Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds.

REFLECTIONS ON SIMULATION OPTIMIZATION

Shane G. Henderson

School of Operations Research and Information Engineering
Cornell University
Ithaca, NY 14850, U.S.A.

ABSTRACT

I provide some perspectives on simulation optimization. First, more attention should be devoted to the finite-time performance of solvers than on ensuring convergence properties that may only arise in asymptotic time scales that may never be reached in practice. Both analytical results and computational experiments can further this goal. Second, so-called sample-path functions can exhibit extremely complex behavior that is well worth understanding in selecting a solver and its parameters. Third, I advocate the use of a layered approach to formulating and solving optimization problems, whereby a sequence of models are built and optimized, rather than first building a simulation model and only later "bolting on" optimization.

1 INTRODUCTION

The simulation optimization (SO) problem as explored in this tutorial is to minimize some real-valued function $f(x)$ over a domain $D \subseteq \mathbb{R}^d$, where $f(x) = \mathbb{E}f(x,\xi)$ can only be estimated through (stochastic) simulation replications $f(x,\xi_1), f(x,\xi_2), \ldots, f(x,\xi_n)$, say. Here $\xi_1, \xi_2, \ldots$ are assumed to be independent random elements that each have the same distribution as ξ, and x is a vector of decision variables. This formulation is very general, since the vector of decision variables x can encompass both distributional and structural parameters, and the sequence $(\xi_1, \xi_2, \ldots)$ can be viewed as the $U(0,1)$ random numbers that drive the simulation. Still, the formulation is not completely general, since it omits non-expectation objectives such as quantiles and other functionals. Many, but certainly not all, of the viewpoints herein extend to such objectives. Also, the formulation excludes stochastic constraints.

My goal is to highlight a few perspectives that I believe merit more attention from the SO research community.

1. Convergence has been over-emphasized in the SO literature; we should strive to design SO solvers (implementations of algorithms) that make rapid progress in reducing the objective function, but not be as concerned with devising solvers to converge to a global minimizer as the iteration count increases without bound. Even convergence to a local minimizer is not, practically speaking, as important as rapid progress in early parts of the search. This perspective may seem odd, but is more fully explicated and defended in Section 2, where I also promote the use of analytical results that shed light on the early stages of SO solver runs, using stochastic approximation and trust-region methods as extended examples to reinforce the key ideas.
2. Analytical results that shed light on early progress of SO solvers can be complemented by simulation experiments that compare a set of SO solvers on a suite of test problems. There has been substantial recent progress in defining metrics for comparing SO solvers (Eckman et al. 2021a) and in improving SimOpt (Pasupathy and Henderson 2006; Pasupathy and Henderson 2011b; Eckman et al. 2019) to support such comparisons (Eckman et al. 2021b). Section 3 surveys some of those metrics.

978-1-6654-3311-2/21/$31.00 ©2021 IEEE

3. SO solvers can often benefit from the use of common random numbers (CRN), because CRN can allow one to more accurately estimate the difference in objective function values of two solutions. The use of CRN across the solution space yields what I call sample-path functions. The behavior of these sample-path functions is very interesting and has implications for the design of, e.g., trust-region methods as explored through examples in Section 4.
4. Finally, Section 5 discusses some modeling philosophy, proposing that SO not be considered *the* approach to solving a practical problem. Rather, in line with the views of several experienced modelers, I advocate working with a succession of models that all address the problem under consideration, where not all models need to be simulation models. Moreover, I advocate trying to build optimization-facilitating structure such as convexity into the SO problem from the outset, only abandoning structure when it cannot be maintained without sacrificing essential model fidelity.

As should now be clear, this paper is not a survey. Readers interested in an introduction to SO could consult Jian and Henderson (2015). Fu (2014) provides a more advanced and comprehensive treatment. Fu and Henderson (2017) provides an historical perspective in relatively compact form. Recent tutorials in the Winter Simulation Conference include Sanchez and Sanchez (2020), which provides a perspective on robustness in discrete optimization problems and Newton et al. (2018), which reviews key ideas in stochastic gradient descent that are highly relevant in the SO context.

2 IS CONVERGENCE OVERRATED?

Why should we design SO solvers to be convergent? This question may seem almost nonsensical. After all, an enormous literature is devoted to proving convergence of optimization algorithms. Naturally, we'd like an optimization solver to solve the problem for which it is designed if given sufficient time, and convergence to an optimal solution is certainly a reasonable way to define the term "solve." However, if the relevant time scales over which convergence or near convergence is achieved are so long that we will never reach that point in practice, then why establish convergence results?

Despite posing this question, I believe convergence proofs are important and worthwhile. First, they can represent a serious theoretical challenge that may uncover new avenues of algorithm analysis. Second, they provide some assurance that a solver will behave reasonably; if the solver converges to an optimal solution in the long run, then one might expect it to make progress towards an optimal solution over shorter time scales. And if it fails to converge, how then does it behave? Third, they may help clarify the time scales over which convergence is expected; if this time scale is very long then one might naturally turn attention to the early behavior of the solver. Fourth, they may motivate the specialization of the solver to problems that exhibit structure, such as convexity, under which convergence times are more palatable.

These perspectives are amplified when one draws a contrast between *global* convergence and *local* convergence. Proofs of the former in the absence of structure tend to embed the principle that global convergence arises when all solutions have been sampled infinitely often. (With an appropriate structural modification of this principle for problems with an infinite number of solutions, e.g., through Lipschitz continuity in the case of continuously parameterized problems, and growth conditions outside a compact set for problems with unbounded domains.) The central difficulty with this principle is that the time scales over which all solutions are visited is typically enormous. A delightful word I learned from Peter Glynn to describe such out-of-reach time scales is "asymptopia." In essence, the convergence proof is correct, but is only relevant in asymptopia.

Local convergence proofs do not suffer to the same extent from this time-scale malady. I applaud the effort to develop local convergence proofs that are relevant on practical time scales. Examples of such results include those for COMPASS (Hong and Nelson 2006), ADALINE (Ragavan et al. 2021) and ASTRO-DF (Shashaani et al. 2018). Even so, some SO solvers, when modified to ensure even local convergence, can perform less reliably as discussed below in the case of stochastic approximation.

In light of this discussion, perhaps a better positioning of the question that opened this section, which was deliberately provocative, is instead "Is convergence overrated when the time scale over which we can reasonably expect convergence or near-convergence is likely beyond our computational reach?" I believe the answer is "yes."

When convergence cannot be expected on a practical timeframe, then we should not view it as a critical goal in the design of solvers that are meant to be used in practice. But then, what should be the goal of a solver? A reasonable goal seems to be that a solver should be able to improve over an initial solution at some reasonable rate, at least for some time. (This rather vague statement will be made precise in two examples below.)

This philosophy aligns with many of the SO solvers deployed in off-the-shelf simulation packages. Those solvers are typically based on some meta-heuristic search principle that can be expected to make at least some progress on many problems, but that does not provide convergence guarantees. (Such solvers are also well equipped to tackle problems with little or no structure that are often posed by simulation users.) These solvers have been highly successful, partly because they provide improvements in an objective function over computational budgets that are relevant in practice.

What kind of theory might be developed to yield practical insight for solvers that either don't converge, or that converge only in asymptopia? One class of results relates to the progress that can be made over the initial iterations of the solver. Here we present two extended examples, both of which use arguments closely related to Lyapunov functions.

