STOCHASTIC SEARCH METHODOLOGIES FOR MULTI-OBJECTIVE SIMULATION OPTIMIZATION

LI HAOBIN

NATIONAL UNIVERSITY OF SINGAPORE

2013

STOCHASTIC SEARCH METHODOLOGIES FOR MULTI-OBJECTIVE SIMULATION OPTIMIZATION

LI HAOBIN

(*Ph.D., NUS*)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL & SYSTEMS ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2013

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

[illegible]

Li Haobin

19 August 19, 2013

Name	:	Li Haobin
Degree	:	Doctor of Philosophy
Supervisor(s)	:	Lee Loo Hay, Chew Ek Peng
Department	:	Department of Industrial & Systems Engineering
Thesis Title	:	Stochastic Search Methodologies for Multi-Objective Simulation Optimization

Abstract

For multi-objective simulation optimization problem (MSOP), this thesis proposed a generic framework for designing various types of multi-objective (MO) search algorithms. Based on the framework, two specific algorithms are designed. First is an advanced stochastic search algorithm multi-objective convergent optimization via most-promising-area stochastic search (MO-COMPASS) that is developed with solid theoretical foundations and proof showing convergence to the local optimum. Another is gradient-oriented polar random search (GO-POLARS) that is designed to strengthen the search efficiency, especially to make it suitable for continuous problems and easy to control exploration of the search space. With the dominated hyper volume concept and a unified gradient derived, we are able to incorporate gradient-based techniques such as GO-POLARS, into the MO search framework.

Keywords : stochastic search, simulation, multi-objective, optimization, gradient, hyper volume, Pareto set, convergent, COMPASS, MO-COMPASS, polar coordinate, GO-POLARS

Acknowledgment

I would like to thank my supervisor A/Professor Loo Hay Lee and A/Professor Ek Peng Chew for their guidance and patience to teach during the past eight years since my undergraduate study.

I also want to thank Professor L. Jeff Hong from Department of Industrial Engineering and Logistics Management, The Hong Kong University of Science and Technology, for his assistance in improving the mathematical proof.

Support from Dr. Peter Lendermann and numerous colleagues from D-SIMLAB Pte. Ltd. for conducting experiments in industrial application is also gratefully acknowledged.

Finally, but most importantly, I would like to thank my wife Xiaoxuan and our little Jordan for their companionship and support during my graduate program.

January 22, 2014

Contents

List of Figures

List of Tables

List of Symbols

a, b	step size in a point search
$g(\cdot), \vec{g}(\cdot)$	true objective value, vector for a given decision
$\bar{G}(\cdot), \bar{\vec{G}}(\cdot)$	sample average of objective value, vector for a given decision
H	number of objectives
$J_p, \|J_p\|$	Jacobian matrix and determinant in p dimension
k	indicator of search iteration
l	indicator of dimension in objective space
m	population search batch size
n	largest value on a dimension in decision space
$N(\cdot)$	number of simulation budget assigned to a given decision
p	dimension in the decision space
r	range, distance, or the radius parameter in polar coordinate
$\vec{Y}(\cdot)$	approximated gradient at given decision point
$\vec{x}, \vec{y}, \vec{z}$	vectors in decision space (in Cartesian coordinate)
$\hat{\vec{x}}^*$	observed best solution
$\mathcal{B}$	constrained subset of solution space
$\mathcal{C}$	most-promising-area
$\mathcal{D}(\cdot)$	convex partial-MPA around the given Pareto solution

$\mathcal{H}$	dominated hyper-volume
$\mathcal{N}(\cdot)$	neighborhood of a given point or set
$\mathcal{P}$	local Pareto set
$\mathcal{V}$	set of visited solutions
ϵ	error term
Π	true Pareto set
$\hat{\Pi}$	observed Pareto set
Θ	decision space
θ	angular parameter in polar coordinate
ω	random factor
σ	standard deviation, parameter controlling sampling density
φ	concentrating function for oriented polar distribution
$\vec{\nabla}(\cdot)$	gradient at given decision point
$\mathrm{CS}(A, B)$	event of correct selection of A as the Pareto set on B
$\mathrm{FS}(A, B)$	event of false selection of A as the Pareto set on B
$\mathrm{U}^p_{\mathrm{polar}}$	p-dimension polar uniform distribution
$\mathrm{O}^p_{\mathrm{polar}}$	p-dimension oriented polar distribution

