Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

R-MGSPLINE: RETROSPECTIVE MULTI-GRADIENT SEARCH FOR MULTI-OBJECTIVE SIMULATION OPTIMIZATION ON INTEGER LATTICES

Eric A. Applegate
Susan R. Hunter

School of Industrial Engineering
Purdue University
315 N. Grant St.
West Lafayette, IN 47906, USA

ABSTRACT

We introduce the R-MGSPLINE (Retrospective Multi-Gradient Search with Piecewise Linear Interpolation and Neighborhood Enumeration) algorithm for finding a local efficient point when solving a multi-objective simulation optimization problem on an integer lattice. In this nonlinear optimization problem, each objective can only be observed with stochastic error and the decision variables are integer-valued. R-MGSPLINE uses a retrospective approximation (RA) framework to repeatedly call the MGSPLINE sample-path solver at a sequence of increasing sample sizes, using the solution from the previous RA iteration as a warm start for the current RA iteration. The MGSPLINE algorithm performs a line search along a common descent direction constructed from pseudo-gradients of each objective, followed by a neighborhood enumeration for certification. Numerical experiments demonstrate R-MGSPLINE's empirical convergence to a local weakly efficient point.

1 INTRODUCTION

A simulation optimization (SO) problem is an optimization problem in which the objective functions are expectations whose values can only be observed with stochastic error (Fu 2015). Much of the research in this area over the past thirty years has focused on single-objective SO problems in various types of decision spaces including categorical, integer-ordered, and continuous (Pasupathy and Henderson 2011; Pasupathy and Henderson 2006). Recently, there has been increased interest in algorithms that can be used for multi-objective simulation optimization (MOSO) problems (Fu et al. 2014). In MOSO problems, the multiple simultaneous objectives are often conflicting such that no single feasible point simultaneously minimizes all objectives. Rather, multiple feasible points may be *efficient*, that is, no other feasible point maps to an objective vector that is at least as small on all objectives, and strictly smaller on at least one objective. The collection of all efficient points is called the *efficient set*, which is a solution to the MOSO problem. MOSO problems arise in a variety of areas including spare parts allocation in aviation (Li et al. 2015), facility design and patient flow in healthcare (Wang et al. 2015), and flood control operations (Prakash et al. 2015). Hunter et al. (2019) provide an introduction to the MOSO problem and survey existing methods to solve the MOSO problem in various contexts.

We consider the d-objective MOSO problem in the context of a q-dimensional integer-ordered feasible space:

$$\text{Problem } M\colon\ \min_{\mathbf{x}\in\mathcal{X}}\ \{\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_d(\mathbf{x})) := (\mathbb{E}[G_1(\mathbf{x},\xi)], \ldots, \mathbb{E}[G_d(\mathbf{x},\xi)])\}$$

where $\mathbf{g}: \mathcal{X} \to \mathbb{R}^d$ is an unknown vector-valued function, the nonempty feasible set $\mathcal{X} \subseteq \mathbb{Z}^q$ is a subset of an integer lattice, and ξ is a random variable. Like Pasupathy and Henderson (2011) and Pasupathy and

Henderson (2006), we refer to this type of feasible set as *integer-ordered*. We define any deterministic constraints in Problem M through the feasible set $\mathcal{X}$ and note that constraints may be *hidden* (Digabel and Wild 2015).

We develop a search algorithm called R-MGSPLINE that, given an initial feasible point, finds a *local weakly efficient point* (defined in Section 1.2). The acronym R-MGSPLINE stands for Retrospective Multi-Gradient SPLINE, where as in Wang et al. (2013), SPLINE stands for Search with Piecewise Linear Interpolation (SPLI) and Neighborhood Enumeration (NE). The algorithm treats the objectives simultaneously, without scalarization, and seeks a local efficient point by performing line searches along common descent directions. We provide an overview of each piece of the algorithm below.

The R in R-MGSPLINE refers to retrospective approximation (RA). RA is a version of the Sample Average Approximation (SAA) framework (see, e.g., Pasupathy and Ghosh 2013) that replaces the unknown functions $\mathbf{g}(\cdot)$ in Problem M with their estimators, resulting in the *sample-path problem*

$$\text{Problem } \overline{M}_n\colon \ \min_{\mathbf{x}\in\mathcal{X}} \ \{\bar{\boldsymbol{G}}_n(\mathbf{x}) = (\bar{G}_{1,n}(\mathbf{x}),\ldots,\bar{G}_{d,n}(\mathbf{x})) := (\tfrac{1}{n}\textstyle\sum_{i=1}^{n} G_1(\mathbf{x},\xi_i),\ldots,\tfrac{1}{n}\sum_{i=1}^{n} G_d(\mathbf{x},\xi_i))\}$$

where $\bar{\boldsymbol{G}}_n(\mathbf{x})$ is an estimator of $\mathbf{g}(\mathbf{x})$, and at each feasible point visited by the algorithm, $\mathbf{x}\in\mathcal{X}\subseteq\mathbb{Z}^q$, the oracle generates n copies of the random objective vector $\boldsymbol{G}(\mathbf{x},\xi_i) := (G_1(\mathbf{x},\xi_i),\ldots,G_d(\mathbf{x},\xi_i))$ for all $i=1,\ldots,n$. The RA framework involves solving a sequence of sample-path problems with an increasing sample size sequence $\{m_\nu,\ \nu=1,2,\ldots\}$, where ν is the RA iteration number. The sample-path local efficient point found in RA iteration $\nu-1$ is used as a warm start for RA iteration ν. As the sample size increases, the warm-starts are likely to improve, which ensures that large sample sizes are not wasted on suboptimal points.

Within an RA iteration ν, the Multi-Gradient SPLINE (MGSPLINE) algorithm obtains a sample-path local efficient point for Problem $\overline{M}_{m_\nu}$. The algorithm first determines a common descent direction from individual objective pseudo-gradients. Then, a line search is performed along the descent direction with a series of increasing step sizes to find a better feasible point. A neighborhood enumeration checks whether the resulting feasible point is a sample-path local efficient point. If so, the MGSPLINE algorithm terminates and the current RA iteration ends. If not, the algorithm returns to the multi-gradient line search step. We conjecture that the sample-path local efficient point returned by R-MGSPLINE will converge into the set of local weakly efficient points (see Section 1.2) as the number of RA iterations goes to infinity, under regularity conditions similar to those required by Cooper et al. (2019).

We remark here that any particular local efficient point may or may not belong to a local efficient set (Cooper et al. 2019). However, an algorithm that finds local efficient points can be embedded in another algorithm that locates many local efficient points to construct a local efficient set, or used within some global search routine to locate a member of the global efficient set. Therefore we view our algorithm as something of a building-block procedure which is likely to be useful within the context of another algorithm.

1.1 Related Work

In this section, we briefly discuss prior work that is related to our development of R-MGSPLINE. We follow categorization concepts from Hunter et al. (2019) and limit our scope to non-scalarization methods in which the decision-maker does not provide any preference or interactive input when $\mathcal{X}$ is integer-ordered. In particular, we mention work from both the SO literature and the deterministic multi-objective optimization (MOO) literature that pertains to gradient-based line searches, neighborhood enumerations, and integer-ordered decision spaces.

