AN INVESTIGATION OF MODEL-BASED APPROACHES IN SOLVING A VARIETY OF GLOBAL OPTIMIZATION PROBLEMS

A Dissertation
Presented to
The Academic Faculty

By

Joshua Q Hale

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

May 2017

AN INVESTIGATION OF MODEL-BASED APPROACHES IN SOLVING A VARIETY OF GLOBAL OPTIMIZATION PROBLEMS

Approved by:

Dr. Enlu Zhou, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Natashia Boland
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. David Goldsman
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Seong-Hee Kim
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Jiming Peng
Department of Industrial Engineering
University of Houston

Date Approved: March 31, 2017

To my family and loved ones

ACKNOWLEDGEMENTS

I would like to provide special thanks to my research adviser Dr. Enlu Zhou for giving me the opportunity to pursue my PhD under her supervision. Her mentorship was essential in every step of the process and really pushed my thought and actions regarding several important aspects of my dissertation. I would also like to thank my committee members, Dr. Natashia Boland, Dr. David Goldsman, Dr. Seong-Hee Kim, and Dr. Jiming Peng for their incredibly valuable time and input.

I also want to acknowledge my family, and in particular Alfreda Hale, Raven Hale, and Zaden Hale for supporting and believing in me. I want to especially thank my late father Zerious Hale for instilling in me the importance of education and teaching me the valuable tools needed to be successful. Finally, I would like to thank all my colleagues and friends that help me along the way that includes but are not limited to Ebony Rowe, Julian Williams, Dr. Helin Zhu, Dr. Fan Ye, Di Wu, Tianyi Liu, Toyya Pujol, Junia Findlay, and all my friends from Sylacauga, Auburn, Georgia Tech, and UIUC.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Model-based optimization methods are a class of random search methods that are useful for solving global optimization problems. These types of methods have shown significant empirical success in solving a multitude of real-world problems in which the objective functions are often highly ill-structured. In this dissertation, we propose new approaches to incorporate and extend model-based methods with the goal of promoting them for future applications.

In the last 20 years, model-based methods have been gaining popularity as a powerful optimization technique among many researchers in different fields. This is due to the increasing need for solving complex problems that are very difficult or infeasible to solve with exact algorithms. The main idea of model-based methods is to iteratively construct a sampling distribution that is biased towards favorable areas in the decision space. Several algorithms have been developed recently that are classified as model-based methods. We provide an overview of these methods and investigate how they are similar in many areas while also having distinctive characteristics.

Part I: We propose a novel algorithm for solving the classical P-median problem. The essential aim is to identify the optimal extended penalty multipliers corresponding to the optimal solution of the underlying problem. For this, we first explore the structure of the data matrix in the P-median problem to recast it as another equivalent global optimization problem over the space of the extended penalty multipliers. Then, we present a model-based algorithm to find the extended penalty multipliers corresponding to the optimal solution of the original P-median problem. Numerical experiments illustrate that the proposed algorithm can effectively find a global optimal or very good suboptimal solution to the underlying P-median problem. It is especially useful for the computationally challenging subclass of P-median problems with a large gap between the optimal solution of the original problem and that of its Lagrangian dual.

Part II: For general multi-objective optimization problems, we propose a new performance metric called domination measure to measure the quality of a solution, which can be intuitively interpreted as the size of the portion of the solution space that dominates that solution. As a result, we reformulate the original multi-objective problem into a stochastic single-objective one and propose a model-based approach to solve it. We show that an ideal version algorithm of the proposed approach converges to a representative set of the global optima of the reformulated problem. We also investigate the numerical performance of an implementable version algorithm by comparing it with numerous existing multi-objective optimization methods on popular benchmark test functions. The numerical results show that the proposed approach is effective in generating a finite and uniformly spread approximation of the Pareto optimal set of the original multi-objective problem and is competitive to the tested existing methods.

Part III: The fields of research for both stochastic optimization and multi-objective optimization are well studied and highly developed. Surprisingly, a paucity of research has attempted to explore the combination of both stochastic and multi-objective optimization. Therefore, we propose a model-based method for solving optimization problems in which a solution performance is measured by multiple objectives that are subject to stochastic noise. Since the objective functions cannot be evaluated directly, the values are estimated via simulation. The proposed method integrates a new allocation heuristic within the model-based framework. This heuristic improves the efficiency of the method by allocating the majority of the computational budget to solutions that play a crucial role in updating the parameters of the sampling distribution used to facilitate the search process. The proposed method is designed to deal with multiple objectives and uncertainty simultaneously. In order to investigate the performance of our proposed algorithm, test functions were constructed by adding stochastic noise to popular benchmark deterministic multi-objective problems. The numerical experiments suggest that the proposed method is quite promising in terms of approximating the Pareto optimal set and in terms of computational efficiency.

CHAPTER 1
INTRODUCTION

1.1 Stochastic Search Methods

Stochastic search methods are algorithms that incorporate some randomized mechanism in generating candidate solutions. They are effective at solving continuous or discrete optimization problems where the objective function is poorly structured, i.e., discontinuous, nonconvex or nondifferentiable. Stochastic search methods perform very well in exploring the entire solution space and are simple to implement since the algorithm is independent of any structural information of the objective function. The randomness of the algorithm helps in preventing the algorithm from getting stalled at local optimal solutions. Typically, stochastic search methods are iterative algorithms that choose candidate solutions by some stochastic mechanism and improve the way those candidate solutions are selected at every iteration. Since many real-world problems are too complex to be solved analytically, stochastic search methods are a desirable alternative to classical optimization methods.

Stochastic search methods are classified as either instance-based or model-based. The major distinctions between instance-based and model-based methods are the procedures in which new candidate solutions are generated and improved. In instance-based methods, the idea is that good solutions have similar structures. Therefore, new solutions are generated by some manipulation of previously generated solutions. Examples of instance-based methods include tabu search [30], simulated annealing [40], and genetic algorithms [62]. In contrast, model-based methods generate new solutions from a parameterized sampling distribution. The idea is to continually update the sampling distribution so that solutions that are close to the global optima are more likely to be sampled as the algorithm progresses. Since solutions are generated from a sampling distribution, model-based methods are con-

sidered to be more robust than their instance-based counterparts. Examples of model-based methods include annealing adaptive search [67], estimation of distribution algorithms [41], gradient-based adaptive search (GASS) [69], model reference adaptive search (MRAS) [34], and cross-entropy (CE) method [23].

1.1.1 Literature Review

Model-based methods have shown empirical success in solving global optimization problems in both continuous and discrete domains. Although model-based methods have the same foundation, there are a variety of differences among different types of model-based approaches. Some methods are tailored for specific problems and are designed for good computational results; whereas, other methods are based on rigorous mathematical theory and provide convergence results. We present a review of the current most popular model-based methods for solving global optimization problems in order to describe the central idea and the flexibility of model-based methods. Although instance-based methods are also considered stochastic search methods, those types of methods are beyond the scope of this dissertation.

The goal in single-objective optimization is to find a feasible solution x^* that achieves the best value of an objective function. More formally, the problem is mathematically described as:

$$x^* \in \arg\max_{x \in \mathscr{X}} H(x), \tag{1.1.1}$$

where x is a vector of n decision variables, $\mathscr{X}$ is a nonempty compact set in $\mathbb{R}^n$, and $H : \mathscr{X} \to \mathbb{R}$ is a real valued bounded function i.e., there exist $H_l > -\infty$ and $H_u < \infty$ such that $H_l < H(x) < H_u$ for all $x \in \mathscr{X}$. We denote the optimal function value as H^*, where there exists a solution x^* such that $H^* \stackrel{\triangle}{=} H(x^*) \geq H(x)$ for all $x \in \mathscr{X}$.

We can reformulate the above optimization problem (1.1.1) by transforming it into an expectation of the objective function under a parameterized probability distribution. Let $\{f(x;\theta) : \theta \in \Theta\}$ denote the family of parameterized probability density functions (pdfs)

on $\mathscr{X}$, where Θ is the parameter space. That is,

$$E_\theta[H(x)] = \int_X H(x) f(x;\theta) dx \leq H^* \ \ \forall \theta \in \Theta. \tag{1.1.2}$$

If there exists an optimal parameter θ^* that causes the parameterized probability distribution $f(x;\theta)$ to assign all of its probability mass on the set of global optimal solutions of (1.1.1), then (1.1.2) is satisfied with equality. As a result, model-based methods seek to find a θ^* such that $E_{\theta^*}[H(x)] = H^*$, where $E_{\theta^*}[\cdot]$ is the expectation with respect to $f(\cdot;\theta^*)$.

The model-based search approach uses the general framework below to solve single-objective optimization problems:

1. Generate candidate solutions from a parameterized probability distribution over the decision space.

2. Update the sampling distribution based on the evaluation of generated candidate solutions.

The fundamental idea is to iteratively construct a probability distribution such that the limiting distribution will allocate most of its mass on a subset of optimal solutions. Essentially, the hope is that the resulting probability distribution is concentrated on the set of optimal solutions. Typically, the top performing samples are used to update the sampling distribution. This ensures that after each iteration, more weight is assigned to the area that contains the best samples from the previous iteration. Clearly, the two components that all model-based methods have are 1) a probabilistic model that has the ability to easily generate candidate solutions and 2) an update rule for the parameters of the probabilistic model. In order to discuss how different methods approach these two components, we provide a brief review of three model-based algorithms: CE method, MRAS, and GASS.

Cross Entropy Method

The CE method was first proposed by Rubinstein [60] as an adaptive procedure that uses importance sampling to estimate rare events, which are events that have an extremely low probability of occurring. Subsequent work by Rubinstein showed that the CE method can be modified to solve continuous and combinatorial optimization problems. In other words, locating an optimal solution via stochastic search can be considered a rare event. As a result, the CE method seeks to find the parameter of a family of parameterized distributions that minimizes the Kullback-Leibler (KL) divergence between that family of distributions and the optimal importance sampling distribution.

To describe how the CE method is used for optimization, we first consider the following estimation problem similar to the explanation found in [23]. Let $\mathbf{X} = [X_1, \ldots, X_N]$ be a random vector from the feasible space $\mathscr{X}$. Suppose we want to calculate the probability l that $H(\mathbf{X})$ is greater than or equal to a value $\gamma \in \mathbb{R}$ under $f(\cdot, u)$ on $\mathscr{X}$. This probability is defined as

$$l = P(H(\mathbf{X}) \geq \gamma) = E_u[I_{\{H(\mathbf{X}) \geq \gamma\}}], \tag{1.1.3}$$

where $E_u[\cdot]$ is the expectation with respect to $f(\cdot, u)$ and $I_{\{H(\mathbf{y}) \geq \gamma\}} = 1$ if $H(\mathbf{y}) \geq \gamma$ and $I_{\{H(\mathbf{y}) \geq \gamma\}} = 0$ otherwise. We assume $P(H(\mathbf{X}) \geq \gamma)$ is very small, i.e., $P(H(\mathbf{x}) \geq \gamma) < 10^{-5}$ and thus the event $\{H(\mathbf{x}) \geq \gamma\}$ can be considered as a rare event. A simple way to calculate an unbiased estimator $\hat{l}$ of the above probability is to use Monte Carlo simulation as follows: Draw independent and identically distributed (i.i.d.) samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from $f(\cdot; u)$. Then calculate

$$\hat{l} = \frac{1}{N} \sum_{i=1}^{N} I_{\{H(\mathbf{X}_i) \geq \gamma\}}. \tag{1.1.4}$$

A large number of simulations are required to get an accurate estimate $\hat{l}$ noticing that l is a rare event. In other words, it would take on average 10^5 simulations to get one random sample $\mathbf{X}'$ that causes $I_{\{H(\mathbf{X}') \geq \gamma\}} = 1$. Alternatively, importance sampling can be used to obtain a good estimate of l. This technique consists of drawing i.i.d samples $\mathbf{X_1}, \ldots, \mathbf{X_N}$

from a different pdf $g(\cdot)$ and estimating l by

$$\hat{l} = \frac{1}{N}\sum_{i=1}^{N} I_{\{H(\mathbf{X_i})\geq\gamma\}}\frac{f(\mathbf{X}_i;\theta)}{g(\mathbf{X}_i)} \tag{1.1.5}$$

The pdf $g(\cdot)$ is known as the importance sampling distribution. The optimal importance sampling pdf is the density of x conditioned on the event $\{H(x) \geq \gamma\}$; that is

$$g^*(x) = \frac{I_{\{H(x)\geq\gamma\}}f(x;u)}{l}, \tag{1.1.6}$$

which yields a zero-variance estimator of $\hat{l}$. Obviously, we cannot calculate g^* from the above equation since it depends on the value l. Therefore, the idea is to restrict the choice of the importance sampling distributions $g(\cdot)$ to some parametric distribution family $\{f(\cdot,\theta) : \theta \in \Theta)\}$ and find the "best" parameter θ^* that can be related to the optimal importance sampling distribution. To accomplish this, the CE method seeks to choose the importance sampling pdf $g(\cdot)$ from the family of pdfs $\{f(\cdot;\theta)\}$ such that Kullback-Leibler (KL) divergence between the optimal importance sampling pdf $g^*(\cdot)$ and $f(\cdot;\theta)$ is minimized. The KL divergence between $g^*(x)$ and f is given by

$$D(g^*(x), f(x;\theta)) = E_{g^*}\left[ln\frac{g^*(x)}{f(x;\theta)}\right] = \int_{\mathscr{X}} g^*(x)\ln g^*(x)dx - \int_{\mathscr{X}} g^*(x)\ln f(x;\theta)dx \tag{1.1.7}$$

This minimization procedure reduces to finding an optimal parameter θ^* where

$$\theta^* = \underset{\theta\in\Theta}{\operatorname{argmin}} \int_{\mathscr{X}} g^*(x)\ln g^*(x)dx - \int_{\mathscr{X}} g^*(x)\ln f(x;\theta)dx = \underset{\theta\in\Theta}{\operatorname{argmax}} \int_{\mathscr{X}} g^*(x)\ln f(x;\theta)dx \tag{1.1.8}$$

Substituting g^* from (1.1.6) into (1.1.8) we obtain

$$\theta^* = \max_{\theta\in\Theta} \int_{\mathscr{X}} \frac{I_{\{H(x)\geq\gamma\}}f(x;u)}{l}\ln f(x;\theta)dx = \max_{\theta\in\Theta} E_u[I_{\{H(x)\geq\gamma\}}\ln f(x;\theta)] \tag{1.1.9}$$

It is important to note that the minimization of KL divergence can be carried out in analytical form if $f(\cdot;\theta)$ belongs to the exponential family. Many common probability distributions belong to the exponential family, including Gaussian, Poisson, Binomial, Geometric, etc. The exponential family is defined as follows:

Definition 1.1.1. ***Exponential Famliy***. *A parameterized family* $\{f(x;\theta) : \theta \in \Theta\}$ *is an exponential family of densities if it satisfies*

$$f(x;\theta) = \exp\left\{\theta^T \Gamma(x) - \eta(\theta)\right\}, \tag{1.1.10}$$

where $\Gamma(x) = [\Gamma_1(x), ..., \Gamma_{d_\theta}(x)]^T$ *is the vector of sufficient statistics,* $\eta(\theta) = \ln\{\int \exp(\theta^T \Gamma(x))dx\}$ *is the normalization factor to ensure* $f(x;\theta)$ *is a probability density function, and* $\Theta = \{\theta : |\eta(\theta)| < \infty\}$ *is the natural parameter space with a nonempty interior. We assume that* $\Gamma(\cdot)$ *is a continuous mapping.*

We can now calculate the parameter θ^* such that $f(\cdot,\theta^*)$ minimizes the KL divergence with respect to the optimal importance sampling distribution g^*. Recall that the goal is to find x^* such that

$$H(x^*) = \max_{x \in \mathscr{X}} H(x). \tag{1.1.11}$$

We set $\gamma^* = H(x^*)$ so that we can associate an estimation problem with the above optimization problem. The idea is to use an iterative algorithm to construct a sequence of parameters $\{\theta_k : 1, \ldots\}$ and levels $\{\gamma_k : 1, \ldots\}$ that converges to θ^* and γ^*, respectively. Basically, the choice of γ and θ determines how well we can estimate l while using a moderate number of samples. In other words, the probability of the target event cannot be too small so that the event $\{H(\mathbf{X}) \geq \gamma\}$ rarely happens. Therefore, the CE method guarantees under the density $f(\cdot;\theta_k)$ that the probability of $l_k = E_{\theta_k}[I_{H(x)\geq\gamma_k}]$ is at least ρ, where ρ is chosen not to be too small, i.e., a common choice is $\rho = .1$. This is accomplished by updating γ_k and θ_k at every iteration of the algorithm. The procedure is as follows:

1. Adaptive updating of γ_k: For θ_{k-1}, let γ_k be a $(1-\rho)$ quantile of $H(x)$. That is, γ_k

satisfies

$$P_{\theta_{k-1}}(H(x) \geq \gamma_k) \geq \rho \tag{1.1.12}$$

$$P_{\theta_{k-1}}(H(x) \leq \gamma_k) \geq 1-\rho \tag{1.1.13}$$

where x is generated from the pdf $f(\cdot;\theta_{k-1})$. A estimator $\hat{\gamma}_k$ of γ_k can be obtained by drawing random samples from $f(\cdot;\theta_{k-1})$, calculating $H(\mathbf{X}_i)$ for all i, ordering them in ascending order: $H_{(1)} \leq \ldots \leq H_{(N)}$, and evaluating the $(1-\rho)$ sample quantile as

$$\hat{\gamma}_k = H_{(\lceil (1-p)N \rceil)} \tag{1.1.14}$$

2. Adaptive updating of θ_k: For a fixed $\hat{\gamma}_k$ and θ_{k-1}, determine the sampling distribution θ_k by

$$\hat{\theta}_k = \underset{\theta \in \Theta}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} I_{\{H(\mathbf{X}_i) \geq \hat{\gamma}_k\}} \ln f(\mathbf{X}_i;\theta). \tag{1.1.15}$$

The parameter $\hat{\theta}_k$ can be further updated by using a smoothing function

$$\hat{\theta}_k := \alpha \hat{\theta}_k + (1-\alpha)\hat{\theta}_{k-1}, \tag{1.1.16}$$

where $0 \leq \alpha \leq 1$.

At iteration k, the target event $\{H(\mathbf{X}) \geq \gamma_k\}$, where $\gamma_k < \gamma^*$ is made less rare by using the sample $(1-\rho)$ quantile of solutions generated at k to get a good estimate for θ_{k+1}. The hope is that the value θ_{k+1} will make the event $\{H(X) \geq \gamma_k\}$ more common, so in the next iteration a value $\hat{\gamma}_{k+2}$ is obtained such that $\hat{\gamma}_{k+2} > \hat{\gamma}_{k+1}$ is calculated. The algorithm terminates when γ_k is close to γ^* and the event $\left\{H(X) \geq \gamma_k^*\right\}$ is no longer a rare event.

1. Choose some θ_0. Set $k = 1$

2. Generate a sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from the density $f(\cdot;\theta_{k-1})$ and compute the sample $(1-\rho)$ quantile $\hat{\gamma}_k$ of the performance according to (1.1.14).

3. Use the sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ and solve the deterministic optimization problem in (1.1.15).

4. If the stopping condition is satisfied, terminate the algorithm; otherwise set $k \leftarrow k+1$ and return to Step 2.

In summary, the CE method finds the optimal importance sampling distribution that is concentrated only on the set of optimal solutions, and the key idea is to use an iterative scheme to estimate the optimal parameter of that distribution that minimizes the KL divergence between the optimal distribution and the family of parameterized distributions.

Model Reference Adaptive Search

The model reference adaptive search (MRAS) [34] is similar to the CE method in that it updates the parameters of a family of parameterized distributions by minimizing the KL divergence. Although MRAS uses a parameterized distribution to generate samples similar to the CE method, it also uses another sequence of distribution called reference distributions $h_k(\cdot)$ to update the parameters of the sampling distribution. The method starts by specifying a family of parameterized distributions $\{f(\cdot;\theta) : \theta \in \Theta\}$ and projecting $h_k(\cdot)$ onto the family to obtain a sampling distribution $f(\cdot;\theta_k)$, where the projection is implemented at each iteration by finding an optimal parameter θ_k that minimizes the KL divergence between $h_k(\cdot)$ and the parameterized family

$$\theta_k = \operatorname*{argmin}_{\theta} D(h_k(x), f(x;\theta)) := \operatorname*{argmin}_{\theta} \left(\int_{\mathscr{X}} \ln \frac{h_k(x)}{f(x;\theta)} h_k(dx) \right) \tag{1.1.17}$$

The sequence of reference distributions $\{h_k(\cdot)\}$ can be constructed in numerous ways. One popular method is to use the simple recursive procedure below:

$$h_k(x) = \frac{H(x)h_{k-1}(x)}{\int_{\mathscr{X}} H(x)h_{k-1}(dx)} \tag{1.1.18}$$

where $h_0(x)$ is a given initial distribution on x and under the assumption that $H(x) > 0$ for all $x \in \mathscr{X}$ to prevent negative probabilities. Basically, (1.1.18) weighs the new reference distribution h_k so that more mass is concentrated on solutions that have better performance. As a result, the expectation of $H(x)$ w.r.t the reference distribution improves at each iteration. That is, good solutions are given more probability under the next reference distribution h_{k+1}

$$E_{h_{k+1}}[H(x)] := \int_{\mathscr{X}} H(x)h_{k+1}(dx) = \frac{\int_{\mathscr{X}} H^2(x)h_k(dx)}{H(x)h_k(dx)} \geq E_{h_k}[H(X)] \tag{1.1.19}$$

This results in a sequence $\{g_k\}$ that converges to a degenerate distribution at the optimal solution. A convergence analysis of this method is presented in [34]. We provide a summary of the MRAS method below.

1. Select a sequence of reference distributions $\{h_k\}$ with desired convergence properties and choose a parameterized family $\{f_\theta\}$.

2. Given θ_k, sample N candidate solutions $x_k^1, \ldots, x_k^N$ from f_{θ_k}

3. Update the parameter θ_{k+1} by minimizing the KL divergence

$$\theta_{k+1} = \operatorname*{argmin}_{\theta} D(h_{k+1}, f(\theta)) \tag{1.1.20}$$

   increase k by 1 and reiterate from step 1.

In summary, this method updates the sampling distribution parameter at each iteration by minimizing the KL divergence between a specific reference distribution and a parameterized family of densities. This method has been shown to converge with probability one under some mild assumptions and perform well numerically.

Gradient-based Adaptive Stochastic Search

Similar to the aforementioned model-based methods, the main idea of GASS [69] is to update the parameterized sampling distribution iteratively towards the favorable areas of the solution space. In particular, this method takes advantage of the reformulated problem (1.1.2) being differentiable in θ under mild regularity conditions on $f(x;\theta)$. Furthermore, the gradient is simple to derive as follows:

$$\begin{aligned} \bigtriangledown_\theta L(\theta) &= \bigtriangledown_\theta \int_{\mathscr{X}} H(x) f(x;\theta) dx = \int_{\mathscr{X}} H(x) \frac{\bigtriangledown_\theta f(x;\theta)}{f(x;\theta)} f(x;\theta) dx \\ &= \mathbb{E}_{f(\cdot;\theta)} \left[H(x) \bigtriangledown_\theta \ln f(x;\theta) \right], \end{aligned}$$

where $L(\theta) = E_\theta[H(x)]$. Note that an unbiased estimator of $\bigtriangledown_\theta L(\theta)$ could be obtained by drawing i.i.d. samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from $f(x;\theta)$, and evaluating $H(\mathbf{X}_i) \bigtriangledown_\theta \ln f(\mathbf{X}_i;\theta)$, and taking the sample average of $\{H(\mathbf{X}_i) \bigtriangledown_\theta \ln f(x^i;\theta)\}$. Then, the idea is to solve the reformulated problem via a stochastic gradient-based method. Specifically, the method iteratively carries out the following two steps:

1. Generate candidate solutions according to the sampling distribution.

2. Based on the evaluation of the candidate solutions, update the parameter of the sampling distribution via gradient search.

A shape function S_θ is constructed to satisfy the following conditions: for every θ, $S_\theta(x)$ is strictly increasing in x, $0 < S_\theta(x) \leq M$, where $M < \infty$ for all $x \in \mathscr{X}$, and for every fixed x, $S_\theta(x)$ is continuous in θ. The purpose of the shape function is to make sure the objective function is positive, while preserving the order of the solutions with respect to the original objective function $H(\cdot)$. A common choice of $S_\theta(\cdot)$ is

$$S_\theta(H(x)) = \frac{1}{1 + \exp(-S_o(H(x) - \gamma_\theta))}, \tag{1.1.21}$$

where S_o is a large positive constant, γ_θ is the $(1-\rho)$-quantile, and $S_\theta(\cdot)$ is a continuous approximation of the indicator function $I_{\{H(x)\geq\gamma_\theta\}}$ that gives equal weights to the solutions with function values above γ_θ and eliminates the solutions with function values below γ_θ. For a fixed $\theta' \in \Theta$, define

$$L(\theta;\theta') \stackrel{\triangle}{=} \int S_{\theta'}(L(x)) f(x;\theta) dx, \quad \text{and} \quad L'(\theta;\theta') \stackrel{\triangle}{=} \ln H(\theta;\theta'). \tag{1.1.22}$$

By the conditions on $S_{\theta'}(\cdot)$ and the fact that $\ln(\cdot)$ is a strictly increasing function, the original problem can be transformed to $\max_{\theta\in\Theta} L'(\theta;\theta')$ for any fixed θ'. A summary of the GASS is below:

1. Initialization: choose an exponential family of densities $\{f(\cdot;\theta)\}$ and specify a small positive constant ε, initial parameter θ_0, sample size sequence $\{N_k\}$, and step size sequence $\{\alpha_k\}$. Set $k = 0$.

2. Sampling: draw samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from $f(x;\theta_k)$.

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

$$\hat{w}_k^i = \frac{\hat{S}_{\theta_k}(H(x_k^i))}{\sum_{j=1}^{N_k} \hat{S}_{\theta_k}(H(x_k^j))} \tag{1.1.23}$$

4. Update $\theta_k \to \theta_{k+1}$ using a Newton-like iteration for $\max_\theta L'(\theta;\theta_k)$, where $L'(\theta;\theta_k) = \ln\{\int S_{\theta_k}(H(x)) f(x;\theta) dx\}$.

5. If the stopping condition is satisfied, terminate algorithm, otherwise set $k \leftarrow k+1$ and return to Step 2.

In summary, this algorithm generates candidate solutions from a parameterized sampling distribution over the feasible solution space and uses a Newton-like iteration on the parameter space to update the parameters of the sampling distributions. Convergence re-

sults are established as a result of formulating the algorithm iterations into a generalized stochastic approximation recursion.

1.1.2 Motivation and Research Goals

Model-based methods have proven to be very effective for solving global optimization problems, both in discrete and continuous solution spaces. They are robust and straightforward to implement because they typically only rely on function evaluations, rather than the structure of the objective function. Therefore, model-based methods have been used to solve a multitude of problems. The advantages of these methods are twofold. On one hand, they are flexible in nature and can be tailored to solve a particular type of problem; on the other hand, their performance can also be problem independent because they are firmly based on probabilistic and statistical principles. As a result, modifications of these methods apart from their basic steps can be implemented on a problem basis, which includes but are not limited to:

1. Shape functions can be applied to the objective function so that the sampling distribution tends to favor samples with higher performance more than other samples or not overreact to samples with high values.

2. The initial parameters of the probability distribution and the probability distribution itself can be chosen based on knowledge about a particular problem. That is, if it has been established that some decision variables are related or a subset of decision variables have high quality solutions in a particular range, then the parameters of the sampling distribution can be chosen to exploit this information.

3. Historically good solutions can be stored so that the performance of the elite set is non-decreasing. Many existing model-based algorithms use an auxiliary memory, where they store some additional information collected during the search, which is used to update the sampling distribution.

Essentially, we can use any relevant problem information to inform us of how to modify model-based methods for peak performance. Clearly, the benefits of model-based methods are immense, and it has proven to be a very useful and powerful optimization tool. However, there does exist some shortcomings of the model-based approach. Since the model-based framework does not incorporate any structural information, this approach may lead to an inefficient algorithm when gradient information is available or the underlying structure is convex. Furthermore, model-based methods assume problems have simple feasible regions; therefore, this approach is not suitable for problems where it is difficult to obtain even one feasible solution. Those types of problems are not considered in this dissertation.

Since model-based methods are designed to tackle problems where the objective can only be obtained via black box evaluation, we can redesign problems as a simple input of decision variables and construct an output that relates back to the original problem. We attempt to answer the following questions: 1) is it possible to extend model-based methods to better solve problems that are traditionally tackled by other methods and 2) can we take advantage of the flexibility of the model-based approach to modify existing model-based methods in order to develop more effective algorithms. The hope is that the results and ideas presented in this dissertation will provide useful guidelines for designing new efficient optimization algorithms using the model-based approach.

1.1.3 Research Results and Contributions

To answer the aforesaid research questions, we propose three model-based algorithms that take different approaches from traditional techniques. We first propose an algorithm for solving the classical P-median problem using the model-based approach. We recast the P-median as another equivalent global optimization problem over the space of the extended penalty multipliers. Then, we employ a model-based method to find the extended penalty multipliers corresponding to the optimal solution of the original P- median problem.

Next, we solve the multi-objective optimization problem by using a new concept to

measure the quality of solutions. This new performance metric called domination measure can be intuitively interpreted as the size of the portion of the solution space that dominates that solution. As a result, we reformulate the original multi-objective problem into a stochastic single-objective problem and propose a model-based approach to solve it.

Finally, we propose a method that adapts the model-based approach to stochastic multi-objective optimization by incorporating an allocation heuristic. This heuristic is developed in order to improve the computational efficiency of the model-based approach in the multi-objective stochastic domain. The numerical results suggest that the proposed method is effective at searching for good solutions in the solution space while at the same time allocating the simulation replications in an efficient manner.

The next three chapters provide details on the previously mentioned approaches. In Chapter 2, we present the Lagrangian search method that uses the model-based approach to search for the optimal extended penalty multipliers that correspond to the optimal solution of the original P-median problem. Within this chapter we

- introduce the P-median problem and provide a literature review.
- formally describe the P-median problem and the Lagrangian relaxation.
- present a detailed description of the proposed algorithm.
- report some numerical results of the proposed algorithm compared with other popular methods.

In Chapter 3 we introduce domination measure as a new quality metric for multi-objective optimization problems. As a result, we reformulate the original multi-objective problem into a stochastic single objective one and propose a model-based method to solve it. In this chapter, we

- provide background on multi-objective optimization framework and existing methods.