List of Abbreviations

COMPASS convergent optimization via most-promising-area stochastic search

CS coordinate sampling

DHV dominated hyper-volume

DOvS discrete optimization via simulation

FDSA finite-difference stochastic approximation method

GD generational distance

GO-POLARS gradient-oriented polar random search

GOCS gradient-oriented coordinate sampling

GPS GO-POLARS sampling

HSO hyper-volume by slicing objectives

IGD inverted generational distance

LPS local Pareto set

MDOvS	multi-objective discrete optimization via simulation
MO	multi-objective
MO-COMPASS	multi-objective convergent optimization via most-promising-area stochastic search
MOCBA	multi-objective optimal computing budget allocation
MOEA	multi-objective evolutionary algorithm
MPA	most-promising-area
MSOP	multi-objective simulation optimization problem
NSGA	non-dominated sorting genetic algorithm
OCBA	optimal computing budget allocation
RMD	revised Mix-D
RPG	random perturbation of the gradient
SA	stochastic approximation
SAN	simulated annealing
SAR	simulated allocation rule
SD	steepest descent
SOP	simulation optimization problem
SPSA	simultaneous perturbation stochastic approximation
WPS	weighed pivot selection

Chapter 1

Introduction

Recently, simulation optimization problem (SOP) has attracted more research interests, due to its broad application to solve complicated real-life problems. Many industrial systems, such as supply chain, manufacturing, and financial management, are often difficult to be formulated in mathematical close form because of their complex business models. Simulation becomes the best way to evaluate such kind of systems and help to identify the optimal configurations.

Beside the complex business models, quite often for industrial applications there is also a large number of decisions to be made. For example, in a supply chain system, normally the inventory levels or ordering quantities for different products at every warehouse or retail store are the decisions need to be made. In such cases, the problem dimension is associated to the amount of real-life entities that concern us, which is usually a very big number.

From mathematical point of view, it implies that the optimization problem is configured in a high-dimensional search space. The number of candidate solutions increases exponentially as the dimension becomes larger. So, in order to obtain the optimal solutions, it costs an extreme amount of computational effort and time to sample and evaluate different possibilities. Hence, instead of exhaustive search which is usually impractical, more efficient search algorithms are desired so as to provide in-time decisions for those large-scale industrial problems.

Many research works address this issue from different perspectives. Some works focus on the stochastic search that explores various parts of a solution space specified by a neighborhood structure, e.g., genetic algorithms, the simulated annealing (SAN) algorithm (Kirkpatrick et al., 1983), the nested partitions method (Shi and Olafsson, 2000), and convergent optimization via most-promising-area stochastic search (COMPASS) (Hong and Nelson, 2006). Meanwhile, the others adopt local information, such as gradient or second order derivatives. The example can be the steepest descent (SD) approach (Arfken, 1985), the finite-difference stochastic approximation method (FDSA) (Blum, 1954b; Kiefer and Wolfowitz, 1952), and the simultaneous perturbation stochastic approximation (SPSA) (Spall, 1999, 2003).

Although there are various state-of-art algorithms proposed to solve the the complex SOPs, the challenges could become even more as nowadays systems often have multiple performance measurements to be observed. We classify those problems as MSOP.

Compared with the single-objective case, in an MSOP we are not only concerned about the solution space which is in high dimension, but also the multi-dimensional objective space. The target of the optimization problem is no longer finding a single point that has the optimal objective value, but a bundle of solutions that are representative for the best value of each contradictory objective. Thus, we could not simply adopt the procedure for solving SOP, as many new aspects need to be taken into consideration for design a MO search algorithm.

A typical MSOP in aerospace industry is addressed by D-SIMSPAIR$^{\text{TM}}$, a simulation-based planning and optimization system developed by D-SIMLAB Technologies Pte. Ltd. (Lendermann et al., 2010). The main purpose of the system is to enable OEM companies or their designated service providers to determine the optimal inventory configuration that satisfies targeted service level at minimum cost, by evaluating the performance of different inventory configurations via simulation. Obviously, the inventory levels of all part at all stock locations are the integer decision variables we can manipulate, and at least two performance criteria, namely the achieved service level (or the probability of achieving target level) and the realized cost are the objectives we are interested in. More objectives are to be involved if multiple airlines are served and the service level needs to be considered independently.