2.1 Stochastic Approximation

There are two primary methods for analyzing stochastic approximation (SA). The first is sometimes called the ordinary-differential-equation (ODE) method that exploits the notion that when suitably scaled, the usual SA recursion converges in a suitable sense to the solution to an ODE; see, e.g., Asmussen and Glynn (2007), Ch. VIII. The second method is to directly analyze the error in the estimated solutions and can be found in many places, including Wright and Recht (2021) and Bottou et al. (2018). What follows is a repeat of this argument, very closely following the presentation in Nemirovski et al. (2009).

Suppose we are minimizing $f(x) = \mathbb{E}f(x,\xi)$ over a non-empty convex compact subset $D \subset \mathbb{R}^d$ for some $d < \infty$. We assume that $f(\cdot,\xi)$ is convex and continuously differentiable for each fixed ξ and that $f(\cdot)$ inherits these properties. (I am skipping some technicalities; see Nemirovski et al. 2009 for details.) Assume we have an unbiased estimator $g(x,\xi)$ of the gradient of $f(\cdot)$ at x, i.e., $\mathbb{E}g(x,\xi) = \nabla f(x)$ for all $x \in D$. Since $f(\cdot)$ is continuous, it attains its minimum on D at some point x^* say. Let Π_D denote projection onto the set D, so that $\Pi_D x = \text{argmin}_{y\in D}\|x-y\|$. Unless otherwise specified, we use the usual ℓ_2 norm throughout. Then Π_D is a contraction, i.e., $\|\Pi_D x - \Pi_D y\| \le \|x-y\|$ for all $x,y \in \mathbb{R}^d$. The usual SA recursion, for a given stepsize sequence $(\gamma_j : j \ge 0)$, starting from some initial point $X_0 \in D$ is

$$X_{j+1} = \Pi_D(X_j - \gamma_j g(X_j,\xi_j))$$

for $j \ge 0$, where $\xi_0,\xi_1,\ldots$ are iid replicates of ξ.

Let $A_j = \frac{1}{2}\|X_j - x^*\|^2$ and $a_j = \mathbb{E}A_j$. Then, since Π_D is a contraction, for $j \ge 0$,

$$\begin{aligned}
A_{j+1} &= \frac{1}{2}\|\Pi_D(X_j - \gamma_j g(X_j,\xi_j)) - x^*\|^2 \\
&= \frac{1}{2}\|\Pi_D(X_j - \gamma_j g(X_j,\xi_j)) - \Pi_D x^*\|^2 \\
&\le \frac{1}{2}\|X_j - \gamma_j g(X_j,\xi_j) - x^*\|^2 \\
&= A_j + \frac{1}{2}\gamma_j^2\|g(X_j,\xi_j)\|^2 - \gamma_j(X_j - x^*)^\top g(X_j,\xi_j). \qquad (1)
\end{aligned}$$

Now, for $j \geq 0$, ξ_j is independent of $\mathscr{F}_{j-1} = \sigma(X_0, \xi_0, \xi_1, \ldots, \xi_{j-1})$, taking $\mathscr{F}_{-1} = \sigma(X_0)$. Thus, for $j \geq 0$, ξ_j is independent of X_j, so using the tower property of expectation,

$$\begin{aligned}\mathbb{E}[\gamma_j(X_j - x^*)^\top g(X_j, \xi_j)] &= \gamma_j \mathbb{E}[\mathbb{E}[(X_j - x^*)^\top g(X_j, \xi_j)|\mathscr{F}_{j-1}]] \\ &= \gamma_j \mathbb{E}[(X_j - x^*)^\top \mathbb{E}[g(X_j, \xi_j)|\mathscr{F}_{j-1}]] \\ &= \gamma_j \mathbb{E}[(X_j - x^*)^\top \nabla f(X_j)].\end{aligned}$$

Taking expectations through (1) we therefore get

$$a_{j+1} \leq a_j + \frac{1}{2}\gamma_j^2 \mathbb{E}[\|g(X_j, \xi_j)\|^2] - \gamma_j \mathbb{E}[(X_j - x^*)^\top \nabla f(X_j)].$$

Now assume that the gradient estimators have uniformly bounded second moment in the domain D, so that $\mathbb{E}[\|g(X_j, \xi_j)\|^2] \leq M$ for some $M < \infty$. Also, now suppose that the function $f(\cdot)$ is strongly convex on D, which, together with our earlier assumption of differentiability, ensures that there exists a constant $c > 0$ such that for all $x, y \in D$,

$$f(y) \geq f(x) + (y - x)^\top \nabla f(x) + \frac{c}{2}\|y - x\|^2,$$

or, equivalently, that

$$(y - x)^\top (\nabla f(y) - \nabla f(x)) \geq c\|y - x\|^2. \tag{2}$$

Strong convexity ensures that x^* is unique, and since x^* is optimal

$$(x - x^*)^\top \nabla f(x^*) \geq 0$$

for all $x \in D$. (In the case where x^* lies in the interior of D then $\nabla f(x^*) = 0$, but x^* could lie on the boundary.) Combining this with (2) gives

$$\mathbb{E}[(X_j - x^*)^\top \nabla f(X_j)] \geq \mathbb{E}[(X_j - x^*)^\top (\nabla f(X_j) - \nabla f(x^*)] \geq c\mathbb{E}[\|X_j - x^*\|^2] = 2ca_j.$$

Summarizing, we now have that for $j \geq 0$,

$$a_{j+1} \leq (1 - 2c\gamma_j)a_j + \gamma_j^2 \frac{M}{2}. \tag{3}$$

Typically, one now chooses $\gamma_j = \theta/(j+1)$ for some $\theta > 0$. Instead, fixed-steplength SA chooses γ_j to be constant in j, say $\gamma_j = \gamma > 0$ with γ so small that $2c\gamma < 1$. Then, (3) ensures that the mean-squared error in X_j as an estimator of x^* decreases geometrically, at least for j so small and the initial error a_0 so large that the error measure a_j is still decreasing. Related ideas are mentioned briefly in Newton et al. (2018), attributing the result to Bottou et al. (2018).

To explore this point a little more, we unfold (3) to get, for $j \geq 1$,

$$a_j \leq \beta^j a_0 + \gamma^2 \frac{M}{2}(1 + \beta + \beta^2 + \cdots + \beta^{j-1}),$$

where $\beta = 1 - 2c\gamma$.

This conclusion nicely encapsulates the dynamics of the SA recursion for fixed γ. Indeed, for fixed steplength γ, $(X_j : j \geq 0)$ is a (general state space) Markov chain and the argument leading to (3) is essentially a proof that $V(x) = \frac{1}{2}\|x - x^*\|^2$ is a Lyapunov function satisfying a geometric drift condition. Provided that the chain is appropriately irreducible, it follows that the chain is positive (Harris) recurrent (Meyn and Tweedie 1993, Theorem 16.0.1). This is intuitive, since the dynamics are such that when X_0 is

far from x^*, the chain is attracted to x^* but cannot converge because our steplengths do not decrease to 0. The choice of steplength dictates how quickly errors are reduced initially (through the term $\beta^j a_0$) but also how closely the chain eventually "orbits" the optimal point x^* through the term $\gamma^2 \frac{M}{2}(1+\beta+\beta^2+\cdots)$.