Our algorithm is primarily based on the prior work of Wang et al. (2013) and Fliege and Svaiter (2000). In Wang et al. (2013), the authors develop an RA algorithm called R-SPLINE that uses pseudo-gradients, line search, and neighborhood enumeration to solve an integer-ordered single-objective SO problem. Fliege and Svaiter (2000) examine how to construct a common descent direction from multiple gradients in a continuous MOO setting. By incorporating a common descent direction into the framework of R-SPLINE, our algorithm can be considered a multi-objective version of R-SPLINE that is able to search through a decision space to find one local efficient point. We know of no other MOSO algorithms that accomplish this same goal.

We note that Mercier et al. (2018) develop a MOSO multi-gradient line search technique called SMGDA for finding an efficient point in a continuous decision space. They determine a common descent direction based on their earlier work (Désidéri 2012) that mirrors Fliege and Svaiter (2000). SMGDA requires that analytic expressions for the gradient are available.

Finally, there are MOSO algorithms whose goal is to find an entire local or global efficient set in integer-ordered decision spaces. R-PERLE and R-MinRLE (Cooper et al. 2019) are RA algorithms for finding a local efficient set; R-PERLE is based on the ε-constraint method (see, e.g., Miettinen 1999) and R-MinRLE is a benchmark algorithm. MO-COMPASS (Li et al. 2015) is a multi-objective version of the popular single-objective algorithm COMPASS (Hong and Nelson 2006; Xu et al. 2010) for finding a local efficient set. MO-COMPASS iteratively updates a Most Promising Area by using a Simulation Allocation Rule to efficiently allocate simulation effort. MOPBnB (Huang and Zabinsky 2014) is a multi-objective probabilistic branch and bound algorithm that finds the *global* efficient set. Similarly, multi-objective ranking and selection (MORS) algorithms such as MOCBA (Lee et al. 2010), MO-SCORE (Feldman and Hunter 2018; Applegate et al. 2019), and M-MOBA (Branke and Zhang 2015; Branke et al. 2016) find a global efficient set when the feasible space is finite and the decision variables may be categorical.

We remark here that the literature on deterministic MOO is vast (Ehrgott 2006; Ehrgott and Gandibleux 2000). Custódio et al. (2012) provide a brief survey on direct search and evolutionary methods for multi-objective derivative-free deterministic continuous optimization. In particular, they discuss the direct search method called Direct Multi-Search (DMS) developed in Custódio et al. (2011). DMS involves an optional "search" step followed by a "poll" step which is similar to our neighborhood enumeration. We also recognize that Peitz and Dellnitz (2018) examine multi-objective optimization with inexact gradients as part of a subdivision algorithm similar to MOPBnB. However, the assumptions they make about error bounds on function and gradient values do not align with our problem context.

1.2 Notation and Terminology

With few exceptions, constants are denoted by lower-case letters (a), random variables by capital letters (X), sets by script capital letters ($\mathcal{A}$), vectors by bold ($\mathbf{x}$), random vectors by capital bold ($\boldsymbol{X}$), and operators by blackboard bold ($\mathbb{E}[X]$). We work in a decision space of q-dimensional integer-valued vectors $\mathbb{Z}^q \subset \mathbb{R}^q$ and the image space of all d-dimensional extended real-valued vectors $\overline{\mathbb{R}}^d$. The d-dimensional vector $(0,0,\ldots,0)$ is denoted $\mathbf{0}_d$, the d-dimensional vector $(1,1,\ldots,1)$ is denoted as $\mathbf{1}_d$, and the d-dimensional vector $(\infty,\infty,\ldots,\infty)$ is denoted as $\boldsymbol{\infty}_d$. If $\mathbf{x} = (x_1,x_2,\ldots,x_q)$ is a $1\times q$ vector, then the L_2 norm of $\mathbf{x}$ is defined by $\|\mathbf{x}\| = (x_1^2+x_2^2+\cdots+x_q^2)^{1/2}$. The q-dimensional vector with its ith component equal to 1 is denoted by $e_i = (0,\ldots,1,\ldots,0)$. If D is a set, then $|D|$ is the cardinality of D.

1.3 Optimality Concepts

In this section, we adopt concepts of domination and optimality from Hunter et al. (2019) and Cooper et al. (2019). We illustrate optimality concepts in Figure 1.

To define concepts of optimality in Problem M, first, we define the concept of dominance in the objective function space.

Definition 1 Let $\mathbf{x}_1,\mathbf{x}_2 \in \mathcal{X}$ and $d \geq 2$. For vectors $\mathbf{g}(\mathbf{x}_1)$ and $\mathbf{g}(\mathbf{x}_2)$, we say that

1. $\mathbf{g}(\mathbf{x}_1)$ *weakly dominates* $\mathbf{g}(\mathbf{x}_2)$, written as $\mathbf{g}(\mathbf{x}_1) \leqq \mathbf{g}(\mathbf{x}_2)$, if $g_k(\mathbf{x}_1) \leq g_k(\mathbf{x}_2)$ for all $k = 1,\ldots,d$.
2. $\mathbf{g}(\mathbf{x}_1)$ *dominates* $\mathbf{g}(\mathbf{x}_2)$, written as $\mathbf{g}(\mathbf{x}_1) \leq \mathbf{g}(\mathbf{x}_2)$, if $\mathbf{g}(\mathbf{x}_1) \leqq \mathbf{g}(\mathbf{x}_2)$ and $\mathbf{g}(\mathbf{x}_1) \neq \mathbf{g}(\mathbf{x}_2)$.
3. $\mathbf{g}(\mathbf{x}_1)$ *strictly dominates* $\mathbf{g}(\mathbf{x}_2)$, written as $\mathbf{g}(\mathbf{x}_1) < \mathbf{g}(\mathbf{x}_2)$, if $g_k(\mathbf{x}_1) < g_k(\mathbf{x}_2)$ for all $k = 1,\ldots,d$.

Because we are working in $\mathcal{X} \subseteq \mathbb{Z}^q$, a neighborhood structure can be defined for the feasible points. As in Wang et al. (2013), we define a flexible neighborhood structure based on Euclidean distance for use in the definitions of local optimality.

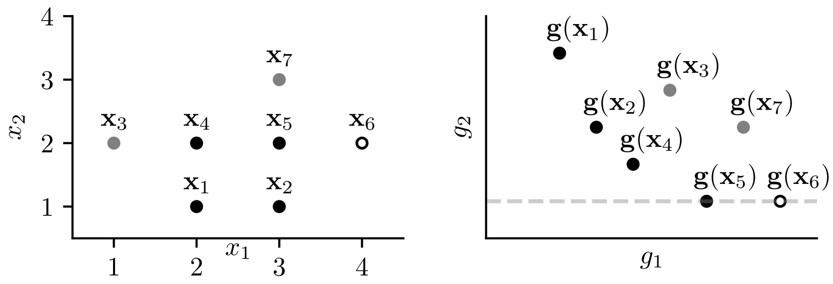

Figure 1: The left panel shows example feasible points in a decision space, and the right panel shows the corresponding images in the objective space. The points $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4, \mathbf{x}_5\}$ are local efficient points that make up the global efficient set. The point $\mathbf{x}_6$ is a local weakly efficient point and a global weakly efficient point. The points $\mathbf{g}(\mathbf{x}_3)$ and $\mathbf{g}(\mathbf{x}_7)$ are dominated. This figure was adapted from Cooper et al. (2019).