For instance, for big component support service providers, it is common that a flight network contains more than five stock locations for a particular part number, and each part number can have an inventory level up to 30 at a location. Clearly, if a full-enumeration scheme were adopted, we would need 30^5 trials to visit all possible solutions, which implies that if each visit takes 1 second to run the simulation, the optimization process will complete only after 281 days. More severe situation arise when we take into consideration more stock locations, for example, with 6 stock locations, 30^6 visits will takes 23 years. D-SIMSPAIR$^{\text{TM}}$ has been adopting heuristics-based approach to reduce this complexity, hence speeding-up the optimization process. But to bring the capability (scalability in problem size) of D-SIMSPAIR$^{\text{TM}}$ to a new level, a revision to the optimization algorithm is required.

By analyzing the industrial problem, we can see that the problem dimension is quite critical in solving an MSOP, especially when the search efficiency is concerned. Moreover, it also shows why the multi-objective optimization come into the picture, which further increases the problem complexity. Note that, the D-SIMSPAIR$^{\text{TM}}$ is only one example showing the strong call for the industrial needs. There could be many more other cases in real-life application as nowadays in a complex system people's interests become more diverse.

Usually, people would like to convert multi-objective problems into single-objective by assigning a weightage to each measurement. For example, the multiple attribute utility method (MAU) (Butler et al., 2001). But when there is no consensus about the weightage or in the situation where measurements are not compromising, e.g. in D-SIMSPAIR$^{\text{TM}}$ no airline would like to sacrifice its own interest for fulfilling the others, it makes more sense to provide the whole Pareto set, i.e., all non-dominated solutions (Lee et al., 2006, 2008, 2010) for the decision makers or a higher-level optimization problem to make the final decision based on more detailed information.

Thus, in this thesis we aim to provide stochastic search methodologies for solving the MSOP in term of providing a set of solutions that well represent the whole Pareto set with required accuracy and efficiency.

In order to achieve the goal, rather than relying on a special design of algorithm, we firstly introduce a stochastic search framework for MSOP and show that it is indeed followed by other MO search algorithms, e.g., non-dominated sorting genetic algorithm (NSGA)-I/II (Deb et al., 2002). So that it provides a guideline for designing specific algorithms with concerns and focus according to various industrial applications. Alternatively, for an existing algorithm we may based on the framework identify where can be potentially improved and redesign it for better performance.

Referring to different MO search algorithms developed in literature and the later chapters of this thesis, we describe the general framework as in Algortihm 1.1.

Algorithm 1.1: Framework of stochastic search for MSOP

```
1 Initialize Pareto set ;
2 while not terminating do
3     Select "pivot solutions" from the interim dominating structure ;
4     Construct Neighborhood ;
5     Sampling from Neighborhood ;
6     Compare the visited solutions and update Pareto set ;
```

According to the framework, first of all, one or a few solutions are chosen to initialize the search. The initial Pareto set can be suggested by domain experts or generated quickly from heuristics.

Then, at each iteration, one or two pivot solutions are randomly selected according to the current dominating structure. Normally, a better quality solution in term of higher level of dominance will have better chance to be selected. As we believe that, the goodness of the solution can be inherited when its neighbors are sampled. Without much consideration, a conventional method to select the pivot solution in the interim Pareto set following uniformity. Meaning that, every interim Pareto solution is treated equally, so as to have the same probability to be chosen as the pivot to initiate the next iteration.

For example, the MO-COMPASS introduced later in Section 3.3.1 and 3.3.2 adopts the uniform selection. However, if we try to fit NSGA-II into the framework, it can be realized that the probabilities of selection are not even but based on a fitness score according to the layer of dominance and crowding distance that indicates the solution density in objective space. Obviously, if we are able to design other pivot selection scheme with valid reasons, a new search algorithm could be developed.

The next important step would be constructing the neighborhood for pivot solutions, within which new candidate solutions can be sampled.

