(2,d)	3.50 ± 0.11	7.2 ± 1.0	1.23	1.28	7.02	0.19
		(4,d)	3.45 ± 0.06	**6.3 ± 0.6***	1.20	1.25	1.96	0.10
		(8,d)	3.53 ± 0.13	7.9 ± 1.7	1.25	1.30	18.71	0.24
	Exp 4: $\Delta_0 = 0.04$ $\gamma_2 = 0.64$	ASTRO-DF	3.53 ± 0.13	7.6 ± 1.6	1.26	1.31	16.38	0.27
		(2,s)	**3.43 ± 0.05***	**6.2 ± 0.5***	**1.18**	**1.23**	**1.16**	**0.11**
		(4,s)	3.50 ± 0.10	7.3 ± 1.2	1.23	1.27	3.06	0.14
		(8,s)	3.71 ± 0.15	10.0 ± 2.1	1.38	1.43	39.41	0.48
		(2,d)	3.44 ± 0.07	6.4 ± 0.8	1.19	1.27	7.02	0.21
		(4,d)	3.50 ± 0.10	7.1 ± 1.0	1.23	1.30	6.64	0.19
		(8,d)	3.71 ± 0.17	10.1 ± 2.2	1.39	1.42	42.20	0.46
$\theta_0 = 0.02$	Exp 5: $\Delta_0 = 0.08$ $\gamma_2 = 0.64$	ASTRO-DF	3.46 ± 0.08	**6.5 ± 0.8**	1.20	1.28	3.77	0.35
		(2,s)	3.48 ± 0.06	7.0 ± 0.7	1.21	1.23	3.52	0.04
		(4,s)	**3.45 ± 0.05**	6.6 ± 0.6	1.20	1.24	2.60	0.04
		(8,s)	3.49 ± 0.07	7.1 ± 0.7	1.22	**1.22**	3.89	0.03
		(2,d)	3.51 ± 0.08	7.0 ± 0.9	1.24	1.26	5.13	0.07
		(4,d)	3.45 ± 0.09	6.6 ± 0.9	**1.19**	**1.22**	4.61	0.03
		(8,d)	3.46 ± 0.06	6.8 ± 0.6	1.20	**1.22**	**2.38**	**0.02**
	Exp 6: $\Delta_0 = 0.08$ $\gamma_2 = 0.8$	ASTRO-DF	3.50 ± 0.08	7.3 ± 0.8	1.23	1.30	4.70	0.40
		(2,s)	3.46 ± 0.06	6.8 ± 0.6	1.20	**1.22**	2.64	**0.03**
		(4,s)	3.44 ± 0.06*	6.6 ± 0.5*	1.19	1.24	1.61	0.05
		(8,s)	3.47 ± 0.05	7.0 ± 0.5	1.21	1.23	2.10	**0.03**
		(2,d)	3.44 ± 0.05	6.6 ± 0.6	1.19	1.24	2.21	0.07
		(4,d)	**3.40 ± 0.06***	**5.9 ± 0.5***	**1.16**	**1.22**	**1.44**	0.04
		(8,d)	3.46 ± 0.06	6.7 ± 0.6	1.20	**1.22**	2.19	**0.03**
	Exp 7: $\Delta_0 = 0.04$ $\gamma_2 = 0.8$	ASTRO-DF	3.48 ± 0.06	6.7 ± 0.8	1.21	1.27	3.85	0.14
		(2,s)	3.43 ± 0.06	6.5 ± 0.5	1.18	1.21	1.63	**0.02**
		(4,s)	**3.41 ± 0.05***	6.2 ± 0.4	**1.16**	1.23	**1.07**	0.05
		(8,s)	3.46 ± 0.09	6.8 ± 0.8	1.20	1.22	4.19	0.04
		(2,d)	3.46 ± 0.07	6.8 ± 0.6	1.20	1.26	2.40	0.08
		(4,d)	3.45 ± 0.08	6.6 ± 0.6	1.19	1.21	2.48	0.03
		(8,d)	**3.41 ± 0.07***	**6.1 ± 0.5**	**1.16**	**1.19**	1.28	**0.02**
	Exp 8: $\Delta_0 = 0.04$ $\gamma_2 = 0.64$	ASTRO-DF	3.49 ± 0.10	7.1 ± 1.0	1.22	1.29	6.86	0.18
		(2,s)	3.44 ± 0.05	6.5 ± 0.4	**1.18**	1.21	**1.32**	0.03
		(4,s)	3.45 ± 0.07	6.8 ± 0.6	1.20	1.22	2.37	0.04
		(8,s)	3.48 ± 0.07	7.0 ± 0.8	1.21	1.23	4.21	0.04
		(2,d)	3.45 ± 0.08	6.5 ± 0.7	1.19	1.23	2,91	0.07
		(4,d)	**3.43 ± 0.06***	6.4 ± 0.5*	**1.18**	**1.20**	1.48	**0.02**
		(8,d)	**3.43 ± 0.06***	**6.1 ± 0.6***	**1.18**	**1.20**	1.82	**0.02**