- introduce the concept of domination measure under the setting of multi-objective optimization and use it to reformulate the original problem into a stochastic single-objective one.
- present a model-based approach to solve the reformulated problem and describe the ideal version and implementable version algorithm.
- conduct numerical experiments to demonstrate the effectiveness and advantages of the proposed approach by comparing it to existing approaches.
- provide conclusions and future directions of research.

Finally, Chapter 4 describes a method that is designed to solve optimization problems where a solution's performance is measured by multiple objectives that are subject to stochastic noise. This proposed method integrates a new allocation heuristic within the model-based framework. In this chapter, we

- describe the need for developing a method specifically designed to solve stochastic multi-objective optimization problems.
- formally describe the stochastic multi-objective problem.
- present a detailed description of the allocation heuristic and the proposed method.
- provide the results of our computational experiments.
- conclude with a discussion of some future research directions.

CHAPTER 2

A LAGRANGIAN SEARCH METHOD FOR THE *P*-MEDIAN PROBLEM

2.1 Background and Motivation

In the *P*-median problem, we are given a set of n potential facility locations for P facilities to serve m clients. The objective is to locate P facilities such that the cost of serving all customers is minimized. Generally, the parameters of this problem can be chosen such that $m \geq n$ or $m \leq n$ and $1 < P < n$. This problem arises from various applications such as clustering [1, 56] and location science [3, 28].

The *P*-median problem is well-known to be NP-hard [2, 65]. Owing to the NP-hardness of the underlying problem, exact solution methods for the P-median problem [11] usually have a formidable complexity that prevents them from solving numerous large-scale instances that have arisen from real world applications. As an alternative, many researchers have switched to the development of heuristic methods for the *P*-median problem [4, 56]. Many algorithms based on linear programming relaxation [49, 66] and methaheurstics [58, 52, 17] have been reported in the literature. Various optimization techniques including filtering [13], local search [3], Lagrangian relaxation [6], constructive methods [21, 56], and primal dual methods [65], have been tried for the *P*-median problem.

Although heuristic methods enjoy a relatively lower complexity than exact algorithms, it has been proven that they cannot guarantee a good approximation to generic *P*-median problems [61]. To circumvent such a challenge, several researchers considered a restricted subclass of the *P*-median problem, the so-called metric *P*-median problem. For the metric *P*-median problem, the assignment cost c_{ij}, the cost of serving client i from a facility at location j, satisfies the triangle inequality, i.e. $c_{ik} \leq c_{ij} + c_{jk}$. Furthermore, the cost is symmetric, i.e. $c_{ij} = c_{ji}$. Due to its nice geometric property, the metric *P*-median problem

has been well studied in the optimization community [61]. On the other hand, we point out that despite the predominant focus on solving the metric P-median problem, there exist many real world instances where the cost matrix does not satisfy the triangle inequality. For example, in the uncapacitated facility location problem, the objective is to minimize the distance weighted by the demand of each client to an open facility. Moreover, text documents are usually clustered by the cosine-similarity measure which is a non-metric measure [1].

It is evident that much work is needed to develop effective algorithms for generic P-median problems. A paucity of work has attempted to explore means of approximating such non-metric cases. It is worth mentioning that the algorithms in [49, 66, 18] obtained some approximation at a cost of violating the constraint of opening only P facilities. For example, [66] finds a solution that opens $O(P\log(m+m/\varepsilon))$ facilities and cost at most $(1+\varepsilon)opt_P$ for minimization problems, where opt_P is the optimal solution for opening P facilities. In [66] the solution is obtained by first solving the linear relaxation problem and then using a randomized rounding scheme on the fractional solutions produced by the linear relaxation problem to get an integer solution. The algorithm in [18] improves the solution obtained in [66] by greedily opening a new facility that minimizes the cost. These methods provide some foundation for solving non-metric P-median problem, but cannot be used in many real world applications where the constraint of opening at most P facilities cannot be violated. Moreover, these methods require solving large-scale linear programs, which incur high computational complexity.

One powerful technique to obtain lower bounds for combinatorial minimization problems is through the Lagrangian relaxation. Given a vector of Lagrangian multipliers, the resulting Lagrangian problem becomes simpler to solve [26]. Similar approaches have been used to find competitive and feasible solutions for problems from various applications [6, 29, 51]. In Lagrangian relaxation-based approaches, the Lagrangian multipliers are iteratively updated to maximize the lower bound as follows: first, a Lagrangian relax-

ation problem is solved to obtain a lower bound; and then a feasible solution to the original P-median problem is extracted from the solution of the Lagrangian relaxation, which provides an upper bound; in the final step, the Lagrangian multipliers are updated to improve the lower bound in the next iteration. Typically, some variant of the sub-gradient method is applied to adjust the Lagrangian multipliers, as in [5, 6]. We point out that the solution to the Lagrangian sub-problem may not be feasible for the original problem, yielding only a bound on the optimal value. Beltran et al. [8] proposed the so-called semi-Lagrangian relaxation method for the P-median problem that can provide an optimal integer solution and close the gap between the original problem and its semi-Lagrangian relaxation. However, the semi-Lagrangian relaxation of the P-median problem itself is still nontrivial to solve.

As an alternative, stochastic search methods have provided an effective approach for finding global optimal solutions for many combinatorial problems [72]. The foundation of stochastic search methods is to "explore" the entire solution space and "exploit" the promising regions of the solution space. Stochastic search methods, such as ant colony optimization [48] and simulated annealing [52, 17], have been shown in some cases to substantially improve the solution quality for larger instances [9] compared with constructive heuristics and local search methods.

2.1.1 Research Results and Contributions

We propose the Lagrangian Search (LS) method, which is able to provide an optimal or near-optimal solution to the original problem from its extended Lagrangian relaxation. The proposed LS method combines the exploration-and-exploitation aspect of stochastic search methods with the efficient approach of obtaining a solution to the extended Lagrangian relaxation via penalty multipliers. We show that the space of the penalty multipliers is usually much smaller than the solution space of the original problem, this allows us to develop an effective search algorithm in the space of the penalty multipliers. To achieve this, we first elaborate on the Lagrangian relaxation to construct a search space U of penalty

multipliers associated with the optimal solution of the relaxed problem. Then, we incorporate a model-based search method to search in the space U to find the penalty multipliers that correspond to the optimal solution of the original problem. We consider model-based search methods [12, 60, 69] because they have demonstrated the robustness in searching the solution space as opposed to their instance-based counterparts. By searching in the space of the penalty multipliers instead of the original solution space, the structure of the cost matrix has almost no bearing on the solution quality. Particularly, we focus on Gap-C instances proposed by Kochetov and Ivanenko in [39], which are defined by non-metric cost matrices where each column and row have the same number of positive integers from a set $\{1,2,\ldots,l\}$. Although there is no restriction on l, the idea is to set l to some integer not too far from zero. (In the examples given in [39] no integer is greater than 4, i.e. $l = 4$.) Compared with other P-median instances, the Gap-C instances have a relatively larger gap between the optimal value of the original problem and its Lagrangian dual [38]. Therefore, these instances have been proven to be computationally difficult for many existing meta-heuristics [39]. To deal with such an issue, we propose to utilize the Lagrangian relaxation in a non-traditional way, i.e., we create a surjective mapping from the penalty multipliers to the solution space of the original problem. In this way, we are able to recast the original P-median problem as another equivalent global optimization problem over the space of the extended penalty multipliers.

In summary, our main contributions are

- We reformulate the P-median problem as another equivalent global optimization problem over a space of the extended penalty multipliers such that there always exist an extended penalty multiplier that corresponds to the optimal solution.

- We introduce a new framework for finding the optimal penalty multipliers via a stochastic search method instead of the commonly used sub-gradient method.

- We conduct numerical experiments to test the performance of the proposed algo-

rithm, and illustrate that the proposed method performs well for a subclass of P-median problems that are known to be computationally difficult for other existing metaheuristics and exact methods in the literature.

2.2 Problem Formulation

We first describe the P-median problem where the goal is to determine the minimum cost of opening P facilities out of n possible facility locations to serve m clients. Let $\mathbf{I} = \{1, \ldots, m\}$ be the finite set of clients and $\mathbf{J} = \{1, \ldots, n\}$ be the finite set of potential facility locations. We define c_{ij} to be the cost of serving client i from a facility at location j. We assume that $c_{ij} > 0$ for all $i \in \mathbf{I}$ and $j \in \mathbf{J}$. Let

$$y_j = \begin{cases} 1 & \text{if a facility is open at location } j \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if client } i \text{ is served from a facility at location } j \\ 0 & \text{otherwise} \end{cases}$$

Therefore, we can formulate the P-median problem as the following integer linear program (ILP)

$$Z = \min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{2.2.1a}$$

$$s.t \quad \sum_{j=1}^{n} y_j = P \tag{2.2.1b}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \ \forall i = 1, \ldots, m \tag{2.2.1c}$$

$$x_{ij} \leq y_j, \quad \forall i = 1, \ldots, m, j = 1, \ldots, n \tag{2.2.1d}$$

$$x_{ij} \in \{0,1\}, \forall i = 1, \ldots, m, j = 1, \ldots, n \tag{2.2.1e}$$

$$y_j \in \{0,1\}, \forall j = 1, \ldots, n \tag{2.2.1f}$$

where constraint (2.2.1b) ensures that exactly P facilities are open, constraint (2.2.1c) guarantees that each client is served by only one facility, and constraint (2.2.1d) guarantees that a client is only served from an open facility. We use C to denote a matrix with entries c_{ij}. We let $\bar{J} = \left\{ j \in \mathbf{J} | y_j = 1 \right\}$ be the set of column indices of C that corresponds to the facilities that are open in a given solution. Because constraint (2.2.1b) must be satisfied, $|\bar{J}| = P$. Given $\bar{J}$, we can easily determine the variable x_{ij} as follows: for each i we set $x_{ij'} = 1$, where j' is an index such that $c_{ij'} = \min_{j \in \bar{J}} c_{ij}$ and $x_{ij} = 0$ for all $j \neq j'$. Essentially, we assign facility j from the set of open facilities $\bar{J}$ that has the minimum cost to client i. Consequently, we are only concerned with determining the value of the variable y_j.

2.2.1 Lagrangian Relaxation

We relax the constraint that each client has to be served by one facility. Therefore, a Lagrangian relaxation is obtained by dualizing constraint (2.2.1c)

$$\begin{aligned} Z_d &= \max_{u_i} \min_{x_{ij}, y_j} \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{i=1}^{m} u_i (1 - \sum_{j=1}^{n} x_{ij}) \\ s.t. &\quad \text{(2.2.1b), (2.2.1d), and (2.2.1e),} \end{aligned} \tag{2.2.2}$$

which is equivalent to

$$\begin{aligned} Z_d &= \max_{u_i} \min_{x_{ij}, y_j} \sum_{i=1}^{m} \sum_{j=1}^{n} (c_{ij} - u_i) x_{ij} + \sum_{i=1}^{m} u_i \\ s.t. &\quad \text{(2.2.1b), (2.2.1d), and (2.2.1e).} \end{aligned} \tag{2.2.3}$$

The objective function (2.2.3) and constraint (2.2.1d) imply the following relationship:

$$x_{ij} = \begin{cases} y_j, & \text{if } c_{ij} - u_i \leq 0 \; \forall \; i = 1, \ldots, m, j = 1, \ldots, n. \\ 0, & \text{otherwise.} \end{cases} \tag{2.2.4}$$

Therefore, given a fixed vector $u = [u_1, \cdots, u_m]$, the Lagrangian problem is transformed into the following integer program

$$\begin{aligned} Z_d(u) &= \min_{y_j} \sum_{j=1}^{n} \sum_{i=1}^{m} \phi_{ij} y_j + \sum_{i=1}^{m} u_i \\ s.t. \quad & \sum_{j=1}^{n} y_j = P, \\ & y_j \in \{0,1\}, \forall j = 1, \ldots, n, \end{aligned} \tag{2.2.5}$$

where $\phi_{ij} = \min(0, c_{ij} - u_i)$. Clearly, the set $\bar{J}$ can be determined by following a simple procedure: because y_j's are the only unknown variables, the y_j's that minimize $Z_d(u)$ are the y_j's corresponding to the P smallest sums $\sum_{i=1}^{m} \phi_{ij}$. This result is the gateway in using the stochastic search method to find the vector u that corresponds to the optimal solution.

2.3 Determining Penalty Multipliers

Throughout this section we use the following notations:

- $c_{i[t]}$ represents the t^{th} order statistic in the i^{th} row of the cost matrix C.
- $M = \left\{ i \in \mathbf{I} | c_{ij^*} < c_{ij} \; \forall \; j \neq j^* : \; j^* = \text{argmin}_k \, c_{ik} \right\}$ is the set of rows with a unique minimal element.
- $J_m = \left\{ j \in \mathbf{J} | \exists i \in M \; s.t. \; c_{ij} = c_{i[1]} \right\}$ is the set of column indices that have a unique minimal cost in at least one row of the cost matrix.
- $I(J) = \left\{ i \in \mathbf{I} | c_{i[1]} \in C_{.J} \right\}$ is the set of row indices where each row's minimal element $c_{i[1]}$ is in a given set of columns J.
- $J'_i = \left\{ j \in \mathbf{J} | c_{ij} = c_{i[1]} \right\}$ is the set of column indices which has a minimum cost in row i.
- $J' = \{ j \in \mathbf{J} | \exists i \; s.t. \; j \in J'_i \}$ is the set of columns that has a minimum element in at least one row.

- $N_i = |J_i'|$ is the number of columns that have a minimal element in row i.
- $\bar{J} = \left\{ j \in \mathbf{J} | y_j = 1 \right\}$ is the set of open facilities.
- $J^* = \left\{ j \in \mathbf{J} | y_j^* = 1 \right\}$ is the optimal set of open facilities to the original P-median problem (2.2.1a)-(2.2.1e).

2.3.1 Constructing Space of Penalty Multipliers

The best choice for u is one that maximizes the Lagrangian dual, i.e. $\max_u Z_d(u)$. Most methods seek to find u that corresponds to the optimal or near-optimal solution to the Lagrangian dual problem [26]. The primary issue with these methods is that, in most cases, the optimal solution to the Lagrangian sub-problem associated with the optimal Lagrangian multipliers may not be feasible for the original problem. In this case, the Lagrangian relaxation can at best find upper and lower bounds of the original objective function. For P-median problems with a large gap between the optimal value of the original problem and its Lagrangian dual such as the Gap-C instances described in [39], the bounds from the Lagrangian relaxation are very weak. This explains the difficulty in the branch-and-bound approach based on Lagrangian relaxation for the Gap-C instances [39].

In this work we plan to take a different course from the traditional Lagrangian relaxation based approach in the literature. We seek to find a penalty multiplier u corresponding to the optimal solution of the original problem (2.2.1a)-(2.2.1e). Essentially, we try to find a vector u whose corresponding solution $\bar{J}$ of (2.2.5) is precisely the optimal set J^*.

In order to construct a space of penalty multipliers, we need to derive a closed-form solution to the Lagrangian dual (2.2.5). To make this an easier task, we relax the Lagrangian dual problem. The extended relaxation causes no effect on the quality of the solution, since we are not concerned with the optimal value of the Lagrangian dual, but rather the connection between the penalty multipliers and some feasible solution to the original P-median problem. The Lagrangian dual sub-problem is relaxed so that every facility is

open, which implies $y_j = 1$ for all $j \in \mathbf{J}$. The Lagrangian sub-problem is described by

$$Z'_d(u) = \sum_{i=1}^{m}\sum_{j=1}^{n}\phi_{ij} + \sum_{i=1}^{m} u_i = \sum_{i=1}^{m}\sum_{j=1}^{n}\min(0, c_{ij} - u_i) + \sum_{i=1}^{m} u_i, \tag{2.3.1}$$

which is concave and continuous [26]. Since every facility is open, the optimal value to the corresponding relaxed version of the original problem, denoted by Z', is computed by summing the minimum cost of each client, that is, $Z' = \sum_i c_{i[1]}$. Each penalty multiplier u_i that maximizes $Z'_d(u)$ can be found by expanding $Z'_d(u)$ into m maximization problems

$$\begin{aligned}
\max_{u_1} Z'_{d_1}(u_1) &\triangleq \sum_{j=1}^{n}\min(0, c_{1j} - u_1) + u_1 \\
\max_{u_2} Z'_{d_2}(u_2) &\triangleq \sum_{j=1}^{n}\min(0, c_{2j} - u_2) + u_2 \\
&\vdots \\
\max_{u_m} Z'_{d_m}(u_m) &\triangleq \sum_{j=1}^{n}\min(0, c_{mj} - u_m) + u_m.
\end{aligned} \tag{2.3.2}$$

We point out that in [21], the authors discussed how to use the above series of problems to obtain a lower bound based on the Lagrangian relaxation. In contrast to [21], our study explores the feasible solution set to the above problems to build a surjective mapping from the penalty multipliers to the set of feasible solutions for the original P-median problem. Proposition 1 below ensures that a vector u maximizes the function $Z'_d(u)$ if and only if each element in vector u lies in the range between the smallest and second smallest elements of each row of the cost matrix.

Proposition 2.3.1. *A vector u maximizes $Z'_d(u)$ if and only if $u_i \in \left[c_{i[1]}, c_{i[2]}\right]$ for $i = 1, \ldots, m$.*

Proof. We first prove that if $u_i \in \left[c_{i[1]}, c_{i[2]}\right]$ for $i = 1, \ldots, m$, then $Z'_d(u)$ is maximized. For any fixed i, let $u_i = c_{i[2]}$. The function $Z'_d(u_i)$ becomes $\sum_j \min(0, c_{ij} - c_{i[2]}) + c_{i[2]}$. For all j where $c_{ij} \geq c_{i[2]}$, $\min(0, c_{ij} - c_{i[2]}) = 0$, and the function $Z'_d(u_i)$ now is reduced to $\min(0, c_{i[1]} - c_{i[2]}) + c_{i[2]} \Rightarrow c_{i[1]}$, which is the minimal element in row i. Thus, $u_i = c_{i[2]}$

obtains the maximal value for $Z'_d(u_i)$ for all i.

Next, for any fixed i, let $u_i = c_{i[1]}$. The function $Z'_d(u_i)$ becomes $\sum_j \min(0, c_{ij} - c_{i[1]}) + c_{i[1]}$. Since $\min(0, c_{ij} - c_{i[1]}) = 0$ for all j, the function $Z'_d(u_i)$ is reduced to $c_{i[1]}$, which is the same result as above. Because the function is concave and continuous, the range between $c_{i[1]}$ and $c_{i[2]}$ will also yield the same function value of $c_{i[1]}$, which is the optimal value for all i.

Lastly, we prove the "only if" part, that is if $\bar{u}$ maximizes $Z'_d(u)$, then $\bar{u}_i \in \left[c_{i[1]}, c_{i[2]}\right]$ for $i = 1, \ldots, m$.

Suppose $\exists\, i$, such that $\bar{u}_i \notin \left[c_{i[1]}, c_{i[2]}\right]$. First, let $\bar{u}_i = \beta$ where $\beta < c_{i[1]}$. The function $Z'_d(u_i)$ becomes $\sum_j \min(0, c_{ij} - \beta) + \beta = \beta$, since $\beta < c_{ij}$ for all $j \in \mathbf{J}$. Therefore, $\sum_j \min(0, c_{ij} - \bar{u}) < c_{i[1]}$, which contradicts the hypothesis that $\bar{u}$ maximizes $Z'_d(u)$. Next, assume $\bar{u}_i = \omega$ where $\omega > c_{i[2]}$. The function $Z'_d(u_i)$ becomes $\sum_j \min(0, c_{ij} - \omega) + \omega \leq c_{i[1]} - \omega + c_{i[2]} - \omega + \omega = c_{i[1]} + c_{i[2]} - \omega$, since $\min(0, c_{ij} - \omega) < 0$ for at least two distinct $j's$. Because $c_{i[2]} - \omega < 0 \Rightarrow Z'_d(\bar{u}_i) \leq c_{i[2]} + c_{i[1]} - \omega < c_{i[1]}$, which contradicts the hypothesis that $\bar{u}$ maximizes $Z'_d(u)$. Therefore, if $\bar{u}$ maximizes $Z'_d(u)$, then $\bar{u}_i \in \left[c_{i[1]}, c_{i[2]}\right]$ for all i. □

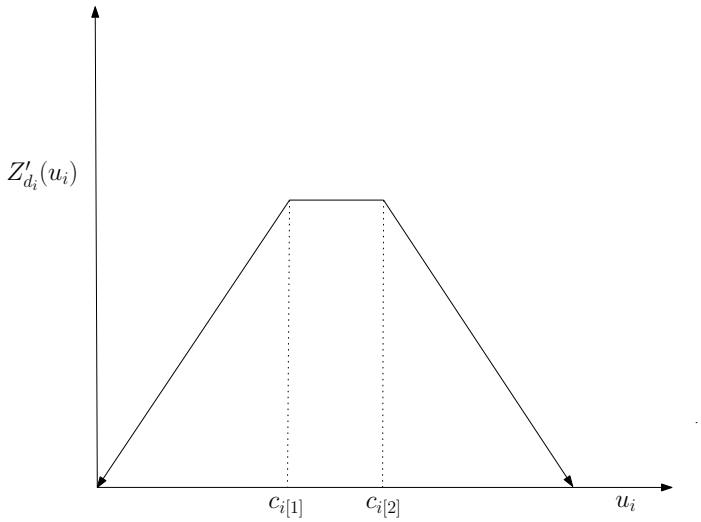


Figure 2.3.1: Illustration of Proposition 2.3.1

From the proof above, the space of penalty multipliers that maximizes the relaxed Lagrangian dual sub-problem is defined by

$U = \left\{ u \triangleq [u_1, u_2 \cdots, u_m] \middle| \; u_i \in [c_{i[1]}, c_{i[2]}], i = 1, \ldots, m \right\}$. Now that we have obtained the space of penalty multipliers that maximizes $Z'_d(u)$, we will solve the original Lagrangian dual (2.2.5) to find $\bar{J}$ (i.e. the set of open facilities) that correspond to any vector $u \in U$. We observed that a vector of penalty multipliers u yields the solution $\bar{J}$ in (2.2.5) if the following sufficient conditions hold:

1. $\sum_i \phi_{ij} < 0$ for all $j \in \bar{J}$.

2. $\sum_i \phi_{ij} = 0$ for all $j \notin \bar{J}$.

Since we want a surjective mapping from the space of penalty multipliers to the feasible set of open facilities, the idea is to have each u_i correspond to a facility j and the value of u_i to indicate whether or not $\phi_{ij} < 0$. To ensure that the space of penalty multipliers U contains the one(s) that corresponds to the optimal set J^*, we show that there exists $u \in U$ for any of all the $\binom{n}{P}$ combinations of P open facilities. First, we consider a cost matrix C where every column in C has one unique minimal element with respect to some row $i \in \mathbf{I}$. Proposition 2 shows that we can always construct a vector $u \in U$ such that the above sufficient conditions hold for any given $\bar{J}$.

Definition 2.3.1. *A cost matrix C is said to be in Class* $\mathbb{I}$*, if every column has a unique minimal element with respect to at least one row. That is,* $\mathbf{J} = J_m$.

Proposition 2.3.2. *For every cost matrix C in Class* $\mathbb{I}$*, every possible set* $\bar{J}$ *is the unique solution to the ILP (2.2.5) for some vector* $u \in U$.

Proof. Given $\bar{J}$, let $u_i = c_{i[2]}$ for all $i \in I(\bar{J})$ and $u_i = c_{i[1]}$ for all $i \notin I(\bar{J})$. We note that by definition of $I(\bar{J})$, if $i \in I(\bar{J})$, then $c_{i[1]} = c_{ij}$ for some $j \in \bar{J}$. Thus, $\sum_i \phi_{ij} y_j = \sum_{i \in I(\bar{J})} \min(0, c_{ij} - c_{i[2]}) y_j + \sum_{i \notin I(\bar{J})} \min(0, c_{ij} - c_{i[1]}) y_j$, where $\sum_{i \notin I(\bar{J})} \min(0, c_{ij} - c_{i[1]}) y_j = 0$ for all $j \in \mathbf{J}$ and $\sum_{i \in I(\bar{J})} \min(0, c_{ij} - c_{i[2]}) y_j = 0$ for all $j \notin \bar{J}$ since $c_{ij} \geq c_{i[2]}$ for all $j \notin \bar{J}$. Thus, $\sum_i \phi_{ij} = 0$ for $j \notin \bar{J}$. Given that for all $j \in \bar{J}$ there exist some $i \in \mathbf{I}$ where $c_{ij} = c_{i[1]}$ and $c_{i[1]} < c_{i[2]}$, $Z_d(u) = \sum_{i \in I(\bar{J})} \min(0, c_{ij} - c_{i[2]}) y_j < 0$ for all $j \in \bar{J}$. Therefore, $\sum_i \phi_{ij} < 0$ for $j \in \bar{J}$. This completes the proof of the proposition. □

Essentially, the strategy of the proof is based on the observation that given $\bar{J}$, $y_j = 1$ in (2.2.5) for all $j \in \bar{J}$ if and only if $\sum_i \phi_{ij} = \sum_i \min(0, c_{ij} - u_i) < 0$. In order for ϕ_{ij} to be nonzero, the unique minimal element of row i must be in column j and the value of u_i must be greater than the minimum element. Therefore, u_i determines the value ϕ_{ij}, where $\phi_{ij} < 0$ if $u_i > c_{i[1]}$ and $c_{i[1]}$ is in column j, and $\phi_{ij} = 0$ otherwise. Thus, y_j can only be assigned the value one if column j has a unique minimal element. Consequently, in order for a possible set $\bar{J}$ to be a unique solution to the ILP (2.2.5) the set $\bar{J}$ must be contained in $J_m = \left\{ j \in \mathbf{J} | \exists i \in M \; s.t. \; c_{ij} = c_{i[1]} \right\}$. To reduce the number of possible values for each u_i, we allow u_i to only take the value $c_{i[1]}$ or $c_{i[2]}$. As a result, the search space is updated to the discrete space $U = \left\{ u \triangleq [u_1, \ldots, u_m] | \; u_i \in \left\{ c_{i[1]}, c_{i[2]} \right\}, i = 1, \ldots, m \right\}$. The following example explains our construction of this search space. We note that for ease of illustration, we let T denote a matrix where each row i has all the possible values for $u_i \in \left\{ c_{i[1]}, c_{i[2]} \right\}$.

Example 1:

$$C = \begin{bmatrix} 12 & 19 & 11 & 17 \\ 16 & 13 & 16 & \mathbf{6} \\ \mathbf{1} & 16 & 9 & 21 \\ 16 & 9 & 17 & 11 \end{bmatrix}, \quad T = \begin{bmatrix} \mathbf{11} & 12 \\ 6 & \mathbf{13} \\ 1 & \mathbf{9} \\ \mathbf{9} & 11 \end{bmatrix} \tag{2.3.3}$$

For this matrix C and $P = 2$, $J^* = \{1,4\}$, $m = n = 4$, and $J_m = \{1,2,3,4\}$. Since, $J^* \subseteq J_m$, we are able to find a vector u that corresponds to J^*. We display all the possible values of u_i and their corresponding ϕ_{ij} values below:

$$\begin{bmatrix} \boldsymbol{u_1 = 11} & \rightarrow & \boldsymbol{\phi_{1\cdot} = [0,0,0,0]} \\ u_2 = 6 & \rightarrow & \phi_{2\cdot} = [0,0,0,0] \\ u_3 = 1 & \rightarrow & \phi_{3\cdot} = [0,0,0,0] \\ \boldsymbol{u_4 = 9} & \rightarrow & \boldsymbol{\phi_{4\cdot} = [0,0,0,0]} \end{bmatrix} \quad \begin{bmatrix} u_1 = 12 & \rightarrow & \phi_{1\cdot} = [0,0,-1,0] \\ \boldsymbol{u_2 = 13} & \rightarrow & \boldsymbol{\phi_{2\cdot} = [0,0,0,-7]} \\ \boldsymbol{u_3 = 9} & \rightarrow & \boldsymbol{\phi_{3\cdot} = [-8,0,0,0]} \\ u_4 = 11 & \rightarrow & \phi_{4\cdot} = [0,-2,0,0] \end{bmatrix}. \tag{2.3.4}$$

To ensure $\bar{J}$ is equal to J^*, according to the construction process in our proof of Proposition

2 we should choose the value of u such that $\sum_i \phi_{i1} < 0$, $\sum_i \phi_{i4} < 0$ and $\sum_i \phi_{i2} = \sum_i \phi_{i3} = 0$. The corresponding penalty multipliers are $u_1 = 11, u_2 = 13, u_3 = 9, u_4 = 9$. As a result, the chosen penalty multipliers applied in (2.2.5) yields the following solution:

$$\begin{aligned} \min_{y_j} \quad & \sum_{j=1}^{4}\sum_{i=1}^{4} \phi_{ij} y_j + \sum_{i=1}^{4} u_i \\ & = -8y_1 + 0y_2 + 0y_3 - 7y_4 + 42 \\ s.t. \quad & \sum_{j=1}^{4} y_j = 2, \end{aligned}$$

which solution corresponds to the optimal solution set

$$\bar{J} = J^* = \{1,4\}.$$

□

Next, we consider another class of cost matrix C where every column in C has a minimal element with respect to some row i, and certain columns have more than one minimal element.

Definition 2.3.2. *A cost matrix C is said to be in Class $\mathbb{III}$ if $\mathbf{J} = J'$ and $\mathbf{J} \neq J_m$.*

The strategy used in Example 1 relies heavily on the observation that we can always choose a $u_i \in U$ such that $u_i > c_{i[1]}$. Specifically, each column in the optimal set must have an unique smallest entry with respect to at least one row. Such a strategy cannot be applied when there are multiple smallest entries in at least one column, i.e. there exists a $j \in J^*$ where for all $i \in \left\{ i \in \mathbf{I} |\ c_{i[1]} \in C_{\cdot j} \right\}$ it is true that $c_{i[1]} = c_{i[2]}$. This is because in the algorithm, we restrict the search space to $u_i \in \left\{ c_{i[1]}, c_{i[2]} \right\}$. Therefore, whenever $c_{i[1]} = c_{i[2]}$, u_i can only take the value $c_{i[1]}$. Example 2 illustrates such a scenario.