Here we are mostly focused on the dynamics for a modest number of iterations, and in that setting the error $\mathbb{E}[\|X_j - x^*\|^2]$ decreases geometrically rapidly.

This is an extremely encouraging result, suggesting rapid improvement in the objective function, with two provisos. First, this rapid improvement can only be expected when we are initially "far" from the optimal point x^*. Still, that is exactly the regime we are focusing on here. Second, the reduction comes provided we can choose a step size $\gamma > 0$ such that $1 - 2c\gamma \geq 0$. It is not at all clear how to choose a step size $\gamma \in (0, 1/2c)$, since c is rarely known, even approximately, which is the essence of the fragility of SA. Moreover, this fragility is clearly on show when attempting to select an adaptive stepsize sequence to ensure (local) convergence; see Asmussen and Glynn (2007), Ch. VIII.

In any event, it should be clear that the error in SA can be analyzed with some fidelity for short runlengths, supporting its use in practice provided a reasonable choice of (fixed) stepsize γ can be identified.

2.2 Trust Region Methods

Trust-region methods are a powerful class of methods for nonlinear optimization methods. They have their origin in deterministic optimization problems, but have seen extension to simulation optimization, e.g., Deng and Ferris (2006), Chang et al. (2007), Shashaani et al. (2016), Shashaani et al. (2018), Blanchet et al. (2019). The idea behind these methods is to solve optimization problems by maintaining, in each iteration, a current iterate $X_k \in D$ and a trust-region radius Δ_k. In each iteration a local metamodel is fit to the objective function $f(\cdot)$ that is assumed to be a useful model in the ball $B(X_k, \Delta_k)$ centered at X_k with radius Δ_k. One then approximately minimizes the metamodel within the ball to generate a proposed new point Y_k say, and the function value $f(Y_k)$ is estimated. If the predicted improvement in objective function value from the metamodel approximately matches the directly estimated improvement in objective function value (the estimated value of $f(X_k) - f(Y_k)$) then the proposed new solution is accepted ($X_{k+1} = Y_k$) and the trust-region radius is enlarged ($\Delta_{k+1} = r\Delta_k$ for some $r > 1$). (The radius is capped at some maximal radius.) If not then the new solution is rejected ($X_{k+1} = X_k$) and the trust-region radius is reduced ($\Delta_{k+1} = \Delta_k / r$).

Trust-region methods may vary slightly from this description, but this is the essence of the idea. The trust region radius is adjusted to ensure a good match between the metamodel and the true function within the ball. Taylor's theorem ensures that for a sufficiently small radius a good match is possible between a linear or quadratic metamodel and the true function, though it is desirable to maintain a larger radius to enable more rapid progress.

Blanchet et al. (2019) develop an elegant approach to the analysis of trust-region methods using supermartingales. They seek to understand the number of iterations $T = \inf\{k \geq 0 : \|\nabla f(X_k)\| \leq \varepsilon\}$ needed to reach a point with a sufficiently small (in norm) gradient, for some fixed $\varepsilon > 0$. They define a potential function, which for one of their analyses takes the form $V(x, \delta) = v f(x) + (1-v)\delta^2$, for some suitable $v \in (0,1)$. It will be convenient for us to adjust this definition slightly to

$$V(x, \delta) = v(f(x) - f(x^*)) + (1-v)\delta^2,$$

where x^* minimizes f. This modified definition differs from theirs only by a constant. Their Theorem 3 shows that under suitable conditions, if $\mathcal{G}_k = \sigma(X_0, X_1, \ldots, X_k, \Delta_0, \Delta_1, \ldots, \Delta_k)$, then

$$\mathbb{E}[V(X_{k+1}, \Delta_{k+1}) - V(X_k, \Delta_k) | \mathcal{G}_k] \leq -b\Delta_k^2, \tag{4}$$

for some $b > 0$ on the event that $k < T$, i.e., on the event that the algorithm is yet to reach a point with small (in norm) gradient. (Strictly speaking, they employ a different sigma field, but this will suffice for our discussion.) Why is this relevant to our discussion? This result yields a bound on $\mathbb{E}T$ through a stopping-time argument. The idea is that $V(X_k, \Delta_k)$ is nonnegative, so cannot go negative. Since, on

average, it decreases by $b\Delta_k^2$ on each step prior to T, the number of iterations T cannot grow so large that $b\mathbb{E}\sum_{k=0}^{T-1}\Delta_k^2 > \mathbb{E}V(X_0,\Delta_0)$.

To give a sense of the more rigorous argument, let x^* be a minimizer of f and define $M_0 = V(X_0,\Delta_0) = v(f(X_0)-f(x^*))+(1-v)\Delta_0^2$. Let $a\wedge b = \min\{a,b\}$. For $k\geq 1$ define

$$M_k = V(X_{T\wedge k},\Delta_{T\wedge k}) + \sum_{j=0}^{T\wedge k-1}\Delta_j^2.$$

Equation 4 implies that $(M_k : k\geq 0)$ is a super martingale with respect to the filtration $(\mathscr{G}_k : k\geq 0)$. This implies that for any $k\geq 0$, $\mathbb{E}M_k \leq \mathbb{E}M_0$, i.e., that

$$\mathbb{E}V(X_{T\wedge k},\Delta_{T\wedge k}) + \mathbb{E}\sum_{j=0}^{T\wedge k-1}\Delta_j^2 \leq \mathbb{E}V(X_0,\Delta_0).$$

Now,

$$\mathbb{E}V(X_{T\wedge k},\Delta_{T\wedge k}) = v(\mathbb{E}f(X_{T\wedge k})-f(x^*))+(1-v)\mathbb{E}\Delta_{T\wedge k}^2 \geq 0,$$

so

$$\mathbb{E}\sum_{j=0}^{T\wedge k-1}\Delta_j^2 \leq \mathbb{E}V(X_0,\Delta_0).$$

Taking the limit as $k\to\infty$, monotone convergence ensures that

$$\mathbb{E}\sum_{j=0}^{T-1}\Delta_j^2 \leq \mathbb{E}V(X_0,\Delta_0).$$

Blanchet et al. (2019) use a separate analysis to show that the trust-region radius stays above some threshold sufficiently frequently that the left-hand-side of this expression is linear in $\mathbb{E}T$ (with explicit constants), and this then yields a bound on $\mathbb{E}T$ that is $O(\varepsilon^{-2})$.

The analysis permits the trust-region method to occasionally make "mistakes" (as all SO solvers will do when simulation noise is involved), provided that the probability of such mistakes is controlled. The probability of a mistake is kept small by choosing the sample size employed to estimate the metamodel on the kth iteration. Essentially, the sample size needs to be of the order Δ_k^{-4} in terms of the trust-region radius Δ_k. The true computational complexity of the trust-region procedure should take into account the number of *samples* required to identify a point with small normed gradient, rather than just the number of *iterations* as discussed here. It seems likely that some such result might be obtained, because the analysis in the paper identifies a biased random-walk-like behavior of the sequence of trust-region radii that keeps them from getting too small, which might prove tractable, but such an analysis is not given therein. A second weakness with this analysis, yet one that seems less central than the first, is that the stopping time T when the norm of the (true) gradient drops below ε is unobservable, so the trust-region method may not *know* that it has identified a high-quality solution.