By constructing an effective neighborhood, we focus on the search in a more promising area which has higher chance to produce non-dominated solutions. This technique is addressed by MO-COMPASS that is introduced in Chapter 3. Another simple neighborhood structure could be the set of all immediate neighbors that have unit Euclidean distance to the current observed Pareto set. Or, in NSGA-II the structure can be concluded as

$$\bigcup_{\vec{z}_1,\vec{z}_2\in\hat{\Pi}_k} \{F_{\text{Mut.}} \circ F_{\text{Cro.}} (\vec{z}_1, \vec{z}_2, \omega)\}.$$

In this presentation, we treat the crossover and mutation procedure as a special designed composite function, i.e., $F_{\text{Mut.}} \circ F_{\text{Cro.}}$, that can be applied on two parent solutions selected from the intermediate Pareto set, with a random factor ω occurring. According to it, all possible outputs are considered as in the neighborhood. Other than the NSGA-II, in another scenario where gradients are known or could be estimated, we may apply a typical gradient search procedure to reach its neighborhood, provided that there is a weighing vector balancing gradients from contradictory objectives.

Sometimes, an adequate solution-sampling scheme is also a key factor that affects the search efficiency for new Pareto solutions. In MO-COMPASS, revised Mix-D (RMD)(Hong and Nelson, 2006) and coordinate sampling (CS) (Hong et al., 2010) are the popular sampling schemes, due to their simplicity and convenient implementation. Other than that, we could also develop different sampling scheme for it. In NSGA-II, the sampling scheme can be seen as the way to play with the random factor ω, e.g., tuning the rate of mutation. However, when a typical gradient search is performed where there is no randomness taking part, the sampling scheme becomes trivial unless we are able to tune the step-size and search direction.

As we can see from the above examples, the sampling scheme is highly coupled with its neighboring structure. Some kinds of neighborhood are suitable for various types of sampling, while the rest only have limited ways of sampling new solutions.

Anyhow, in most of the cases, the basic idea for designing a superior neighborhood structure as well as its sampling scheme is to identify good solutions fast without loosing opportunity to visit the potential better ones.

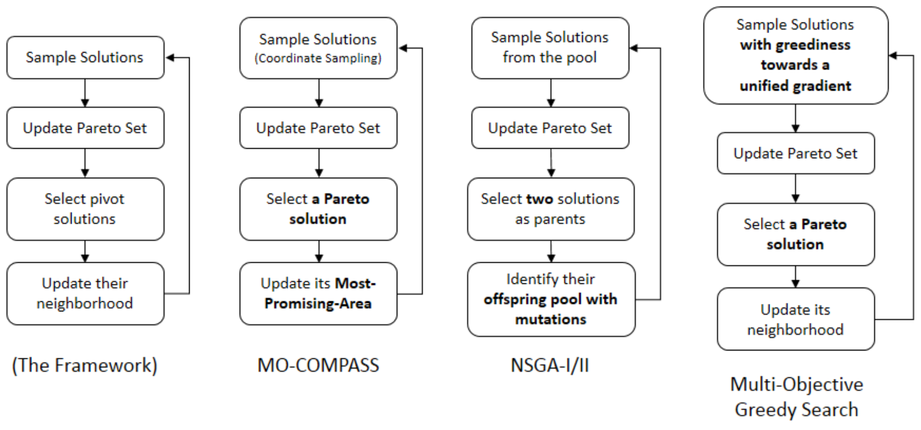

Sample Solutions
Update Pareto Set
Select pivot solutions
Update their neighborhood
(The Framework)
Sample Solutions (Coordinate Sampling)
Update Pareto Set
Select a Pareto solution
Update its Most-Promising-Area
MO-COMPASS
Sample Solutions from the pool
Update Pareto Set
Select two solutions as parents
Identify their offspring pool with mutations
NSGA-I/II
Sample Solutions with greediness towards a unified gradient
Update Pareto Set
Select a Pareto solution
Update its neighborhood
Multi-Objective Greedy Search

Figure 1.1: The MO search framework, examples and new design.

In order to do so, we have to utilize acquired information in a smart way while maintaining sufficient exploration in the solution space.

The iterative process continues until termination conditions are met or we run out of simulation budget. We can see from the framework in Algorithm 1.1 that three steps are critical, namely, 3, 4 and 5, as Step 6 is no more than a straightforward calculation. In Section 5.3 we will elaborate them in details.

The illustration of the framework and typical example of its varieties are shown in Figure 1.1. With the framework, new search algorithm will be designed by modifying one of the steps in the framework. For instance, if we apply greediness in sampling new solution providing that we can find a unified gradient among multiple objectives (as in Section 5.2), we can simply propose the multi-objective greedy search algorithm.