Algorithm	Overall Risk of θ
ASTRO-DF	1.03
(2,s)	0.70
(4,s)	0.65
(8,s)	1.29
(2,d)	0.96
(4,d)	0.80
(8,d)	1.68

Figure 2: Illustration of results from 20 runs of each version, summarized for comparison.

stratified sampling estimator consistently outperform the original ASTRO-DF, and the performance tends to get better as we increase the number of strata.

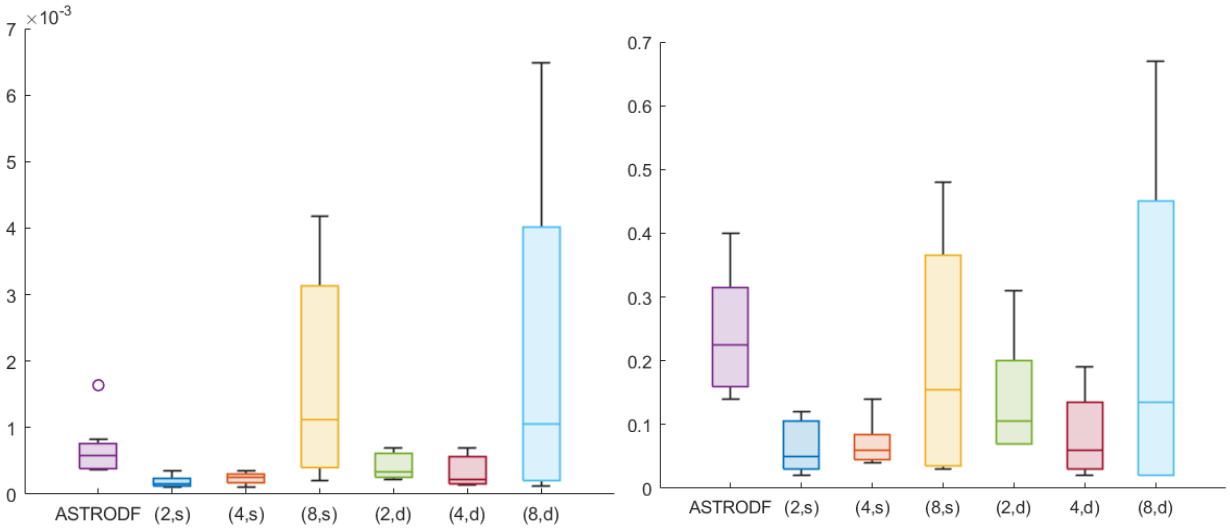
Next, we compare the effect of static vs dynamic stratification (splitting as part of the SO). For experiments 1-4, the static stratification generally has better performance, and vice versa is true for experiments 5-8. This highlights the importance of selecting a good stratification structure. Even for the same problem, the optimal stratification structure can vary based on where we are in the search space. In the experiments that start off from a steep region, a small step size can significantly reduce the objective function value and lead the algorithm to a better point. As a result, all the algorithms perform well for experiments 5-8, but some of the S-ASTRO-DF$(d,\cdot)$ algorithms slightly outperform the rest with a lower risk value. S-ASTRO-DF$(d,\cdot)$ algorithms have some inherent stochasiticity due to using different stratification structures for each macroreplication. But this inherent stochasiticity does not necessarily have a negative effect on the risk. In the experiments starting from a region with a gentle slope, better estimates are needed to make progress, which could be contributing to why algorithms with dynamic stratification do not perform as well for experiments 1-4.

The dynamic stratification employed is a preliminary study that involves adapting the stratification structure based on the search trajectory for a better understanding of the local distribution at the incumbent. However, the changing structure throughout the search will further complicate the complexity analysis.