This supermartingale-style argument looks likely to be useful in analyzing other SO algorithms, as the authors point out. The key insight needed is the identification of an appropriate potential function.

3 FINITE-TIME METRICS

Theoretical analyses such as those we have sketched in the previous section often require strong assumptions to enable the analysis to go through. Such assumptions can weaken the overall conclusion relative to observed practice. Accordingly, these analyses can be buttressed with empirical comparisons of solvers on a library of test problems. In fact, if it proves difficult or impossible to develop theoretical analyses, then empirical comparisons may be the primary method for comparison of solvers.

Eckman et al. (2021a, 2021b) develop, respectively, metrics for empirical comparisons of multiple solvers on multiple test problems and a new version of the testbed SimOpt (Pasupathy and Henderson 2006, 2011b, 2011a; Eckman et al. 2019). The new version is a significant improvement over previous versions, with the main innovations being

1. a conversion to python;
2. the use of github;
3. development of support for random instances of problems;
4. multiple SO problems built from a single simulation model;
5. careful exercise and automation of CRN control through wrapper functions that set up and use a newly implemented version of the generator described in L'Ecuyer et al. (2002) with 3 levels of streams instead of 2; and
6. the introduction and automation of many plots that provide an evaluation of the performance of multiple solvers on multiple problems.

Here we focus on the last point above, giving examples of the plots and discuss how they can be interpreted, referring the reader to Eckman et al. (2021a) for a more complete discussion. The plots below are inspired by related ideas from the literature, with important references including Dolan and Moré (2002), Gould and Scott (2016), Moré and Wild (2009), Ali et al. (2005) and Beiranvand et al. (2017).

We use the term "macro-replication" to refer to a single run of a single solver on a single problem, restricting the solver to use at most a predefined problem-specific budget, b say, of simulation replications. The progress of the solver as it works to solve the problem can be visualized through a *progress curve*, which is essentially a rescaled version of plots that are commonly used to indicate progress. We rescale the horizontal axis to indicate the *fraction* of the budget b expended, so that it runs from 0 to 1. We rescale the vertical axis to indicate the fraction of the initial optimality gap that remains, so that it also runs from 0 to 1. More precisely, let X_t denote the (random) solution the solver recommends after a fraction $t \in [0,1]$ of the budget b has been expended, starting from a fixed initial solution x_0. Let $f(x^*)$ indicate the optimal objective function value of the problem. (If $f(x^*)$ is unknown then a bound or estimate can be used in lieu.) A *progress curve* plots $\nu(t) = (f(X_t) - f(x^*))/(f(x_0) - f(x^*))$ as a function of t. If exact function values are not available, then estimates are used; those estimated function values are obtained from so-called *post-replications* that are needed to avoid optimization bias as described in, e.g., Mak et al. (1999). Macro-replications are needed to gain a sense of the variability in performance of the solver on the problem. We can aggregate the performance of the solver on multiple macro-replications through *aggregated* progress curves that indicate the mean or a quantile at each time t. In the hypothetical plot in Figure 1a the solver makes steady progress, eventually reducing the optimality gap to 20% of its initial value.

There are two levels of simulation in these evaluations: macro-replications and post-replications. Accordingly, the estimators of aggregated progress curves are two-level estimators. Eckman et al. (2021a) discusses the implications of the use of two-level estimators and advocates the use of bootstrapping to obtain error estimates.

The (random) area under a progress curve, $A = \int_0^1 \nu(t)\,dt$, provides a sense of how rapidly a solver solves a problem; if the area A is small, then the solver rapidly identifies high-quality solutions. These areas depend heavily on the choice of budget b due to the use of rescaling in the plots; if b is too large then the progress curve will drop rapidly to 0 and the area A will be nearly 0. If b is too small then the progress curve will not decrease much and the area A will be nearly 1. Thus, the budget b needs to be chosen with some care. The mean and standard deviation of A can be estimated from multiple macro-replications, with the mean indicating overall performance and the standard deviation indicating variability in performance. Two-level simulation is needed when the function $f(\cdot)$ cannot be evaluated exactly. The performance of a solver on a suite of problems can be summarized in a scatter plot, where the coordinates of each point in the plot are the mean and standard deviation of the area under a progress curve for a single problem.

Multiple solvers are readily compared using these scatter plots or superimposed versions thereof. In the hypothetical plot in Figure 1b, the solver indicated by blue x's tends to have both a lower mean and a lower standard deviation over 10 problems, but the standard deviation varies more from problem to problem.

Progress curves provide a sense of how a solver is progressing over time, but they do not directly indicate how long a solver takes to solve a problem. To that end we can define $\tau(\alpha) = \inf\{t \in [0,1] : \nu(t) \leq \alpha\}$ to be the α-solve time, i.e., the fraction of the budget b needed to obtain a solution with an optimality gap that is at most a fraction α of the initial optimality gap. If no such solution is found then $\tau(\alpha) = \infty$. Thus, $\tau(\alpha)$ is an extended random variable.

In comparing solvers on a suite of problems $\mathscr{P}$ we can compute, or estimate, the distribution of $\tau^{p,s}(\alpha)$, the α-solve time for each solver s on each problem $p \in \mathscr{P}$. Then we can compute the average likelihood of Solver s solving problems in the set $\mathscr{P}$ within a fraction $t \in [0,1]$ of each problem's budget by

$$\rho^s(t) = \frac{1}{|\mathscr{P}|} \sum_{p \in \mathscr{P}} \mathbb{P}(\tau^{p,s}(\alpha) \leq t),$$

using estimators in place of exact values when needed. We call the curve $(\rho^s(t) : t \in [0,1])$ the *solvability profile of the cdf (cumulative distribution function) of the α-solve times*. Like the area scatter plot mentioned above it provides a summary of Solver s performance across the entire suite of problems $\mathscr{P}$. In the hypothetical plot in Figure 1c, after half the budget is expended, the solver has on average (over problems) a 20% chance of α-solving the problem, with this value increasing to 85% after a little over 80% of the budget is expended. Beyond this point the solver makes no further progress.

Sometimes we want to single out a solver, s_0 say, for special attention. For example, s_0 may be a new solver we have developed that we want to compare to a collection of existing solvers. Or s_0 may be a benchmark solver that has, in past experiments, exhibited good overall performance. In such cases, it may be of interest to look at *solvability difference profiles*, henceforth termed "difference profiles." Difference profiles are simply differences of solvability profiles. For solvers s and s_0, the difference profile is $(\rho^s(t) - \rho^{s_0}(t) : t \in [0,1])$, which at each value of t takes values in $[-1,1]$. Solver s does better than the benchmark s_0 if the difference profile mostly lies above 0. In the hypothetical plot in Figure 1d, Solver s_0 makes better progress than Solver s up until about 60% of the budget is expended, but then Solver s outperforms Solver s_0 from then on.

Difference profiles allow us to compare two solvers in detail. The same information can be obtained by looking at two solvability profiles, one for each solver, so why introduce difference profiles? Mainly, they highlight differences between solvers. Also, they allow us to exploit CRN in estimating the difference in solver performance, reducing estimator variance; see Eckman et al. (2021a).