Example 2.

$$C' = \begin{bmatrix} 4 & 0 & 0 & 1 \\ 5 & 0 & 8 & 0 \\ 0 & 7 & 6 & 0 \\ 0 & 3 & 0 & 8 \end{bmatrix}, \quad T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{2.3.5}$$

The matrix C' belongs to the Gap-C instances since each column and row has exactly 2 nonzero integers from the set $\{1,2,\ldots,8\}$. For this matrix C' with the parameters $P = 2$ and $m = n = 4$, the optimal solution of the original problem is $J^* = \{1,4\}$. Since $J^* \nsubseteq J_m = \emptyset$, we are not able to find a vector u that corresponds to J^*. We display all the possible values of u_i and their corresponding ϕ_{ij} values below:

$$\begin{bmatrix} u_1 = 0 & \rightarrow & \phi_{1\cdot} = [0,0,0,0] \\ u_2 = 0 & \rightarrow & \phi_{2\cdot} = [0,0,0,0] \\ u_3 = 0 & \rightarrow & \phi_{3\cdot} = [0,0,0,0] \\ u_4 = 0 & \rightarrow & \phi_{4\cdot} = [0,0,0,0] \end{bmatrix}$$

Because all the penalty multipliers in the search space are equal to zero, we are forced to let $u_i = 0$ for all i and the resulting $u = [0,0,0,0]$ applied in (2.2.5) yields the following solution:

$$\begin{aligned} \min_{y_j} \quad & \sum_{j=1}^{4}\sum_{i=1}^{4} \phi_{ij} y_j + \sum_{i=1}^{4} u_i \\ & = 0y_1 + 0y_2 + 0y_3 + 0y_4 + 0 \\ s.t. \quad & \sum_{j=1}^{4} y_j = 2, \end{aligned} \tag{2.3.6}$$

which has multiple optimal solutions corresponding to (2.2.5). Specifically, the set $\bar{J}$ that corresponds to the multiple optimal solutions can be any of the $\binom{4}{2}$ combinations of 2 open facilities.

□

The multiplicity of the optimal solutions in (2.3.6) is caused by the non-uniqueness of the minimal element in every row. As previously stated, in order for a column j to be a solution for some $u \in U$ the column j must have a unique minimal element with respect to at least one row. Therefore, a cost matrix in which there exists a column that does not contain a unique minimal element presents problems with our current search space. Recall that u_i determines the value ϕ_{ij}, where $\phi_{ij} < 0$, if $u_i = c_{i[2]} > c_{i[1]}$ and the minimal element $c_{i[1]}$ is located in column j. For cost matrices in Class $\mathbb{I}$, we always can find a $c_{i[2]} > c_{i[1]}$ for all i. Now we consider cost matrices in Class $\mathbb{III}$ where there exists an i such that $c_{i[2]} = c_{i[1]}$. In order to have u_i determine the value ϕ_{ij} when $c_{i[2]} = c_{i[1]}$, we expand each penalty multiplier u_i into a row vector $u_{i\cdot}$ such that $u_{i\cdot}$ can take multiple possible values. For each column j' that has a minimum element in row i there exists a $u_{i\cdot}^{j'}$ such that $\phi_{ij'} < 0$. This is accomplished by setting

$$u_{ij}^{j'} = \begin{cases} c_{i[2']} & \text{if } j = j'. \\ c_{i[1]} & \forall j \in \mathbf{J} \setminus \{j'\}, \end{cases}$$

where $c_{i[2']}$ denotes the second smallest element in row i with a distinct value from the smallest element. Given this new representation of u we replace u_i with u_{ij} in (2.2.5). Thus, for a given matrix $u = [u_{1\cdot}, \cdots, u_{m\cdot}]$, we solve the Lagrangian dual problem

$$\begin{aligned} Z_d^*(u) &= \min_{y_j} \sum_{j=1}^{n} \sum_{i=1}^{m} \phi_{ij} y_j + \sum_{j=1}^{n} \sum_{i=1}^{m} u_{ij} \\ s.t. \quad & \sum_{j=1}^{n} y_j = P, \\ & y_j \in \{0,1\}, \forall j = 1, \ldots, n, \end{aligned} \tag{2.3.7}$$

where ϕ_{ij} is updated to $\phi_{ij} = \min(0, c_{ij} - u_{ij})$. Therefore, $u_{i\cdot}^{j'}$ determines the value $\phi_{i\cdot}$, where $\phi_{ij} < 0$ for $j = j'$ and $\phi_{ij} = 0$ for all $j \neq j'$. In the case where we need $\phi_{ij} = 0$ for all

j we add the penalty multiplier

$$u_{ij}^0 \quad = \quad c_{i[1]} \qquad \forall j \in \mathbf{J}.$$

This penalty multiplier is added because $\sum_i \phi_{ij}$ must equal zero for all $j \notin \bar{J}$. With this new formulation $\bar{J}$, we can ensure that $\sum_i \min(0, c_{ij} - u_{ij}) < 0$ for all $j \in \bar{J}$ and $\sum_i \min(0, c_{ij} - u_{ij}) = 0$ for all $j \notin \bar{J}$, which satisfies the sufficient conditions mentioned previously. As a result, for Class $\mathbb{III}$ matrices, we now have a surjective mapping from the space of extended penalty multipliers to the feasible set of open facilities.

In the case where all the entries of a row are equal, we can delete that row in prepossessing, because the facility that is assigned to that client can be arbitrarily chosen. Therefore, the inequality $c_{i[2']} > c_{i[1]}$ will always hold. In consequence, the search space is updated to $U = \left\{ u \triangleq [u_{1\cdot}; \ldots; u_{m\cdot}] \middle| \, u_{i\cdot} \in \left\{ u_{i\cdot}^0, u_{i\cdot}^{J_i'(1)}, \cdots, u_{i\cdot}^{J_i'(N_i)} \right\}, i = 1, \ldots, m \right\}$. Due to the new construction of the search space, the vector u is replaced by an m by n matrix

$$u = \begin{bmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & & \vdots \\ u_{m1} & \cdots & u_{mn} \end{bmatrix},$$

where we slightly abuse the notation u to denote a matrix.

Proposition 2.3.3. *For every cost matrix C in Class $\mathbb{III}$, each possible set $\bar{J}$ is the unique solution to $Z_d^*(u)$ for some matrix $u \in U$.*

Proof. Let $u_{i\cdot} = u_{i\cdot}^0$ for all $i \notin I(\bar{J})$ and $u_{i\cdot} = u_{i\cdot}^{j'}$ for all $i \in I(\bar{J})$ where $j' \in \bar{J}$. Thus, $\sum_i \min(0, c_{ij} - u_{ij}) y_j = \sum_{i \in I(\bar{J})} \min(0, c_{ij} - u_{ij}^0) y_j + \sum_{i \notin I(\bar{J})} \min(0, c_{ij} - u_{ij}^{j'}) y_j$, where $\sum_{i \notin I(\bar{J})} \min(0, c_{ij} - u_{ij}^0) y_j = 0$ for all $j \in \mathbf{J}$ and $\sum_{i \in I(\bar{J})} \min(0, c_{ij} - u_{ij}^{j'}) y_j = 0$ for all $j \notin \bar{J}$ by construction of $u_{i\cdot}^{j'}$, leading to $\sum_i \phi_{ij} < 0$ for $j \in \bar{J}$ and $\sum_i \phi_{ij} = 0$ for $j \notin \bar{J}$. This completes the proof of the proposition. □

It is clear that Class $\mathbb{I}$ can be considered a special case of Class $\mathbb{III}$. Therefore, the

space of U can also be applied for cost matrices in Class $\mathbb{I}$. The effectiveness of this search space is demonstrated by Example 3, which further illustrates that for a given matrix C' belonging to the Gap-C instances, the extended representation of the penalty multipliers allows the method to find the penalty multipliers corresponding to the optimal solution.

Example 3.

$$C' = \begin{bmatrix} 4 & 0 & 0 & 1 \\ 5 & 0 & 8 & \mathbf{0} \\ \mathbf{0} & 7 & 6 & 0 \\ \mathbf{0} & 3 & 0 & 8 \end{bmatrix}, \quad T = \begin{bmatrix} \boldsymbol{u^0_{1.}} & u^2_{1.} & u^3_{1.} \\ u^0_{2.} & u^2_{2.} & \boldsymbol{u^4_{2.}} \\ u^0_{3.} & \boldsymbol{u^1_{3.}} & u^4_{3.} \\ u^0_{4.} & \boldsymbol{u^1_{4.}} & u^3_{4.} \end{bmatrix} \tag{2.3.8}$$

Note that the matrix C' remains the same as in Example 2. We display all the possible values of u and their corresponding ϕ_{ij} values below:

$$\begin{bmatrix} \boldsymbol{u^0_{1.} = [0,0,0,0]} & \rightarrow & \boldsymbol{\phi_{1.} = [0,0,0,0]} \\ u^0_{2.} = [0,0,0,0] & \rightarrow & \phi_{2.} = [0,0,0,0] \\ u^0_{3.} = [0,0,0,0] & \rightarrow & \phi_{3.} = [0,0,0,0] \\ u^0_{4.} = [0,0,0,0] & \rightarrow & \phi_{4.} = [0,0,0,0] \end{bmatrix}$$

$$\begin{bmatrix} u^2_{1.} = [0,1,0,0] & \rightarrow & \phi_{1.} = [0,-1,0,0] \\ u^2_{2.} = [0,5,0,0] & \rightarrow & \phi_{2.} = [0,-5,0,0] \\ \boldsymbol{u^1_{3.} = [6,0,0,0]} & \rightarrow & \boldsymbol{\phi_{3.} = [-6,0,0,0]} \\ \boldsymbol{u^1_{4.} = [3,0,0,0]} & \rightarrow & \boldsymbol{\phi_{4.} = [-3,0,0,0]} \end{bmatrix}$$

$$\begin{bmatrix} u^3_{1\cdot} = [0,0,1,0] & \rightarrow & \phi_{1\cdot} = [0,0,-1,0] \\ \boldsymbol{u^4_{2\cdot} = [0,0,0,5]} & \boldsymbol{\rightarrow} & \boldsymbol{\phi_{2\cdot} = [0,0,0,-5]} \\ u^4_{3\cdot} = [0,0,0,6] & \rightarrow & \phi_{3\cdot} = [0,0,0,-6] \\ u^3_{4\cdot} = [0,0,3,0] & \rightarrow & \phi_{4\cdot} = [0,0,-3,0] \end{bmatrix}$$

To ensure $\bar{J}$ is equal to J^*, we select the penalty multipliers according to the construction process in our proof of Proposition 3. Since $J^* = \{1,4\}$ and $I(J^*) = \{2,3,4\}$, we choose the penalty multipliers $u^4_{2\cdot}$, $u^1_{3\cdot}$, and $u^1_{4\cdot}$ which assures that $\sum_i \phi_{i1} < 0$ and $\sum_i \phi_{i4} < 0$. Furthermore, we choose the last penalty multiplier $u^0_{1\cdot}$. As a result of the chosen penalty multipliers, $\sum_i \phi_{i2} = \sum_i \phi_{i3} = 0$. Therefore, the value for (2.3.7) becomes

$$\begin{aligned} \min_{y_j} \quad & \sum_{i=1}^{4}\sum_{j=1}^{4} \min(0, c_{ij} - u_{ij}) y_j + \sum_{j=1}^{4}\sum_{i=1}^{4} u_{ij} \\ & = -9y_1 + 0y_2 + 0y_3 - 5y_4 + 14 \\ s.t. \quad & \sum_{j=1}^{4} y_j = 2 \\ \bar{J} = J^* \quad & = \{1,4\}, \end{aligned}$$

which is the desired result.

□

The strategy used above depends on the assumption that every column has a minimum element with respect to some row $i \in \mathbf{I}$. Therefore, cost matrices where there exists a column that does not contain a minimum element presents complications with our current search space. We can circumvent this issue by converting such a cost matrix into a matrix where every column has a minimal element of a certain row.

Definition 2.3.3. *A cost matrix C is said to be in Class* $\mathbb{III}$ *if* $\mathbf{J} \neq J'$*, i.e.,* J' *is a strict subset of* $\mathbf{J}$.

We can transform a cost matrix C in Class $\mathbb{III}$ into a cost matrix C' in Class $\mathbb{II}$ by modifying the cost matrix so that a minimum element is included in every column. One

way to accomplish this transformation is to perturb the cost matrix such that every column has a minimal element and use the original cost matrix to determine the objective value. This allows us to solve (2.3.7) for any possible $\bar{J}$ while not affecting the original objective value.

Let J'^c be the set of column indices that are not in J', i.e., a column in J'^c does not contain a minimal element of any row. First, we define $C_{.J'^c}$ as a sub-matrix such that each column has no minimum element respect to any row i of the original cost matrix C. Next, for each column j in $C_{.J'^c}$, we subtract $c_{ij} - c_{i[1]}$ from an entry c_{ij}. This procedure ensures that there is a $c_{i[1]}$ in every column. The perturbed cost matrix is only used in the Lagrangian sub-problem to obtain the solution $\bar{J}$, and the original cost matrix is used to obtain the value of $\bar{J}$. Therefore, the optimal solution is unaffected by this perturbation.

Example 4.

$$C = \begin{bmatrix} 6 & 3 & \mathbf{1} & 4 \\ 5 & 2 & \mathbf{1} & 5 \\ 4 & \mathbf{1} & 6 & 6 \\ \mathbf{1} & 6 & 6 & 2 \end{bmatrix} \Rightarrow C' = \begin{bmatrix} 6 & 3 & \mathbf{1} & 4 \\ 5 & 2 & \mathbf{1} & \mathbf{1} \\ 4 & \mathbf{1} & 6 & 6 \\ \mathbf{1} & 6 & 6 & 2 \end{bmatrix} \quad or \; C' = \begin{bmatrix} 6 & 3 & \mathbf{1} & \mathbf{1} \\ 5 & 2 & \mathbf{1} & 5 \\ 4 & \mathbf{1} & 6 & 6 \\ \mathbf{1} & 6 & 6 & 2 \end{bmatrix}$$

For this example $J'^c = 4$, which results in $C_{.J'^c} = [4\ 5\ 6\ 2]^T$. We choose a minimum element from column 3, since it has a minimum element with respect to rows 1 and 2. We either subtract $c_{24} - c_{23}$ from entry c_{24} or $c_{14} - c_{13}$ from c_{14} to construct the perturbed cost matrix C'.

□

We point out that we generally cannot transform a cost matrix in Class $\mathbb{III}$ into a cost matrix in Class $\mathbb{I}$. In particular, when a cost matrix has more columns than rows there exists no perturbation of that matrix that will cause each column to have a unique minimal element. Therefore, our construction of matrix u is needed to solve this problem for cost

matrices with arbitrary dimensions.

Since any cost matrix can be transformed, or be considered to belong in Class $\mathbb{III}$, Proposition 3 plays a critical role in the effectiveness of our method. Specifically, for P-median problems we extend the Lagrangian relaxation by representing the penalty multipliers via a matrix and reduce the space of penalty multipliers to the search space U. Proposition 3 guarantees this reduction retains a $u \in U$ that corresponds to the optimal solution of the original problem. Fortunately, the expansion of the penalty multipliers from a vector to a matrix does not significantly increase the size of the search space U or the memory requirements. This yields a novel way to reformulate the original P-median problem as another global optimization problem of the extended penalty multipliers. In implementation, we only need to store $N_i + 1$ elements (each of which is a vector) in each row of matrix T as displayed in (2.3.8). Below, we explain how to find the matrix of optimal penalty multipliers u^* that corresponds to the optimal solution of the original problem.

2.3.2 Finding Optimal Penalty Multipliers

We convert the original optimization problem into a new optimization problem. This new problem is to find a solution u^* to the problem

$$\operatorname*{argmin}_{u \in U} h(u), \tag{2.3.9}$$

where $h(u) : \mathbb{R}^{mxn} \to \mathbb{R}$ is a function whose image is the value of the original objective function $\sum_{i,j} c_{ij} x_{ij}$. The function $h(u)$ is evaluated in the following steps:

1. For u, find a solution $\bar{J}$ in the ILP (2.3.7). If the cost matrix C falls in Class $\mathbb{IIII}$, use the transformed matrix C'.

2. Compute the corresponding $x'_{ij}s$ as follows: for each i set $x_{ij'} = 1$, where j' is an index such that $c_{ij'} = \min_{j \in \bar{J}} c_{ij}$ and $x_{ij} = 0$ for all $j \neq j'$.

3. Compute the objective function $\sum_{i,j} c_{ij} x_{ij}$ from $x'_{ij}s$ determined in step 2.

In cases where the cost matrix C is augmented to C', matrix C' is only used in step 1 to obtain the corresponding columns. The original cost matrix is used to compute the objective function value. For this reason, the optimal solution for the original problem is not influenced by the transformation of C to C'.

Due to the lack of structure of the objective function $h(u)$, a stochastic search method is desirable for finding the global optimal solution. We incorporate a model-based search algorithm to search in U. The model-based search algorithm is broken up into two primary steps at each iteration: 1) generate candidate solutions from a sampling distribution, and 2) update the sampling distribution such that future sampling is biased towards high quality solutions in the next iteration. In our context, we try to find high quality solutions of the penalty multiplier u in the search space U we constructed. The sampling distribution is parameterized by a probability matrix F whose $(i,j)^{\text{th}}$ element represents the probability of selecting the $(i,j)^{\text{th}}$ element of matrix T. Matrix T is constructed such that each possible penalty multiplier $u_{i\cdot}$ is located in row i. (For the cost matrix in Example 3, $F_k(3,1)$ is the probability that $u_{3\cdot} = u^0_{3\cdot}$ at iteration k.) This probability matrix is used to determine the probability distribution over all possible matrices of penalty multipliers u. We note that matrix T is not a matrix in the typical sense, because its elements are row vectors and it does not necessarily have the same number of elements in every row. However, we use this notation for the convenience of explaining the implementation of our algorithm. Because the probability of selecting a matrix of penalty multipliers for each $i \in \mathbf{I}$ is independent, the probability of a specific matrix of penalty multipliers is the product of the probabilities of selecting $u_{i\cdot}$ for each $i \in \mathbf{I}$. Consequently, the probability mass function $f(\cdot;F_k)$ of the penalty multiplier is given by:

$$f(u;F_k) = \prod_{i=1}^{m} \sum_{j=1}^{n} F_k(i,j) I_{[u_{i\cdot}=T_{ij}]}, \tag{2.3.10}$$

where the indicator function $I_{[u_{i\cdot}=T_{ij}]} = 1$ if $u_{i\cdot} = T_{ij}$ and $I_{[u_{i\cdot}=T_{ij}]} = 0$ otherwise. The probability matrix can be updated according to a particular model-based optimization algorithm such as the Cross Entropy Method [60], Model Reference Adaptive Search [12], and Gradient-Based Adaptive Stochastic Search [69]. Implementation details will be discussed in the next section.

In summary, we significantly reduce the search space of penalty multipliers by constructing a space U where each penalty multiplier maximizes the dual function. Then, we incorporate a stochastic search method to search in U to find the penalty multiplier that minimizes the objective function $h(u)$. The description of the Lagrangian Search Method is presented below.

Algorithm 1 Lagrangian Search Method

Input: Cost matrix C, and value for P.
Output: Value $\sum_{i,j} c_{ij}x_{ij}$, and assignments x_{ij} for all i and j.

1. Make appropriate transformation of cost matrix C to ensure each column has a minimal element with respect to some row.

2. For each row i, construct the set of columns J_i' which have a minimum cost in row i.

2. Construct search space of all possible penalty multiplier matrices: $U = \left\{u \triangleq [u_{1\cdot};\ldots;u_{m\cdot}] |\ u_{i\cdot} \in \left\{u_{i\cdot}^0, u_{i\cdot}^{J_i'(1)}, \ldots, u_{i\cdot}^{J_i'(N_i)}\right\}\right\}$, where u is a matrix of penalty multipliers.

3. Implement a stochastic search method to find the solution to $\min_{u\in U} h(u)$.

2.3.3 Optimal Penalty Multipliers vs Optimal Lagrangian Multipliers

It is important to distinguish the differences between the optimal penalty multipliers and the optimal Lagrangian multipliers. Although both are motivated by Lagrangian relaxation, the two optimal multipliers are derived from two different objectives. As mentioned previously, the optimal penalty multipliers u^* are those that map to the optimal set of open facilities of

the original P-median problem. On the other hand, the optimal Lagrangian multipliers $\bar{u}^*$ maximizes the Lagrangian dual

$$Z_d^* = \max_{\bar{u}} Z_d(\bar{u}), \tag{2.3.11}$$

which can be solve in a straightforward way [26]. Let the set

$$X = \left\{ x \middle| \sum_{j=1}^{n} y_j = P, \sum_{j=1}^{n} x_{ij} > 0, x_{ij} \leq y_j, \text{and } x_{ij} \in \{0,1\} \ \forall i,j \right\}$$

contain all feasible solutions to the Lagrangian dual problem. Due to X being a finite set, all feasible solutions can be enumerated . Let $|X| = T$ and denote x^t as a solution belonging to X, where $t = 1,...,T$. Following this, we can express (2.3.11) as the following linear program

$$\begin{aligned} Z_d^* &= \max w, \\ s.t.\ w &\leq \sum_{i=1}^{4}\sum_{j=1}^{4} c_{ij}x_{ij}^t + \sum_{i=1}^{4} \bar{u}_i(1 - \sum_{j=1}^{4} x_{ij}^t),\ \ t = 1,...,T. \end{aligned} \tag{2.3.12}$$

To explain the difference between the two optimal multipliers we execute CPLEX to find the optimal Lagrangian multipliers of the cost matrices C and C' that correspond to Example 1 and Example 3 respectively.

Example 5

$$C = \begin{bmatrix} 12 & 19 & 11 & 17 \\ 16 & 13 & 16 & 6 \\ 1 & 16 & 9 & 21 \\ 16 & 9 & 17 & 11 \end{bmatrix}, \quad C' = \begin{bmatrix} 4 & 0 & 0 & 1 \\ 5 & 0 & 8 & 0 \\ 0 & 7 & 6 & 0 \\ 0 & 3 & 0 & 8 \end{bmatrix}$$

For matrix C and $P = 2$, the optimal value $Z^* = 30$ and for matrix C' and $P = 2$, the optimal value $Z^* = 1$. Since the dimensionality of C and C' is small the set of feasible solutions

can easily be determined. Recognize that the feasible set of solutions are equivalent for both cost matrices. Solving (2.3.11) we obtain the optimal Lagrangian multipliers and the corresponding Lagrangian dual value for both C and C' via CPLEX, which are given below.

- Matrix C: $\bar{u}^* = [14, 16, 21, 16]$ and $Z_d(\bar{u}^*) = 30$
- Matrix C': $\bar{u}^* = [1, 3, 1, 3]$ and $Z_d(\bar{u}^*) = 0 < Z^*$

□

Although both the optimal Lagrangian multipliers and the optimal penalty multipliers obtain the optimal value Z^* for matrix C, the multipliers themselves are different; that is, $u^* \neq \bar{u}^*$. Since, matrix C' causes a duality gap, the optimal Lagrangian multipliers are only able to obtain a lower bound on the optimal value Z^*. In contrast, Example 3 shows that the extended penalty multipliers are able to obtain the optimal value Z^*. In conclusion, the main difference between the two optimal multipliers is that in the case where their exist a duality gap by definition the optimal Lagrangian multipliers do not achieve the optimal value of the original problem. Yet, by construction there always exist an optimal extended penalty multiplier that corresponds to the optimal solution of the original problem.

2.4 Numerical Results

We tested our method on instances from the Discrete Location Problems library available at <http://math.nsc.ru/AP/benchmarks/P-median/p-med_eng.html>, which are instances with large gap (Gap-C) between the original P-median problem and its Lagrangian relaxation, instances on perfect codes (PCodes), instances on chess boards (Chess), and instances on finite protective planes (FPP). The other methods considered in the comparison are the following:

- Hybrid Heuristic [56], which is based on a greedy randomized adaptive search procedure, swap-based local search, and path-relinking for the P-median problem.

- The method presented in [54], which uses a cooperative exact approach where tabu search procedure in the primal is combined with a Lagrangian branch and bound search in the dual.

- CPLEX, which to our knowledge is one of the most effective commercial mixed integer program solver.

For all problems, we incorporated the Monte Carlo version of the Model Reference Adaptive Search (MRAS) method [34] as the stochastic search method in step 3 of our algorithm. The MRAS method is competitive in solving global optimization problems in combinatorial and continuous domains and has been proven to converge asymptotically to an optimal solution. The MRAS method generates candidate solutions from a parameterized probability distribution on the solution space at each iteration, and uses a sequence of reference distributions to lead the updating process of the parameterized distribution such that it assigns higher probability on the set of optimal solutions. For more details of the MRAS method, we refer the reader to [34]. We incorporate the MRAS method in the following manner:

Initialization: Set $k = 0$ and let F_0 be a uniform distribution over T.

1. Generate N random penalty multipliers matrices $u^1, \ldots, u^N$ from probability matrix F_k. For each i, sample the penalty multiplier located in the $(i,j)^{\text{th}}$ entry of T with probability $F_k(i,j), j = 1, \ldots, N_i$.

2. Calculate the objective value $h(u^i)$ for all i and order them from smallest to largest, $h_{[1]} \leq \ldots \leq h_{[N]}$. Let $\gamma_k = h_{[\lceil pN \rceil]}$ be the p sample quantile of the objective values, where $p \in [0,1]$.

3. Update the probability matrix in the k^{th} iteration by

$$F_k(i,j) = \frac{\sum_{n=1}^{N} S(h(u^n))^k / f(u^n; F_{k-1}) I_{[h(u^n) \leq \gamma_k]} I_{[u_{i\cdot}^n \in T_{ij}]}}{\sum_{n=1}^{N} S(h(u^n))^k / f(u^n; F_{k-1}) I_{[h(u^n) \leq \gamma_k]}}, \tag{2.4.1}$$

where $S(\cdot): \mathbb{R} \to \mathbb{R}^+$ is a strictly decreasing function.

4. Perform smoothing update for the probability matrix

$$F_k := \psi F_k + (1-\psi)F_{k-1}, \tag{2.4.2}$$

where $0 < \psi \leq 1$.

5. If the stopping criteria is not met, set $k := k+1$ and go back to step 1.

For each problem instance, we performed 30 independent replications of our algorithm implemented in MATLAB. We report the best solution value (V^*), worst solution value (V'), standard error (δ), and average running time in seconds obtained out of the 30 trials. We chose the following initial parameters: sample size $N = 1000$, percent quantile $p = .05$, smoothing constant $\psi = .8$, $S(h(u)) = e^{-kh(u)}$, and the initial probability matrix F is uniformly distributed. The algorithm was terminated when the top performing sample was the same value for four consecutive iterations. The computational times for the exact cooperative method were found in [54], which only reports the statistics of the times on all 30 instances for each problem class (Table 5). For the Gap-C instances, the time for our LS method was found to be significantly shorter than CPLEX and the minimum time of the exact cooperative method, while achieving the optimal value for 28 out of 30 Gap-C instances. Furthermore, the LS method obtained better solutions than the Hybrid Heuristic method for all of the Gap-C instances. For the other classes of problem instances, CPLEX and the exact cooperative method outperformed the LS method in computational time while achieving the same optimal value as the LS method. These results were expected due to the the small gap between the underlying problem and its Lagrangian relaxation. We point out that CPLEX and the exact cooperative method obtained the optimal solution for every instance. The numerical experiments presented in Table 1-4 were performed on a computer with an Intel Core i5-4210U 2.40 GHz processor, 5.91 GB of memory, and a Windows 8 operating system. The results in Table 5 are based on implementation in C and testing on

workstations equipped with an Intel i7-2600 CPUs 3.4 GHz processor [54].

In summary, although the Hybrid Heuristic was notably faster than the LS Method, the Hybrid Heuristic could not consistently achieve values as good as the LS method. Moreover, the LS method solves the Gap-C instances in significantly shorter time than both CPLEX and the exact cooperative method. The low standard error δ and the mean value V' of the instances indicates that our algorithm is robust with respect to the randomness in the algorithm. The numerical results below demonstrate that for Gap-C instances, the LS method outperforms the Hybrid Heuristic with respect to solution quality. Furthermore, although the exact cooperative method and the LS method were implemented on different machines, due to the large difference in computational times, it is fair to conclude that the LS method outperforms the exact cooperative method in terms of computational time for Gap-C instances.

2.5 Conclusion

In this chapter, we presented a method that solves the P-median problem by searching in the space of the extended penalty multipliers. We substantially reduce the space of the extended penalty multipliers by only considering the penalty multipliers that are solutions to the extended Lagrangian relaxation. The combination of constructing and searching in the space of the extended penalty multipliers provides a solid foundation for our algorithm which yields satisfactory solutions in competitive computational experiments. This provides a new framework within the Lagrangian-type methods by selecting the optimal penalty multipliers via a stochastic search method. Furthermore, this general framework has the potential to yield computationally effective heuristic algorithms for other combinatorial optimization problems.