By changing the other parameters in the experiments, we also explore the sensitivity of different stratification settings. Figure 3 plots the distribution of the algorithmic risk in terms of solution (final and overall) for the eight experiments. It shows that the median of the overall risk for algorithms with stratified sampling is lower than that without stratification. The interquartile range for algorithms with eight strata is large compared to the rest, mainly due to their poor performance in the first four experiments. Even though S-ASTRO-DF$(4,s)$ does not give the best result for a particular experiment, it has the most consistent performance for all the scenarios considered in this study.

5 CONCLUSION

This paper explores using stratification for the derivative-free trust-region-based SO with adaptive sampling. The analytical and numerical results reveal that while stratified sampling enhances the performance of the algorithms, care is needed in the stratification structure to enhance robustness. Though it is expected that increasing the number of strata would give more precise estimates and further reduce the variance in theory, it does not necessarily help in practice. Here, we only consider static and dynamic stratification with fixed



(a) Boxplots for algorithmic risk (in θ) at termination.

(b) Boxplots for overall risk (in θ).

Figure 3: Comparison of the algorithmic risk in terms of the final solution under different settings.

structures. An adaptive stratification structure during each iteration is an open question, an interesting aspect of which involves determining the appropriate number of strata in each iteration. The stratification variable also affects the performance of a stratified sampling-based SO. We use turbulence intensity for the wind power case study to stratify the input domain. In the future, we seek guidelines that identify influential inputs to be used in the stratification in a bid to maximize the SO performance.

REFERENCES

Bollapragada, R., R. Byrd, and J. Nocedal. 2018. "Adaptive Sampling Strategies for Stochastic Optimization". *SIAM Journal on Optimization* 28(4):3312–3343.

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen. 1984. *Classification and Regression Trees*. Boca Raton: CRC press.

Bretthauer, K. M., A. Ross, and B. Shetty. 1999. "Nonlinear Integer Programming for Optimal Allocation in Stratified Sampling". *European Journal of Operational Research* 116(3):667–680.

Chaddha, R., W. Hardgrave, D. Hudson, M. Segal, and J. Suurballe. 1971. "Allocation of Total Sample Size When only the Stratum Means are of Interest". *Technometrics* 13(4):817–831.

Curtis, F. E., and K. Scheinberg. 2020. "Adaptive Stochastic Optimization: A Framework for Analyzing Stochastic Optimization Algorithms". *IEEE Signal Processing Magazine* 37(5):32–42.

Eckman, D. J., S. G. Henderson, and S. Shashaani. 2022. "Diagnostic Tools for Evaluating and Comparing Simulation-Optimization Algorithms". accessed October 5, 2022. https://people.orie.cornell.edu/shane/pubs/comparingsimopt.pdf.

Eckman, D. J., S. G. Henderson, S. Shashaani, and R. Pasupathy. 2020. *Simulation Optimization Library*. http://github.com/simopt-admin/simopt, accessed 1st May 2020.

Espath, L., S. Krumscheid, R. Tempone, and P. Vilanova. 2021. "On the Equivalence of Different Adaptive Batch Size Selection Strategies for Stochastic Gradient Descent Methods". *arXiv preprint arXiv:2109.10933*. https://doi.org/10.48550/arXiv.2109.10933, accessed 23rd May 2022.

Etoré, P., and B. Jourdain. 2010. "Adaptive Optimal Allocation in Stratified Sampling Methods". *Methodology and Computing in Applied Probability* 12(3):335–360.

Farias, F., T. Ludermir, and C. Bastos-Filho. 2020. "Similarity Based Stratified Splitting: An Approach to Train Better Classifiers". *arXiv preprint arXiv:2010.06099*. https://doi.org/10.48550/arXiv.2010.06099, accessed 3^{rd} May 2022.

Glynn, P. W., and W. Whitt. 1992. "The Asymptotic Efficiency of Simulation Estimators". *Operations Research* 40(3):505–520.

Glynn, P. W., and Z. Zheng. 2021. "Efficient Computation for Stratified Splitting". In *Proceedings of the 2021 Winter Simulation Conference*, edited by S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, 1–8. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Ha, Y., S. Shashaani, and Q. Tran-Dinh. 2021. "Improved Complexity of Trust-Region Optimization for Zeroth-Order Stochastic Oracles with Adaptive Sampling". In *Proceedings of the 2021 Winter Simulation Conference*, edited by S. Kim, B. Feng,

K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, 1–12. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Huddleston, H., P. Claypool, and R. Hocking. 1970. "Optimal Sample Allocation to Strata using Convex Programming". *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 19(3):273–278.