Definition 2 For $a \geq 0$, the $\mathcal{N}_a$-neighborhood of the point $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{Z}^q$ is $\mathcal{N}_a(\mathbf{x}) := \{\mathbf{x}' \in \mathbb{Z}^q \colon ||\mathbf{x} - \mathbf{x}'|| \leq a\}$.

In this paper, we focus on the 1-unit neighborhood of $\mathbf{x}$, denoted $\mathcal{N}_1(\mathbf{x})$, which is comprised of the $2q+1$ integer points that lie within one unit of $\mathbf{x}$. Henceforth for readability, we drop the subscript. Given this neighborhood defined in the decision space, we now define local optimality concepts.

Definition 3 Let $\mathbf{x}^* \in \mathcal{X}$. We say $\mathbf{x}^*$ is

1. a *local efficient point* on $\mathcal{N}$ if there does not exist $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*) \cap \mathcal{X}$ such that $\mathbf{g}(\mathbf{x}) \leq \mathbf{g}(\mathbf{x}^*)$.
2. a *local weakly efficient point* on $\mathcal{N}$ if there does not exist $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*) \cap \mathcal{X}$ such that $\mathbf{g}(\mathbf{x}) < \mathbf{g}(\mathbf{x}^*)$.

Local Pareto points and *local weakly Pareto points* are the images of local efficient points and local weakly efficient points, respectively. In Section 5 we refer to the *global efficient set*, which is the set of all local efficient points when the neighborhood radius is infinity. We denote the set of all local weakly efficient points as $\mathcal{X}^{\mathrm{w}}$, which, by definition, is a superset of the global efficient set. In Section 5 we also refer to a *local weakly efficient set*, $\mathcal{W}$, which is a collection of local weakly efficient points such that no points in the set have images that strictly dominate the images of other points in the set, and the image of each neighborhood point not in the set is dominated by the image of a point in the set.

Since we work with the sample-path Problem $\overline{M}_n$, we define *sample-path* versions of the above optimality concepts by replacing the objective function values $\mathbf{g}(\mathbf{x})$ and $g_k(\mathbf{x})$ with $\bar{\boldsymbol{G}}_n(\mathbf{x})$ and $\bar{G}_{k,n}(\mathbf{x})$, respectively, $k \in \{1, \ldots, d\}$. A local solution to Problem $\overline{M}_n$ is a sample-path local weakly efficient point, denoted $\boldsymbol{X}^*$.

1.4 Problem Statement

We consider the following problem statement: Given a simulation oracle capable of producing estimators $\bar{\boldsymbol{G}}_n(\mathbf{x})$ of $\mathbf{g}(\mathbf{x})$ such that $\bar{\boldsymbol{G}}_n(\mathbf{x}) \to \mathbf{g}(\mathbf{x})$ w.p.1 as the sampling effort $n \to \infty$ for each $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{Z}^q$, find a local efficient point for Problem M. We note that the simulation oracle defines both the function $\mathbf{g}$ and the feasible region $\mathcal{X}$. Thus, if $\mathbf{x} \notin \mathcal{X}$, we assume the oracle will return an infeasible indicator.

2 DETERMINING A COMMON DESCENT DIRECTION

We now discuss details surrounding finding a common descent direction in MGSPLINE. For discussion, in this section only, suppose $\mathbf{g} : \mathbb{R}^q \to \mathbb{R}^d$ is a vector-valued objective function where g_k is continuously differentiable for each $k \in \{1, \ldots, d\}$. We further suppose that there are no constraints, so that $\mathcal{X} = \mathbb{R}^q$. Let $\nabla g_k(\mathbf{x})$ denote the gradient of the kth objective at $\mathbf{x}$ for each $k \in \{1, \ldots, d\}$ and each $\mathbf{x} \in \mathbb{R}^q$. Then let

$\nabla \mathbf{g}(\mathbf{x})$ denote the transpose of the Jacobian matrix at $\mathbf{x}$; that is,

$$\nabla \mathbf{g}(\mathbf{x}) = \begin{pmatrix}\nabla g_1(\mathbf{x}) & \dots & \nabla g_d(\mathbf{x})\end{pmatrix} = \begin{pmatrix} \frac{\partial g_1}{\partial x_1}(\mathbf{x}) & \dots & \frac{\partial g_d}{\partial x_1}(\mathbf{x}) \\ \vdots & & \vdots \\ \frac{\partial g_1}{\partial x_q}(\mathbf{x}) & \dots & \frac{\partial g_d}{\partial x_q}(\mathbf{x}) \end{pmatrix}.$$

Then if our goal is to move from the current point $\mathbf{x}$ in a direction that simultaneously improve all objectives, we wish to find a direction $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_q) \in \mathbb{R}^q$ such that $\nabla g_k(\mathbf{x})^\intercal \boldsymbol{\gamma} < 0$ for each $k \in \{1, \dots, d\}$, assuming such a direction exists; in matrix notation, this implies $\nabla \mathbf{g}(\mathbf{x})^\intercal \boldsymbol{\gamma} < \mathbf{0}_d$.

We follow Fliege and Svaiter (2000), who compute a common descent direction as follows. First, notice that a common descent direction can be found by finding the solution to

$$\text{Problem } C\colon \quad \underset{\boldsymbol{\gamma} \in \mathbb{R}^q}{\text{minimize}} \quad \left(\max\{\nabla g_k(\mathbf{x})^\intercal \boldsymbol{\gamma} \colon k \in \{1, \dots, d\}\}\right) + \frac{1}{2} \|\boldsymbol{\gamma}\|^2.$$

In the case of $d = 1$ objective, solving this problem yields the direction of steepest descent, $\boldsymbol{\gamma} = -\nabla g(\mathbf{x})$. To remove the max operator, we write this problem as a convex quadratic program with linear constraints,

$$\begin{aligned} \text{Problem } D\colon \quad \text{minimize} \quad & \alpha + \frac{1}{2} \|\boldsymbol{\gamma}\|^2 \\ \text{s.t.} \quad & \nabla g_k(\mathbf{x})^\intercal \boldsymbol{\gamma} \le \alpha \quad \text{for all } k = 1, \dots, d; \quad \alpha \in \mathbb{R},\ \boldsymbol{\gamma} \in \mathbb{R}^q. \end{aligned}$$

If a descent direction exists, the optimal value of Problem D is less than zero. We note that solving Problem D is only one way to find a common descent direction, if one exists. We consider finding other common descent directions and evaluating their effects on the resulting algorithm as future research.

To implement these ideas in R-MGSPLINE, we form pseudo-gradients, $\hat{\nabla} G_k(\mathbf{x})$ for each $k \in \{1, \dots, d\}$, collect them into a Jacobian matrix, and take its transpose to form $\hat{\nabla} \boldsymbol{G}(\mathbf{x})$. As in Fliege and Svaiter (2000), for implementation, we solve the dual formulation of an estimated version of Problem D,

$$\begin{aligned} \text{Problem } \hat{D}\colon \quad \text{maximize} \quad & -\frac{1}{2} \left\| \hat{\nabla} \boldsymbol{G}(\mathbf{x}) \boldsymbol{\lambda} \right\|^2 \\ \text{s.t.} \quad & \boldsymbol{\lambda}^T \mathbf{1}_d = 1; \quad \lambda_i \ge 0 \quad \text{for all } k = 1, \dots, d, \end{aligned}$$

and return $\widehat{\boldsymbol{\gamma}} = -\hat{\nabla} \boldsymbol{G}(\mathbf{x}) \boldsymbol{\lambda}^*$ as the common descent direction, where $\boldsymbol{\lambda}^*$ is the solution to Problem $\hat{D}$.