Table 2.4.1: Gap-C: $n = m = 100$ and $P = 14$

Gap	Optimal Value	Lagrangian Search				Hybrid Heuristic		CPLEX
		V_L^*	V'	δ	Time (sec)	V_H^*	Time (sec)	Time (sec)
39.73	147	147	149	0.33	54	153	.28	17025
58.77	145	145	147	0.01	53	147	.17	25607
59.56	142	142	146	0.42	67	148	.27	17131
41.91	144	144	144	0	48	158	.27	20550
41.52	137	137	140	0.02	52	146	.11	20532
34.83	139	139	142	0.01	59	1146	.11	18567
37.65	130	130	133	0.2	63	1120	.13	20815
43.06	138	138	143	0.67	65	143	.20	22832
43.36	147	147	150	0.23	72	164	.13	26225
41.17	142	142	144	0	70	146	.13	21867
44.39	140	140	147	0.51	67	172	.11	22412
44.20	152	152	158	0.67	62	172	.13	23548
41.94	133	133	137	0.31	65	148	.16	20042
45.24	136	136	144	0.2	63	144	.23	26106
43.98	134	134	138	0.41	69	141	.17	17073
45.56	136	136	136	0	74	154	.22	25665
38.16	137	137	148	0.35	59	145	.16	23591
41.56	140	140	146	0	61	163	.11	56783
36.30	138	140	150	0.2	50	157	.13	23458
40.94	121	121	128	0.1	67	121	.13	20055
42.29	133	133	133	0	63	149	.11	24691
43.92	139	139	145	0.28	61	145	.11	15434
42.17	131	131	139	0	69	142	.11	41641
40.18	132	132	145	0.82	68	156	.11	11852
37.31	136	136	136	0	76	139	.11	32014
41.92	137	137	141	0.61	81	159	.11	11902
37.33	124	124	124	0	52	136	.11	36892
39.29	137	137	137	0	59	142	.17	22212
47.59	141	141	141	0	54	163	.22	26951
40.96	129	130	136	0.51	67	1127	.13	39681

Table 2.4.2: PCodes: $n = m = 128$ and $P = 7$

Gap	Optimal Value	Lagrangian Search				Hybrid Heuristic		CPLEX
		V_L^*	V'	δ	Time (sec)	V_H^*	Time (sec)	Time (sec)
5.98	232	232	234	0.33	62	232	.16	2.24
2.72	217	217	225	0.43	69	218	.16	1.49
1.25	227	227	234	1.21	56	227	.16	1.53
1.96	219	219	225	.39	58	219	.16	1.86
4.14	223	223	227	2.82	52	227	.16	2.24
1.46	215	215	224	1.21	72	215	.16	1.75
1.92	221	221	226	1.2	78	229	.20	1.70
2.48	217	217	228	0.27	68	220	.16	1.88
4.40	220	220	220	0.01	74	221	.20	2.34
1.51	223	223	223	0	79	223	.16	1.60
1.51	211	211	219	1.01	81	211	.16	1.66
2.77	226	226	230	0.82	84	226	.16	2.15
0.17	209	209	215	0.91	75	209	.16	1.64
0.17	226	226	230	0.63	71	230	.16	2.01
0.17	221	221	221	0.0	73	221	.16	2.11
1.95	225	225	235	0.42	81	225	.16	1.72
1.74	222	222	226	0.12	68	222	.20	1.66
0.00	205	205	221	1.98	67	205	.22	1.57
0.77	226	226	233	0.87	62	232	.16	1.72
2.79	229	229	239	0.46	68	229	.20	1.87
0.74	215	215	215	0.0	65	215	.14	1.42
2.83	212	212	216	0.23	71	213	.22	2.06
0.80	209	209	209	0.0	76	209	.16	1.73
1.31	207	207	211	0.51	84	207	.14	1.89
.48	220	220	220	0	.72	220	.16	1.89
2.06	220	220	227	0.61	76	224	.16	1.98
0.00	207	207	212	0.35	72	207	.16	1.57
0.00	222	222	231	0.24	89	229	.16	1.73
2.37	223	223	228	0.21	92	223	.26	2.04
3.44	204	204	204	0.0	78	206	.14	1.90

Table 2.4.3: FPP: $n = m = 133$ and $P = 11$

Gap	Optimal Value	Lagrangian Search				Hybrid Heuristic		CPLEX
		V_L^*	V'	δ	Time (sec)	V_H^*	Time (sec)	Time (sec)
12.20	230	230	230	0.03	62	233	.17	12.45
11.59	220	220	224.32	2.32	60	223	.19	8.52
11.33	233	233	234.78	1.42	69	233	.19	11.26
14.62	221	221	222.63	1.56	56	221	.16	17.37
10.68	226	226	227.49	1.32	59	226	.19	14.93
11.42	236	236	237.83	1.01	61	240	.19	11.38
7.02	211	211	211.71	0.83	68	224	.20	2.71
10.47	220	220	220.82	0.69	62	220	.17	3.50
10.00	225	225	226.31	0.92	78	231	.17	8.64
5.40	228	228	228.91	0.73	80	228	.19	4.45
12.86	219	219	220.61	1.31	69	219	.17	15.20
10.66	239	239	240.18	0.67	65	239	.17	4.69
11.90	226	226	227.33	1.11	75	226	.17	11.25
6.24	224	224	224.67	0.35	63	241	.19	10.08
12.23	236	236	236	0.0	79	236	.17	15.01
7.69	219	219	219.58	0.89	79	219	.17	4.52
7.76	234	234	234.61	0.47	69	234	.19	14.23
11.02	224	224	224.99	0.71	67	224	.17	15.53
5.16	234	234	234.72	0.53	70	244	.19	4.22
9.12	228	228	228.56	0.19	62	228	.19	7.18
9.48	221	221	221.78	0.43	73	221	.17	8.49
14.47	223	223	224.31	0.88	81	223	.19	4.14
6.79	223	223	225.02	1.98	66	230	.17	2.90
13.04	216	216	218.43	2.82	78	216	.17	4.02
10.48	216	216	216.76	0.31	86	216	.17	3.84
9.30	232	232	233.98	1.61	71	232	.17	4.23
6.76	222	222	224.67	2.47	62	222	.19	12.43
10.42	227	227	228.32	.93	89	227	.19	4.12
6.59	213	213	213.43	0.36	71	213	.19	4.00
8.50	231	231	231.72	0.48	77	231	.17	9.21

Table 2.4.4: Chess: $n = m = 144$ and $P = 4$

Gap	Optimal Value	Lagrangian Search				Hybrid Heuristic		CPLEX
		V_L^*	V'	δ	Time (sec)	V_H^*	Time (sec)	Time (sec)
7.17	258	258	259.31	1.23	72	261	.20	2.49
4.82	247	247	247.67	0.48	61	247	.27	2.13
1.89	246	246	249.21	2.48	59	246	.20	2.54
26.38	235	235	236.30	1.20	55	235	.20	2.15
9.96	257	257	257.67	0.89	64	257	.20	3.04
3.76	236	236	237.23	0.91	71	236	.27	2.23
6.21	249	249	251.03	2.22	83	251	.34	2.66
23.01	239	239	239.92	1.47	69	239	.36	2.42
28.88	232	232	232.76	0.83	76	232	.27	2.65
6.23	244	244	244.68	0.45	78	244	.27	2.50
6.44	249	249	251.22	2.51	97	249	.34	2.90
6.43	247	247	248.11	1.13	84	247	.20	3.66
3.01	247	247	247.81	1.31	81	247	.25	2.22
23.20	250	250	250.25	0.28	76	252	.19	2.12
8.79	247	247	247.56	0.32	91	247	.28	2.59
27.67	253	253	253.72	0.78	93	256	.27	2.49
34.66	251	251	251.33	0.57	76	252	.20	2.58
29.22	243	243	243.28	0.41	68	243	.20	2.94
8.48	242	242	242.41	0.29	61	248	.20	2.31
8.01	249	249	250.12	1.08	72	249	.20	2.28
28.00	250	250	251.22	1.52	73	259	.19	3.00
30.12	249	249	249.52	0.78	65	254	.36	2.77
9.58	256	256	256.49	0.63	62	259	.28	2.46
7.74	239	239	241.07	1.73	73	239	.19	2.03
5.07	248	248	248.67	.81	79	250	.19	3.09
26.72	232	232	233.31	0.92	87	232	.28	2.73
28.45	239	239	239.29	0.35	62	239	.19	1.92
31.17	247	247	247.18	0.22	68	251	.27	2.13
2.73	239	239	239.34	0.29	59	239	.28	2.46
3.32	226	226	226	0.0	82	226	.20	2.77

Table 2.4.5: Results from [54]

Exact Cooperative Method	Time (sec)				
Instances	Min	Median	Max	Mean	St.Dev
Gap-C	4213	5040	7035	5198	565
PCodes	1	1	1	1	1
FPP	1	1	1	1	1
Chess	1	1	1	1	1

CHAPTER 3
SOLVING MULTI-OBJECTIVE OPTIMIZATION VIA ADAPTIVE STOCHASTIC SEARCH WITH DOMINATION MEASURE

3.1 Background and Motivation

Problems that require optimizing several objectives concurrently are known as multi-objective optimization problems. Obviously, this type of problems arises in many real-world applications, including construction science [42], economics [63], medical treatments [55], and logistics [44], in which incommensurable and conflicting objectives need to be optimized. Therefore, it is often unlikely to have a solution that optimizes all objectives simultaneously. A more reasonable goal is to obtain a set of solutions, where the quality of each solution is incomparable without any prior knowledge of preference. These solutions are known as Pareto optimal solutions, which are "optimal" in the sense that no other solutions in the solution space are superior to them while taking into account all of the objectives. The set of Pareto optimal solutions is called the Pareto optimal set, and its image in the objective space is called the Pareto front.

Numerous methods have been developed to find or approximate the Pareto optimal set of a general multi-objective optimization problem, among which perhaps the most popular ones are evolutionary algorithms [24] that use iterative selection, mutation and crossover operations to generate multiple Pareto optimal solutions in parallel. Other methods include stochastic search methods [68] that choose candidate solutions from some probability distribution and improve the way those candidate solutions are selected in each iteration, and particle swarm methods [50] that keep a population of potential solutions (particles) that are manipulated by a velocity vector, which changes the position of the particles at each iteration. Furthermore, simulated annealing [22] has also been implemented in the multi-

objective domain by using multiple weight vectors to convert the problem into several single-objective problems. A major downside of using these types of methods is that in most cases the performance of the algorithm is very sensitive to the parameters or weights one chooses.

Although model-based algorithms have mostly been used to solve single-objective optimization problems, there are some methods that incorporate the Cross Entropy (CE) method, a model-based approach, to solve multi-objective problems, see e.g., [64] and [7]. In [64] samples are generated from a fixed number of sampling distributions, which allows the algorithm to be very fast and simple to implement. However, since a fixed number of sampling distributions are used, this method may have difficulty with constructing a sampling distribution that has majority of its mass on isolated points of the Pareto optimal set. [7] propose a histogram approach in which candidate solutions are drawn independently along each dimension of the solution space. By sampling in this manner the Cross Entropy method can be applied separately in each dimension, which allows the algorithm to produce satisfactory results in few evaluations. Although this method performs very well on some problems, it could have difficulty in capturing the entire Pareto optimal set that consists of highly correlated solutions.

In view of the existing methods in the literature, a method that explores the superiority or the dominance relationship among solutions by simple quality metrics is still lacking. Incorporating such metrics reduces problem dimensionality since the multi-dimensional objective space is mapped onto a single-dimensional one through direct comparisons among solutions. Thus, the original multi-objective problem is transformed into a single-objective one, in which the objective is to find the solutions that optimize a particular quality metric. Of course, as pointed out by [71], a reduction of problem dimensionality will inevitably cause the loss of information. That is, generally the global optimal set of the reformulated problem will not be the same as the Pareto optimal set. In particular, [71] prove that in order for a metric to retain the Pareto dominance relation the dimension of the metric and

the objective space must be equivalent. There are some initial explorations along this line of research. For example, a quality metric can be derived by a weighted aggregation of the objective functions. The issue with this quality metric is that a single choice of weights leads to at most one point in the Pareto optimal set. Although multiple points could be obtained via changing the weights, the final approximation could be unsatisfactory since a uniform spread of the weighting coefficients does not necessarily produce a uniform spread of Pareto optimal solutions. Moreover, this technique is infeasible for problems with a large amount of objectives, since the total number of weight combinations grows exponentially w.r.t the number of objectives.

In [71] quality metric of a solution set termed hypervolume was proposed, which is defined as the total size of the area that is dominated by that set in the objective space w.r.t. a reference point. It is desirable for performance assessment of a solution set since it has the ability to measure how close solutions are to the Pareto front as well as how evenly spread the solutions are in the Pareto optimal set. Therefore, it can qualitatively compare two different solution sets by the use of one value. Since the hypervolume metric is influenced by any type of progress to the true Pareto front, many methods have used this metric as a way to guide the search to favorable areas of the solution space, see [27], etc. Although the hypervolume metric has many favorable properties, there are also some drawbacks. In particular, the choice of the reference point could affect progression of the search to promising areas of the solution space, the hypervolume is biased towards convex regions of the objective space, and as the number of objectives increase so does the computational complexity of calculating the hypervolume.

3.1.1 Research Results and Contribution

We introduce a new parameter-free and unary performance metric to measure the quality of solutions, termed as *domination measure*. Practically, the domination measure of a solution can be viewed as the measure of the region in the solution space that dominates

that solution. Unlike hypervolume mentioned previously, it is a global performance metric since all the solutions in the solution space are taken into account for its computation. To the best of our knowledge, this work is among the very first to incorporate such a performance metric for solving multi-objective problems.

There are many advantages for using domination measure as a performance metric. Foremost, unlike the aforementioned scalarization methods, this metric does not require any tuning of parameters. Moreover, the domination measure of a solution is a rigorous quantification of the quality of a solution. That is, the lower the domination measure, the better the solution. Finally, if a solution is Pareto optimal, then it has a domination measure of zero since no solution dominates it. When the solution space consists of finite solutions, the set of Pareto optimal solutions is exactly the set of solutions with domination measure of zero. Precisely, a solution being Pareto optimal is equivalent to the solution having a domination measure of zero. In contrast, for a continuous solution space a solution that is not Pareto optimal can also have a domination measure of zero, which is not surprising since some of the dominance relation is lost by reducing the dimensionality of the objective space through domination measure. Although theoretically the set of solutions with domination of zero is not equivalent to the Pareto optimal set, it is a sufficiently good approximation in the sense that no solution dominates it almost surely.

By employing domination measure as a performance metric, we are able to transform the original multi-objective optimization problem into a single-objective problem, where the goal is to find solutions that have a domination measure of zero. Note that this objective is also stochastic since domination measure is an expectation of an indicator function on the dominance relation w.r.t. a uniform probability measure. While we benefit from a significant reduction in problem complexity, we make the compromise from finding Pareto optimal solutions to finding solutions with a domination measure of zero. Nevertheless, we will propose an approach tailored for the reformulated problem and show empirically that it is not prone to converge to a non-Pareto optimal solution with a domination measure of

zero.

To solve the reformulated stochastic single-objective problem, we propose a model-based approach that finds multiple global optimal solutions. The idea is to introduce a mixed sampling distributions with components from a parameterized family of densities, and iteratively update the sampling distribution parameters by minimizing the Kullback-Leibler divergence between properly constructed reference distributions and the parameterized sampling distributions. The sampling distributions are updated until they are degenerate distributions concentrated on a global optima. Our approach is similar to existing model-based approaches such as the CE method and MRAS in the sense that reference distributions are introduced to guide the sampling process while the actual sampling is achieved by the parameterized distributions. A major difference from the aforementioned methods is that in every iteration our approach keeps track of a population of sampling distributions with an adaptive number of components instead of a single sampling distribution. This ensures that multiple uniformly spread global optimal solutions are generated rather than a single optimal solution.

Based on the proposed approach we design an algorithm in an ideal version and an implementable version. For the ideal version algorithm, we show that for every solution that has a domination measure of zero, there exists a sequence of parameterized sampling distributions that converges to the degenerate distribution on that solution. For the implementable version algorithm, we show empirically that it is competitive to many existing multi-objective optimization methods by testing them on several classic benchmark problems. In particular, we observe that in all the cases tested, our approach is able to generate solutions that are approximately uniformly spread across the Pareto optimal set by a user-specified threshold distance.

In summary, the contributions are as follows:

- For multi-objective optimization problems, we introduce a novel performance metric termed domination measure to determine the quality of a solution, and we reformu-

late the original problem into a stochastic single-objective one that seeks to minimize the domination measure.

- We propose a novel model-based approach to solve the reformulated problem, leading to an ideal version algorithm that possesses nice convergence properties and an implementable version algorithm that performs well numerically.

- We show that our proposed approach produces a finite and uniformly spread approximation of the Pareto optimal set and performs competitively to or even outperforms many existing approaches.

3.2 Problem Formulation

A general multi-objective problem consists of minimizing (or maximizing) multiple objectives over a defined solution space, which can be formulated as follows:

$$\begin{aligned} \min \quad & \mathbf{f}(x) = \{f_1(x), f_2(x), ..., f_n(x)\} \\ \text{s.t.} \quad & x \in \mathscr{X}, \end{aligned} \tag{3.2.1}$$

where $\mathscr{X}$ denotes the solution space that might be described by constraints, and $\{f_i(\cdot) : \mathscr{X} \to \mathbb{R},\ i = 1, ..., n\}$ are scalar functions. Without loss of generality, we assume $\mathscr{X}$ is a bounded subset of $\mathbb{R}^d$ and all objective are minimized.

Since it is rarely the case that there exists a solution that minimizes all the objectives simultaneously, a reasonably compromised goal is to find all the solutions that are not dominated by any other solutions in $\mathscr{X}$ in terms of the objective values. Specifically, a solution $x \in \mathscr{X}$ is (Pareto) dominated by another solution $y \in \mathscr{X}$ if $f_i(y) \leq f_i(x)$ for all $i = 1, ..., n$, and there exists one $j \in \{1, ..., n\}$ such that $f_j(y) < f_j(x)$. In other words, a solution x is dominated by another solution y if all the objectives evaluated at y are better than (less than or equal to) the ones evaluated at x, and at least one objective evaluated at y is strictly better than (less than) the one evaluated at x. Note that Pareto dominance is

a (strict) partial order defined on $\mathscr{X}$ since it is irreflexive, asymmetric and transitive. For simplicity, we use $y \prec_d x$ to denote that y dominates x. Following this, a solution $x \in \mathscr{X}$ is called Pareto optimal if all other solutions in $\mathscr{X}$ do not dominate x, and thus the goal is to find the Pareto optimal set or a uniformly spread subset representation of the Pareto optimal set.

In general, this problem is difficult because the dominance relation is a partial order defined on the solution space, and the problem is essentially a combinatorial problem over a solution space that is often continuous. As mentioned before, many approaches in the literature are developed for solving various approximations or reductions of a multi-objective optimization problem. In particular, one type of approach is to reformulate the original problem into a single-objective one (e.g., through a weighting scheme on the objective functions) and apply algorithms that are designed for problems with a single objective. As a result of the reformulation, some information of the original problem (e.g., the dominance relationship or the defined partial order among solutions) is lost.

In the next subsection, we will propose a performance metric called domination measure that quantifies the dominance relationship between a solution of interest and all other solutions in the solution space, and use it to reformulate the original multi-objective problem into a stochastic single-objective one. In contrast to most of the existing reformulation techniques, the proposed reformulation technique relies on the combinatorial dominance relationship between solutions rather than the absolute objective values of the solutions. We will show that it leads to a close approximation of the original problem, and is solvable by a proposed model-based optimization method.

3.2.1 Domination Measure

Roughly speaking, domination measure of a solution describes the portion of the solutions in the solution space that dominates that solution. To ease presentation, let us consider defining domination measure of a solution $x \in \mathscr{X}$, denoted by $D(x)$, in the following two

cases: 1) the solution space $\mathscr{X} \in \mathbb{R}^d$ has a non-zero and finite measure w.r.t. the Lebesgue measure ν of the d-dimensional Euclidean space $\mathbb{R}^d$; 2) $\mathscr{X}$ is finite. For the second case, we extend the definition of Lebesgue measure on $\mathbb{R}^d$ to any finite set S with cardinality $|S|$ as follows: Suppose $S_1 \subseteq S$ is a subset of S, then we define the Lebesgue measure of S_1 by $\nu(S_1) = |S_1|/|S|$, i.e., $\nu(S_1)$ is the cardinality of S_1 divided by the cardinality of $|S|$; in particular, $\nu(S) = 1$. We formally define domination measure as follows:

Definition 3.2.1. ***(Domination Measure)****. Assume $\mathscr{X}$ in problem (3.2.1) is either a subset of $\mathbb{R}^d$ that has a non-zero and finite Lebesgue measure or a finite set. Further assume for all $x \in \mathscr{X}$ the set of solutions that dominates x, denoted by $\mathscr{D}_x$, is Lebesgue measurable. Then, for all $x \in \mathscr{X}$, the domination measure $D(x)$ of x is defined as*

$$D(x) \triangleq \frac{\nu(\mathscr{D}_x)}{\nu(\mathscr{X})}, \tag{3.2.2}$$

where $\nu(A)$ is the Lebesgue measure of A.

Remark 3.2.1. *It is easy to extend Definition 3.2.1 to the case where the set $\mathscr{X}$ is (countable or uncountable) infinite and has Lebesgue measure of zero. Specifically, let $D(x) = 1$ if the cardinality of D_x is infinite and 0 otherwise. Since for this case the domination measure of a solution only has two trivial values (0 and 1), we omit the discussion about this case.*

Intuitively, the domination measure $D(x)$ of x is the ratio of the measure of $\mathscr{D}_x$ to the measure of the entire solution space $\mathscr{X}$. Therefore, $\forall x \in \mathscr{X}$ we have $0 \leq D(x) \leq 1$. Furthermore, it is a rigorous performance metric for the quality of a solution. If the domination measure of a solution is close to zero, then that solution is dominated by a small number of solutions in the solution space.

Lemma 3.2.1. *For any Pareto optimal solution $x^* \in \mathscr{X}$, its domination measure $D(x^*) = 0$.*

The other direction of the statement in Lemma 3.2.1 might not be true, which means a solution with domination measure of zero might not be Pareto optimal. To illustrate

this, consider a simple example with two objectives and a two-dimensional solution space $[0,1] \times [0,1] \in \mathbb{R}^2$. The two objectives are $f_1(x_1,x_2) = x_1$ and $f_2(x_1,x_2) = x_2$. Therefore, (f_1, f_2) is a mapping from $[0,1] \times [0,1]$ to itself. The solution space for this example is displayed in Figure 3.2.1. Although $(0,0)$ is the lone Pareto optimal solution, all solutions in red have a domination measure of zero. For example, point $(0.4,0)$ is dominated by all points where $x \in [0,0.4)$ and $y = 0$, but the measure of those points is equal to zero. Conversely, for a finite solution space, no information is lost using domination measure.

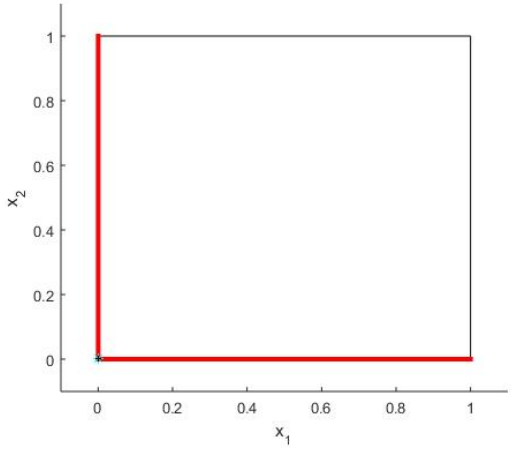


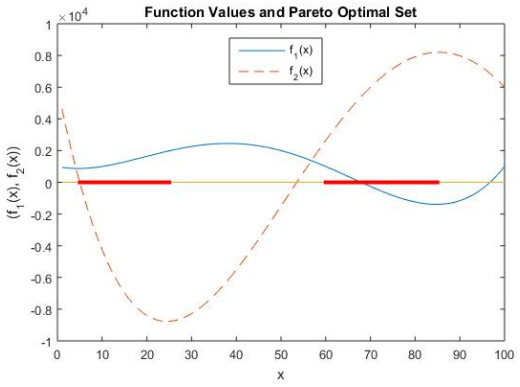
Figure 3.2.1: Illustration of Domination Measure.

Since a Pareto optimal solution achieves a minimum domination measure of zero, domination measure retains part of the information from the partial order dominance relation. The benefit is that the original multi-dimensional combinatorial problem can be reduced to a simple single-objective one with a known minimum objective value of zero.

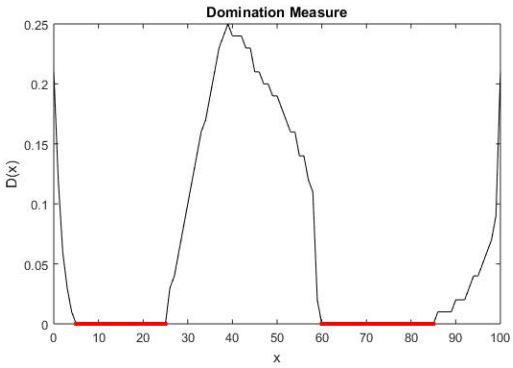
To demonstrate the effectiveness of this reduction, we consider an example from [50] as follows: The solution space is $\mathbb{Z} \cap [0,100]$, i.e., the set of all the integers between 0 and 100, which is a finite set. There are two objective functions:

$$f_1(x) = 0.001x(x-10)(x-60)(x-100)+1000,\ f_2(x) = 0.001x(x-70)(x-100)(x-200)+6000.$$

The Pareto optimal set is $P^* = \mathbb{Z} \cap ([5,25] \cup [60,85])$, and it is highlighted in Figure 3.2.2.



(a) Pareto Optimal Set by Function Values.



(b) Pareto Optimal Set by Domination Measure

Figure 3.2.2: Pareto Optimal Set by Function Values and Domination Measure.

Locating the Pareto optimal set directly using function values (see Figure 3.2.2a) is difficult, whereas the Pareto optimal set is easily identifiable using domination measure (see Figure 3.2.2b). That is, the solutions with domination measure zero are the Pareto optimal set for a finite solution space.

We minimize the domination measure instead of solving the original combinatorial problem. Note that we can reformulate (3.2.2) as

$$D(x) = \frac{\nu(\mathscr{D}_x)}{\nu(\mathscr{X})} = \frac{\int_{\mathscr{X}} \mathbb{1}\{y \prec_d x\}\, \nu(dy)}{\int_{\mathscr{X}} \nu(dy)} = \int_{\mathscr{X}} \mathbb{1}\{y \prec_d x\}\, U_{\mathscr{X}}\, \nu(dy) = \mathbb{E}_U\left[\mathbb{1}\{y \prec_d x\}\right], \tag{3.2.3}$$

where $\mathbb{1}\{E\} = 1$ if the event E is true and $\mathbb{1}\{E\} = 0$ otherwise, $U_{\mathscr{X}}(\cdot)$ is the uniform probability measure (induced by the uniform distribution) on $\mathscr{X}$, and $\mathbb{E}_U[\cdot]$ denotes the expectation w.r.t. $U_{\mathscr{X}}(\cdot)$. Consequently, we solve the following single-objective stochastic optimization problem:

$$\min_{x\in\mathscr{X}} D(x) = \mathbb{E}_U\left[\mathbb{1}\{y \prec_d x\}\right], \quad \text{or equivalently} \quad \max_{x\in\mathscr{X}} -D(x) = \mathbb{E}_U\left[-\mathbb{1}\{y \prec_d x\}\right]. \tag{3.2.4}$$

Typically, the goal is to find multiple global optima of problem (4.2.2) such that they approximately form a uniform distribution on the Pareto optimal set of the original problem (3.2.1). Suppose the Pareto optimal set is nonempty, then by Lemma 3.2.1, there always

exists an $x \in \mathscr{X}$ such that $D(x) = 0$. As a result, problem (4.2.2) always admits at least one global optimal. For simplicity, we denote the set of global optima of problem (4.2.2) by A^*. Therefore, $\forall x \in \mathscr{X}$, $D(x) = 0$ if and only if $x \in A^*$.

Explicitly calculating $D(x)$ is usually computationally expensive or infeasible, especially in the case of a continuous solution space. Alternatively, since Monte Carlo simulation is straightforward to implement and scales well with the dimension of the solution space, it is a good choice for estimating $D(x)$. The general procedure is as follows: Draw independent and identically distributed (i.i.d.) samples according to the uniform distribution $U_{\mathscr{X}}(\cdot)$ on the solution space, which is denoted by $\{x^1, ..., x^N\}$. Then calucalte

$$\widetilde{D}(x^i) \triangleq \frac{1}{N} \sum_{j=1}^{N} \mathbb{1}\left\{x^j \prec_d x^i\right\}, \tag{3.2.5}$$

which is an unbiased estimator of $D(x^i)$. We also point out that the sampling distribution could be of other forms via the principle of importance sampling as long as it is fully supported on $\mathscr{X}$.

3.3 A Framework of Model-based Approach

Recall that we aim to find multiple global minimizers of the domination measure $D(x)$ such that they approximately form a uniform distribution on the Pareto optimal set of problem (3.2.1). Since $D(x)$ is often evaluated via simulation, its structural properties such as convexity or differentiability are unknown. Thus, traditional gradient-based optimization methods might not be applicable for the minimization of $D(x)$. In contrast, model-based optimization methods are good alternatives as they impose minimal requirements on the problem structure. Common model-based methods include Annealing Adaptive Search (ASS) ([57]), the Cross-Entropy (CE) method ([59]), Model Reference Adaptive Search (MRAS) ([34]), and Gradient-based Adaptive Stochastic Search (GASS) ([69]), etc.

The main idea of model-based methods is to introduce a sampling distribution, which

often belongs to a parameterized family of densities, over the solution space, and iteratively update the parameters of the sampling distribution by generating and evaluating candidate solutions. Specifically, the methods iteratively carry out the following two steps:

1. Generate candidate solutions according to the sampling distribution.

2. Based on the evaluation of the candidate solutions, update the parameter of the sampling distribution.

The hope is to have the sampling distribution more and more concentrated on the promising region of the solution space where the optimal solutions are located, and eventually become a degenerate distribution on one of the global optima. Therefore, finding an optimal solution in the solution space is transformed to finding an optimal sampling distribution parameter in the parameter space. A key difference among the aforementioned model-based methods lies in how to update the sampling distribution parameter. For example, in CE and MRAS the updating rule is derived by minimizing the Kullback-Leibler (KL) divergences between a converging sequence of reference distributions and a chosen exponential family of densities. For another example, in GASS the updating rule is derived by converting the original (possibly non-differentiable) deterministic optimization problem into a differentiable stochastic optimization problem on the sampling distribution parameter, and then applying a Newton-like scheme.

Compared with gradient-based methods, model-based methods are more robust in the sense that at every iteration they exploit the promising region of the solution space that has already been identified, while maintaining exploration of the entire solution space. The updating rule on the sampling distribution parameter controls the balance between exploration and exploitation.

In principle, we could extend all the aforementioned model-based methods to our problem setting in a direct manner. However, there are two main issues with implementing existing model-based methods: 1) all these methods are designed for producing a single

global optimal solution which corresponds to a degenerate parameterized distribution, but multiple evenly spread global optimal solutions are needed to represent the Pareto optimal set; 2) the methods that are convergent require the uniqueness of the global optimal solution, which in general is not satisfied for problem (4.2.2).

To address the issues mentioned above, we propose a model-based method with a mixed sampling distribution that consists of a number of distributions from the same parameterized family of densities, and iteratively update the parameters of the mixed sampling distribution. The hope is that each component of the mixed sampling distribution will concentrate on a promising region of the solution space and explore that region exclusively. Eventually, all the components of the mixed sampling distribution will be become degenerate distributions concentrated on distinct global optimal solutions.

The choice of the number of components in the mixed sampling distribution and the method for updating each component are important. Since the promising region of the solution space is unknown and can only be explored via sample evaluations, the number of components needs to be determined adaptively so that the all the areas containing the global optima are eventually explored. The updating scheme on the components needs to achieve convergence so that degenerate sampling distributions on individual global optimal solutions are obtained.