Jain, P., S. Shashaani, and E. Byon. 2021. "Wake Effect Calibration in Wind Power Systems with Adaptive Sampling based Optimization". In *Proceedings of the IISE Annual Conference*, edited by A. Ghate, K. Krishnaiyer, and K. Paynabar, 43–48. Red Hook, NY: Institute of Industrial and Systems Engineers.

Jensen, N. O. 1983. *A Note on Wind Generator Interaction*, Volume 2411. Roskilde: Risø National Laboratory.

Kawai, R. 2010. "Asymptotically Optimal Allocation of Stratified Sampling with Adaptive Variance Reduction by Strata". *ACM Transactions on Modeling and Computer Simulation* 20(2):1–17.

Kennedy, M. C., and A. O'Hagan. 2001. "Bayesian Calibration of Computer Models". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63(3):425–464.

Kim, S., R. Pasupathy, and S. G. Henderson. 2015. "A Guide to Sample Average Approximation". In *Handbook of Simulation Optimization*, edited by M. C. Fu, 207–243. New York: Springer.

Liu, B., X. Yue, E. Byon, and R. Al Kontar. 2022. "Parameter Calibration in Wake Effect Simulation Model with Stochastic Gradient Descent and Stratified Sampling". *The Annals of Applied Statistics* 16(3):1795–1821.

Mak, W. K., D. P. Morton, and R. K. Wood. 1999. "Monte Carlo Bounding Techniques for Determining Solution Quality in Stochastic Programs". *Operations Research Letters* 24(1):47–56.

Mulvey, J. M. 1983. "Multivariate Stratified Sampling by Optimization". *Management Science* 29(6):715–724.

Nemirovski, A., A. Juditsky, G. Lan, and A. Shapiro. 2009. "Robust Stochastic Approximation Approach to Stochastic Programming". *SIAM Journal on Optimization* 19(4):1574–1609.

Pettersson, P., and S. Krumscheid. 2021. "Adaptive Stratified Sampling for Non-smooth Problems". *arXiv preprint arXiv:2107.01355*. https://doi.org/10.48550/arXiv.2107.01355, accessed 4th April 2022.

Ross, S. 2013. "Chapter 9 - Variance Reduction Techniques". In *Simulation* (Fifth Edition ed.)., edited by S. Ross, 153 – 231. New York: Academic Press.

Sanchez, S. M., and P. J. Sanchez. 2020. "Robustness Revisited: Simulation Optimization Viewed through a Different Lens". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K. G. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 60–74. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Shashaani, S., F. S. Hashemi, and R. Pasupathy. 2018. "ASTRO-DF: A Class of Adaptive Sampling Trust-Region Algorithms for Derivative-Free Stochastic Optimization". *SIAM Journal on Optimization* 28(4):3145–3176.

Tong, C. 2006. "Refinement Strategies for Stratified Sampling Methods". *Reliability Engineering & System Safety* 91(10-11):1257–1265.

You, M., E. Byon, J. Jin, and G. Lee. 2017. "When Wind Travels Through Turbines: A New Statistical Approach for Characterizing Heterogeneous Wake Effects in Multi-turbine Wind Farms". *IISE Transactions* 49(1):84–95.

Zhang, D.-g., C.-h. Ni, J. Zhang, T. Zhang, P. Yang, J.-x. Wang, and H.-r. Yan. 2022. "A Novel Edge Computing Architecture Based on Adaptive Stratified Sampling". *Computer Communications* 183:121–135.

Zhao, P., and T. Zhang. 2014. "Accelerating Minibatch Stochastic Gradient Descent using Stratified Sampling". *arXiv preprint arXiv:1405.3080*. https://doi.org/10.48550/arXiv.1405.3080, accessed 12th March 2022.

AUTHOR BIOGRAPHIES

PRANAV JAIN is a third year Ph.D. student in the Edward P. Fitts Department of Industrial and Systems Engineering at North Carolina State University. His main research is in data-driven simulation optimization. His email address is pjain23@ncsu.edu.

SARA SHASHAANI is an Assistant Professor in the Edward P. Fitts Department of Industrial and System Engineering at North Carolina State University. Her research interests are analysis and design of stochastic models, Monte Carlo methodology, and simulation optimization. Her email address is sshasha2@ncsu.edu and her homepage is https://shashaani.wordpress.ncsu.edu/.

EUNSHIN BYON is an Associate Professor in the Department of Industrial and Operations Engineering at the University of Michigan. Her research interests include data science, energy and sustainability, and quality and reliability engineering. Her email address is ebyon@umich.edu. Her website is https://ebyon.engin.umich.edu.