3 ALGORITHM OVERVIEW

Algorithm 1 provides an overview of R-MGSPLINE. In the νth RA iteration, m_ν is the sample size at each visited point in the MGSPLINE function, where common random numbers (CRN) can be used across visited points. The output from MGSPLINE is then used as the starting point for the next iteration. The sample sizes, m_ν, and oracle call limits, b_ν, are increased for each retrospective iteration. We believe that showing R-MGSPLINE converges to a local weakly efficient point as ν increases to infinity requires a straightforward modification of the proofs contained in Cooper et al. (2019).

Algorithm 1: R-MGSPLINE

```
Input: initial point $\mathbf{x}_0 \in \mathcal{X}$; sequence of sample sizes $\{m_\nu\}$; sequence of limits on oracle calls $\{b_\nu\}$
1 Initialize $\boldsymbol{X}_0^* = \mathbf{x}_0$
2 for $\nu = 1, 2, \dots$ with CRN do
3     $[\boldsymbol{X}_\nu^*, \bar{\boldsymbol{G}}_{m_\nu}(\boldsymbol{X}_\nu^*)]$ = MGSPLINE($\boldsymbol{X}_{\nu-1}^*, m_\nu, b_\nu$)
```

4 MULTI-GRADIENT SPLINE (MGSPLINE)

MGSPLINE is a search routine that begins at an input point, $\boldsymbol{X}_{\text{old}}$, and returns a point, $\boldsymbol{X}_{\text{new}}$, such that $\bar{\boldsymbol{G}}_{m_v}(\boldsymbol{X}_{\text{new}}) \leq \bar{\boldsymbol{G}}_{m_v}(\boldsymbol{X}_{\text{old}})$, or possibly $\bar{\boldsymbol{G}}_{m_v}(\boldsymbol{X}_{\text{new}}) = \bar{\boldsymbol{G}}_{m_v}(\boldsymbol{X}_{\text{old}})$. Algorithm 2 displays pseudocode for the MGSPLINE function. MGSPLINE is composed of two subroutines: Multi-Gradient Search with Piecewise Linear Interpolation (MGSPLI) and Neighborhood Enumeration (NE). The MGSPLI function performs a line search along a descent direction formed from individual objective pseudo-gradients near the input point and returns a point whose image is sample-path non-dominated among points encountered during the search, $\boldsymbol{X}_{\text{SPLI}}$. The NE subroutine then searches the neighborhood of $\boldsymbol{X}_{\text{SPLI}}$ to determine if the images of its neighbors sample-path dominate $\boldsymbol{X}_{\text{SPLI}}$. If a neighbor's image sample-path dominates $\boldsymbol{X}_{\text{SPLI}}$, the neighbor is set to $\boldsymbol{X}_{\text{NE}}$ and another line search is performed. This process repeats until no neighbors have images that sample-path dominate $\boldsymbol{X}_{\text{SPLI}}$, or a limit on oracle calls is exceeded.

Algorithm 2: $[\boldsymbol{X}_{\text{new}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{new}})]$ = MGSPLINE($\boldsymbol{X}_{\text{old}}, m, b$)

```
  Input: initial point $\boldsymbol{X}_{\text{old}} \in \mathcal{X}$; sample size $m$; limit on oracle calls $b$
  Output: sample-path local efficient point $\boldsymbol{X}_{\text{new}} \in \mathcal{X}$ and estimated objective vector $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{new}})$
1 Set $n = 0$, $\boldsymbol{X}_{\text{NE}} \leftarrow \boldsymbol{X}_{\text{old}}$
2 repeat
3     $[n', \boldsymbol{X}_{\text{SPLI}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{SPLI}})]$ = MGSPLI($\boldsymbol{X}_{\text{NE}}, m, b$)        /descent direction and line search
4     if $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{NE}}) \leq \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{SPLI}})$ then $\boldsymbol{X}_{\text{SPLI}} \leftarrow \boldsymbol{X}_{\text{NE}}$     /ensure not dominated by input point
5     $[n'', \boldsymbol{X}_{\text{NE}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{NE}})]$ = NE($\boldsymbol{X}_{\text{SPLI}}, m$)                  /neighborhood enumeration
6     Set $n = n + n' + n''$
7 until $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{SPLI}}) = \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{NE}})$ or $n > b$
```

4.1 Multi-Gradient Search with Piecewise Linear Interpolation (MGSPLI)

Algorithm 3 displays the MGSPLI function that determines a common descent direction using subroutine Piecewise Linear Interpolation (PLI) (Algorithm 4) and then performs a line search along the common descent direction. We now explain MGSPLI in detail.

Given an input point $\boldsymbol{X}_{\text{in}} \in \mathcal{X}$, MGSPLI first adds a small random perturbation (smaller than 1 w.p.1) independently to each dimension of $\boldsymbol{X}_{\text{in}}$ using the PERTURB function and passes the resulting non-integer-valued point, $\boldsymbol{X}_{\text{P}}$, to PLI. The PLI subroutine first forms a simplex around the perturbed point (Algorithm 4, Lines 1-5) and estimates objective values at each feasible simplex point, keeping track of which simplex point's image is sample-path non-dominated (Algorithm 4, Lines 5-10). If at least one of the simplex points is infeasible, PLI checks to see whether it can locate a naive common descent direction; if not, it sets the descent direction to *undefined*. Otherwise, the objective values of the simplex points are used to determine pseudo-gradients for each objective (Algorithm 4, line 14), and these pseudo-gradients are assembled into the transposed Jacobian matrix $\hat{\nabla}\boldsymbol{G}(\mathbf{x})$. Problem $\hat{D}$ is solved to determine a common descent direction, $\widehat{\boldsymbol{\gamma}}$. PLI returns the best simplex point and the descent direction to MGSPLI.

Back in MGSPLI (Algorithm 3, Line 6), $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}})$ is compared to the current value of $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$, and $\boldsymbol{X}_{\text{best}}$ may be updated. If PLI did not return a valid descent direction or the oracle call limit has been met, no line search is done and $\boldsymbol{X}_{\text{best}}$ is returned to MGSPLINE as input to NE. Barring no early exit criteria, MGSPLI begins a line search by taking a step from $\boldsymbol{X}_0 = \boldsymbol{X}_{\text{best}}$ along descent direction $\widehat{\boldsymbol{\gamma}}$ which results in a point $\boldsymbol{X}_1$. It is highly likely that $\boldsymbol{X}_1$ will not be an integer-valued point. Thus, $\boldsymbol{X}_1$ is shifted to the nearest integer solution, currently implemented by using the floor function on each of its components. If $\boldsymbol{X}_1$ is feasible and its image sample-path dominates the image of $\boldsymbol{X}_{\text{best}}$, then $\boldsymbol{X}_{\text{best}}$ is updated and a longer line search step is taken from $\boldsymbol{X}_0$. This process continues until $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_1)$ does not dominate $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$, or $\boldsymbol{X}_1$ is infeasible, or a simulation limit is met. At this point, MGSPLI returns to the perturbation step (Algorithm 3, Line 3) and starts the descent direction and line search process again. Note, however, that if a line search stops "early," within 2 steps, MGSPLI returns $\boldsymbol{X}_{\text{best}}$ to MGSPLINE as input to NE.