In this thesis, based on the framework mentioned above, two innovative MO search algorithms are developed.

With the ideas of pure stochastic local search proposed in COMPASS (Hong and Nelson, 2006), for the first algorithm we develop a new MO-COMPASS for solving in MO circumstance that is proven to have convergent property.

Secondly, for utilizing the gradient information, we developed GO-POLARS based on a brand new hyper polar coordinate system and corresponding random distributions, in which we can perturb the search direction from the gradient in a well-controlled manner.

To apply GO-POLARS in MO circumstance, we extend the concept of a singular indicator for measuring a Pareto set, i.e., the dominated hyper-volume (DHV) (Bradstreet et al., 2008; Nebro et al., 2008; Zitzler et al., 2003), so as to invent a unified gradient for MSOPs. We note that the methodology developed can also be applied to other gradient-based techniques other than the GO-POLARS.

Following are the main contributions we have achieved in this thesis:

- The local Pareto set (LPS) is defined for the multi-objective discrete optimization via simulation (MDOvS) which fills the gap of theoretical work in this research field. It serves as a fundamental knowledge for deriving the convergence property.

- The MO-COMPASS algorithm is developed and it is shown by numerical experiments that the algorithm is superior in solving benchmark MDOvS problems compared with existing algorithms. It improves the search efficiency for solving MSOPs.

- The MO-COMPASS algorithm is rigourously proven to have strong convergent property to LPS. It is the first algorithm that is claimed to have such property.

- The hyper polar coordinate and corresponding random distributions are rigourously defined and applied for solving optimization problem, in which the perturbation of search direction can be easily controlled. It is the first time polar coordinate is used in designing optimization search algorithm.

- The GO-POLARS is mathematically proven to have local convergent property as stochastic approximation (SA).

- The DHV concept and hyper-volume by slicing objectives (HSO) methods are extended so that they can be applied in identifying a unified gradient for an MSOP, which is new in the area of multi-objective optimization.

In subsequent parts of this thesis, we will firstly have a literature review of various type of search algorithms in Chapter 2, discussing about the search methodologies that have been developed by previous research works especially for the simulation optimization.

For development of specific algorithms, the convergent method MO-COMPASS is to be discussed in Chapter 3, and the gradient oriented random search GO-POLARS is described in Chapter 4 and 5.

Noted that, the GO-POLARS is a brand new concept that even for a single objective case. So in Chapter 4 we first discuss it for single-objective problems, and then extend it into multi-objective circumstance in Chapter 5.

Chapter 2

Literature Review

In literature, there are many optimization problems aiming to analyze a complex system. In generic form, the problems can be described as

$$\min_{\vec{x}} \vec{g}\,(\vec{x}), \text{ for } \vec{x} \in \Theta.$$

When $\vec{g}\,(\cdot)$ has one dimension, we classify it as single-objective problem; otherwise, it is multi-objective problem. From the other prospective, according to the solution space Θ, the problems can be classified as discrete and continuous.

Quite often the mathematical programming methods cannot be applied, either because the loss function is too complicated to be analyzed or the problem is formulated by a simulation model where the close form of loss function does not occur. In such cases, adaptive search algorithms becomes better choice as only local information is required which can be easily obtained from loss functions or evaluated by simulation models.

2.1 For Single-Objective Problems

We first look at the single-objective optimization problems, where there is plenty of search algorithms that have been developed, but mainly are included in two categories.

One category of well-known search algorithms is driven by gradient information. The oldest method is the SD approach (Debye, 1909) which assumes that the gradient is known at each search iterate, so the search always moves towards its opposite direction with a step-size proportional to its magnitude and a given gain sequence. When the ideal situation occurs where the gradient can be directly measure, the steepest descent shows a very good performance as the search iterate converges to the optimum fastest comparing with the others.

However, in the case where the gradient cannot be directly measured, a FDSA should be used for estimating the gradient information (Blum, 1954b; Kiefer and Wolfowitz, 1952). The basic idea of FDSA is to perturb the decision variable with an infinitesimal distance at every coordinate in both positive and negative direction and observe its corresponding change in the objective value. A simple calculation shows that for a p-dimension decision space, it takes $2 \cdot p$ evaluations to approximate a gradient, which could be expensive if the evaluation is done by simulation which normally takes time and there is many gradients need to be estimated.