4 SAMPLE-PATH FUNCTIONS CAN BE MESSY

A fundamental problem in SO involves comparing two solutions, x_1 and x_2 say, to see which has the lower objective function value. Such comparisons are usually greatly facilitated by the use of CRN, where we estimate $f(x_1) - f(x_2)$ by

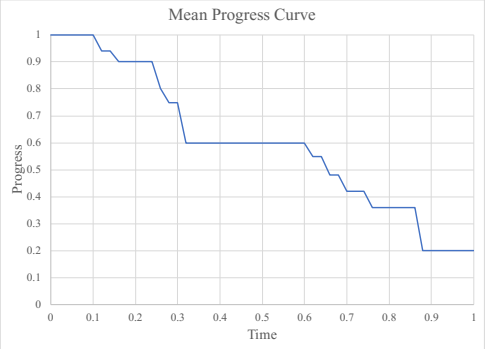
$$\frac{1}{n} \sum_{i=1}^{n} [f(x_1, \xi_i) - f(x_2, \xi_i)],$$

with $(\xi_1, \xi_2, \ldots, \xi_n)$ being an iid sample from the distribution of ξ. This is an unbiased estimator of $f(x_1) - f(x_2)$, the sign of which can be used to assess which of x_1 and x_2 is the better solution.

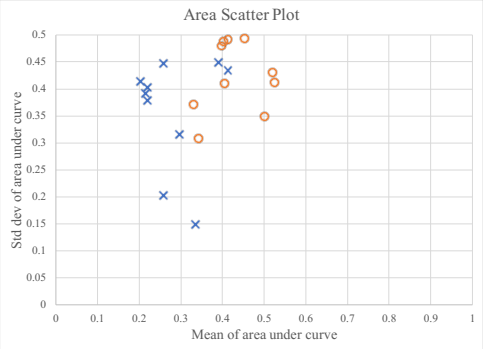
Taking this idea to an extreme, we can estimate the entire function $f(\cdot)$ at any point x by

$$\frac{1}{n} \sum_{i=1}^{n} f(x, \xi_i),$$

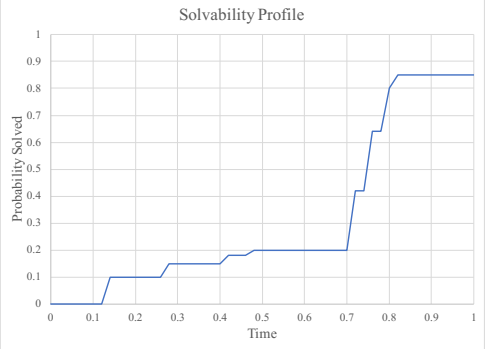
using a single iid sample $(\xi_1, \xi_2, \ldots, \xi_n)$ that is common to all points x. This is the central idea behind sample-average approximation (Shapiro 2003; Kim et al. 2015), where one then minimizes this sample-



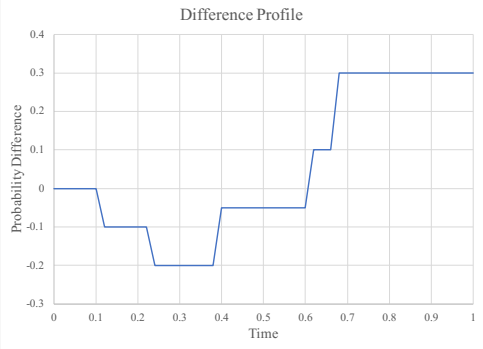
(a) A mean progress curve.



(b) An area scatter plot.



(c) A solvability profile.



(d) A difference profile.

Figure 1: Example plots in SimOpt (hypothetical).

path function to obtain an estimator X_n^* of a true minimizer x^* of $f(\cdot)$. It is also closely related to the score-function optimization method (Rubinstein and Shapiro 1990) where one uses a change of measure to estimate the entire function $f(\cdot)$ using a single sample. There are also close links to recent work in the use of regularization and related ideas in stochastic optimization, e.g., Sutter et al. (2021). Moreover, trust-region methods use simulation replications at x values in some local neighborhood to fit a local metamodel; if CRN is used then these metamodels are built from the sample-path function values in that neighborhood. Accordingly, it is well worth understanding how these sample-path functions behave.

It turns out that the sample-path functions often, or even usually, exhibit discontinuities in x. Such discontinuities can arise when events in a simulation model change their order in time as x is perturbed; see discussion of the commuting condition in Glasserman (1991) and Kim et al. (2015) for a recent exploration. In some models, an exchange in the order of events does not bring about a discontinuity. This happens, for example, in the newsvendor model and many others; see Glasserman (1991) and Kim et al. (2015). This is an important observation underlying much of the development in Fu and Hu (1997). However, the typical situation in practice is that the sample path functions are discontinuous.

In the remainder of this section we explore such discontinuities in two examples, with the goal of improving our understanding of the nature of such discontinuities. In these examples, the discontinuities become dense in the domain D as the number of samples n grows, but the size of the discontinuities is of stochastic order n^{-1}. Intuitively, we say that a sequence of random variables $(\zeta_n : n \geq 1)$ is of stochastic order $g(n)$ if ζ_n "looks like" $g(n)\tilde{\zeta}_n$, where the random variables $(\tilde{\zeta}_n : n \geq 1)$ are uniformly bounded. More rigorously, we say that a sequence of random variables $(\zeta_n : n \geq 1)$ is of stochastic order $g(n)$ for some

deterministic positive-valued function $(g(n) : n \geq 1)$ and write $\zeta_n = O_p(g(n))$ if the set of random variables $(\zeta_n/g(n) : n \geq 1)$ is tight, i.e., for all $\varepsilon > 0$ there exists an $M > 0$ such that $P(|\zeta_n/g(n)| > M) \leq \varepsilon$ for all $n \geq 1$. We will see that in these examples, the sample-path functions exhibit many discontinuities, but the size of those discontinuities shrinks as the sample size n grows. Moreover, any point in the domain will typically have a discontinuity within a distance that is $O_p(n^{-1})$.

This is an important observation in the context of, e.g., trust region methods using CRN, since one should ensure that the solutions on which one builds the metamodel should be spaced to "step over" some number of these discontinuities to avoid being misled by very local curvature that is not indicative of the overall shape of the function. These observations suggest that the neighborhood should have a radius that is of larger order than n^{-1}. The central issue is precisely the same as that which leads to derivative estimators such as those obtained from infinitesimal perturbation analysis (IPA) being biased. In that case, one can think of the discontinuities in the sample-path function as being "corrections" to account for the bias in the gradient estimators. Closely related ideas are explored from a different perspective in Eckman and Henderson (2020).

4.1 Bus Departures

This is a standard example; see, e.g., Ross (1996), p. 68 for the derivation of the exact function value and Kim et al. (2015) for a closely related discussion. Passengers arrive to a bus terminal over the interval $[0,1]$ according to a Poisson process with constant rate λ. One bus is scheduled to depart at time 1 and we are selecting the time, x, at which to schedule a second bus to depart. Both buses have infinite capacity. We want to choose x to minimize the expected value of the sum of the expected waiting times of passengers, where the wait for a passenger begins when they arrive and ends when their bus leaves. The order-statistic property allows us to conclude, as in Ross (1996), that

$$f(x) = \frac{\lambda}{2}(x^2 + (1-x)^2),$$

so that $f(\cdot)$ is smooth and convex.