Algorithm 3: $[n', \boldsymbol{X}_{\text{best}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})]$ = MGSPLI($\boldsymbol{X}_{\text{in}}, m, b$)

```
Input: solution $\boldsymbol{X}_{\text{in}} \in \mathcal{X}$; sample size $m$, oracle call limit $b$
Output: number of oracle calls used $n'$, best feasible point $\boldsymbol{X}_{\text{best}} \in \mathcal{X}$ and $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$
1  Initialize: initial step size $s_0 \leftarrow 2.0$ and multiplier $c \leftarrow 2.0$; $\boldsymbol{X}_{\text{best}} \leftarrow \boldsymbol{X}_{\text{in}}$ and $n' \leftarrow 0$
2  repeat
3      $\boldsymbol{X}_{\text{P}} \leftarrow \text{PERTURB}(\boldsymbol{X}_{\text{best}})$                /shift to non-integer point
4      $[n_{\text{pli}}, \boldsymbol{X}_{\text{simp}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}), \widehat{\boldsymbol{\gamma}}] = \text{PLI}(\boldsymbol{X}_{\text{P}}, \boldsymbol{X}_{\text{best}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}}), m)$   /determine descent direction
5      $n' \leftarrow n' + n_{\text{pli}}$                /update oracle calls
6      if $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}) \le \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$ then $\boldsymbol{X}_{\text{best}} \leftarrow \boldsymbol{X}_{\text{simp}}$   /update $\boldsymbol{X}_{\text{best}}$ from simplex points
7      if $\|\widehat{\boldsymbol{\gamma}}\|$ is undefined or $n' > b$ then
8          $i \leftarrow 0$                /exit if no direction or too many oracle calls
9      else
10         Initialize: $i \leftarrow 0$, $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}_{\text{best}}$
11         repeat
12             $i \leftarrow i+1$, $s \leftarrow c^{i-1} \times s_0$, and $\boldsymbol{X}_1 \leftarrow \boldsymbol{X}_0 + s \times \widehat{\boldsymbol{\gamma}} / \|\widehat{\boldsymbol{\gamma}}\|$      /take a step
13             Shift $\boldsymbol{X}_1$ to its nearest integer solution
14             if $\boldsymbol{X}_1$ is feasible then
15                 Simulate at $\boldsymbol{X}_1$ to observe $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_1)$
16                 $n' \leftarrow n' + m$                /update oracle calls
17                 if $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_1) \le \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$ then $\boldsymbol{X}_{\text{best}} \leftarrow \boldsymbol{X}_1$   /update $\boldsymbol{X}_{\text{best}}$ or stop line search
18         until $\boldsymbol{X}_1 \neq \boldsymbol{X}_{\text{best}}$ or $n' > b$
19 until $i \le 2$                /exit if line search stops quickly
20 return $n', \boldsymbol{X}_{\text{best}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$
```

Algorithm 4: $[n_{\text{pli}}, \boldsymbol{X}_{\text{simp}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}), \widehat{\boldsymbol{\gamma}}] = \text{PLI}(\boldsymbol{X}_{\text{P}}, \boldsymbol{X}_{\text{best}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}}), m)$

```
Input: point $\boldsymbol{X}_{\text{P}} = (X_1, X_2, \ldots, X_q) \in \mathbb{R}^q \setminus \mathbb{Z}^q$; sample size $m$
Output: number of oracle calls used $n_{pli}$, best simplex point $\boldsymbol{X}_{\text{simp}}$ and $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}})$, descent direction $\widehat{\boldsymbol{\gamma}}$
1  Set $\boldsymbol{S}_0 = \lfloor \boldsymbol{X}_{\text{P}} \rfloor$          /element-wise floor, find initial simplex point
2  Set $\boldsymbol{Z} = \boldsymbol{X}_{\text{P}} - \boldsymbol{S}_0$ /element-wise fractional differences to initial simplex point
3  Sort $\boldsymbol{Z} = (Z_1, Z_2, \ldots, Z_q)$ to get $1 = Z_{p(0)} > Z_{p(1)} \ge \cdots \ge Z_{p(q)} \ge Z_{p(q+1)} = 0$
4  Set $\boldsymbol{S}_i = \boldsymbol{S}_{i-1} + \mathbf{e}_{p(i)}$, for $i = 1, 2, \ldots, q$          /find other simplex points
5  Set $n_{\text{pli}} = 0$, $\tilde{q} = 0$, $\boldsymbol{X}_{\text{simp}} = \mathbf{0}_q$, and $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}) = \boldsymbol{\infty}_d$
6  for $i = 0, 1, \ldots, q$ do
7      if $\boldsymbol{S}_i$ is feasible then
8          Obtain $m$ simulation replications at $\boldsymbol{S}_i$ and update $n_{\text{pli}} \leftarrow n_{\text{pli}} + m$
9          Set $\tilde{q} \leftarrow \tilde{q} + 1$          /update number of feasible simplex points
10         if $\bar{\boldsymbol{G}}_m(\boldsymbol{S}_i) \le \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}})$ then $\boldsymbol{X}_{\text{simp}} \leftarrow \boldsymbol{S}_i$, $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}) \leftarrow \bar{\boldsymbol{G}}_m(\boldsymbol{S}_i)$     /update best point
11 if $\tilde{q} < q + 1$ then
12     if $\bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}) \le \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{best}})$ then
13         $\widehat{\boldsymbol{\gamma}} \leftarrow \boldsymbol{X}_{\text{simp}} - \boldsymbol{X}_{\text{best}}$
14     else
15         $\widehat{\boldsymbol{\gamma}} \leftarrow undefined$
16 else
17     Form matrix $\hat{\nabla}\boldsymbol{G}(\boldsymbol{X}_{\text{P}})$ with $p(i)$th row $\hat{\nabla}\boldsymbol{G}_{p(i)}(\boldsymbol{X}_{\text{P}}) = \bar{\boldsymbol{G}}_{m_k}(\boldsymbol{S}_i) - \bar{\boldsymbol{G}}_{m_k}(\boldsymbol{S}_{i-1})$ for $i = 1, 2, \ldots, q$
18     Solve Problem $\hat{D}$ to find $\widehat{\boldsymbol{\gamma}}$          /determine common descent direction
19 return $n_{\text{pli}}, \boldsymbol{X}_{\text{simp}}, \bar{\boldsymbol{G}}_m(\boldsymbol{X}_{\text{simp}}), \widehat{\boldsymbol{\gamma}}$
```

4.2 Neighborhood Enumeration (NE)

Neighborhood enumeration is performed to certify that the current best point is a sample-path local efficient point. Recall from Section 1.3 that $\mathcal{N}(\boldsymbol{X})$ consists of the $2q+1$ neighbors that are within Euclidean distance 1 from the point $\boldsymbol{X}$. NE determines the set of neighbors in $\mathcal{N}(\boldsymbol{X}_{\text{SPLI}})$, retrieves m_ν samples from each neighbor, and determines whether the estimated objective values of each neighbor dominate the estimated objective values of $\boldsymbol{X}_{\text{SPLI}}$. If a neighbor's image sample-path dominates $\boldsymbol{X}_{\text{SPLI}}$, the neighbor is returned as $\boldsymbol{X}_{\text{NE}}$. Otherwise, $\boldsymbol{X}_{\text{SPLI}}$ is returned as a sample-path local efficient point.