In essence, our updating rule on the sampling distributions is similar to the one in the CE or MRAS, in which the sampling distribution parameter at each iteration is derived by minimizing the KL divergence between a specific reference distribution and a parameterized family of densities. However, our updating rule keeps track of an increasing population of reference distributions instead of a single one at each iteration. A common choice of the parameterized family of densities is the exponential family defined as follows:

Definition 3.3.1. ***Exponential Famliy***. *A parameterized family* $\{g(x;\theta) : \theta \in \Theta\}$ *is an*

exponential family of densities if it satisfies

$$g(x;\theta) = \exp\left\{\theta^T \Gamma(x) - \eta(\theta)\right\}, \tag{3.3.1}$$

where $\Gamma(x) = [\Gamma_1(x), ..., \Gamma_{d_\theta}(x)]^T$ *is the vector of sufficient statistics,* $\eta(\theta) = \ln\{\int \exp(\theta^T \Gamma(x))dx\}$ *is the normalization factor to ensure* $g(x;\theta)$ *is a probability density function, and* $\Theta = \{\theta : |\eta(\theta)| < \infty\}$ *is the natural parameter space with a nonempty interior. We assume that* $\Gamma(\cdot)$ *is a continuous mapping.*

We point out that many common probability distributions belong to the exponential family, including Gaussian, Poisson, Binomial, Geometric, etc. Within the exponential family of densities, the updating of the sampling parameter using a reference sampling distribution is carried out as follows: Suppose $h(x)$ is a reference sampling distribution of interest, then the corresponding sampling distribution $g(x;\theta_*)$ within the exponential family of densities could be found via

$$\theta_* \stackrel{\triangle}{=} \operatorname*{argmin}_{\theta\in\Theta} KL(h(\cdot), g(\cdot;\theta)) = \operatorname*{argmin}_{\theta\in\Theta} \mathbb{E}_h\left[\ln\frac{h(x)}{g(x;\theta)}\right] = \operatorname*{argmax}_{\theta\in\Theta} \mathbb{E}_h\left[\ln g(x;\theta)\right] \tag{3.3.2}$$

where $\mathbb{E}_h[\cdot]$ denotes the expectation w.r.t. $h(\cdot)$. It is important to note that the problem (3.3.2), i.e., the minimization of KL divergence, can be carried out in analytical form since $g(\cdot;\theta)$ belongs to the exponential family. Specifically, by Definition 3.3.1, problem (3.3.2) is equivalent to

$$\operatorname*{argmax}_{\theta\in\Theta} \int_{x\in\mathscr{X}} (\theta^T \Gamma(x) - \eta(\theta))h(x)dx.$$

Note that $(\theta^T\Gamma(x) - \eta(\theta))$ is strictly concave in θ (see, e.g., [47]). It follows that $\int_{x\in\mathscr{X}}(\theta^T\Gamma(x) - \eta(\theta))h(x)dx$ is also strictly concave in θ. Therefore, (3.3.2) admits a unique optimal solution θ_* that satisfies the first-order condition as follows:

$$\int \left(\Gamma_j(x) - \frac{\int \Gamma_j(x)\exp\left(\theta_*^T\Gamma(x)\right)dx}{\int \exp\left(\theta_*^T\Gamma(x)\right)dx}\right) h(x)dx = 0, \quad j = 1, ..., d_\theta,$$

or equivalently,

$$\mathbb{E}_h[\Gamma(x)] = \mathbb{E}_{\theta_*}[\Gamma(x)], \tag{3.3.3}$$

where $\forall\theta \in \Theta$, $\mathbb{E}_\theta[\cdot]$ denotes the expectation w.r.t. $g(x;\theta)$.

In practice, finding the exact value of θ_* can be difficult since $\mathbb{E}_h[\Gamma(x)]$ usually does not admit a closed-form expression; however, an good estimate of θ_* could be obtained via the principle of importance sampling, noting that

$$\mathbb{E}_{\theta_*}[\Gamma(x)] = \mathbb{E}_h[\Gamma(x)] = \mathbb{E}_{\theta'}\left[\frac{\Gamma(x)h(x)}{g(x;\theta')}\right],$$

and i.i.d. samples can be drawn according to $g(\cdot;\theta')$ to estimate the right hand side of above equation.

For example, if $\{g(\cdot;x)\}$ is the family of multivariate Gaussian distributions $\mathcal{N}(\mu,\Sigma)$, where μ is the mean parameter, Σ is the covariance matrix, and $\theta = (\mu,\Sigma)$. Then $\theta_* = (\mu_*,\Sigma_*)$ can be solved as

$$\mu_* = \frac{\mathbb{E}_{\theta'}[h(x)x/g(x;\theta')]}{\mathbb{E}_{\theta'}[h(x)/g(x;\theta')]} \text{ and } \Sigma_* = \frac{\mathbb{E}_{\theta'}\left[h(x)(x-\mu_*)(x-\mu_*)^T/g(x;\theta')\right]}{\mathbb{E}_{\theta'}[h(x)/g(x;\theta')]},$$

and estimated via sampling according to $g(\cdot;\theta')$.

3.3.1 Algorithm—An Ideal Version

We first present an ideal version algorithm of the proposed approach. Although it is not implementable in practice, its merit lies in revealing the mathematical intuition of an implementable version algorithm, which will be introduced later.

Denote the population of reference sampling distribution at the k^{th} iteration by $\{h_{i,k}(x) : i = 1,...,I_k\}$, where I_k is the number of sampling distributions in the population. Let $\{\gamma_k : k = 1,...\}$ be a decreasing sequence of reference values that satisfies $\gamma_k \in (0,1]$ and $\lim_{k\to\infty}\gamma_k = 0$. We will use $\{\gamma_k\}$ as reference values for characterizing elite solutions and

promising regions of the solution space in terms of the domination measure. Specifically, let $A_k := \{x \in \mathscr{X} : D(x) \leq \gamma_k\}$ be the set of solutions with domination measure value below γ_k. Loosely speaking, A_k will approach A^* as $k \to \infty$, A^* is the set of solutions with domination measure 0.

Let us further introduce a partition on the set A_k, denoted by $\pi_k := \{A_{1,k}, ..., A_{I_k,k}\}$, such that each $A_{i,k}$ is U-measurable (recall that U is the uniform measure on $\mathscr{X}$), and

$$A_k = \cup_{i=1}^{I_k} A_{i,k} \text{ and } A_{i,k} \cap A_{j,k} = \emptyset, \forall i \neq j.$$

Note that the cardinality of π_k is equal to I_k, which is the number of the sampling distributions at the k^{th} iteration. Define the magnitude $|\pi_k|$ of the partition π_k by $|\pi_k| := \max_i diam(A_{i,k})$, where $diam(A) := \sup_{x,y} \|x-y\|$ is the "diameter" of set A and $\|\cdot\|$ is the vector Euclidean norm. We impose a requirement on $\{\pi_k : k = 1, ...\}$ as follows:

$$\lim_{k\to\infty} |\pi_k| = 0. \tag{3.3.4}$$

Requirement 3.3.4 forces the partition to become arbitrarily fine for sufficiently large k's. Equivalently, each partitioning subset $A_{i,k}$ is shrinking to a degenerate point in A^* or a empty set as $k \to \infty$. Such a partition always exists, since $\mathscr{X}$ is bounded

Let the reference distribution $h_{i,k}(x) \propto \mathbb{1}\{x \in A_{i,k}\}$, where $h_{i,k}(x)$ is the uniform distribution supported on $A_{i,k}$. Denote the corresponding sampling distribution by $g(x;\theta_{i,k})$, where

$$\theta_{i,k} = \operatorname*{argmin}_{\theta\in\Theta} KL(h_{i,k}(\cdot), g(\cdot;\theta)).$$

Note that the way to construct the reference sampling distribution is similar to the one in the CE method (see [59]). Of course, one could also use more sophisticated reference sampling distributions by an introduction of a shape function to put non-equal weights for $x \in A_{i,k}$ (see [34]). The mixed sampling distribution at the k^{th} iteration, denoted by $g_k(x)$, consists

of $g(x;\theta_{i,k})$ with equal weights. That is,

$$g_k(x) = \frac{1}{I_k}\sum_{i=1}^{I_k} g(x;\theta_{i,k}). \tag{3.3.5}$$

The complete algorithm is summarized in the following Algorithm 2, which is also referred to as Algorithm "**SASMO$_0$**".

Algorithm 2 Stochastic **A**daptive **S**earch for **M**ulti-objective **O**ptimization—Ideal

---

1. **Initialization**: Choose a parameterized family of densities $\{g(x;\theta) : \theta \in \Theta\}$. Specify a sequence of reference values $\{\gamma_k : k = 1,...\}$.

2. **Iteration**: For the k^{th} iteration, choose a partition $\pi_k = \{A_{1,k},...,A_{I_k,k}\}$ on A_k, where $A_k = \{x \in \mathscr{X} : D(x) \leq \gamma_k\}$. Then determine the sampling distribution parameters $\{\theta_{i,k}\}$ by

$$\theta_{i,k} = \underset{\theta\in\Theta}{\operatorname{argmin}}\, KL(h_{i,k}(\cdot), g(\cdot;\theta)).$$

3. **Termination**: Check if some stopping criterion is satisfied. If yes, stop and return the means of the currents sampling distributions; else, set $k := k+1$ and go back to step 2.

Let us analyze the convergence properties of Algorithm "**SASMO$_0$**", which provides an intuitive understanding towards the convergence of its implementable version. In particular, we will show that the parameterized sampling distributions $\{g(x;\theta_{i,k}\}$ converge to degenerate distributions on the global optima of the reformulated problem (4.2.2) if $g(x;\theta)$ belongs to an exponential family of densities, as summarized in the following Theorem 3.3.1.

Theorem 3.3.1. *Suppose the parameterized family of densities $\{g(x;\theta) : \theta \in \Theta\}$ is an exponential family defined by Definition 3.3.1. Further suppose that the sequence of reference values $\{\gamma_k \in (0,1] : k = 1,...\}$ and the sequence of partitions $\{\pi_k : k = 1,...\}$ satisfy, respectively,*

$$\lim_{k\to\infty} \gamma_k = 0 \ \textit{and} \ \lim_{k\to\infty} |\pi_k| = 0. \tag{3.3.6}$$

Then $\forall x^* \in A^*$, there exists a sequence of sampling distributions $\{g(x;\theta_{i_k,k}) : k = 1,...\}$,

where $i_k \in \{1, ..., I_k\}$*, such that*

$$\lim_{k\to\infty} \mathbb{E}_{\theta_{i_k,k}}[\Gamma(x)] = \Gamma(x^*). \tag{3.3.7}$$

Proof. Proof of Theorem 3.3.1. Notice that $\forall x^* \in A^*$, there exists a sequence of partitioning sets $\{A_{i_k,k} : k = 1, ...\}$ s.t. $x^* \in A_{i_k,k}$ and $A_{i_k,k} \in \pi_k$, since the sets in each partition π_k completely cover A_k and hence A^*. Furthermore, by the properties of the exponential family of densities (see (3.3.3)), we have that

$$\mathbb{E}_{h_{i_k,k}}[\Gamma(x)] = \mathbb{E}_{\theta_{i_k,k}}[\Gamma(x)],$$

where recall that $h_{i_k,k} \propto \mathbb{1}\{x \in A_{i_k,k}\}$.

Therefore, to show (3.3.7), it remains to show

$$\lim_{k\to\infty} \mathbb{E}_{h_{i_k,k}}[\Gamma(x)] = \Gamma(x^*), \quad \text{or equivalently,} \quad \lim_{k\to\infty} \mathbb{E}_{h_{i_k,k}}[\Gamma(x) - \Gamma(x^*)] = 0.$$

That is, to show

$$\lim_{k\to\infty} \int_{A_{i_k,k}} (\Gamma(x) - \Gamma(x^*))\, h_{i_k,k}(x)dx = 0. \tag{3.3.8}$$

Given that $\Gamma(\cdot)$ is continuous on $\mathscr{X}$, we have that $\forall \varepsilon > 0$, $\exists \delta > 0$ s.t.

$$|\Gamma(x) - \Gamma(x^*)| \leq \varepsilon, \ \ \forall x \in B_\delta(x^*),$$

where $B_\delta(x^*) := \{x \in \mathscr{X} : \|x - x^*\| \leq \delta\}$ represents the neighborhood ball centered at x^* with radius δ. Further note that $\lim_{k\to\infty} |\pi_k| = 0$, we have

$$\lim_{k\to\infty} diam(A_{i_k,k}) = 0.$$

Therefore, there exists a large integer K_ε depending on ε such that for all $k \geq K_\varepsilon$, $A_{i_k,k} \subseteq$

$B_\delta(x^*)$, we note that $x^* \in A_{i_k,k}$. It follows that for all $k \geq K_\varepsilon$,

$$\int_{A_{i_k,k}} (\Gamma(x) - \Gamma(x^*))\, h_{i_k,k}(x)dx \leq \varepsilon \int_{A_{i_k,k}} h_{i_k,k}(x)dx = \varepsilon.$$

Therefore, (3.3.8) and hence Theorem 3.3.1 holds. □

Theorem 3.3.1 implies that for any solution with domination measure zero, there exists a sequence of exponential sampling distributions that converges to a degenerate distribution on that solution.

In practice, Algorithm "**SASMO$_0$**" is not implementable for the following reasons: 1) the set A_k, which is regarded as the promising region of the solution space, could not be constructed explicitly since the domination measure $D(x)$ is unknown; 2) solving for the sampling parameters exactly through minimizing KL divergence is unlikely since the reference sampling distribution $h_{i,k}(\cdot)$ do not have an explicit characterization.

3.3.2 Algorithm—An Implementable Version

To have an implementable version of Algorithm "**SASMO$_0$**", we integrate a sampling step at each iteration, in which multiple i.i.d. candidate solutions are drawn according to a certain sampling distribution and the corresponding values of domination measure are evaluated or estimated.

There many reasons why the sampling step is integral in constructing an implementable version of Algorithm "**SASMO$_0$**". First, the domination measure $D(x)$ can be approximated for all the candidate solutions, which will then be used to determine the reference value γ_k and characterize the set of promising solutions A_k. In particular, suppose $\mathbf{g}_k(\cdot)$ is the sampling distribution at the k^{th} iteration, and N_k i.i.d. candidate solutions $\{x_k^1, ..., x_k^{N_k}\}$ are drawn according to $\mathbf{g}_k(\cdot)$. Then the domination measure for each candidate solution can be estimated by

$$\widetilde{D}(x_k^i) = \frac{1}{N_k \cdot \nu(\mathscr{X})} \sum_{j=1}^{N_k} \frac{1}{\mathbf{g}_k(x_k^j)} \mathbb{1}\left\{x_k^j \prec_d x_k^i\right\}$$

via the principle of importance sampling. Second, the partition π_k will be determined based on the evaluations of candidate solutions; that is, the partitioning sets $\{A_{i_k,k}\}$ will be characterized by clusters of the elite candidate solutions from a certain clustering algorithm. Thus, the reference sampling distributions $\{h_{i_k,k}(\cdot)\}$ will be characterized by empirical distributions consisting of the elite candidate solutions in those clusters. In this case the magnitude of π_k will be determined by the parameters of the clustering algorithm. Third, the sampling parameters $\{\theta_{i_k,k}\}$ will be solved by minimizing the KL divergence between the constructed empirical sampling distributions and the parameterized sampling distributions. To this end, let us describe the following Algorithm 3, which is referred to as Algorithm "**SASMO$_1$**", for simulation optimization of multi-objective problem (3.2.1).

In the initialization step (step 1) of Algorithm "**SASMO$_1$**", a common choice of the parameterized family of densities is the exponential family. The initial sampling parameter $\theta_{1,0}$ should be chosen in such a way that the resulted sampling distribution is close to the uniform distribution on $\mathscr{X}$, so that the entire solution space will be evenly explored in the early iterations. For example, if the parameterized family of densities is the family of multi-variate Gaussian distributions, then $\theta_{1,0}$ is characterized by the mean vector $\mu_{1,0}$ and covariance matrix $\Sigma_{1,0}$. To enforce global exploration of the entire solution space, $\Sigma_{1,0}$ needs to be relatively large. The mixed coefficient α, the percent quantile ρ, as well as the sample size sequence $\{N_k\}$ will affect the robustness and convergence of the algorithm. We will discuss their selections in the Numerical Experiments Section.

Algorithm 3 **S**tochastic **A**daptive **S**earch for **M**ulti-objective **O**ptimization—**I**mplementable

1. **Initialization**: Choose a parameterized family of densities $\{g(x;\theta) : \theta \in \Theta\}$ with initial parameter $\theta_{1,0}$ with $I_0 = 1$. Specify a mixing coefficient $\alpha \in (0,1)$, a percent quantile ρ, a sample size sequence $\{N_k\}$. Set $k = 0$.

2. **Sampling**: Draw N_k i.i.d. candidate solutions $\{x_k^i : i = 1,2,...,N_k\}$ according to $\mathbf{g}(\cdot;\theta_k)$, where

$$\mathbf{g}_k(\cdot) \stackrel{\triangle}{=} (1-\alpha)g_k(\cdot) + \alpha U_{\mathscr{X}}(\cdot) \tag{3.3.9}$$

is a mixed sampling distribution, $g_k(x) = \frac{1}{I_k}\sum_{i=1}^{I_k} g(x;\theta_{i,k})$, and $U_{\mathscr{X}}(\cdot)$ is the uniform distribution on $\mathscr{X}$.

3. **Estimation**: For $i = 1,...,N_k$, estimate the domination measure $D(x_k^i)$ at x_k^i by

$$\widetilde{D}(x_k^i) = \frac{1}{N_k \cdot \nu(\mathscr{X})} \sum_{j=1}^{N_k} \frac{1}{\mathbf{g}_k(x_k^j)} \mathbb{1}\left\{x_k^j \prec_d x_k^i\right\}. \tag{3.3.10}$$

Sort $\{\widetilde{D}(x_k^i)\}$ in ascending order, denoted by $\widetilde{D}(x_1^{(1)}) \le \widetilde{D}(x_1^{(2)}) \le \cdots \le \widetilde{D}(x_1^{(N_k)})$. Set the reference value $\widetilde{\gamma}_k$ to be the sample ρ-percent quantile $\widetilde{D}(x_1^{(\lceil \rho N_k \rceil)})$, i.e., $\widetilde{\gamma}_k = \widetilde{D}(x_1^{(\lceil \rho N_k \rceil)})$, where $\lceil \rho N_k \rceil$ is the smallest integer that is greater than or equal to ρN_k.

4. **Updating**: Construct the set of elite candidate solutions by $\widetilde{A}_k := \{x_k^i : \widetilde{D}(x_k^i) \le \widetilde{\gamma}_k\}$. Using a clustering algorithm (e.g., Algorithm 4) to cluster $\widetilde{A}_k$ into clusters $\widetilde{\pi}_k := \{\widetilde{A}_{1,k},...,\widetilde{A}_{I_k,k}\}$. Update the parameter $\theta_{i,k}$ based on the set of elite candidate solutions $\widetilde{A}_{i,k}$ by solving

$$\theta_{i,k} \stackrel{\triangle}{=} \operatorname*{argmax}_{\theta\in\Theta} \frac{1}{|\widetilde{A}_{i,k}|} \sum_{x\in\widetilde{A}_{i,k}} \frac{\ln g(x;\theta)}{\mathbf{g}_k(x)}. \tag{3.3.11}$$

5. **Stopping**. Check if some stopping criterion is satisfied. If yes, stop and return the means of the current parameterized sampling distributions; else, set $k := k+1$ and go back to step 2.

In the sampling step (step 2), note that in (6) the sampling distribution $\mathbf{g}_k(\cdot)$ is derived by mixing the uniform distribution on $\mathscr{X}$ with the equal-weighted combination of the parameterized sampling distributions obtained from the previous iteration. The uniform distribution component in $\mathbf{g}_k(\cdot)$ helps maintain a global exploration of the entire solution space. Furthermore, since the candidate solutions drawn from $\mathbf{g}_k(\cdot)$ will be used to estimate the domination measure, the uniform distribution component controls the variance in the estimations. In particular, the variances will be bounded and the bounds only depend on the choice of N_k. Lastly, the choice of mixing coefficient α affects the robustness of the algorithm since it determines how much global exploration is achieved and the amount of variance reduction. A typical choice of α is $\alpha = 0.1$.

In the estimation step (step 3), for the sake of convergence, the sample size N_k is either set to be a large constant or in a way such that $N_{k+1} = \tau N_k$ for certain $\tau > 1$. The domination measure of the sampled candidate solutions are estimated via the principle of importance sampling, where the importance sampling distribution is also the mixed distribution $\mathbf{g}_k(\cdot)$. As mentioned earlier, due to the uniform distribution component in the sampling distribution, the variances of the resulted estimators are bounded. The quantile level ρ controls the reference value $\widetilde{\gamma}_k$, i.e., determines the number of elite candidate solutions that are used to update the sampling distribution in the next iteration, and the trade-offs between the exploitation of the neighborhood of current best solutions and the exploration of the entire solution space. For example, when a smaller ρ is used, less elite candidate solutions are used in updating the sampling distribution, which results in less exploration in the solution space. Furthermore, note that the simple evolution of $\widetilde{\gamma}_k$ does not guarantee that it convergences to zero. However, it performs well in numerical tests and its performance is competitive to the more sophisticated methods of constructing $\widetilde{\gamma}_k$ in other model-based optimization methods, e.g., the one in SMRAS from [35].

In the updating step (step 4), the promising region A_k is characterized by the set of elite candidate solutions $\widetilde{A}_k$, and the partition π_k is characterized by the set of clusters $\widetilde{\pi}_k$ yielded

from applying a clustering algorithm on $\widetilde{A}_k$. Note that $\widetilde{\pi}_k$ also needs to satisfy

$$\lim_{k\to\infty} |\widetilde{\pi}_k| = 0, \text{ where } |\widetilde{\pi}_k| = \max_{1\le i\le I_k} diam(\widetilde{A}_{i,k}).$$

If a threshold-based clustering algorithm is used as the clustering algorithm, then the sequence of the threshold distances should decrease to zero. We defer the discussion of the details of this clustering algorithm (Algorithm 4). The updating rule (3.3.11) on θ can be regarded as a sampled version of the updating rule in Algorithm (2), in which the reference sampling distribution $h_{i,k}(\cdot)$ is replaced by the corresponding empirical distribution. In particular, if the parameterized family of densities is the family of multivariate Gaussian family with $\theta = (\mu, \Sigma)$, then the mean $\mu_{i,k}$ and the covariance matrix $\Sigma_{i,k}$ computed from (3.3.11) is the sample mean and covariance matrix given by the candidate solutions in $\widetilde{A}_{i,k}$. We point out the resulted sampling distribution parameter could be further projected onto a properly constructed subset of Θ for numerical stability.

In the stopping step (step 5), a common stopping criterion is when the threshold distance in the clustering algorithm falls below a pre-specified threshold bound $\bar{\Delta}$ or a maximum number of iterations t_{max} is reached. The resulted means of the current parameterized sampling distributions will form a finite and approximate uniform representation of the solutions with domination measure of zero, with the hope that these solutions are close to the Pareto optimal set. Moreover, the pre-specified threshold bound or maximum number of iterations will largely determine the number of solutions in the resulted approximation of the Pareto optimal set.

n a broader sense, we note that Algorithm "**SASMO$_1$**" can be used to generate a finite and an approximate uniform representation of the optimal solution set for a single-objective optimization problem with multiple (possible uncountable) global optima, with a deterministic or stochastic objective. Here the advantage of using Algorithm "**SASMO$_1$**" for finding multiple minima of the stochastic minimization problem (4.2.2) lies in that only

a single-layer of simulation is needed instead of a nested one, since the sampling is used to search the solution space and estimate the domination measure. This is in contrast to using a model-based algorithm to solve a general stochastic optimization problem, because an outer-layer simulation is employed to sample on the solution space and an inner-layer simulation is needed for estimation of the function values.

The convergence analysis of Algorithm "**SASMO$_1$**" is more complicated due to the involvement of sampling. Strong requirements on the sample size sequence, the reference value sequence, the choice of clustering algorithm, and even the updating rule might need to be imposed to guarantee the convergence of the algorithm. We leave the convergence analysis of Algorithm "**SASMO$_1$**" for future research. Nevertheless, we will show that Algorithm "**SASMO$_1$**" performs competitively to or outperforms some of the existing algorithms. Note that similar to Algorithm "**SASMO$_1$**", these methods do not have convergence results.

We conclude this section by introducing the clustering algorithm used in the updating step of Algorithm "**SASMO$_1$**", in which a threshold distance Δ_0 and a threshold distance shrinking factor $C > 1$ are chosen. Of course, other clustering algorithms can be used as long as it satisfies the aforementioned guidelines.

In the iteration step of Algorithm 4, we randomized the order in which the distance from the selected solution and the centroids of the existing clusters is compared to prevent one cluster from becoming significantly larger than the other. In the termination step, the threshold distance at the next iteration Δ_{k+1} is decreasing adaptively, noting that $1/I_k \cdot \sum_{i=1}^{I_k} Tr(\widetilde{\Sigma}_{i,k})$ is a empirical measure of how solutions within each cluster are close to each other. If this measure is still large, then the threshold distance is forced to decrease by at least a factor of C. Therefore, the shrinking factor C determines how fast Algorithm "**SASMO$_1$**" terminates and how many solutions are generated in the final finite representation approximation of the Pareto optimal set.

Algorithm 4 A Threshold-Based Clustering Algorithm

Input: Threshold distance Δ_k and elite solution set $\widetilde{A}_k$.
Output: The set of clusters $\widetilde{\pi}_k$ and threshold distance Δ_{k+1}.

1. **Initialization**: Randomly select a solution from $\widetilde{A}_k$. This solution is defined as the centroid of cluster $\widetilde{A}_{1,k}$.

2. **Iteration**: Randomly select a solution from $\widetilde{A}_k$ that has not been assigned to any cluster. Compute the Euclidean distances from the solution to the centroids of existing clusters in a randomized order. Assign the solution to first cluster found with distance less than Δ_k and update the centroid of the cluster as the average of the solutions in the cluster. If no such cluster is found, create a new cluster where the solution is the centroid of the new cluster.

3. **Termination**: Check if there is solution from $\widetilde{A}_k$ that has not been clustered. If yes, go to step 2; otherwise return the set of clusters $\widetilde{\pi}_k$ and the threshold distance at next iteration by

$$\Delta_{k+1} = \min \left[\frac{1}{CI_k} \sum_{i=1}^{I_k} Tr(\widetilde{\Sigma}_{i,k}), \frac{\Delta_k}{C} \right], \tag{3.3.12}$$

where recall that I_k is the number of clusters, $\widetilde{\Sigma}_{i,k}$ is the sample variance of cluster $\widetilde{A}_{i,k}$, and $Tr(\cdot)$ is the trace of a matrix.

3.4 Numerical Experiments

The objective of our numerical results is to show that Algorithm "**SASMO$_1$**" is 1) not sensitive to the geometry of the Pareto optimal set or the Pareto front, 2) scalable in terms of decision variables and objective functions, and 3) competitive to existing methods in terms of how close the solutions are to the true Pareto optimal set and how evenly spread the solutions are in the solution space. To accomplish these goals, we evaluate the performance of Algorithm "**SASMO$_1$**" on test functions from the ZDT ([24]), DTLZ ([24]), and Van Veldhuizen's ([19]) test suites and compare our results with the following existing methods:

- Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) ([24])
- Strength Pareto Evolutionary Algorithm (SPEA-II) ([37])
- Pareto Envelope-based Selection Algorithm (PSEA-II) ([20])

- Multi-objective Particle Swarm Optimization (MOPSO) ([19])

The above methods are all evolutionary algorithms and they differ in how the population of candidate solutions are selected and maintained. In NSGA-II the population of solutions is divided based on the following rule: the first group of solutions is all non-dominated and the second group of solutions is only dominated by the solutions in the first group. The grouping is continued in this manner until all solutions are classified. Once the solutions are divided, a crowding distance is calculated, which measures how close a solution is to its neighbors. Solutions are selected based on their group classification and crowding distance, and new solutions are generated from crossover and mutation operators. SPEA-II is an extended version of the original SPEA algorithm ([70]), which includes a specialized ranking system to order the solutions based on their fitness value, which is an objective function that summaries how close a given solution is to the Pareto front. SPEA-II keeps an archive of all solutions generated starting from the initialization of the algorithm and constructs a population of solutions that combines the archived solutions with the solutions generated at the current iteration. All non-dominated solutions in the population are assigned fitness values such that the search is directed towards the true Pareto front. PSEA-II introduces a new selection technique where the objective space is divided into hyperboxes and solutions are randomly selected from those hyperboxes. The fitness value of a non-dominated solution depends on the number of non-dominated solutions that occupy that same hyperbox. This method of selection is shown to result in a good spread of solutions in the objective space. MOPSO is a particle swarm method that includes a constraint-handling mechanism and a mutation operator that substantially improves the exploration ability of the original algorithm.

The problems in the ZDT test suite have a scalable number of parameters. Therefore, this test suite tests the ability of an algorithm to converge to the Pareto front and obtain diverse solutions in a high-dimensional solution space. The challenge in dealing with a high dimensional solution space is that it is more difficult to get an evenly spread of solutions in

the solution space. In practice the structure of the Pareto front is unknown; therefore, it is important to see how our method performs with different Pareto front geometries. Consequently, we consider test function from the Van Veldhuizen's test suite since they present a variety of Pareto front geometries. The different structures of the Pareto front can be convex, concave, degenerate, mixed, continuous, discontinuous, or contain flat regions. In this context, flat regions are areas of the objective space where relative small perturbations of parameters in the decision space do not affect the objective values. Each geometric structure presents is own difficulty. For instance, problems with isolated points are difficult to solve because usually there is no information in the surrounding region that indicates a Pareto optimal solution is nearby. Furthermore, a function that has a many to one mapping between the decision space and the objective space are difficult to solve due to the flat regions in the objective space. The last set of problems that we consider is the DLTZ test suite, which has a scalable number of objectives while also having complicated Pareto front geometries. The increase in dimensionality of the objective space causes problems with selecting the best solutions. When the number of objectives are large it causes a majority of solutions to be non-dominated by each other, which may throttle an algorithm's convergence to the true Pareto front. The problems chosen from the aforementioned test suites are listed below and more details on the problems properties and challenges can be found in [36].