As for the sample-path functions, let $\xi = (N, T_1, T_2, \ldots, T_N)$ include the random number N of passenger arrivals over $[0,1]$ and the ordered arrival times of those N passengers. Define $\mathbb{I}(\cdot)$ to equal 1 if its argument is true and 0 otherwise. The sample-path function, $f(x,\xi)$, for a single realization of ξ can be written as

$$\sum_{i=1}^{N} [x\mathbb{I}(T_i \leq x) + \mathbb{I}(T_i > x) - T_i].$$

The behavior of the sample-path function $f(x,\xi)$ when viewed as a function of x is interesting; see the left panel of Figure 2. The function is piecewise linear with jumps at each arrival time $T_i, i = 1,2,\ldots,N$. Within the interval $[T_i, T_{i+1})$ the function increases with slope equal to i, $i = 0,1,2,\ldots,N$, where, for convenience, we define $T_0 = 0$ and $T_{N+1} = 1$. The jump at time T_i is a decrease of size $1 - T_i$, for $i = 1,\ldots,N$, because the passenger arriving at time T_i has to wait till time 1 when the bus departs just before their arrival, but leaves immediately when the bus departs at their arrival time.

Now consider what happens when we compute the sample-path function averaged over n iid replications of ξ, as in Figure 2. Again there is a discontinuity of size $-(1-x)/n$ at all points x at which a passenger arrives on one of the n simulated paths, and at points x in between passenger arrivals on any of the sample paths the function is linear with slope $N_n(x)/n$, where $N_n(x)$ is the number of passengers who arrive by time x over all n realizations. Thus, there is a jump discontinuity of size $O_p(1/n)$ at the time of any passenger arrival over the n paths. The number of such jumps has a Poisson(λn) distribution which is $O_p(n)$, and the individual passenger arrival times are uniformly distributed over $[0,1]$. Thus, the distance from any arbitrary point to the nearest discontinuity is $O_p(n^{-1})$.

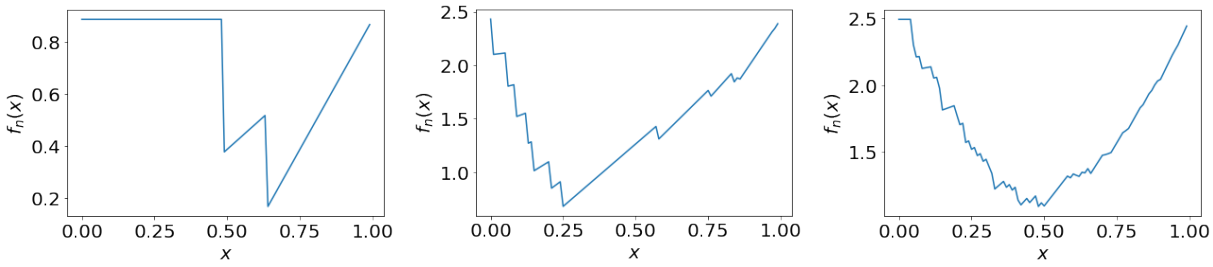


Figure 2: Sample-path function in the bus-scheduling problem for $n = 1, 3, 10$ with $\lambda = 5$.

4.2 Ambulances

The following example has only a tenuous connection to the reality of ambulance simulation, but the name is at least suggestive. Calls for ambulance service arise according to a Poisson process at constant rate λ calls per hour (h). Call locations are uniformly distributed in the square $[0,20]^2$ where distances are measured in kilometers (km). Two ambulances serve this demand. When a call is received, the closer available ambulance travels to the call at a speed of s km/h, traveling in a straight line. If both ambulances are unavailable at the time of the call, the call is queued and calls are then served in first-in-first-out order. Ambulances spend a random amount of time at the scene of the call. Scene times are assumed to be iid. After the scene time is complete the ambulance returns to its base, being instantaneously redirected to the next call if the next call is received before the ambulance reaches its base. Upon arrival at the base it becomes available to handle additional calls. Ambulance 1 is based at the point $(15,15)$, while the position of Ambulance 2, (x_1,x_2) is to be selected so as to minimize the expected value of the sum of the response times over all calls received over a 24 hour period. The response time of a call is the length of the time interval from when the call arrives till an ambulance arrives at scene. I have used very similar examples in previous articles, most recently in Eckman and Henderson (2020).

If we use just one ambulance in this problem, then the sample-path functions are continuous. The sample-path behavior of this model with two ambulances is much more interesting. In essence, we get discontinuities in the sample-path functions whenever 1) both ambulances are available when a call is received; 2) the location of the second ambulance base is such that both ambulances are the same travel distance from the call; and 3) The next call arises before the current call is completed and the responding ambulance can return to base. With reference to the left panel of Figure 3, the orange point is the location of Base 1 at (15, 15). Assume that both ambulances are available at their bases when a call is received at the plus sign $(+)$. If Base 2 lies on the indicated circle, then both bases are equidistant from the call, and by perturbing the location of Base 2 we can change the choice of ambulance that will attend the call. Suppose Ambulance 2 attends the call, and while that ambulance is busy, a second call is received that is closer to Base 2 than to Base 1. Then, since Ambulance 2 is unavailable, Ambulance 1 responds to the new call. If we had instead perturbed Base 2 so that Ambulance 1 responded to the first call, then Ambulance 2 would attend to the second call. In these two perturbed situations, the response time for the second call is different almost surely, since with probability one the second call is not equidistant from both bases. Any further calls that are received during this "busy period" may also have different response times, and the cumulative effect of the difference in response times registers as a discontinuity in the sample function. The size of this discontinuity is bounded by the sum of the potential change in response times within the replication, which is $O_p(1)$, i.e., it is stochastically bounded. When we then average over n replications, the ensuing discontinuities in the sample-path function are thus $O_p(n^{-1})$, just as we saw with the bus-scheduling example. Away from the set of discontinuities, the sample-path function is smooth.

This discussion should make clear that discontinuities will arise in the sample function on a union of circle circumferences or portions thereof within the square; see Figure 3. The number of such circumferences is bounded above by the number of calls that are received on the sample path. Thus, the set of discontinuities