We implement NE by evaluating the neighbors one at a time and stopping the procedure as soon as the estimated objective values of a neighboring point dominate $\bar{\boldsymbol{G}}_{m_\nu}(\boldsymbol{X}_{\text{SPLI}})$. There are other ways of performing neighborhood enumeration, including evaluating *all* neighbors and choosing one to pass back to MGSPLINE. However, evaluating all neighbors may become inefficient in problems with a high-dimensional decision space.

5 NUMERICAL EXPERIMENTS

In this section, we provide preliminary results on the performance of R-MGSPLINE on three test problems. Since we know of no other algorithms that directly address our problem statement of finding local efficient points in the integer-ordered MOSO context, we focus on evaluating our algorithm. We evaluate our algorithm in terms of the distance between the true image of the estimated local efficient point, $\mathbf{g}(\boldsymbol{X}^*(t))$, and the set of true global or local weakly Pareto points, $\mathcal{E}$ or $\mathcal{X}^{\text{w}}$, respectively, as the amount of simulation effort, measured as the total number of simulation oracle calls t, increases. Mathematically, we write $d(\mathbf{x},\mathcal{B}) = \inf_{\mathbf{x}'\in\mathcal{B}} \|\mathbf{x}-\mathbf{x}'\|$ as the distance from the point $\mathbf{x}\in\mathbb{R}^q$ to the set $\mathcal{B}$. We use two bi-objective test problems from Cooper et al. (2019) that are available in the PyMOSO software (Cooper and Hunter 2019) and a 3-objective problem adapted from a survey paper of deterministic test problems to which we add randomness.

Our current implementation of R-MGSPLINE is through the Python software package of PyMOSO (Cooper and Hunter 2019). By default, RA algorithms currently implemented in PyMOSO use a static sample size increase of $m_\nu = \lceil m_0 * 1.1^\nu \rceil$ and $b_\nu = \lceil b_0 * 1.2^\nu \rceil$. We use the default PyMOSO sequences with parameters $m_0 = 2$ and $b_0 = 8(q-1)$. We perform 1,000 independent runs of our algorithm on each test problem and compute our performance metrics at the end of each retrospective iteration within each test run. We use CRN in each RA iteration and specify a total budget of 6,000 simulation oracle calls for each test run.

Our first test problem, Problem T_A in PyMOSO, is a modified version of a problem that appears in Kim and Ryu (2011):

$$\text{Problem } T_A: \quad \text{minimize}_{\mathbf{x}\in\mathcal{X}} \begin{cases} g_1(\mathbf{x}) = \mathrm{E}[(x_1/10-2\xi_1)^2 + (x_2/10-\xi_2)^2] \\ g_2(\mathbf{x}) = \mathrm{E}[x_1^2/100 + (x_2/10-2\xi_3)^2], \end{cases}$$

where $\mathcal{X} = \tilde{\mathcal{X}}_{A1}\times\tilde{\mathcal{X}}_{A2}$ and $\tilde{\mathcal{X}}_{A1} = \tilde{\mathcal{X}}_{A2} = \{0,1,2,\ldots,50\}$, $|\mathcal{X}| = 2601$, and ξ_i are independent chi-squared random variables with one degree of freedom so that $\mathrm{E}[\xi_i] = 1$ and $\mathrm{Var}(\xi_i) = 2$ for all $i\in\{1,2,3\}$.

In Problem T_A, there are 231 local weakly efficient points that exist due the discretization of the problem, of which 50 are in the global efficient set, which we denote as $\mathcal{E}$. Figure 2 displays the decision space and objective spaces of Problem T_A in the left and middle panels. The performance of R-MGSPLINE on Problem T_A is shown in the right panel of Figure 2 in terms of the distance from the true objectives of the current best point to the global Pareto set, $\mathbf{g}(\mathcal{E})$, as well as the distance to the set of local weakly Pareto points, $\mathbf{g}(\mathcal{X}^{\text{w}})$. We note that while R-MGSPLINE appears to converge to a point in the global efficient set eventually, the convergence to any of the local weakly efficient points seems faster.

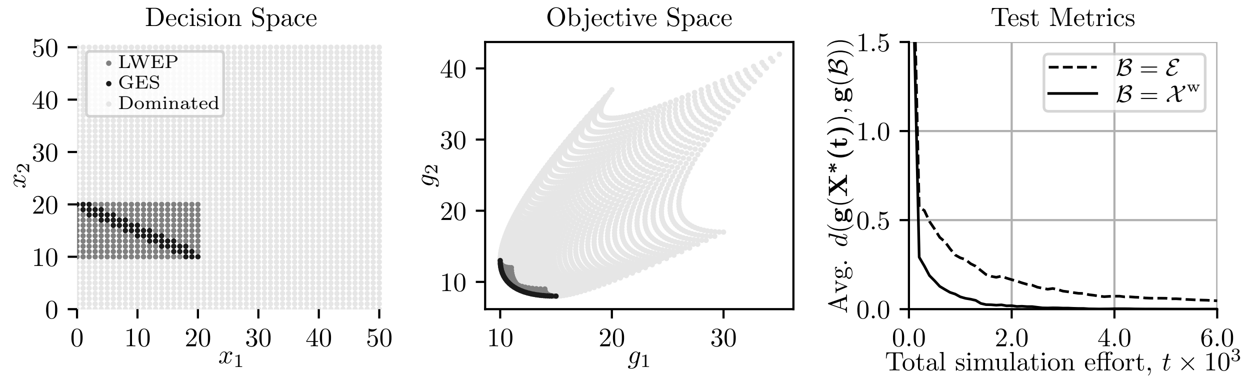

Figure 2: Test Problem A: The global efficient set (GES, black) and local weakly efficient points (LWEP, dark gray) and their images are highlighted in the decision space and objective space in the left and middle panels, respectively. The distances to the global Pareto set and to the set of local weakly Pareto points are shown in the right panel, averaged across 1,000 independent sample paths.

Our second test problem, Problem T_C in PyMOSO, is a modified version of a test problem that appears in Ryu and Kim (2014):

$$\text{Problem } T_C: \quad \text{minimize}_{\mathbf{x}\in\mathcal{X}} \begin{cases} g_1(\mathbf{x}) = \mathrm{E}\left[\sum_{i=1}^{2} -10\xi_i \exp\left\{-0.2\sqrt{x_i^2 + x_{i+1}^2}\right\}\right] \\ g_2(\mathbf{x}) = \mathrm{E}\left[\sum_{i=1}^{3} \xi_i(|x_i|^{0.8} + 5\sin^3(x_i))\right] \end{cases}$$

where $\mathcal{X} = \tilde{\mathcal{X}}_{C1} \times \tilde{\mathcal{X}}_{C2} \times \tilde{\mathcal{X}}_{C3}$, $\tilde{\mathcal{X}}_{Ci} = \{-5, -4.5, -4.0, -3.5, \ldots, 5\}$ for all $i \in \{1,2,3\}$, $|\mathcal{X}| = 9{,}261$, and ξ_1, ξ_2, and ξ_3 are independent chi-squared random variables with one degree of freedom so that $\mathrm{E}[\xi_i] = 1$ and $\mathrm{Var}(\xi_i) = 2$ for all $i \in \{1,2,3\}$. Problem T_C has dependence between the random objective function values returned by the simulation oracle.