- ZDT2

$$
\begin{aligned}
\min \ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\
f_1(x) &= x_1 \\
f_2(x) &= g(x)[1-(x_1/g(x))^2], \ \text{where} \\
g(x) &= 1+9(\sum_{i=2}^{30} x_i)/(30-1)] \\
x_i &\in [0,1], \ i=1,\ldots,30.
\end{aligned}
$$

- ZDT3

$$\begin{aligned}
\min\ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\
f_1(x) &= x_1 \\
f_2(x) &= g(x)[1-(x_1/g(x))^2 - \frac{x_1}{g(x)}\sin(10\pi x_1)],\ \text{where} \\
g(x) &= 1+9(\sum_{i=2}^{30} x_i)/(30-1) \\
x_i &\in [0,1],\ i=1,\ldots,30.
\end{aligned}$$

- ZDT4

$$\begin{aligned}
\min\ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\
f_1(x) &= x_1 \\
f_2(x) &= g(x)(1-\sqrt{x_1/g(x)}),\ \text{where} \\
g(x) &= 1+10(10-1)+\sum_{i=2}^{10}(x_i^2-10\cos(4\pi x_i)) \\
x_i &\in [-5,5],\ i=1,\ldots 10.
\end{aligned}$$

- MOP3

$$\begin{aligned}
\max\ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\
f_1(x) &= -[1+(A_1-B_1)^2+(A_2-B_2)^2] \\
f_2(x) &= -[(x_1+3)^2+(x_2+1)^2],\ \text{where} \\
A_1 &= 0.5\sin 1-\cos 1+2\sin 2-1.5\cos 2 \\
A_2 &= 1.5\sin 1-\cos 1+2\sin 2-0.5\cos 2 \\
B_1 &= 0.5\sin x_1-2\cos x_1+\sin x_2-1.5\cos x_2 \\
B_2 &= 1.5\sin x_1-\cos x_1+2\sin x_2-0.5\cos x_2
\end{aligned}$$

$$x_i \in [-\pi, \pi], \ i = 1 \ldots 2.$$

- MOP4

$$\begin{aligned} \min \ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\ f_1(x) &= \sum_{i=1}^{2} (-10 \exp(-0.2\sqrt{x_i^2 + x_{i+1}^2})) \\ f_2(x) &= \sum_{i=1}^{3} (|x_i|^{0.8} + 5\sin(x_i)^3) \\ x_i &\in [-5, 5], \ i = 1, \ldots, 3. \end{aligned}$$

- MOP5

$$\begin{aligned} \min \ \mathbf{f}(x) &= (f_1(x), f_2(x), f_3(x)) \\ f_1(x) &= 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \\ f_2(x) &= \frac{(3x_1 - 2x_2 + 4)^2}{8} + \frac{(x_1 - x_2 + 1)^2}{27} + 15 \\ f_3(x) &= \frac{1}{x_1^2 + x_2^2 + 1} - 1.1 \exp(-x_1^2 - x_2^2) \\ x_1, x_2 &\in [-30, 30]. \end{aligned}$$

- MOP6

$$\begin{aligned} \min \ \mathbf{f}(x) &= (f_1(x), f_2(x)) \\ f_1(x) &= x_1 \\ f_2(x) &= (1 + 10x_2)[1 - (\frac{x_1}{1 + 10x_2})^2 - \frac{x_1}{1 + 10x_2} sin(8\pi x_1)] \\ x_1, x_2 &\in [0, 1]. \end{aligned}$$

- DTLZ1

$$\begin{aligned}
\min\ \mathbf{f}(x) &= (f_1(x), f_2(x), f_3(x)) \\
f_1(x) &= 1/2 x_1 x_2 (1+g(x)) \\
f_2(x) &= 1/2 x_1 (1-x_2)(1+g(x)) \\
f_3(x) &= 1/2 (1-x_1)(1+g(x)), \ \text{where} \\
g(x) &= 100(\|x\| - 2 + \sum_{i=3}^{7}((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))) \\
x_i &\in [0,1] \ , i = 1, \ldots, 7.
\end{aligned}$$

- DTLZ2

$$\begin{aligned}
\min\ \mathbf{f}(x) &= (f_1(x), f_2(x), f_3(x)) \\
f_1(x) &= \cos(x_1\pi/2)\cos(x_2\pi/2)(1+g(x)) \\
f_2(x) &= \cos(x_1\pi/2)\sin(x_2\pi/2)(1+g(x)) \\
f_3(x) &= \sin(x_1\pi/2)(1+g(x)), \ \text{where} \\
g(x) &= \sum_{i=3}^{12}(x_i - 0.5)^2 \\
x_i &\in [0,1], \ i = 1, \ldots, 12.
\end{aligned}$$

We solve all problems with Algorithm "**SASMO$_1$**", which terminates when the threshold distance falls below the threshold bound $\bar{\Delta}$ or when the maximum number of iterations t_{max} is reached. We choose the following parameters: an initial sample size $N_0 = 1000$ and $N_k = k^{1.01}N_0$, a percent quantile $\rho = 0.10$, a mixed coefficient $\alpha = 0.1$, a threshold bound $\bar{\Delta} = 0.001$, a threshold distance shrinking factor $C = 1.1$, an initial mean $\mu_0 = \mathbf{0}$, a maximum number of iterations $t_{max} = 100$, and an initial covariance matrix $\Sigma_0 = 1000\mathbf{I}_d$. The resulted approximations of the Pareto fronts for the tested problems are illustrated in Figure 3.4.1. Note that for each subfigure of Figure 3.4.1, the cyan thick curve represents

the true Pareto front and the black thin curve represents the approximated Pareto front produced by Algorithm "**SASMO$_1$**" in the objective space (f_1, f_2). For the tested cases, we observe that Algorithm "**SASMO$_1$**" is capable of obtaining isolated Pareto optimal points and capturing the entire Pareto front for problems that have multiple discontinuous Pareto curves. These results also demonstrate that relaxing the concept of Pareto dominance to domination measure does not affect the solution quality of our algorithm.

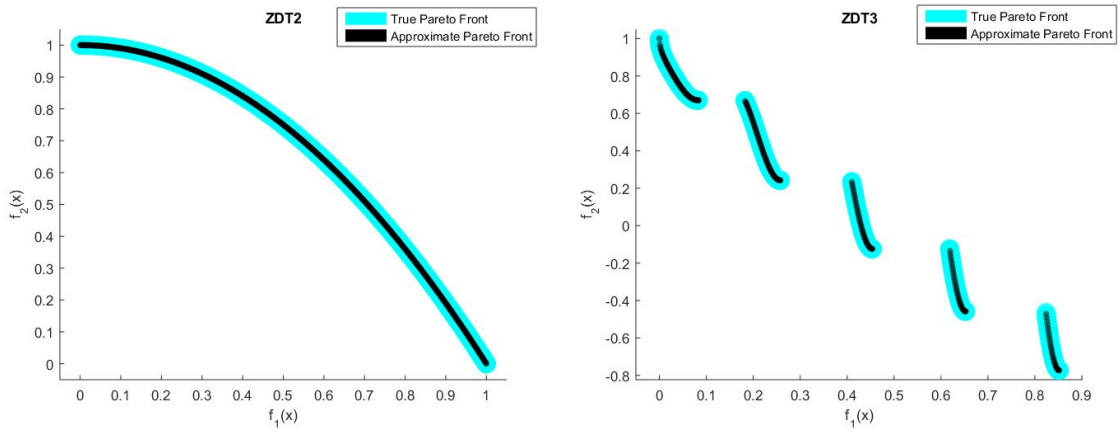
In order to qualitatively quantify the performance of Algorithm "**SASMO$_1$**" and compare it with the performances of the aforementioned existing algorithms, we use the convergence metric Λ and diversity metric Υ defined in [24].

The convergence metric Λ in the objective space is defined by

$$\Lambda \stackrel{\triangle}{=} \frac{1}{|R|} \sum_{y:\ \mathbf{f}(y)\in R} \left\{ \min_{x\in Z} \|\mathbf{f}(x) - \mathbf{f}(y)\| \right\},$$

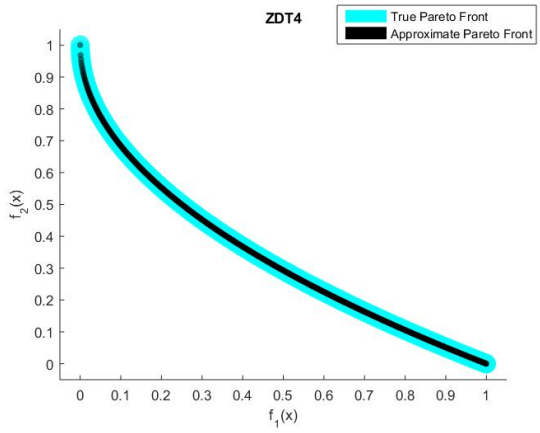
where R is a pre-specified reference set consisting of $|R|$ uniformly spread points from the true Pareto front and we choose $R = 500$, $\mathbf{f}(\cdot) = (f_1(\cdot), ..., f_n(\cdot))$ is the vector of objective functions, and Z is the set of approximate Pareto front generated by the algorithm of interest. In other words, the convergence metric Λ can be regarded as the average distance from all points in the reference set to the approximate Pareto front, which measures the closeness of the approximate Pareto front to the true Pareto front. Therefore, the smaller the value is for Λ, the closer the approximate Pareto front is to the true Pareto front.

Before introducing the diversity metric Υ in the solution space, let us first order the obtained $|Z|$ approximate Pareto optimal solutions $\{x^1, ..., x^{|Z|}\}$ generated from an algorithm of interest by $\{x^{(1)}, ..., x^{(|Z|)}\}$ such that $x_1^{(1)} \leq \cdots \leq x_1^{(|Z|)}$. That is, they are ordered by the values of their first components. We also let $x^{(1)}$ and $x^{(|Z|)}$ be the left and right boundary points of the approximate Pareto optimal set, and let x^l and x^r be the left and right boundary of the true Pareto optimal set also in terms of the value of a solution's first component. The

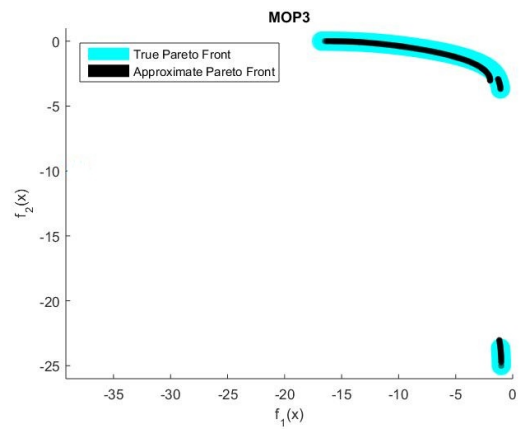


(a) ZDT2

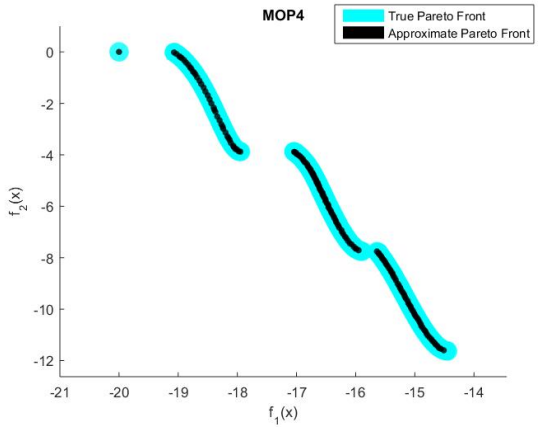
(b) ZDT3



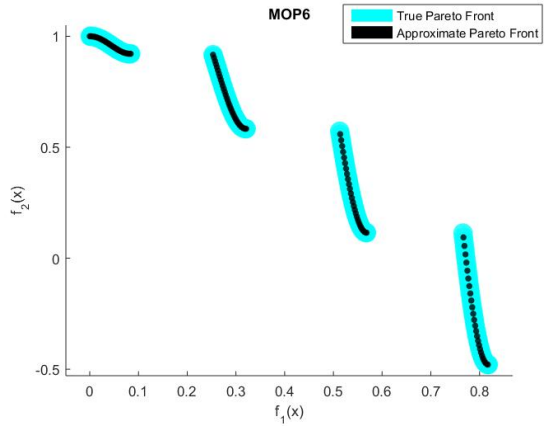
(c) ZDT4



(d) MOP3



(e) MOP4



(f) MOP6

Figure 3.4.1: Approximate Pareto Front V.S. True Pareto Front.

diversity metric Υ is defined by

$$\Upsilon = \frac{d_l + d_r + \sum_{i=1}^{|Z|-1} |d_i - \bar{d}|}{d_l + d_r + (|Z|-1)\bar{d}},$$

where $d_l := \|x^l - x^{(1)}\|$ ($d_i := \|x^{(i+1)} - x^{(i)}\|$) is the distance between the left (right) boundary point of the true Pareto optimal set and the left (right) boundary point of the approximate Pareto optimal set, $d_i := \|x^{(i+1)} - x^{(i)}\|$ is the distance between an approximate Pareto optimal solution and its closest neighbor, and $\bar{d} := 1/(|Z|-1)\sum_{i=1}^{|Z|-1} d_i$ is the average of these distances. Essentially, Υ measures how well the solutions are evenly spaced in the solution space. The smaller the value is for this metric, the closer the approximate Pareto optimal set is from being uniformly distributed.

For each problem instance, we perform 30 independent replications of each method implemented in MATLAB. The codes for the existing methods can be found at `http://yarpiz.com/category/multiobjective-optimization`. We report the best values for the convergence metric Λ and the diversity metric Υ obtained out of the 30 trials for each method. For the existing methods, we choose the following parameters:

- Number of generations: 100
- Population size: 1000
- Archive size: 1000.

The results are summarized in Table 4.4.1 and Table 4.4.3.

We can see that Algorithm "**SASMO$_1$**" outperforms the existing methods on over half of the problems in respect to both the convergence and diversity metrics. The favorable results w.r.t. the diversity metric is likely due to the fact that the center of each cluster represents an estimated Pareto optimal solution and it is at least the threshold distance away from the closest cluster. As a result, the distance between each estimated Pareto optimal solution is close to the threshold bound for most of the problems. We point out

Table 3.4.1: Comparisons of Convergence Metric Λ.

Problem	**SASMO$_1$**	NSGA-II	SPEA-II	MOPSO	PSEA-II
	Λ	Λ	Λ	Λ	Λ
ZDT2	**.0023**	.1064	.0765	.2134	.1863
ZDT3	**.1145**	.5657	.3312	.1821	.2765
ZDT4	**4.321**	7.5432	9.8756	10.321	9.216
MOP3	**.0221**	.0265	.0456	.8973	.0821
MOP4	.0987	.0345	.0673	.1268	.0531
MOP5	**.0299**	.0321	.0312	.0589	.0439
MOP6	.0532	.0582	.0439	1.2956	.9882
DTLZ1	2.9831	2.4531	2.8742	3.9814	3.7124
DTLZ2	**2.4389**	2.9875	2.6192	4.8621	4.0921

Table 3.4.2: Comparisons of Diversity Metric Υ.

Problem	**SASMO$_1$**	NSGA-II	SPEA-II	MOPSO	PSEA-II
	Υ	Υ	Υ	Υ	Υ
ZDT2	**.5641**	.9987	1.2534	1.9874	1.8761
ZDT3	.6015	.5654	1.7543	.9479	1.0113
ZDT4	**.3490**	.3986	.9014	.9851	1.1421
MOP3	**.0289**	.5467	1.4543	1.2156	2.1949
MOP4	**.0354**	.3916	.6989	.7126	.9511
MOP5	**.0211**	.4040	.9431	1.1164	.9346
MOP6	**.0216**	.4510	.6961	.5174	.6920
DTLZ1	.8721	.8142	1.6380	1.9722	.9974
DTLZ2	.9955	.9013	1.6724	1.9882	2.0521

that as the dimension of the solution space increases so does the diversity metric for all the algorithms. Another takeaway from the numerical results is how well our algorithm performed on problems that had discontinuous Pareto fronts. This is likely due to the fact that an adaptive number of components is used in the mixed sampling distribution so that each promising region of the solution space is thoroughly explored. In conclusion, we show that Algorithm "**SASMO$_1$**" gives satisfactory results regardless of the geometry of the Pareto front, and is competitive to several existing algorithms.

3.5 Conclusion and Future Research

In this chapter, we introduce a novel performance metric called domination measure to measure the quality of a solution in a multi-objective problem. Instead of solving the original problem, we propose a model-based approach to find a finite and approximately uniformly spread representation of the solutions with domination measure of zero, which is close to a finite and approximately uniformly spread representation of the Pareto optimal set. We present an ideal version algorithm that has nice convergence properties, and an implementable version of the algorithm that has competitive numerical performances compared to many existing approaches. More sophisticated approaches and algorithms based on domination measure can be incorporated to further improve the results on theoretically convergence and numerical performance. Another future direction of research is to extend Algorithm "$\mathbf{SASMO_1}$" to the setting of stochastic multi-objective optimization, where the sample size allocations across the candidate solutions could to be studied.

CHAPTER 4
SOLVING STOCHASTIC MULTI-OBJECTIVE OPTIMIZATION WITH AN ALLOCATION SCHEME USING DOMINATION PROBABILITY

4.1 Background and Motivation

Stochastic multi-objective optimization (SMO) applies to a wide range of fields, such as inventory management [46], scheduling [53], transportation logistics [31], and medicine allocation [16]. Although the methods of stochastic optimization and multi-objective optimization are well established, the intersection of the two is less developed. Nevertheless, for many real-life problems, it is desirable to have an algorithm that can optimize multiple objectives in the presence of uncertainty. For example, consider the classic (s,S) inventory policy with stochastic demand where s and S denote the reorder point and order up to point, respectively. In this example, the goal may be to maximize the service level while minimizing the inventory cost where both performance measures depend on customer demand that can only be realized through simulation. Developing an efficient SMO method is important due to the increasing necessity for solving problems that seek to optimize multiple objectives that can only be evaluated through noisy simulation outputs or real world observations.

The added difficulty of stochastic multi-objective problems when compared to their deterministic version presented in Chapter 3 lies in the fact that the true performance of any solution is unknown. In other words, these problems assume that obtaining an exact evaluation of the objective functions is infeasible or too computationally expensive. As a result, the performance of a solution has to be approximated by independent simulation runs, where the outputs from the simulation are not deterministic but follow some unknown distribution. The error in computing a solution's true performance is unavoidable and re-

sults in uncertainty in determining superiority among the solutions. Therefore, it becomes more difficult to differentiate between dominated solutions and non-dominated solutions due to the stochastic noise in the objective functions.

We consider multi-objective problems where the decision space is continuous and the objective values can only be estimated from some noisy observation. This setting requires expending computational budget on both searching in the solution space for Pareto optimal solutions and obtaining good estimates of the objective functions at promising solutions. Essentially, the primary question for this problem is how to distribute the simulation budget between searching the solution space for better solutions and getting a more precise estimate of a solution's performance. Due to the empirical success of the model-based approach in obtaining Pareto optimal solutions in the deterministic setting shown in chapter 3, we adapt this model-based approach for the stochastic variant of multi-objective optimization.

Although model-based approaches for stochastic single-objective optimization problems are not studied as much as their deterministic counterparts, there do exist a few methods that have been extended to handle the stochastic case [33]. The Stochastic Model Reference Adaptive Search (SMRAS) [35] is a generalization of the MRAS method that was designed for single-objective deterministic optimization. Recall that in the MRAS method a sequence of reference distributions that depend solely on the performance of candidate solutions is constructed to update parameters of the sampling distribution. Since, in the stochastic case, the exact performance of a solution is unknown, the SMRAS method uses sample averages of the objective function evaluations to approximate the true sequence of reference distributions. The SMRAS method is shown to perform well numerically and has convergence results that illustrate that iteratively constructed sampling distributions converge to a degenerate distribution that is concentrated on the global optimal solution. Although this method performs well with respect to single-objective optimization, the extension of this approach to multi-objective optimization is difficult. That is, most of the

results depend on the uniqueness of an optimal solution, which is clearly not the case in SMO. Furthermore, SMRAS uses a simple allocation rule that requires each solution to get an equal number of simulation replications that increases as the algorithm progresses. If one simulation run is expensive in terms of computational time, the implementation of this algorithm could be impractical. To improve the efficiency of model-based approaches applied to stochastic single-objective optimization, He et al [32] combines the idea of optimal computing budget allocation within each step of the CE method to develop the CEOCBA method. The efficiency of the CE method is enhanced by allocating replications to solutions such that the mean squared error of the estimate used to update the sampling distribution is minimized. Alternatively, Chen et al. [14] developed a procedure that determines the best subset of a population of solutions. This procedure could be useful for modifying existing deterministic model-based approaches to handle a stochastic objective. The aforementioned methods work quite well for stochastic single-objective optimization, but their extensions to SMO seem to be problematic due to the dependency of their mathematical analyses and algorithmic development on the assumption of a single objective.

Despite that we consider a continuous solution space, the number of candidate solutions generated at every iteration of a model-based method is finite. As a result, the problem reduce to how to allocate replications to candidate solutions to maximize the performance of the model-based approach when applied to SMO. Within an iteration, the performance of the model-based approach can directly link to how efficiently and accurately the elite set can be identified, since typically the solutions in the elite set are used to update the parameters of the sampling distribution. The problem of optimizing the number of allocations within an iteration of a model-based method applied to SMO can be thought of as a multi-objective ranking and selection problem.

There are numerous ranking and selection procedures for multi-objective problems [10, 25, 45, 43]. The most popular method MOCBA [43] seeks to determine the Pareto optimal solutions out of a set of solutions by minimizing two types of errors dealing with misclas-

sification. That is, either misidentifying a Pareto optimal solution as non-Pareto optimal or misidentifying a non-Pareto optimal solution as Pareto optimal. The MOCBA method is shown to be significantly more efficient than the use of equal allocation. Similar to the MOCBA method, other methods in the multi-objective ranking and selection literature focus on correctly identifying the Pareto optimal set. As a consequence, implementing these types of ranking and selection methods within the model-based approach may not result in an effective algorithm. Although we are considering discrete sets of solutions, the original solution space is continuous. Therefore, there may not exist any Pareto optimal solutions in the set of candidate solutions at the current iteration. Instead of trying to find the Pareto optimal solutions, we develop an allocation heuristic that aims to correctly select the solutions that are used to guide the search process.

4.1.1 Research Results and Contribution

In this chapter we propose a method that is an extension of the SASMO algorithm for SMO denoted as SASMO-II. SASMO-II adapts the model-based approach presented in SASMO by incorporating an allocation heuristic that improves the efficiency of the original model-based approach. Specifically, the allocation heuristic assigns a larger portion of the computing budget to solutions that are used to identify the solutions that play a critical role in the search process of the algorithm. Since the best solutions, denoted as the "elite set," are used to update the parameters of the distribution, the goal of the allocation heuristic is to minimize the number of simulation replications required to correctly choose the elite set. Due to the uncertainty in computing the objective functions, establishing superiority among solutions is more difficult than in the deterministic setting.

We employ a performance metric to approximate the domination measure, termed *domination probability*, that calculates the probability that a solution dominates another solution given the current number of simulation replications allocated to both solutions. The idea is to allocate additional replications to solutions that cause a change in our belief re-

garding which solutions belong to the elite set. If additional replications of a solution have no impact on the landscape of the elite set, then we assumed that employing more replications to that solution is not helpful in guiding the search, and thus, is not necessary. The idea is to allocate replications to solutions that either cause a solution to leave the elite set or to join the elite set. Furthermore, similar to SMRAS, SASMO-II approximates the reference distributions used to influence the search process. The advantage of our algorithm is that instead of focusing on determining the Pareto optimal solution at every iteration, we design the allocation heuristic to improve the performance of the search process in the case where the objective functions cannot be computed exactly. Apart from the aforementioned multi-objective ranking and selection methods, the proposed allocation heuristic is specifically tailored to improve the efficiency of the model-based approach in solving stochastic multi-objective problems. In spite of this allocation heuristic being designed for implementation within the SASMO algorithm framework, the ideas associated with this heuristic can also be used to extend other model-based approaches to SMO.

The contributions of this proposed algorithm are twofold: 1) We develop an allocation heuristic that is designed to select a top subset of solutions, where the quality of a solution depends on multiple stochastic objectives, and 2) We propose a model-based algorithm that is designed specifically to stochastic multi-objective problems and performs well numerically in terms of efficiency and approximating the true Pareto optimal set. We measure the performance of our algorithm by adding artificial noise to popular deterministic multi-objective problems. We compare our results with other approaches based on convergence and diversity metrics.

4.2 Problem Formulation

A general stochastic multi-objective problem consists of minimizing multiple objectives over a defined solution space, which can be formulated as follows:

$$\min \left\{ \mathbb{E}_{\tau_{x,1}}[f_1(x,\tau_{x,1})], \ldots, \mathbb{E}_{\tau_{x,n}}[f_n(x,\tau_{x,n})] \right\} \tag{4.2.1}$$
$$s.t.\ \ x \in \mathscr{X},$$

where $\mathscr{X} \subset \mathbb{R}^d$ denotes the deterministic feasible region, $\{f_z(\cdot) : \mathscr{X} \to \mathbb{R},\ z = 1,...,n\}$ are scalar functions, and $\tau_{x,z}$ is the stochastic noise in the system for objective z that may or may not depend on x. We assume that $\mathscr{X}$ is a bounded subset of $\mathbb{R}^d$ and $f_z(x,\tau_{x,z})$ is measurable and integrable with respect to the distribution of $\tau_{x,z}$ for $x \in \mathscr{X}$ and for $z = 1,...,n$.

Since there rarely exists one solution that solves (4.2.1), we seek to find a set of non-dominated solutions, namely Pareto optimal solutions. Analogous to the deterministic setting, a solution x is characterized as a non-dominated solution if there does not exist a solution $y \in \mathscr{X}$ such that $f_z(y) \leq f_z(x)$ for all $z = 1,...,n$, and there exists one $j \in \{1,...,n\}$ such that $f_j(y) < f_j(x)$, where $f_z(x) := \mathbb{E}_{\tau_{x,z}}[f_z(x,\tau_{x,z})]$ for $z = 1,...,n$. In other words, non-dominated solutions are optimal in the sense that there does not exist another feasible solution whose true mean is preferred over them while taking into account all of the objectives.

To alleviate the complexity of solving (4.2.1) directly we use the concept of domination measure introduced in Chapter 3 to reformulate (4.2.1) into a stochastic single-objective problem. Recall that the domination measure of a solution can be intuitively interpreted as the size of the portion of the solution space that dominates that solution. Domination measure is advantageous for performance assessment of a solution since the performance of this metric does not depend on any fine tuning of parameters, and if a solution is Pareto optimal, it has a domination measure of zero. As pointed out in Chapter 3, reducing the dimensionality of the multi-objective problem via domination measure inevitably causes a

loss of information. However, this performance metric has shown to perform well empirically with respect to deterministic objectives. Therefore, we seek to minimize the domination measure instead of solving the original stochastic multi-objective problem. The goal is to find multiple solutions to the minimization problem

$$\min_{x \in \mathscr{X}} D(x) = \mathbb{E}_U \left[\mathbb{1} \left\{ y \prec_d x \right\} \right] \tag{4.2.2}$$

in order to construct an approximate finite representation of the Pareto optimal set.

Since calculating the true domination measure is usually computationally expensive or computationally infeasible in the case of a continuous decision space, we can estimate the domination measure by the use of Monte Carlo simulation. This is accomplished by generating random samples where every sample in the decision space is equally likely to be selected. Thus, given N independent and identically distributed (i.i.d.) samples $\{x^1, ..., x^N\}$ according to $U_{\mathscr{X}}(\cdot)$ on the solution space, the domination measure of x can be approximated by

$$\widetilde{D}(x^i) \triangleq \frac{1}{N} \sum_{j=1}^{N} \mathbb{1} \left\{ x^j \prec_d x^i \right\}, \tag{4.2.3}$$

which is an unbiased estimator of $D(x^i)$. Similar to the domination measure, the indicator function $\mathbb{1} \left\{ x^j \prec_d x^i \right\}$ cannot be computed exactly since we cannot obtain the mean $f_z(x)$ directly. We will discuss how to estimate the indicator functions in the subsequent section.

4.3 Model Based Approach for Stochastic Multi-objective Problems

The SASMO algorithm presented in Chapter 3 was originally designed for finding Pareto optimal solutions of deterministic objectives. SASMO is a model-based method that generates candidate solutions from a mixed sampling distribution from the same parameterized family of densities. It iteratively updates the parameters of the probabilistic model so that it is biased toward solutions with a low domination measure. The hope is that each component of the mixed sampling distribution will converge to degenerated sampling distributions

that are concentrated on Pareto optimal solutions.

SASMO-II is designed to integrate a modified version of SASMO with the proposed allocation heuristic. First, we explain some shortcomings of SASMO when applied to the stochastic setting and why a new algorithm is needed to tackle stochastic noise in the objectives. Then we give a full description of SASMO-II.

4.3.1 SASMO Algorithm

In every iteration of SASMO, a population of solutions is generated from a mixed sampling distribution. The set of elite solutions are selected based on their domination measure estimate and are used to update the probabilistic model. The main steps of the SASMO Algorithm are as follows:

1. Select a set of parameterized family of densities $\{g(x;\theta) : \theta \in \Theta\}$.

2. Generate candidate solutions from a mixed sampling distribution.

3. Update the parameters of the distribution by minimizing the KL divergence between the empirical distribution of the elite set of solutions and the parameterized family of densities.

The elite set at iteration k is determined by $A_k := \{x_k^i : \tilde{D}(x_k^i) \leq \gamma_k\}$, where γ_k is some threshold. Consequently, only candidate solutions that have a domination measure that is no worse than γ_k will be used in updating the parameters of the sampling distribution. Clearly, the updating step plays an integral role in the performance of the algorithm, since the updated parameters are in charge of guiding the search into promising regions of the solution space. An accurate characterization of the elite set in the deterministic setting is straightforward since the true objective values are known.

For problems in which the performance of a solution can only be observed via a noisy simulation output, one simulation is not enough. Instead, multiple simulation replications

are required to obtain a good estimate of a solution's true performance. Although the presence of stochastic noise in the system will impact the search process, the simulation error can be reduced if the elite set is correctly identified. More accurate objective evaluations can be obtained by increasing the number of replications, but increasing replications in an inefficient manner may lead to a computationally expensive algorithm.

A straightforward extension of the SASMO algorithm that solves stochastic multiple objective problems can be developed by using equal allocation in estimating the objective values of a solution (i.e., an identical number of replications are used to evaluate every solution's performance). The uncertainty in the objectives can be nearly eliminated by performing many simulations at every candidate solution to diminish the effects of the stochastic noise. The main disadvantage with implementing this strategy within the SASMO algorithm is that it could potentially be highly inefficient. For example, if the sample means of a solution have very low variance, then only a few replications would be needed to provide a good estimate. Furthermore, running several simulations equally for all candidate solutions at every iteration would require a large computational budget. Thus, it is of interest to design an allocation scheme that will allocate a larger portion of the computing budget to candidate solutions that play a bigger role in the updating process. Essentially, the allocation scheme needs to identify a subset of candidate solutions that correspond to the top solutions based on their domination measure. If the top solutions are not determined correctly, it may result in an updated parameter that directs the search in an unfavorable direction. In the next section, we describe an efficient approach for finding the top solutions based on some threshold value among a finite set of solutions.

4.3.2 Allocation Heuristic

Throughout this section we use the following notations:

- $S \subset \mathscr{X}$: A finite set of solutions where $|S| = N$.
- M^i: The number of replications allocated to solution i.