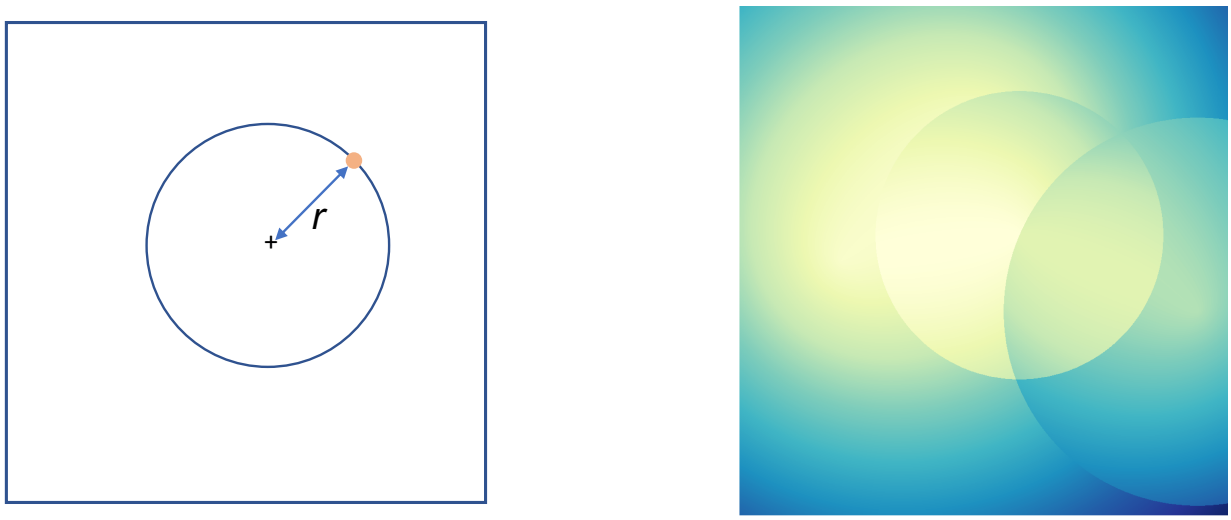


Figure 3: Left: The fixed ambulance base is located at the orange dot at the point (15, 15). A call arises at the location of the plus sign (+) at a distance r from the fixed base. Discontinuities in the sample path function can then arise anywhere on the circumference of the circle centered at the call location with radius r. Right: Heat plot of a sample-path function involving 4 calls, where the first and third calls arrive when both ambulances are available.

is at most a countable union of circle circumferences and has measure 0, just as we saw in the bus-scheduling example. Moreover, the conditions for a discontinuity to arise have positive probability in any replication, so with n replications we will see $O_p(n)$ circles on which discontinuities will arise. These circles are centered on call locations and have radius equal to the distance between the call location and the fixed ambulance base. The union of all such circles completely covers the square, so the discontinuities that arise will become dense in the square as $n \to \infty$, just as we saw in the bus-scheduling example. Finally, the distance from any fixed point $\tilde{x}$ in the square to a discontinuity is $O_p(n^{-1})$. To see why, note that discontinuities arise on a circle under the 3 conditions mentioned above. Such a circle will intersect the ball of radius δ centered at $\tilde{x}$ provided that the location of the initial call mentioned in the 3 conditions falls on the union of the perpendicular bisectors of the line segments that join the fixed base to points in the ball centered at $\tilde{x}$ with radius δ. This event has positive probability that is of order δ, and therefore $O_p(1/\delta)$ replications are required to get a single instance. Taking $n = 1/\delta$ yields the observation.

Thus, the ambulance example shares features with the bus-scheduling example. We use only two ambulances in this example because we can then visualize the underlying geometry; with more ambulances we expect similar sample-path behavior but visualization is difficult owing to the number of dimensions.

These two examples do not prove a general principle, but they are at least suggestive that we might expect similar behavior of sample-path functions in general, namely that discontinuities have magnitude $O_p(n^{-1})$, are at a distance at most $O_p(n^{-1})$ from any fixed point in the domain and thus become dense in the domain as $n \to \infty$, and away from the set of discontinuities the sample-path functions are piecewise smooth. Such sample-path functions are highly irregular on a fine scale. Nevertheless, the law of large numbers ensures that they converge to $f(x)$ at any point x in the domain almost surely as $n \to \infty$, so assuming $f(\cdot)$ is reasonably well structured, e.g., smooth, then the overall "shape" of the sample-path functions cannot be too poor. This suggests that trust-region methods are a suitable tool for optimizing such functions. In this setting, and assuming one uses CRN in fitting the metamodel in each iteration, one needs to exercise caution if the trust-region radius shrinks to $O_p(n^{-1})$, since then there is a nontrivial chance that the metamodel is

being fit to the (very) local structure of the sample-path function that is potentially a poor indicator of the true nature of the function $f(\cdot)$.

It is worthwhile placing the above observations in context by pointing out that the error in the point estimate of $f(x)$ will typically be $O_p(n^{-1/2})$ as assured by the central limit theorem under a finite second moment assumption. Such an error is large compared to the sizes of the discontinuities we have discussed, which are $O_p(n^{-1})$, at least for a large enough sample size n. Accordingly, the discontinuities we have mentioned are, for large enough n, small compared to the overall error in the estimated function values. Yet it is still important to be aware of this sample-path behavior for the reasons we have discussed.

5 MODELING CONSIDERATIONS

Many simulation users first build a simulation model and only afterwards consider the use of optimization. This can yield highly intractable SO problems with little structure to be exploited by SO solvers. This observation partially explains the predominance of SO solvers that do not require problem structure.

I instead advocate building a succession of models, analyzing each model in detail including optimization as appropriate before moving on to a more complex model. This approach has a number of advantages over "one and done" modeling where a single model is designed and built. First, a succession of models allows for each model's predictions to be compared against those of earlier models, which helps verify model implementation and adds insight. Second, one might be able to stop modeling at an early stage before building more complex models that can be expensive to build and prone to error. Third, one always has a model on hand that can provide answers to the questions that originally motivated the modeling effort, which is a comfort as deadlines approach and more complex models are yet to be completed.

A criticism of this approach is that one may need to perform more modeling and coding effort. This concern is valid, but consider that if successive model complexity grows rapidly, then the early models may not represent much of an overhead. For example, if the time to build each successive model grows exponentially, then the time needed to generate the final model dominates the total time needed to generate all previous models.

The idea of building a sequence of models has been espoused many times, so in some sense I am merely echoing the perspective of others. What may be new here is that I advocate thinking about optimization with every model that is built. Perhaps the principles are clearest through an example, which is only sketched due to space concerns.

In bike sharing, important quantities include the target inventory levels of bikes in stations at the start of each day, and, perhaps annually, the number of racks to place at each station. One can begin using deterministic flow models at each individual station, ignoring station interactions, where the flow rates are time dependent. A next model could make those flows stochastic, yielding independent queueing models at each station. The next round of modeling can incorporate station interactions through network fluid models, and perhaps a final step can move to full simulation of a network of bike stations. These models represent progressive increases in model complexity and vary in terms of convexity properties and tractability of optimization. The results from each model can be interpreted in light of the results from the others (Freund et al. 2019; Freund et al. 2018; Freund et al. 2021; Jian et al. 2016; Jian and Henderson 2015). Space constraints prevent a full discussion.

ACKNOWLEDGMENTS

My thanks to colleagues who have co-authored related work or influenced my thinking in other ways; they are too numerous to mention individually. Special thanks to Russell Barton and Uday Shanbhag for an invitation to present a tutorial on this topic at the 2021 Simulation Society Workshop in State College, PA. This work was supported in part by National Science Foundation grants TRIPODS+X DMS-1839346 and CMMI 2035086.

REFERENCES

Ali, M. M., C. Khompatraporn, and Z. B. Zabinsky. 2005. "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems". *Journal of Global Optimization* 31(4):635–672.

Asmussen, S., and P. W. Glynn. 2007. *Stochastic Simulation: Algorithms and Analysis*, Volume 57 of *Stochastic Modeling and Applied Probability*. New York: Springer.

Beiranvand, V., W. Hare, and Y. Lucet. 2017. "Best practices for comparing optimization algorithms". *Optimization and Engineering* 18(4):815–848.