Cooper et al. (2019) note that Problem T_C has multiple local weakly efficient sets (see Section 1.3). In their analysis, they find a collection of 516 unique local weakly efficient sets that contain just 73 points. We denote the collection of these local weakly efficient sets as $\cup_{i=1}^{516}\mathcal{W}_i$. There are a total of 512 local weakly efficient points; recall that not all local weakly efficient points belong to a local weakly efficient set (Cooper et al. 2019). Figure 3 displays the decision space, objective space, and performance of R-MGSPLINE on Problem T_C. The test metrics consider the distance from the current best point to the set of 516 unique weakly Pareto sets, $\mathbf{g}(\cup_{i=1}^{516}\mathcal{W}_i)$, as well as the distance to the set of 512 local weakly Pareto points, $\mathbf{g}(\mathcal{X}^{\mathrm{w}})$. We note that some sample paths get "stuck" at a local weakly efficient point, which leads to the leveling-off behavior in the distance to the local weakly efficient sets, whereas the convergence to a local weakly efficient point occurs rather quickly.

Our third test problem, Problem T_D, is a modified version of test problem ZLT1 that appears in Huband et al. (2006).

$$\text{Problem } T_D: \quad \text{minimize}_{\mathbf{x}\in\mathcal{X}} \begin{cases} g_1(\mathbf{x}) = \mathrm{E}[(x_1/5 - \xi_1)^2 + (x_2/5)^2 + (x_3/5)^2] \\ g_2(\mathbf{x}) = \mathrm{E}[(x_1/5)^2 + (x_2/5 - \xi_2)^2 + (x_3/5)^2] \\ g_3(\mathbf{x}) = \mathrm{E}[(x_1/5)^2 + (x_2/5)^2 + (x_3/5 - \xi_3)^2] \end{cases}$$

where $\mathcal{X} = \tilde{\mathcal{X}}_{C1} \times \tilde{\mathcal{X}}_{C2} \times \tilde{\mathcal{X}}_{C3}$, $\tilde{\mathcal{X}}_{Ci} = \{-25, -24, \ldots, -1, 0, 1, \ldots, 24, 25\}$ for all $i \in \{1,2,3\}$, $|\mathcal{X}| = 132{,}651$, and ξ_1, ξ_2, and ξ_3 are independent uniform random variables in the range $[-1,3]$, so that $\mathrm{E}[\xi_i] = 1$ and $\mathrm{Var}(\xi_i) = \frac{4}{3}$ for all $i \in \{1,2,3\}$.

We note that Problem T_D is similar to Problem T_A except there are three objectives. There are a total of 216 local weakly efficient points, 46 of which are in a local efficient set. Figure 4 displays the decision

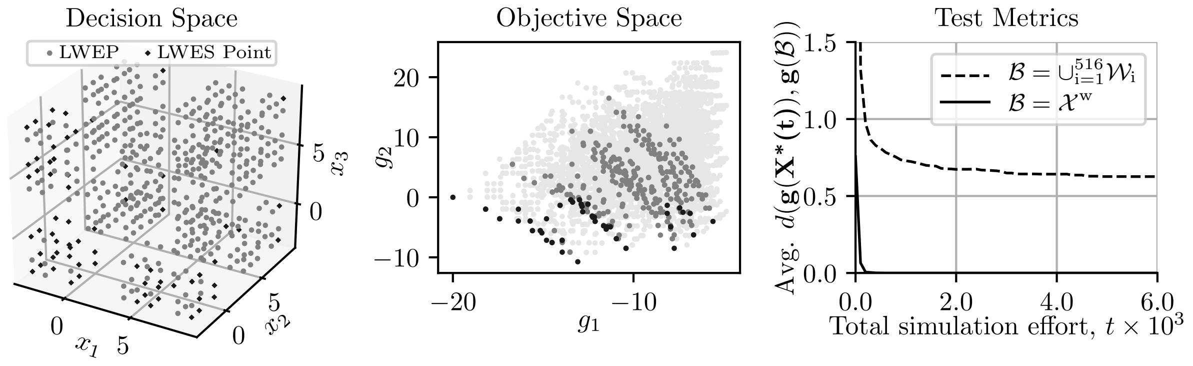

Figure 3: Test Problem C: The points in the 516 unique local weakly efficient sets (LWES points, black) and all local weakly efficient points (LWEP, dark gray) and their images are highlighted in the decision space and objective space in the left and middle panels, respectively. The distances to the union of 516 unique local weakly Pareto sets and to the set of local weakly Pareto points are shown in the right panel, averaged across 1,000 independent sample paths.

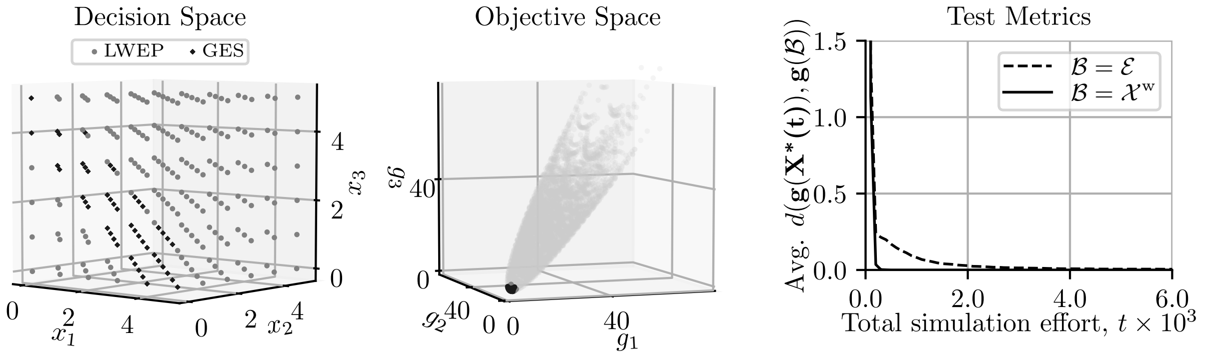

Figure 4: Test Problem D: The global efficient set (GES, black) and local weakly efficient points (LWEP, gray) and their images are highlighted in the decision space and objective space in the left and middle panels, respectively. The distances to the global Pareto set and to the set of local weakly Pareto points are shown in the right panel, averaged across 1,000 independent sample paths.

space and objective space of Problem T_D in the left and middle panels, respectively. The performance of R-MGSPLINE on Problem T_D is shown in the right panel of Figure 4 in terms of distance from the true objectives of the current best point to the global Pareto set, $\mathbf{g}(\mathcal{E})$, as well as the distance to the set of local weakly Pareto points, $\mathbf{g}(\mathcal{X}^w)$. As in Problem T_A, while R-MGSPLINE seems to converge into the global efficient set, the convergence to a local weakly efficient point seems faster.

6 CONCLUDING REMARKS

We develop the R-MGSPLINE algorithm with the goal of finding a local efficient point in the context of solving a MOSO problem. Within each RA iteration, the MGSPLINE subroutine computes a common descent direction from estimated pseudo-gradients of each objective and conducts a line search. The point returned by MGSPLINE is certified to be sample-path non-dominated in its neighborhood by means of a neighborhood enumeration. We have shown empirically that R-MGSPLINE converges into the set of local weakly efficient points on our test problems, and we believe that a convergence proof is a straightforward extension of the convergence proofs found in related work (Cooper et al. 2019).