- $M^{\max}$: The total number of simulation replications available.
- A: The set of elite solutions.
- $f_z(x^i, \tau_{i,z}^s)$: The sth simulation sample for the zth objective of solution i.
- $f_z(x^i) := E[f_z(x^i, \tau_{i,z})]$: The mean for the zth objective of solution i.
- $\bar{f}_z(x^i) = \frac{1}{M_i} \sum_{s=1}^{M_i} f_z(x^i, \tau_{i,z}^s)$: The sample mean of solution i for the zth objective.
- σ_{iz}^2: The variance for the random variable $\tau_{i,z}$ that represents the noise for the zth objective for solution i.
- $\tilde{f}_z(x^i) = N(\bar{f}_z(x^i), \frac{\sigma_{i,z}^2}{M^i})$: A random variable whose probability distribution represents the posterior distribution of the mean for the zth objective for solution i.
- $\hat{f}_z(x^i) \approx N(\bar{f}_z(x^i), \frac{\sigma_{i,z}^2}{M^i + \tilde{m}^i})$: A random variable whose probability distribution represents the posterior distribution of the mean for the zth objective for solution i after $\tilde{m}^i$ additional replications have been allocated to solution i. For more details of this approximation refer to equation (4.3.8).
- $\tilde{D}(x)$: An unbiased estimator of the domination measure $D(x)$.
- $\bar{D}(x)$: The domination count of solution x.
- $\hat{D}(x)$: The domination probability of solution x.
- γ: The elite threshold value based on domination measure.
- $\bar{\gamma}$: The elite threshold value based on domination count.

The objective is to find an efficient allocation of the total computing budget of simulation replications such that the set of elite solutions A is selected. Specifically, the elite set is define as all solutions whose estimated domination measure is less than or equal to some threshold. Since we are only concerned with determining the solutions that belong to

the elite set, the rank among those solutions is irrelevant. Considering that the domination measure is estimated by a finite set of solutions and the value of the true domination measure is not important for determining the elite set, we can instead calculate the domination count of the solutions. The domination count $\bar{D}(x)$ of a solution x is defined as

$$\bar{D}(x) = \sum_{y \in S} \mathbb{1}\{y \prec_d x\}, \tag{4.3.1}$$

where $S \subset \mathscr{X}$ such that $|S| = N$. We point out that given a finite set S the domination count of a solution is equal to N times the domination measure estimate of x; thus, $\bar{D}(x) = N \times \tilde{D}(x)$. As a result, the elite set based on the domination count is equivalent to the elite set corresponding to the domination measure estimate. That is, $A = \left\{x | \tilde{D}(x) \leq \tilde{\gamma}\right\} = \{x | \bar{D}(x) \leq \bar{\gamma}\}$, where $\bar{\gamma} = N \times \tilde{\gamma}$.

Under the concept of domination count, the performance of one solution is measured in terms of its dominance relationship with respect to all other solutions in S. Since we cannot obtain $f_z(x^i)$ for $i = 1, ..., N$ and $z = 1, ..., n$, the true dominance relationship is unknown. However, we can observe random variables $f_z(x^i, \tau_{i,z}^s)$ via simulation. Therefore, given that we generate M^i i.i.d random samples $\left\{f_z(x^i, \tau_{i,z}^1), \ldots, f_z(x^i, \tau_{i,z}^{M^i})\right\}$ we can approximate the expectation $f_z(x^i) := \mathbb{E}[f_z(x^i, \tau_{i,z})]$ by the sample mean $\bar{f}_z(x^i)$. We assume that the random variable $\tau_{i,z}$ has a normal distribution with mean zero and variance $\sigma_{i,z}^2$. Typically, this normality assumption is satisfied in practice since the simulation output is usually determined from batch means or average performance; therefore, the Central Limit Theorem usually holds. If the variance $\sigma_{i,z}^2$ is known then the posterior distribution of $f_z(x^i)$ after M^i simulation replications is represented by the random variable $\tilde{f}_z(x^i) \sim N(\bar{f}_z(x^i), \frac{\sigma_{i,z}^2}{M^i})$. As M^i increases, the confidence interval of the true mean becomes more narrow. In other words, as the number of simulation replications increase we are more confident that the sample mean estimator $\bar{f}_z(x^i)$ is close to the true mean $f_z(x^i)$.

Using the sample mean, we want to estimate the domination count for all solutions in S. For simplicity in representing the dominance relation, we assume that there do not exist two distinct solutions in S that have the same mean for all objectives. Specifically, for any solutions $x, y \in S$, there exists at least one objective q such that $f_q(x) \neq f_q(y)$. Following this assumption, it is true that $\mathbb{1}\{y \prec_d x\} = \prod_{z=1}^{n} \mathbb{1}\{f_z(y) \leq f_z(x)\}$. Then, the dominance count of solution x can be expressed as

$$\bar{D}(x) = \sum_{j \in S} \mathbb{1}\{y \prec_d x\} = \sum_{j \in S \setminus x} \prod_{z=1}^{n} \mathbb{1}\{f_z(y) \leq f_z(x)\}. \tag{4.3.2}$$

As mentioned previously, the computation of the indicator function $\mathbb{1}\{f_z(y) \leq f_z(x)\}$ requires knowledge of the unknown mean $f_z(x)$ for all objectives $z = 1, ..., n$. To approximate this indicator function, we can replace the true mean $f_z(x)$ by the sample mean $\bar{f}_z(x)$ or the random variable $\tilde{f}_z(x)$. Substituting the sample mean in (4.3.2) may give partial information. That is, unless the sample means of two solutions' objectives are relatively far from each other, we cannot say for certain if a solution dominates another solution or not. For example, if the sample means of all of solution x objectives are less than the corresponding sample means of solution y objectives, there could exist one objective such that the true mean of y is less than the true mean of x. Therefore, we would misclassify x as dominating y. The sole use of sample means does not provide any information about how close the estimated domination count is to the true domination count. In contrast, if $f_z(x)$ is replaced by $\tilde{f}_z(x)$, then we can take the expectation of the indicator function

$$\mathbb{E}[\mathbb{1}\left\{\tilde{f}_z(x^j) \leq \tilde{f}_z(x^i)\right\}] = P(\tilde{f}_z(x^j) \leq \tilde{f}_z(x^i)) = P(\tilde{f}_z(x^j) - \tilde{f}_z(x^i) \leq 0), \tag{4.3.3}$$

which gives the probability that a solution dominates another solution based on the simulations performed so far. The equation (4.3.3) assumes that the random variable $\tau_{i,z}$ associated with each solution and objective are mutually independent. Therefore, we can estimate the

domination probability for solution x by

$$\hat{D}(x) = \sum_{j \in S \setminus x} \prod_{z=1}^{n} P(\tilde{f}_z(y) - \tilde{f}_z(x) \leq 0). \tag{4.3.4}$$

Basically, the domination probability can be interpreted as a continuous approximation of the domination count. In particular, the closer $P(\tilde{f}_z(y) - \tilde{f}_z(x) \leq 0)$ is to one, the higher the probability that y dominates x. Moreover, as the number of simulation replications goes to infinity, the domination probability converges to the true domination count.

Lemma 4.3.1. $\lim_{M \to \infty} \hat{D}(x) = \bar{D}(x)$, *where* $M = M^1 = M^2 = ... = M^N$

Proof. It is sufficient to show that

$$\lim_{M \to \infty} P(\tilde{f}_z(y) - \tilde{f}_z(x) \leq 0) = P(f_z(y) - f_z(x) \leq 0) \tag{4.3.5}$$

for any objective z and solution $x, y \in S$. Notice that (4.3.5) is the definition of convergence in distribution. Therefore, we need to show that $\tilde{f}_z(y) - \tilde{f}_z(x)$ converges in distribution to $f_z(y) - f_z(x)$.

By the strong law of large numbers $\lim_{M \to \infty} \frac{1}{M} \sum_{s=1}^{M} f_z(y, \tau_{y,z}^s) \overset{a.s.}{\to} f_z(y)$ and $\lim_{M \to \infty} \frac{1}{M} \sum_{s=1}^{M} f_z(x, \tau_{x,z}^s) \overset{a.s.}{\to} f_z(x)$. Since almost sure convergence implies convergence in distribution, it is true that $\lim_{M \to \infty} \frac{1}{M} \sum_{s=1}^{M} f_z(y, \tau_{y,z}^s) - \frac{1}{M} \sum_{s=1}^{M} f_z(x, \tau_{x,z}^s) \overset{d}{\to} f_z(y) - f_z(x)$. As a result,

$$\lim_{M \to \infty} P(\tilde{f}_z(y) - \tilde{f}_z(x) \leq 0) = P(f_z(y) - f_z(x) \leq 0)$$

□

Following the above lemma, we conclude that although the domination probability is not an unbiased estimator of the domination count, it is a consistent estimator. Therefore, it is rational to use the domination probability to approximate the domination count. The probabilities necessary to compute the domination probability can be easily calculated,

since the difference between two Normal distributions

$$\tilde{f}_z(y) - \tilde{f}_z(x) \sim N\left(\bar{f}_z(y) - \bar{f}_z(x), \frac{\sigma^2_{y,z}}{M^y} + \frac{\sigma^2_{x,z}}{M^x}\right) \tag{4.3.6}$$

is also a Normal distribution. To improve the accuracy of the domination probability in describing the domination count, we have to increase the number of simulation replications. The key question is how to determine the number of replications for each solution in an efficient manner. The path to answer this question starts with determining how additional replications impact the domination probability of solutions. The impact of additional replications before those replications are actually performed can be estimated by constructing a predictive posterior distribution [15]. A predictive posterior distribution is used to examine how sensitive the domination probability of a solution is to additional replications of a particular solution. Before $\tilde{m}^j$ simulation replications are allocated to solution j, the predictive posterior distribution can be represented by a random variable

$$\hat{f}_z(x^j) \sim N\left(\frac{1}{M^j + \tilde{m}^j} \sum_{s=1}^{M^j+\tilde{m}^j} f_z(x, \tau^s_{j,z}), \frac{\sigma^2_{j,z}}{M^j + \tilde{m}^j}\right). \tag{4.3.7}$$

If $\tilde{m}^j$ is relatively small when compared to M^j, we can expect that the change in the sample mean is insignificant, and thus, we can approximate the predictive posterior distribution for solution j and objective z by

$$\hat{f}_z(x^j) \approx N\left(\frac{1}{M^j} \sum_{s=1}^{M^j} f_z(x, \tau^s_{j,z}), \frac{\sigma^2_{j,z}}{M^j + \tilde{m}^j}\right). \tag{4.3.8}$$

Then, the probability that x^j dominates x^i given $\tilde{m}^j$ additional replications of solution j can be expressed as $P(x^j \prec_d x^i) = \prod_{z=1}^{n} P(\hat{f}_z(x^j) - \tilde{f}_z(x^i) \leq 0)$, where the distribution of $\hat{f}_z(x^j) - \tilde{f}_z(x^i)$ can be approximated by

$$\hat{f}_z(x^j) - \tilde{f}_z(x^i) \approx N\left(\bar{f}_z(x^j) - \bar{f}_z(x^i), \frac{\sigma^2_{j,z}}{M^j + \tilde{m}^j} + \frac{\sigma^2_{i,z}}{M^i}\right) \tag{4.3.9}$$

As a result, we can calculate the change in the probability that x^j dominates x^i given $\tilde{m}^j$ additional replications are allocated to solution j as

$$\Delta P(x^j \prec_d x^i)_{x_j} = \prod_{z=1}^{n} P(\hat{f}_z(x^j) \leq \tilde{f}_z(x^i)) - \prod_{z=1}^{n} P(\tilde{f}_z(x^j) \leq \tilde{f}_z(x^i)), \tag{4.3.10}$$

where $i \neq j$. The above equation for $\Delta P(x^j \prec_d x^i)_{x^j}$ can be interpreted as how the probability that j dominates i will be affected if the variance of $\tilde{f}_z(x^j)$ decreases for all z. The magnitude of the change in probability is comparable to the signal noise ratio in the OCBA literature. That is, if the magnitude of the change is small, then that indicates that either the sample means of one solution is much worse than the other solution or the sum of their variances are low. In either case, we are confident in whether x^j dominates x^i. This implies that additional replications of x^j are not needed to get a more accurate estimate of the domination count for solution x^i. On the other hand, if the difference of the sample means is close to zero, then the domination probability will change if the variance decreases. The magnitude of this change will depend on which solution is assigned additional replications. In particular, if $\sigma^2_{j,z} > \sigma^2_{i,z}$, then additional replications of solution j will result in a bigger change of probability than additional replications of solution i. Essentially, if we are more certain about the true mean of a particular solution, then additional replications of that solution will not affect the domination probability as much as a solution that we are less confident about.

Additional replications allocated to a solution may impact its own domination probability as well as the domination probability of other solutions. Therefore, we can examine how more replications of a solution influence the domination probability of every solution in S. Unlike in single objective problems where more replications of a certain solution only affects the function evaluation of that solution, in this case, the domination probability of a solution could be affected by more replications of all the solutions in S. Thus, we want to consider how the change in domination probability for each solution would affect the pro-

file of the elite set. We employ an allocation scheme that allocates replications to samples that cause the most elite solutions to leave the elite set and the most non-elite solutions to enter the elite set.

First, we determine how the domination probability of a solution changes with additional replications of all the solutions in S. Lets consider the case where $j \neq i$, then the change in domination probability of solution i is calculated by

$$\Delta\hat{D}(x^i)_{x^j} = \prod_{z=1}^{n} P(\hat{f}_z(x^j) \leq \tilde{f}_z(x^i)) - \prod_{z=1}^{n} P(\tilde{f}_z(x^j) \leq \tilde{f}_z(x^i)). \tag{4.3.11}$$

From the above equation, recognize that more replications of solution j can only change the domination probability of i at most by one. In contrast, when $j = i$, the change in domination probability is given by

$$\Delta\hat{D}(x^i)_{x^i} = \sum_{j \in S \setminus i} \prod_{z=1}^{n} P(\hat{f}_z(x^j) \leq \bar{f}_z(x^i)) - \sum_{j \in S \setminus i} \prod_{z=1}^{n} P(\bar{f}_z(x^j) \leq \bar{f}_z(x^i)). \tag{4.3.12}$$

In this case, the domination probability of i can change with respect to all $j \in S \setminus i$.

From equations (4.3.11) and (4.3.12), we can calculate how additional replications of solutions in S will change any solution's domination probability. Following the logic of OCBA approaches, we aim to allocate the majority of the computational budget to solutions that are critical to the process of identifying the best solutions. In this context, critical solutions are defined as solutions that are close to the elite threshold $\bar{\gamma}$. We classify solutions as being critical if there are likely to change from being an elite solution to a non-elite solution or vice versa. That is, if additional replications cause current elite solutions' domination probabilities to be close to or greater than the threshold, then those solutions are considered critical. Similarly, a non-elite solution is considered critical if additional replications cause its domination probability to be close to or less than the threshold. As result, we construct the indicator function $\delta^i = 1$ if solution i is a critical solution and $\delta^i = 0$

otherwise as follows

$$\delta_{\varepsilon}^{i} = \mathbb{1}\left\{\hat{D}(x^i) + \sum_{j=1}^{N} |\Delta\hat{D}(x^i)_{x^j}| \geq \bar{\gamma} - \varepsilon\right\} \forall i \in A. \tag{4.3.13}$$

$$\delta_{\varepsilon}^{i} = \mathbb{1}\left\{\hat{D}(x^i) + \sum_{j=1}^{N} |\Delta\hat{D}(x^i)_{x^j}| \leq \bar{\gamma} + \varepsilon\right\} \forall i \notin A. \tag{4.3.14}$$

The value $\bar{\gamma}$ can be thought as a barrier dividing the elite solutions from the non-elite solutions. The further away a solution in the elite set is from this barrier, the more we are confident that that solution actually belongs to the elite set. We measure how close solutions are to $\bar{\gamma}$ by adding their change in domination probability given additional replications to their current domination probability. If the change in domination probability causes a solution performance to become at least ε away from γ, then we classify that solution as a critical solution.

Now that critical solutions have been identified, the idea is to allocate more replications to solutions that change the critical solutions' domination probabilities the most. The number of replications we allocate to solution j is proportional to the magnitude of the change in domination probability of the critical solutions given additional replications of solution j. Additional replications M^j of solution j are calculated by

$$M^j = \frac{W^j}{\sum_i W^i} M^{max}, \quad W^j = \sum_{i=1} \delta_{\varepsilon}^{i} \Delta\tilde{D}(x^i)_{x^j}. \tag{4.3.15}$$

The above allocation equation gives the majority of the computational effort to solutions that have the biggest effect on the domination probability of the critical solutions. The idea is that we do not want to waste our computational resources on solutions that will have no impact on changing the elite set.

We present an effective sequential allocation algorithm denoted as the Domination Probability Allocation (DPA) heuristic that seeks to characterize the elite set $A = \{x|\bar{D}(x) \leq \bar{\gamma}\}$ by utilizing the change in domination probability. The sequential aspect of the algorithm

allows the use of updated sample means and variances to play a role in deciding the allocation of additional replications. At each iteration of the algorithm we update the sample means and variances of every solution and construct the observed elite set A accordingly. The propose heuristic given initial replications m, maximum number of replications M^{max}, threshold $\bar{\gamma}$, constant Δ_t, and a sequence $\{\varepsilon_t : t = 1, ...\}$ is outlined as follows:

Algorithm 5 Dominaiton **P**robability **A**llocation Heuristic

Step 1: Perform m simulation replications for all candidate solutions $M^1 = M^2 = ... = M^N = m$. Set $t = 0$.

Step 2: Calculate the domination probability $\hat{D}(x^i)$ for $i = 1, 2, ..., N$ and construct the observed elite set $A = \left\{x^i | \hat{D}(x^i) \leq \bar{\gamma}\right\}$ and determine δ^i.

Step 3: If $\sum_{i=1}^{N} M^i < M^{max}$ and $\sum_{i=1}^{N} \delta_{\varepsilon_t}^i > 0$, set $t \rightarrow t+1$ and go to step 4, otherwise terminate algorithm and output the final observed elite set A.

Step 4: Increase the total computing budget by Δ_t (s.t. $\Delta_t \leq M^{max} - \sum_{i=1}^{N} M^i$) and determine the new allocation $M_t^1, ..., M_t^N$, where $M_t^i = \lfloor \frac{W^i}{\sum_j W^j} \Delta_t \rfloor$.

Step 5: Perform M_t^i additional replications of solution i update the total number of replications $M^i = M^i + M_t^i$ for all i and go to Step 2.

First, m initial simulation replications are conducted for all N solutions and the domination probabilities are calculated from those preliminary simulations. At every iteration of the heuristic the observed elite set is constructed and the number of additional replications for every solution is calculated. The sequence ε_t is required to go to zero as t goes to infinity. This requirement ensures that, at the beginning of the algorithm, we are more conservative in identifying solutions as critical, but as the algorithm progresses, we are more aggressive in classifying solutions as critical. The algorithm terminates when the number of simulation replications reaches the maximum computing budget or when no critical solutions have been identified.

To provide some insight into the effectiveness of this allocation scheme, we compare the DPA heuristic with the Uniform Computing Budget Allocation (UCBA) algorithm, which allocates the same number of replications to all solutions. We consider the multi-objective

problem

$$[f_1(x), f_2(x)] = [1 - \exp(-\sum_{i=1}^{3}(x_i - \frac{1}{\sqrt{3}})^2) + \tau_1, 1 - \exp(-\sum_{i=1}^{3}(x_i + \frac{1}{\sqrt{3}})^2) + \tau_2], \quad (4.3.16)$$

where $\tau_1, \tau_2 \sim N(0,1)$, $\mathscr{X} := [-4,4]^3$ and $N = 10$. The elite set A is characterized by $\bar{\gamma} = 1.5$. The true domination count of the solutions considered is illustrated in Table 4.3.1, in which the elite solutions are bolded.

Table 4.3.1: Solution's True Domination Count

Solution	$f(x)$	$\bar{D}(x)$	Solution	$f(x)$	$\bar{D}(x)$
1. **(.45,.15,-.23)**	[.5729,.8183]	0	6. **(-.11,.03,-.44)**	[.8359,.4546]	0
2. **(.26,.30,0)**	[.4001,.8354]	0	7. (-.83,.73,1)	[.8872,.9859]	4
3. **(-.58,-.51,-.98)**	[.9929,.1535]	0	8. (3,.97,-.41)	[.9991,1.000]	9
4. **(-.31,.26,-.02)**	[.7120,.6615]	0	9. **(-.61,-.98,-.46)**	[.9926,.1622]	0
5. (-.68,-.93,-.91)	[.9977,.2177]	2	10. (2,-.49,.18)	[.9639,.9993]	5

To determine an approximate number of replications for the UCBA algorithm, we apply a brute force search. Given that we must allocate $\frac{M^{max}}{10}$ replications for every solution, we determine the maximum computing budget that results in the UCBA algorithm identifying all solutions that have a true domination count less than 1.5. We initialize the brute force search such that $M_{max} = 10,000$ and increment M^{max} by 1000 replications until the algorithm correctly selects the elite set.

We generate 20 replications of the DPA heuristic and the brute force search applied to the UCBA algorithm. The parameters selected for the DPA heuristic are $M^{max} = 20,000$, $m = 20$, $\varepsilon_t = .9^t$, $\tilde{m}^j = 10$ for $j = 1,..,10$, and $\Delta_t = 500$. We point out that the proposed allocation algorithm correctly chose the elite set in every experiment. Table 4.3.2 and Figure 4.3.1 displays the average number of the replications that are allocated to each solution for both the DPA heuristic $\bar{M}_{DPA}$ and UCBA $\bar{M}_{UCBA}$ from 20 independent numerical exper-

iments. The DPA heuristic took a total of 2,000 simulation replications compared to 4,250 simulation replications required by the UCBA algorithm to obtain the elite set. The results indicate that the DPA heursitic is more efficient in correctly identifying the true set of elite solutions than the UCBA algorithm. Intuitively, this is due to the fact that the DPA heuristic only allocates replications to solutions that are sensitive to calculating the elite set.

Table 4.3.2: UCBA vs Allocation Algorithm Results

Solution	$\bar{M}_{DPA}$	$\bar{M}_{UCBA}$	Solution	$\bar{M}_{DPA}$	$\bar{M}_{UCBA}$
1. **(.45,.15,-.23)**	712	425	6. **(-.11,.03,-.44)**	85	425
2. **(.26,.30,0)**	249	425	7. (-.83,.73,1)	20	425
3. **(-.58,-.51,-.98)**	294	425	8. (3,.97,-.41)	20	425
4. **(-.31,.26,-.02)**	224	425	9. **(-.61,-.98,-.46)**	54	425
5. (-.68,-.93,-.91)	320	425	10. (2,-.49,.18)	22	425

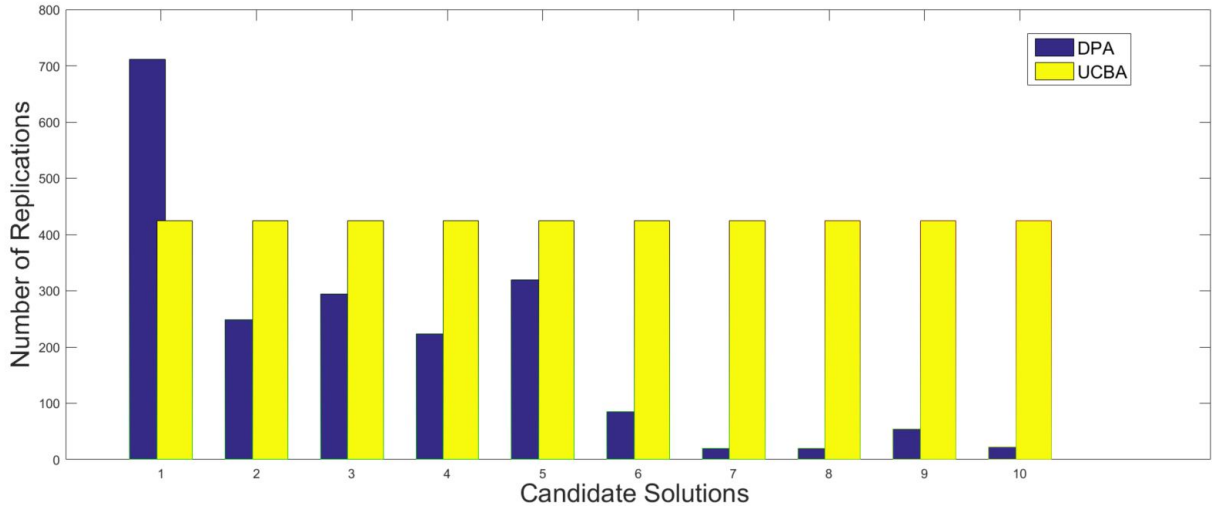


Figure 4.3.1: UCBA vs Allocation Algorithm Results

Since the DPA heuristic is designed for selecting a subset of the top solutions, its use would be beneficial for model-based algorithms that require a characterization of an elite set. When model-based methods are applied to a stochastic setting, the efficiency of these methods is strongly linked with how efficient the elite set can be determined. We combine

this heuristic with a modified version of the SASMO method introduced in chapter 3 to develop SASMO-II. Along with some modification to the SASMO method for the stochastic setting, the DPA heuristic is employed to minimize the total computing budget necessary to evaluate the performance of candidate solutions and correctly identify the elite set at every iteration of the algorithm. Although this heuristic is implemented in the framework of the SAMSO algorithm, it could also be applied to any model-based algorithm for stochastic multi-objective optimization problems.

4.3.3 SASMO-II Algorithm

As mentioned before, extending the SASMO algorithm to simulation multi-objective problems requires handling the simulation error in an efficient manner. A naive approach would be to allocate simulation replications uniformly to each candidate solution. In the previous subsection, we show how inefficient an equal allocation procedure can be for determining the elite set of solutions. Instead of using equal allocation for estimating the performance of the candidate solutions, we apply the DPA heuristic to enhance the computational efficiency of a model-based approach in the multi-objective stochastic domain. Other key modifications of the SASMO algorithm are required to adapt the SASMO framework to stochastic multi-objective problems. SASMO-II extends the SASMO algorithm by using approximations for the reference distribution and integrating the DPA heuristic. A detailed description of SASMO-II is discussed below.

The algorithm is initialized by choosing a parameterized family of densities $\{g(\cdot,\theta) : \theta \in \Theta\}$, where the initial parameters of the sampling distribution are chosen arbitrarily if no knowledge about where good solutions are located in the decision space is available. For example, if the family of multi-variate Gaussian distributions is chosen, then the mean vector can arbitrarily be chosen and the trace of the covariance matrix should be large relative to the size of the feasible region so that all solutions have a significant probability of being sampled. Since we must maintain a tradeoff between searching in the solution space (exploration)

Algorithm 6 SASMO-II

1. **Initialization**: Choose a parameterized family of densities $\{g(x;\theta) : \theta \in \Theta\}$ with initial parameter. Determine values for the sample size N_0, maximum computing budget M_0^{max} percent quantile ρ, initial sampling parameters $\theta_{1,0}$, maximum number of iterations $t_{\max} > 1$, mixing coefficient $\alpha \in (0,1)$, and constants $\omega_1 > 1$,$\omega_2 > 1$, $\eta > 0$. Specify sequence of initial simulation replications $\{m_k : 1,...\}$ and the sequence required for the DPA heuristic $\{\varepsilon_t : 1,...\}$. Set $k = 0$ and $I_0 = 1$.

2. **Sampling**: Draw N_k i.i.d. candidate solutions $\{x_k^i : i = 1,2,...,N_k\}$ according to $\mathbf{g}_k(\cdot)$, where

$$\mathbf{g}_k(\cdot) \stackrel{\triangle}{=} (1-\alpha)g_k(\cdot) + \alpha U_{\mathscr{X}}(\cdot)$$

where, $g_k(x) = \frac{1}{I_k}\sum_{i=1}^{I_k} g(x;\theta_{i,k})$, and $U_{\mathscr{X}}(\cdot)$ is the uniform distribution on $\mathscr{X}$.

3. **Simulating**: Calculate the threshold value $\bar{\gamma}_k$ and determine the sample size N_{k+1}. Determine the number of replications M_k^i of solution x_k^i and the elite set $\tilde{A}_k$ from the DPA heuristic with input parameters M_k^{max}, $\{\varepsilon_t : 1,...\}$, and $\bar{\gamma}_k$. Set $M_{k+1}^m ax = \omega_2 * M_k^m ax$.

4. **Updating**: Update the parameter of the sampling distribution by θ_{k+1} according to the SASMO algorithm.

5. **Terminating**: Check if some terminating criterion is satisfied. If yes, stop and return the means of the current parameterized sampling distributions; else, set $k := k+1$ and go back to step 2.

and obtaining an accurate estimate of a good solution (exploitation), the algorithm has two allocation rules. The first allocation rule denoted by N_k deals with the exploration part of the algorithm, where the number of candidate solutions generated for the sampling distribution in iteration k is N_k. The second allocation rule $\left\{M_k^i, i = 1,...,N_k\right\}$ is associated with the exploration aspect of the algorithm which is determined by the DPA heuristic where M_k^{max} determines the total computing budget for iteration k. To be clear, without stochastic noise, increasing N_k ensures that our domination measure estimate is close to the true domination measure. Whereas increasing M_k^{max} improves our certainty that our comparison of any two solutions is accurate. Therefore, as both N_k and M_k^{max} increase our domination measure, estimates get closer to the true domination measure. Essentially, the stochastic noise in the objectives requires another level of simulation when compared to the deterministic setting. That is, the outer-layer simulation corresponds to sampling on the solution space, whereas

the inner-layer simulation corresponds to improving the certainty of determining the best solutions. The constants ω_1 and ω_2 control the rate of increase in N_k and M_{max}, respectively. The percent quantile ρ_k is used to calculate the threshold $\bar{\gamma}_k$. Lastly, α facilitates the proportion of samples that are generated from a uniform distribution.

In the sampling step of our algorithm, we generate candidate solutions according to a mixture distribution, where solutions are generated from $\mathbf{g}(\cdot)$ with probability $(1-\alpha)$ and a uniform distribution with probability α. Generating candidate solutions from a mixture distribution allows the probabilistic model to have the ability to concentrate on specific regions of the decision space. The former component is updated at every iteration based on the solutions that are in the set of elite solutions. The latter component is used to ensure that solutions from the entire solution space are used to calculate the domination probability. Given enough simulation replications, this produces an accurate estimate of the domination measure, regardless of how biased the first component is to a specific region of the decision space. Furthermore, the latter component is especially necessary in the later stages of the algorithm, since the first component will be concentrated on particular regions of the decision space.