Blanchet, J., C. Cartis, M. Menickelly, and K. Scheinberg. 2019. "Convergence Rate Analysis of a Stochastic Trust-Region Method via Supermartingales". *INFORMS Journal on Optimization* 1(2):92–119.

Bottou, L., F. E. Curtis, and J. Nocedal. 2018. "Optimization methods for large-scale machine learning". *SIAM Review* 60(2):223–311.

Chang, K. H., L. J. Hong, and H. Wan. 2007. "Stochastic Trust Region Gradient-free Method (STRONG)- A New Response-Surface-Based Algorithm in Simulation Optimization". In *Proceedings of the 2007 Winter Simulation Conference*, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 346–354. Piscataway, New Jersey: IEEE.

Deng, G., and M. C. Ferris. 2006. "Adaptation of the UOBYQA algorithm for noisy functions". In *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, B. Lawson, J. Liu, and F. Wieland, 312–319. IEEE.

Dolan, E. D., and J. J. Moré. 2002. "Benchmarking optimization software with performance profiles". *Mathematical Programming* 91(2):201–213.

Eckman, D. J., and S. G. Henderson. 2020. "Biased gradient estimators in simulation optimization". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 2935–2946. Piscataway NJ: IEEE.

Eckman, D. J., S. G. Henderson, and R. Pasupathy. 2019. "Redesigning a testbed of simulation-optimization problems and solvers for experimental comparisons". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee, K.-H. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 3457–3467. Piscataway NJ: IEEE.

Eckman, D. J., S. G. Henderson, and S. Shashaani. 2021a. "Evaluating and comparing simulation optimization algorithms". Working paper.

Eckman, D. J., S. G. Henderson, and S. Shashaani. 2021b. "SimOpt: Software for a Testbed of Simulation Optimization Problems and Solvers". Working paper.

Freund, D., S. G. Henderson, E. O'Mahony, and D. B. Shmoys. 2019. "Analytics and bikes: riding tandem with Motivate to improve mobility". *INFORMS Journal on Applied Analytics* 49(5):310–323.

Freund, D., S. G. Henderson, and D. B. Shmoys. 2018. "Minimizing multimodular functions and allocating capacity in bike-sharing systems". *Production and Operations Management* 27(12):2346–2349. Extended abstract.

Freund, D., S. G. Henderson, and D. B. Shmoys. 2021. "Minimizing multimodular functions and allocating capacity in bike-sharing systems". Submitted.

Fu, M. 2014. *Handbook of Simulation Optimization*. International Series in Operations Research & Management Science. Springer New York.

Fu, M. C., and S. G. Henderson. 2017. "History of seeking better solutions, AKA simulation optimization". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, 131–157. Piscataway NJ: IEEE.

Fu, M. C., and J.-Q. Hu. 1997. *Conditional Monte Carlo: Gradient Estimation and Optimization Applications*. Boston: Kluwer.

Glasserman, P. 1991. *Gradient Estimation Via Perturbation Analysis*. The Netherlands: Kluwer.

Gould, N., and J. Scott. 2016. "A note on performance profiles for benchmarking software". *ACM Transactions on Mathematical Software (TOMS)* 43(2):1–5.

Hong, L. J., and B. L. Nelson. 2006. "Discrete Optimization via Simulation Using COMPASS". *Operations Research* 54(1):115–129.

Jian, N., D. Freund, H. Wiberg, and S. G. Henderson. 2016. "Simulation optimization for a large-scale bike-sharing system". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. K. Roeder, P. I. Frazier, R. Szechtman, and E. Zhou, 602–613. Piscataway NJ: IEEE.

Jian, N., and S. G. Henderson. 2015. "An introduction to simulation optimization". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, T. M. K. Roeder, C. Macal, and M. Rosetti, 1780–1794. Piscataway NJ: IEEE.

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, Volume 216 of *International Series in Operations Research & Management Science*, Chapter 8, 207–244. Springer New York.

L'Ecuyer, P., R. Simard, E. J. Chen, and W. D. Kelton. 2002. "An Objected-Oriented Random-Number Package with Many Long Streams and Substreams". *Operations Research* 50(6):1073–1075.

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* 24:47–56.

Meyn, S. P., and R. L. Tweedie. 1993. *Markov Chains and Stochastic Stability*. London: Springer-Verlag.

Moré, J. J., and S. M. Wild. 2009. "Benchmarking derivative-free optimization algorithms". *SIAM Journal on Optimization* 20(1):172–191.

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.

Newton, D., R. Pasupathy, and F. Yousefian. 2018. "Recent trends in stochastic gradient descent for machine learning and big data". In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 366–380. Piscataway NJ: IEEE.

Pasupathy, R., and S. G. Henderson. 2006. "A testbed of simulation-optimization problems". In *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 255–263. Piscataway NJ: IEEE.

Pasupathy, R., and S. G. Henderson. 2011a. *SimOpt*. http://www.simopt.org. Accessed 4/2/2021.

Pasupathy, R., and S. G. Henderson. 2011b. "SimOpt : A library of simulation optimization problems". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 4080–4090. Piscataway NJ: IEEE.

Ragavan, P. K., S. R. Hunter, R. Pasupathy, and M. R. Taaffe. 2021. "Adaptive Sampling Line Search for Stochastic Optimization with Integer Variables". Manuscript.

Ross, S. M. 1996. *Stochastic Processes*. 2nd ed. New York: Wiley.

Rubinstein, R. Y., and A. Shapiro. 1990. "Optimization of static simulation models by the score function method". *Mathematics and Computers in Simulation* 32:373–392.

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.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 60–74. Piscataway NJ: IEEE.

Shapiro, A. 2003. "Monte Carlo sampling methods". In *Stochastic Programming*, edited by A. Ruszczynski and A. Shapiro, Handbooks in Operations Research and Management Science, 353–425. Amsterdam: Elsevier.

Shashaani, S., F. S. Hashemi, and R. Pasupathy. 2018. "ASTRO-DF: A class of adaptive sampling trust-region algorithms for derivative-free simulation optimization". *SIAM Journal on Optimization* 28(4):3145–3176.

Shashaani, S., S. R. Hunter, and R. Pasupathy. 2016. "ASTRO-DF: Adaptive Sampling Trust-Region Optimization Algorithms, Heuristics, and Numerical Experience.". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, 554–565. Piscataway, NJ: IEEE.

Sutter, T., B. Van Parys, and D. Kuhn. 2021. "A General Framework for Optimal Data-Driven Optimization". Manuscript.

Wright, S. J., and B. Recht. 2021. *Optimization for Data Analysis*. Cambridge University Press.

AUTHOR BIOGRAPHIES

SHANE G. HENDERSON holds the Charles W. Lake, Jr. Chair in Productivity in the School of Operations Research and Information Engineering at Cornell University. His research interests include a range of topics in discrete-event simulation and simulation optimization, and he has worked for some time with emergency services, bike-sharing and other transportation applications. Recently he has worked intensively on modeling efforts to support pandemic decision making by Cornell University leadership. He is a co-creator of SimOpt, a testbed of simulation optimization problems and solvers. He co-edited the Proceedings of the 2007 Winter Simulation Conference. He is an INFORMS Fellow. His web page is http://people.orie.cornell.edu/shane. and his email address is sgh9@cornell.edu.