ACKNOWLEDGMENTS

The authors thank Kyle Cooper for his assistance with PyMOSO. This work was supported by the National Science Foundation through grant CMMI-1554144.

REFERENCES

Applegate, E. A., G. Feldman, S. R. Hunter, and R. Pasupathy. 2019. “Multi-objective Ranking and Selection: Optimal Sampling Laws and Tractable Approximations via SCORE”. *Journal of Simulation*. doi: 10.1080/17477778.2019.1633891.

Branke, J., and W. Zhang. 2015. “A New Myopic Sequential Sampling Algorithm for Multi-Objective Problems”. In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 3589–3598. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Branke, J., W. Zhang, and Y. Tao. 2016. “Multiobjective Ranking and Selection Based on Hypervolume”. 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, 859–870. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Cooper, K., and S. R. Hunter. 2019. “PyMOSO: Software for Multi-Objective Simulation Optimization with R-PERLE and R-MinRLE”. *INFORMS Journal on Computing*. doi: 10.1287/ijoc.2019.0902.

Cooper, K., S. R. Hunter, and K. Nagaraj. 2019. “Bi-objective simulation optimization on integer lattices using the epsilon-constraint method in a retrospective approximation framework”. *INFORMS Journal on Computing*. doi: 10.1287/ijoc.2019.0918.

Custódio, A., M. Emmerich, and J. Madeira. 2012. “Recent Developments in Derivative-Free Multiobjective Optimization”. *Computational Technology Reviews* 5(1):1–31.

Custódio, A. L., J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente. 2011. “Direct Multisearch for Multiobjective Optimization”. *SIAM Journal on Optimization* 21(3):1109–1140.

Désidéri, J. A. 2012. “Multiple-Gradient Descent Algorithm (MGDA) for Multiobjective Optimization”. *Comptes Rendus Mathematique* 350(5–6):313–318.

Digabel, S. L., and S. M. Wild. 2015. “A Taxonomy of Constraints in Simulation-Based Optimization”. *arXiv preprint arXiv:1505.07881*.

Ehrgott, M. 2006. “A Discussion of Scalarization Techniques for Multiple Objective Integer Programming”. *Annals of Operations Research* 147(1):343–360.

Ehrgott, M., and X. Gandibleux. 2000. “A Survey and Annotated Bibliography of Multiobjective Combinatorial Optimization”. *OR-Spektrum* 22(4):425–460.

Feldman, G., and S. R. Hunter. 2018, January. “SCORE Allocations for Bi-Objective Ranking and Selection”. *ACM Transactions on Modeling and Computer Simulation* 28(1):7:1–7:28.

Fliege, J., and B. F. Svaiter. 2000. “Steepest Descent Methods for Multicriteria Optimization”. *Mathematical Methods of Operations Research* 51(3):479–494.

Fu, M. C. 2015. *Handbook of Simulation Optimization*, Volume 216 of *International Series in Operations Research & Management Science*. New York: Springer.

Fu, M. C., G. Bayraksan, S. G. Henderson, B. L. Nelson, W. B. Powell, I. O. Ryzhov, and B. Thengvall. 2014. “Simulation Optimization: A Panel on the State of the Art in Research and Practice”. In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 3696–3706. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

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

Huang, H., and Z. B. Zabinsky. 2014. “Multiple Objective Probabilistic Branch and Bound for Pareto Optimal Approximation”. In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 3916–3927. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Huband, S., P. Hingston, L. Barone, and L. While. 2006. “A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit”. *IEEE Transactions on Evolutionary Computation* 10(5):477–506.

Hunter, S. R., E. A. Applegate, V. Arora, B. Chong, K. Cooper, O. Rincón-Guevara, and C. Vivas-Valencia. 2019, January. “An Introduction to Multi-Objective Simulation Optimization”. *ACM Transactions on Modeling and Computer Simulation* 29(1):7:1–7:36.

Kim, S., and J. Ryu. 2011. "The Sample Average Approximation Method for Multi-Objective Stochastic Optimization". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 4026–4037. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Lee, L. H., E. P. Chew, S. Teng, and D. Goldsman. 2010. "Finding the Non-Dominated Pareto Set for Multi-Objective Simulation Models". *IIE Transactions* 42(9):656–674.

Li, H., L. H. Lee, E. P. Chew, and P. Lendermann. 2015. "MO-COMPASS: A Fast Convergent Search Algorithm for Multi-Objective Discrete Optimization via Simulation". *IIE Transactions* 47(11):1153–1169.

Li, H., Y. Zhu, Y. Chen, G. Pedrielli, N. A. Pujowidianto, and Y. Chen. 2015. "The Object-Oriented Discrete Event Simulation Modeling: A Case Study on Aircraft Spare Part Management". 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, 3514–3525. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Mercier, Q., F. Poirion, and J. Désidéri. 2018. "A Stochastic Multiple Gradient Descent Algorithm". *European Journal of Operational Research* 271(3):808–817.

Miettinen, K. 1999. *Nonlinear Multiobjective Optimization*. Boston: Kluwer Academic Publishers.

Pasupathy, R., and S. Ghosh. 2013. "Simulation Optimization: A Concise Overview and Implementation Guide". In *TutORials in Operations Research*, edited by H. Topaloglu, Chapter 7, 122–150. Catonsville, Maryland: Institute for Operations Research and the Management Sciences.

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, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pasupathy, R., and S. G. Henderson. 2011. "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, 4075–4085. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Peitz, S., and M. Dellnitz. 2018. "Gradient-Based Multiobjective Optimization with Uncertainties". In *NEO 2016*, 159–182. Cham, Switzerland: Springer.

Prakash, O., K. Srinivasan, and K. P. Sudheer. 2015. "Adaptive Multi-Objective Simulation-Optimization Framework for Dynamic Flood Control Operation in a River-Reservoir System". *Hydrology Research* 46(6):893–911.

Ryu, J., and S. Kim. 2014. "A Derivative-Free Trust-Region Method for Biobjective Optimization". *SIAM Journal on Optimization* 24(1):334–362.

Wang, H., R. Pasupathy, and B. W. Schmeiser. 2013. "Integer-Ordered Simulation Optimization Using R-SPLINE: Retrospective Search using Piecewise-Linear Interpolation and Neighborhood Enumeration". *ACM Transactions on Modeling and Computer Simulation* 23(3):17:1–17:24.

Wang, Y., L. H. Lee, E. P. Chew, S. S. W. Lam, S. K. Low, M. E. H. Ong, and H. Li. 2015. "Multi-Objective Optimization for a Hospital Inpatient Flow Process via Discrete Event Simulation". 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, 3622–3631. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Xu, J., B. L. Nelson, and L. J. Hong. 2010. "Industrial Strength COMPASS: A Comprehensive Algorithm and Software for Optimization via Simulation". *ACM Transactions on Modeling and Computer Simulation* 20(1):1–29.

AUTHOR BIOGRAPHIES

ERIC A. APPLEGATE is a Ph.D. student in the School of Industrial Engineering at Purdue University. His research interests include applied operations research, particularly simulation optimization methods and applications, in addition to teaching methodology. His email address is applegae@purdue.edu.

SUSAN R. HUNTER is an assistant professor in the School of Industrial Engineering at Purdue University. Her research interests include Monte Carlo methods and simulation optimization, especially in the presence of multiple performance measures. Her email address is susanhunter@purdue.edu. Her website is https://web.ics.purdue.edu/~hunter63.