In the simulation step, the DPA heuristic is used to select the elite set of solutions with parameter M_k^{max}. The threshold value $\bar{\gamma}_k$ is calculated by the quantile estimate of the domination probability from the initial simulation replications m_k in iteration k. The quantile estimate $\gamma_k = \hat{D}_{\lceil \rho_k * N_k \rceil}$ depends on both the sample size N_k and percent quantile ρ_k. Therefore, the choice of N_k and ρ_k are both important issues for empirical performance. We construct the threshold value such that it is strictly decreasing with respect to k as follows:

- **If** the algorithm is in the first iteration; i.e $k = 0$, then use the current quantile estimate $\hat{D}_{\lceil \rho_k * N_k \rceil}$ as the threshold value, i.e $\bar{\gamma}_k = \gamma_k$, set $\rho_{k+1} = \rho_k$ and increase the sample size by ω_1, $N_{k+1} = \omega_1 N_k$.

- **Elseif** use the current quantile estimate γ_k as the threshold value only if there was some decrease in the current quantile estimate with respect to the previous threshold

value, $\gamma_k \leq \max\{0, \bar{\gamma}_{k-1} - \eta\}$ then set $\bar{\gamma}_k = \gamma_k$, $\rho_{k+1} = \rho_k$, and $N_{k+1} = \omega_1 N_t$.

- **Else** find the largest $\hat{\rho} \in (0, \rho_k)$ such that the quantile estimate $\hat{D}_{\lceil \rho_k * N_k \rceil}$ is less than the previous threshold value, $\hat{D}_{[\lceil \hat{\rho} N_k \rceil]} \leq \max\{0, \bar{\gamma}_{k-1} - \eta\}$ then set $\bar{\gamma}_k = \hat{D}_{[\lceil \hat{\rho} N_k \rceil]}$, $\rho_{k+1} = \hat{\rho}$, and $N_{k+1} = \omega_1 N_k$. If such a $\hat{\rho}$ cannot be found, increase the initial number of simulation replications until a $\hat{\rho}$ can be found. Since there always exists a solution that has a true domination count of zero, we can always find such a $\hat{\rho}$ with enough simulation replications.

At every iteration, the sample size strictly increases and the threshold value decreases until it reaches its lower bound of zero. These conditions are needed to ensure that the quality of the elite solutions improves at every iteration. Furthermore, the improving threshold value allows the sampling distribution to quickly direct the search to promising areas of the decision space.

In the final step of our algorithm, we update the parameters of the sampling distribution according to SASMO, where the reference sampling distribution is approximated by the empirical distributions of the elite solutions based on their domination probability. Due to both allocation rules increasing, the domination probability will become closer to N times the domination measure as the number of iterations increases. The hope is that the updating process causes each component of the sampling distribution to converge to a degenerate distribution concentrated only on Pareto optimal solutions.

4.4 Numerical Results

In this section, we test SASMO-II with some modified deterministic multi-objective optimization problems, where the objectives are altered to mimic stochastic ones. In particular, we develop test problems by incorporating Gaussian noise to the deterministic objectives. This Gaussian noise is denoted by a random variable τ which is normally distributed with mean 0 and variance 10. The objectives are altered to minimize the expectation of the

multiple functions with respect to τ. The constructed test functions are listed below:

- Modified MOP1

$$\begin{aligned} \min \ \mathbf{f}(x) &= (\mathbb{E}_\tau[f_1(x,\tau)], \mathbb{E}_\tau[f_2(x,\tau)]) \\ f_1(x) &= (\tau x)^2 \\ f_2(x) &= (x+\tau-2)^2 \\ x &\in [-10^3, 10^3] \end{aligned}$$

- Modified MOP2

$$\begin{aligned} \min \ \mathbf{f}(x) &= (\mathbb{E}_\tau[f_1(x,\tau)], \mathbb{E}_\tau[f_2(x,\tau)]) \\ f_1(x) &= 1-\exp(-\sum_{i=1}^{3}(x_i-\frac{1}{\sqrt{3}})^2)+\tau \\ f_2(x) &= 1-\exp(-\sum_{i=1}^{3}(x_i+\frac{1}{\sqrt{3}})^2)+\tau \\ x_i &\in [-4,4], \quad i=1,2,3 \end{aligned}$$

- Modified ZDT3

$$\begin{aligned} \min \ \mathbf{f}(x) &= (\mathbb{E}_\tau[f_1(x,\tau)], \mathbb{E}_\tau[f_2(x,\tau)]) \\ f_1(x) &= x_1+\tau \\ f_2(x) &= g(x)[1-(x_1/g(x))^2-\frac{x_1}{g(x)}\sin(10\pi x_1)]+\tau, \ \text{where} \\ g(x) &= 1+9(\sum_{i=2}^{30}x_i)/(30-1) \\ x_i &\in [0,1], \ i=1,\ldots,30. \end{aligned}$$

- Modified ZDT4

$$\min \ \mathbf{f}(x) = (\mathbb{E}_\tau[f_1(x,\tau)], \mathbb{E}_\tau[f_2(x,\tau)])$$

$$
\begin{aligned}
f_1(x) &= x_1 + \tau \\
f_2(x) &= g(x)(1 - \sqrt{x_1/g(x)}) + \tau, \text{ where} \\
g(x) &= 1 + 10(10-1) + \sum_{i=2}^{10}(x_i^2 - 10\cos(4\pi x_i)) \\
x_i &\in [-5,5], \ i = 1, \ldots 10.
\end{aligned}
$$

The above test problems combine the complexity of the original multi-objective problems with the additional challenge of dealing with uncertainty in the objective functions. These problems are developed to imitate a stochastic multi-objective problem in the sense that the expectations must be approximated via simulation.

In order to quantify the performance of our algorithm, we use the convergence metric Λ and diversity metric Υ as seen in Chapter 3 and the total number of objective evaluations T. As a reminder, the convergence metric measures the distance between the estimated Pareto optimal solutions and the true set of Pareto optimal solutions, and the diversity metric measures how well spread the solutions are in the decision space.

We compared the performance of SASMO-II with NSGA-II, MOPSO, and SPEA-II implemented with an equal allocation rule. For the aforementioned methods, we choose the following parameters:

- Number of generations: 250
- Population size: 100
- Archive size: 100
- Simulation Allocations: 10.

For each problem instance, we performed 30 independent replications of each method implemented in MATLAB. We reported the best value for the convergence metric (Λ) and the average number of total objective evaluations (T) obtained out of the 30 trials for each method.

SASMO-II terminates when the current iteration number equals the maximum number of iterations k_{max} or when the threshold distance of the clustering algorithm is less than the threshold bound. We choose the following parameters for SASMO-II: initial sample size $N_0 = 500$, initial simulation replications $m_0 = 5$ maximum computing budget $M_0^{max} = 500$, percent quantile $\rho_0 = .10$, initial mean $\mu_0 = \mathbf{0}$, initial covariance matrix $\Sigma_0 = 1000\mathbf{I}_d$, $t_{max} = 25$, mixing coefficient $\alpha = 0.1$, $\tilde{m}^j = 10$ for $j = 1,..,N$, constants $\omega_1 = 1.01$, $\omega_2 = 1.10$, $\eta = .01$, $\Delta = 300$ and sequences $m_k = m_{k-1} + 2$ and $\varepsilon_t = .9^t$.

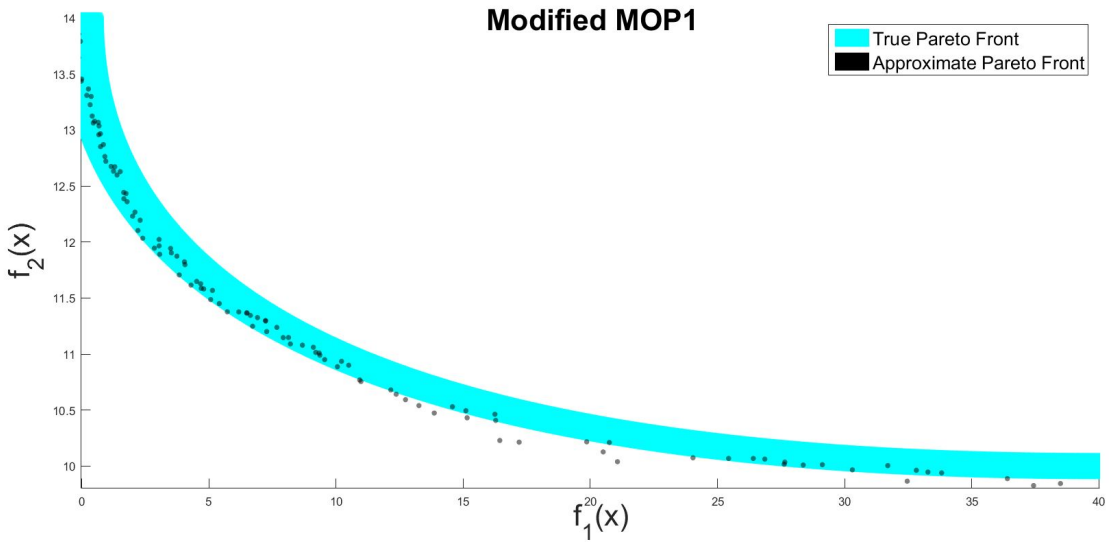


Figure 4.4.1: Modified MOP1 Results.

The results are summarized in Table 4.4.1 - 4.4.3, and the resulted approximations for the Pareto fronts produced by SASMO-II are illustrated in Figures 4.4.1 - 4.4.4. We can see that SASMO-II outperforms the existing methods for all the problems with respect to the convergence metric and total number of objective evaluations. Although the SASMO-II performed the least amount of objective evaluations with respect to all test problems, the approximated Pareto front produced by SASMO-II was the best compared to the other methods. We contribute this to the integration of the DPA heuristic within SASMO-II. The DPA heuristic causes our proposed method to be less sensitive to simulation error caused by stochastic noise in the objective functions. We point out that the diversity metric results are similar to those yielded by the original SASMO algorithm, since the distance between

the estimated solutions in the decision space is not directly affected by the stochastic nature of the problem.

In summary, the numerical results indicate that SASMO-II has the ability to obtain approximate Pareto solutions for problems with complex Pareto front structures with the addition of stochastic noise. The results also demonstrate that approximating the reference distributions by the domination probability has little effect on the capability of the proposed method to find Pareto optimal solutions. Additionally, the DPA heuristic outperforms equal allocation in terms of the number of simulation replications required to successfully identify the elite set of solutions.

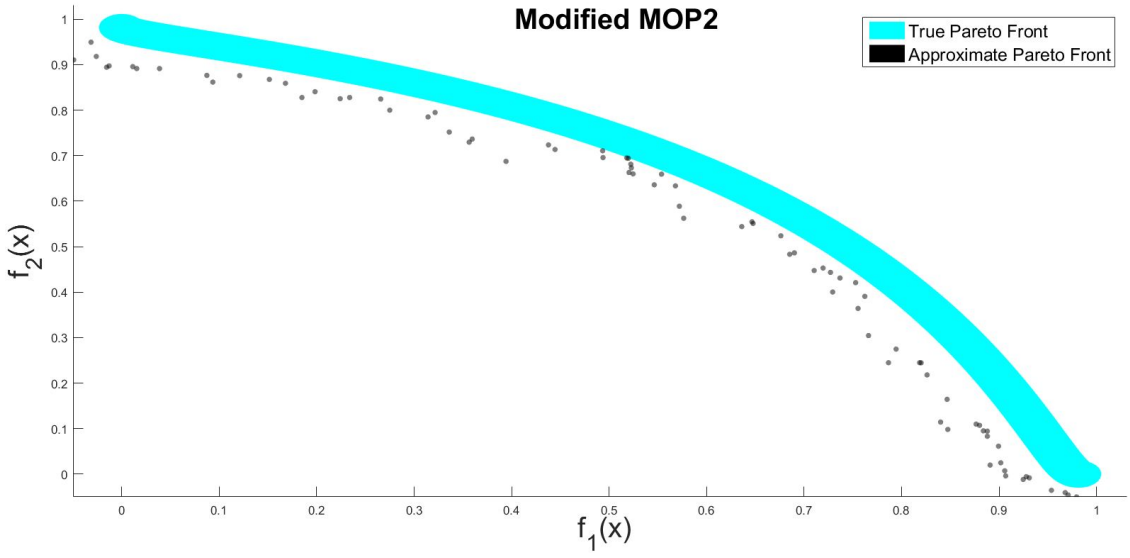


Figure 4.4.2: Modified MOP2 Results.

Table 4.4.1: Comparisons of Convergence Metric Λ.

Problem	SASMO-II	NSGA-II	SPEA-II	MOPSO
	Λ	Λ	Λ	Λ
MOP1	.8276	3.942	4.011	4.284
MOP2	1.962	5.287	6.422	7.023
ZDT3	3.178	12.982	14.981	14.502
ZDT4	1.434	11.2173	12.763	13.301

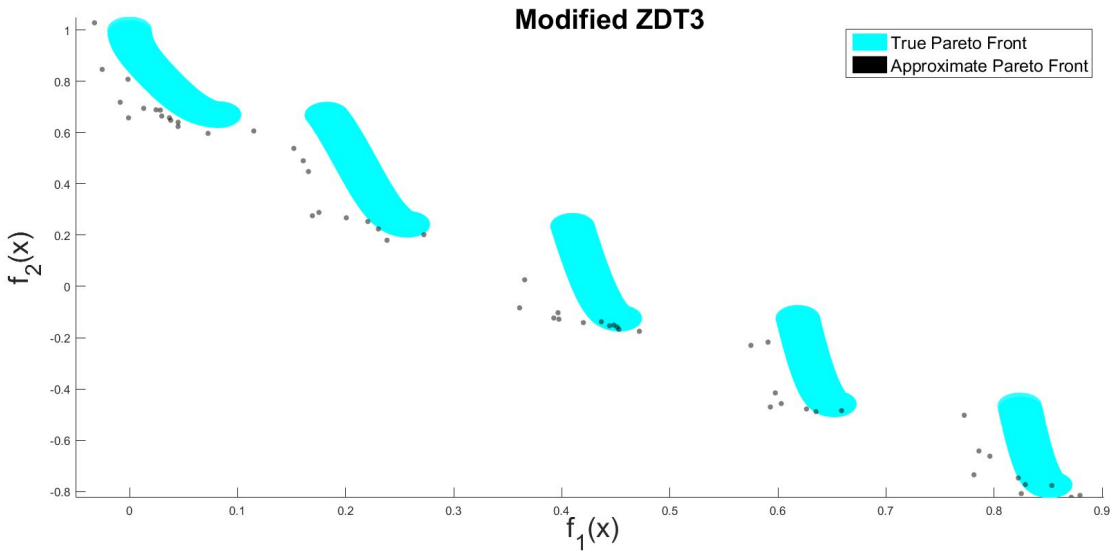


Figure 4.4.3: Modified ZDT3 Results.

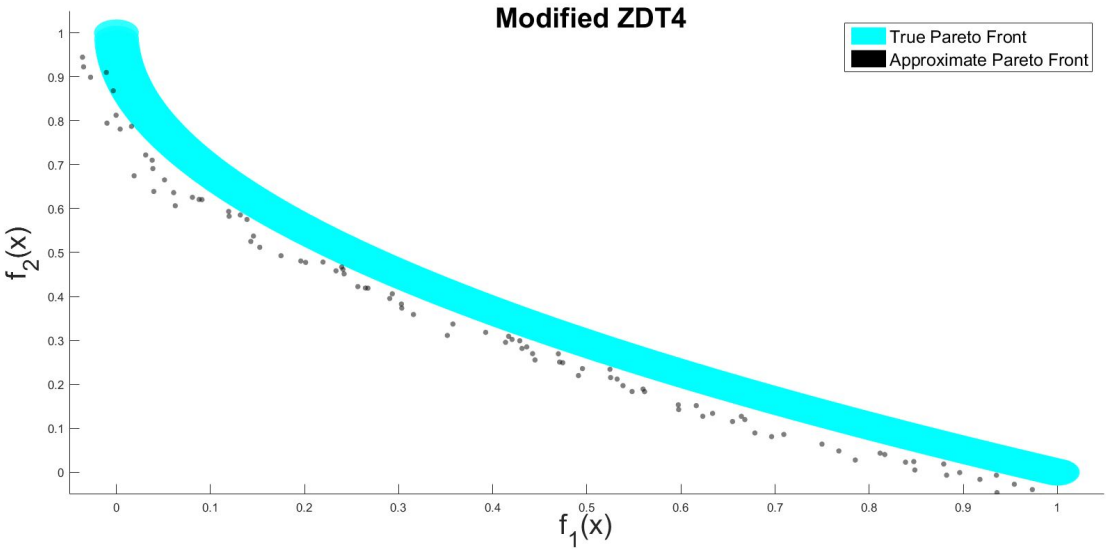


Figure 4.4.4: Modified ZDT4 Results.

4.5 Conclusions and Future Research

We have proposed SASMO-II for solving stochastic optimization problems in the multi-objective domain. This method extends the SMASO algorithm presented in Chapter 3 and incorporates an allocation heuristic within every iteration. Instead of finding the Pareto optimal solutions, the DPA heuristic is designed to determine how the simulation replications should be allocated in order to successfully identify the elite solutions. As a result,

Table 4.4.2: Comparisons of Diversity Metric Υ.

Problem	SASMO-II	NSGA-II	SPEA-II	MOPSO
	Υ	Υ	Υ	Υ
MOP1	.02143	.0031	.0289	.4415
MOP2	.0781	.0632	.1621	.9136
ZDT3	.7699	.6214	.9822	1.003
ZDT4	.4967	.5812	1.275	1.344

Table 4.4.3: Total Number of Objective Evaluations T.

Problem	SASMO-II	NSGA-II	SPEA-II	MOPSO
	T	T	T	T
MOP1	90,842	250,000	250,000	250,000
MOP2	101,779	250,000	250,000	250,000
ZDT3	153,913	250,000	250,000	250,000
ZDT4	149,868	250,000	250,000	250,000

the SASMO-II algorithm is effective at searching for good solutions in the solution space while at the same time allocating the number of simulation replications to every solution in order to enhance the computational efficiency of the algorithm.

The numerical results indicate that implementing an allocation methodology specifically to improve the performance of the model-based framework results in an efficient algorithm designed for SMO. It should be straightforward to extend other model-based algorithms designed for deterministic multi-objective optimization by implementing the DPA heuristic in the evaluation step of the algorithms. It is important to recognize that the DPA heuristic does not improve the search process but attempts to minimize the computational budget required to alleviate the simulation error in the search process due to the stochastic nature of the problem.

To enhance the performance of SASMO-II, one future research direction is to consider the effects the parameters have on the performance of the algorithm and to investigate the theoretical convergence of SASMO-II. Furthermore, determining the optimal number of samples required for both the inner-layer and outer-layer simulation is still an open area of

research. Another future research topic is extending this method to solve multi-objective problems with constraints under uncertainty.

REFERENCES

[1] M. R. Ackermann, J. Blömer, and C. Sohler, "Clustering for metric and nonmetric distance measures," *ACM Transactions on Algorithms*, vol. 6, no. 4, pp. 1–26, 2010.

[2] S. Arora, P. Raghavan, and S. Rao, "Approximation schemes for euclidean k-medians and related problems," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, ser. STOC '98, Dallas, Texas, USA: ACM, 1998, pp. 106–113, ISBN: 0-89791-962-9.

[3] V. Arya, "Local search heuristics for k-median and facility location problems," *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 21–29, 2001.

[4] P. Avella, A. Sassano, and I. Vasil'ev, "Computational study of large-scale p-median problems," *Mathematical Programming*, vol. 109, no. 1, pp. 89–114, 2007.

[5] F. Barahona and R. Anbil, "The volume algorithm: producing primal solutions with a subgradient method," *Mathematical Programming*, vol. 87, no. 3, pp. 385–399, 2000.

[6] J. E. Beasley, "Lagrangean heuristics for location problems," *European Journal of Operational Research*, vol. 65, no. 3, pp. 383–399, 1993.

[7] J. Bekker and C. Aldrich, "The cross-entropy method in multi-objective optimisation: an assessment," *European Journal of Operational Research*, vol. 211, no. 1, pp. 112–121, 2011.

[8] C. Beltran, C. Tadonki, and J. Vial, "Solving the p-median problem with a semi-lagrangian relaxation," *Computational Optimization and Applications*, vol. 35, no. 2, pp. 239–260, 2006.

[9] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2008.

[10] J. Branke and W. Zhang, "A new myopic sequential sampling algorithm for multi-objective problems," in *Winter Simulation Conference (WSC), 2015*, IEEE, 2015, pp. 3589–3598.

[11] A. Ceselli, "Two exact algorithms for the capacitated p-median problem," *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, vol. 1, no. 4, pp. 319–340, 2003.

[12] H. Chang, J. Hu, M. Fu, and S. Marcus, "Model reference adaptive search," in *Simulation-Based Algorithms for Markov Decision Processes*, ser. Communications and Control Engineering, Springer London, 2013, pp. 89–177, ISBN: 978-1-4471-5021-3.

[13] M. Charikar, S. Guha, va Tardos, and D. B. Shmoys, "A constant-factor approximation algorithm for the k-median problem," *Journal of Computer and System Sciences*, vol. 65, no. 1, pp. 129 –149, 2002.

[14] C.-H. Chen, D. He, M. Fu, and L. H. Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579–595, 2008.

[15] C.-h. Chen and L. H. Lee, *Stochastic simulation optimization: an optimal computing budget allocation*. World scientific, 2011, vol. 1.

[16] T. Chen and C. Wang, "Multi-objective simulation optimization for medical capacity allocation in emergency department," *Journal of Simulation*, vol. 10, no. 1, pp. 50–68, 2016.

[17] F. Chiyoshi and R. Galvo, "A statistical analysis of simulated annealing applied to the p-median problem," *Annals of Operations Research*, vol. 96, no. 1-4, pp. 61–74, 2000.

[18] M. Chrobak, C. Kenyon, and N. Young, "The reverse greedy algorithm for the metric k-median problem," *Information Processing Letters*, vol. 97, no. 2, pp. 68–72, 2006.

[19] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2002, vol. 242.

[20] D. W. Corne, N. R. Jerram, J. D. Knowles, M. J. Oates, *et al.*, "Pesa-ii: region-based selection in evolutionary multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO2001*, Citeseer, 2001.

[21] G. Corneuejols, "Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms," *Management Science*, vol. 23, pp. 789–810, 1977.

[22] P. Czyżżak and A. Jaszkiewicz, "Pareto simulated annealinga metaheuristic technique for multiple-objective combinatorial optimization," *Journal of Multi-Criteria Decision Analysis*, vol. 7, no. 1, pp. 34–47, 1998.

[23] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.

[24] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16.

[25] G. Feldman, S. R. Hunter, and R. Pasupathy, "Multi-objective simulation optimization on finite sets: optimal allocation via scalarization," in *2015 Winter Simulation Conference (WSC)*, IEEE, 2015, pp. 3610–3621.

[26] M. Fisher, "Lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, pp. 1–18, 1981.

[27] M. Fleischer, "The measure of pareto optima applications to multi-objective metaheuristics," in *Evolutionary multi-criterion optimization*, Springer, 2003, pp. 519–533.

[28] G. N. G. Cornuejols and L. Wolsey, *The uncapacitated facility location problem*. Wiley, New York, 1989.

[29] R. Galvao and C ReVelle, "A Lagrangean heuristic for the maximal covering location problem," *European Journal of Operational Research*, 1996.

[30] F. Glover and M. Laguna, *Tabu Search*. Springer, 2013.

[31] H. Hassan-Pour, M Mosadegh-Khah, and R Tavakkoli-Moghaddam, "Solving a multi-objective multi-depot stochastic location-routing problem by a hybrid simulated annealing algorithm," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 223, no. 8, pp. 1045–1054, 2009.

[32] D. He, L. H. Lee, C.-H. Chen, M. C. Fu, and S. Wasserkrug, "Simulation optimization using the cross-entropy method with optimal computing budget allocation," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 20, no. 1, p. 4, 2010.

[33] J. Hu, "Model-based stochastic search methods," in *Handbook of Simulation Optimization*, Springer, 2015, pp. 319–340.

[34] J. Hu, M. C. Fu, and S. I. Marcus, "A model reference adaptive search method for global optimization," *Operations Research*, vol. 55, no. 3, pp. 549–568, 2007.

[35] J. Hu, M. C. Fu, S. I. Marcus, *et al.*, "A model reference adaptive search method for stochastic global optimization," *Communications in Information & Systems*, vol. 8, no. 3, pp. 245–276, 2008.

[36] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 5, pp. 477–506, 2006.

[37] M. Kim, T. Hiroyasu, M. Miki, and S. Watanabe, "Spea2: improving the performance of the strength pareto evolutionary algorithm 2," in *Parallel problem solving from nature-PPSN VIII*, Springer, 2004, pp. 742–751.

[38] Y. Kochetov, E. Alekseeva, T. Levanova, and M. Loresh, "Large neighborhood local search for the p-median problem," *Yugoslav Journal of Operations Research ISSN: 0354-0243 EISSN: 2334-6043*, vol. 15, no. 1, 2005.

[39] Y. Kochetov and D. Ivanenko, "Computationally difficult instances for the uncapacitated facility location problem," in *Metaheuristics: Progress as real problem solvers*, Springer, 2005, pp. 351–367.

[40] C. Koulamas, S. Antony, and R Jaen, "A survey of simulated annealing applications to operations research problems," *Omega*, vol. 22, no. 1, pp. 41–56, 1994.

[41] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2002, vol. 2.

[42] R. Le Riche, A. Saouab, and J. Bréard, "Coupled compression rtm and composite layup optimization," *Composites Science and Technology*, vol. 63, no. 15, pp. 2277–2287, 2003.

[43] E. P. Lee Loo Hay and D. Goldsman, "Finding the non-dominated pareto set for multi-objective simulation models," *IIE Transactions*, vol. 42, no. 9, pp. 656–674, 2010.

[44] L. H. Lee, E. P. Chew, S. Teng, and Y. Chen, "Multi-objective simulation-based evolutionary algorithm for an aircraft spare parts allocation problem," *European Journal of Operational Research*, vol. 189, no. 2, pp. 476–491, 2008.

[45] L. H. Lee, E. P. Chew, S. Teng, and D. Goldsman, "Optimal computing budget allocation for multi-objective simulation models," in *Simulation Conference, 2004. Proceedings of the 2004 Winter*, IEEE, vol. 1, 2004.

[46] L. H. Lee, S. Teng, E. P. Chew, I. A. Karimi, K. W. Lye, P. Lendermann, Y. Chen, and C. H. Koh, "Application of multi-objective simulation-optimization techniques to inventory management problems," in *Proceedings of the 37th conference on Winter simulation*, Winter Simulation Conference, 2005, pp. 1684–1691.

[47] E. L. Lehmann and G. Casella, *Theory of point estimation*. Springer Science & Business Media, 2006.

[48] T. Levanova and M. Loresh, "Algorithms of ant system and simulated annealing for the p-median problem," *Automation and Remote Control*, vol. 65, no. 3, pp. 431–438, 2004.

[49] J.-H. Lin and J. S. Vitter, "Approximations with minimum packing constraint violation," *In Proceedings of the 24th Annuial ACM Symposium on Theory of computing*, pp. 771–782, 1992.

[50] H. O. Mete and Z. B. Zabinsky, "Multiobjective interacting particle algorithm for global optimization," *INFORMS Journal on Computing*, vol. 26, no. 3, pp. 500–513, 2014.

[51] J. M. Mulvey and H. P. Crowder, "Cluster analysis: an application of lagrangian relaxation," *Management Science*, vol. 25, no. 4, pp. 329–340, 1979.

[52] A. T. Murray and R. Church, "Applying simulated annealing to location-planning models," *Journal of Heuristics*, vol. 2, no. 1, pp. 31–53, 1996.

[53] A. Persson, H. Grimm, A. Ng, T. Lezama, J. Ekberg, S. Falk, and P. Stablum, "Simulation-based multi-objective optimization of a real-world scheduling problem," in *Simulation Conference, 2006. WSC 06. Proceedings of the Winter*, IEEE, 2006, pp. 1757–1764.

[54] M. Posta, J. A. Ferland, and P. Michelon, "An exact cooperative method for the uncapacitated facility location problem," *Mathematical Programming Computation*, pp. 1–33, 2014.

[55] S. N. Qasem and S. M. Shamsuddin, "Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis," *Applied Soft Computing*, vol. 11, no. 1, pp. 1427–1438, 2011.

[56] M. G. Resende and R. F. Werneck, "A hybrid heuristic for the p-median problem," *Journal of Heuristics*, vol. 10, no. 1, pp. 59–88, 2004.

[57] H. E. Romeijn and R. L. Smith, "Simulated annealing for constrained global optimization," *Journal of Global Optimization*, vol. 5, no. 2, pp. 101–126, 1994.

[58] K. Rosing, C. ReVelle, and D. Schilling, "A gamma heuristic for the p-median problem," *European Journal of Operational Research*, vol. 117, no. 3, pp. 522 –532, 1999.

[59] R. Y. Rubinstein, "Combinatorial optimization, cross-entropy, ants and rare events," in *Stochastic optimization: algorithms and applications*, Springer, 2001, pp. 303–363.

[60] R. Y. Rubinstein and D. P. Kroese, *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer, 2004.

[61] R. Solis-Oba, "Approximation algorithms for the k-median problem," in *Efficient Approximation and Online Algorithms*, ser. Lecture Notes in Computer Science, E. Bampis, K. Jansen, and C. Kenyon, Eds., vol. 3484, Springer Berlin Heidelberg, pp. 292–320.

[62] M. Srinivas and L. M. Patnaik, "Genetic algorithms: a survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994.

[63] A. Toffolo and A. Lazzaretto, "Evolutionary algorithms for multi-objective energetic and economic optimization in thermal system design," *Energy*, vol. 27, no. 6, pp. 549–567, 2002.

[64] A. Unveren and A. Acan, "Multi-objective optimization with cross entropy method: stochastic learning with clustered pareto fronts," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, IEEE, 2007, pp. 3065–3071.

[65] K. V. J. Vijay, "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation," *Journal of the Association for Computing Machinery*, vol. 48, pp. 274–296, 2001.

[66] N. E. Young, *K-medians, facility location, and the chernoff-wald bound*, 2000.

[67] Z. B. Zabinsky, "Annealing adaptive search," in *Stochastic Adaptive Search for Global Optimization*, Springer, 2003, pp. 83–104.

[68] ——, *Stochastic adaptive search for global optimization*. Springer Science & Business Media, 2013, vol. 72.

[69] E. Zhou and J. Hu, "Gradient-based Adaptive Stochastic Search for Non-differentiable Optimization," *IEEE Transactions on Automatic Control*, vol. 59, pp. 1818–1832, 2014.

[70] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithmsa comparative case study," in *International Conference on Parallel Problem Solving from Nature*, Springer, 1998, pp. 292–301.

[71] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *Evolutionary Computation, IEEE Transactions on*, vol. 7, no. 2, pp. 117–132, 2003.

[72] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo, "Model-based search for combinatorial optimization: a critical survey," *Annals of Operations Research*, vol. 131, no. 1-4, pp. 373–395, 2004.