As FDSA is costly especially when dimension is high, Spall (1999, 2003) propose the SPSA that increases the estimation efficiency. Which is different from FDSA, in SPSA the perturbation to the decision vector is conducted simultaneously at all dimensions and in each dimension either take the positive or the negative direction. And the complement of the perturbation is also taken. Then, the gradient is approximated as the rate of objective change to the perturbation value at each dimension. Obviously, the evaluation time spent on SPSA becomes independent to the dimension, to be more specific for each gradient only 2 evaluations are required.

Although it has been shown that under certain conditions, the SPSA converges to a local optimal point almost surely (Spall, 2003), the global convergence can only be ensured under strict conditions by Maryak and Chin (2008) as in a relax setting the greedy use of gradient information sacrifices the exploration on the whole solution space. It can be argued that, in FDSA and SPSA the gradient is approximated with certain noise, which unintentionally increases the variety of the search directions. However, since the noise cannot be controlled explicitly, it is difficult to balance search exploration at a desired level.

Another category, often referred as metaheuristics local search, mainly depend on stochastic sampling within carefully designed neighborhood structures. For example, the SAN algorithm (Kirkpatrick et al., 1983), Tabu search (Glover, 1990), genetic algorithms, the nested partitions method (Shi and Olafsson, 2000), and COMPASS (Hong and Nelson, 2006).

Compared to gradient-driven algorithms, the neighborhood structure often ensures a better exploration on the search space. For example, in the developed algorithms, the SAN applies a temperature parameter setting as a threshold to randomly decide whether a subordinate solution should be accepted so as to continue the next iteration of search; while the genetic algorithms randomly select parent solutions to perform crossover and mutation is performed in addition in order to make solutions diverse so as to further enhance the exploration in the search space.

But on the other hand, there is certainly some room for improvement in term of search efficiency, as the gradient information or other domain related knowledge which can probably be measured or approximated is not utilized at all.

It is obvious that if the search direction in a gradient-based algorithm can be randomized with desired variation, or the stochastic sampling in a metaheuristic can be oriented by the gradient information, we can design a new search algorithm that believes to have better performance than both. Although Pogu and Souza De Cursi (1994) proposed a method for random perturbation of the gradient (RPG), we noticed that the perturbation within a region surrounding the targeted point cannot control the search direction explicitly. For example, when the step-size is sufficiently large or small, the same amount of perturbation may incur much difference in the search direction.

Thus, there is a lack of research work on the stochastic search that is enabled with explicit control on the search direction so as to make it diverse but towards the gradient direction with desired concentration.

2.2 For Multi-Objective Problems

However, there are limited works on the search algorithms for multi-objective problems. For example, Czyzak and Jaskiewicz (1998) extends the simulated annealing method into multi-objective circumstance. According to Fleischer (2002); Marler and Arora (2004), most of the rest are the multi-objective evolutionary algorithm (MOEA)s, e.g., the Pareto-archived evolution strategy (PAES) (Knowles and Corne, 1999), the strength-Pareto EA (SPEA) (Zitzler, 1999), and the NSGA and its variation (Deb et al., 2002; Srinivas and Deb, 1995). The main reason is because genetic algorithm is able to keep a population of diverse solutions so that it is capable to contain a Pareto set that covers the full objective range.

In the typical NSGA, all solutions in a population are divided into layers according to their dominance relationship. For example, the overall non-dominated solutions are concluded into the first layer, and within the remaining solutions the non-dominated set are treated as the second layer. The rest can be done in the same manner. All solutions are sorted either by layers, or within a layer according to their neighboring distance in the descending order so as to ensure a well spread solution set. And thereby, parent solutions will be selected according to the fitness score which is proportional to their rank in the population. By crossing over and mutation, new offspring will be generated to form the next generation.

Comparing with the original NSGA which is normally referred as NSGA-I, the improved version NSGA-II adopts the Crowding-distance to rank the solutions in a dominance layer so as to decrease the time complexity and has elitism enabled so that it become more efficient in searching for the optimal Pareto set. Thus, NSGA-II is the most popular algorithm in the field.

While most of the works for MOEAs are focusing on reduction of computational time complexity, there are seldom multi-objective search algorithms that utilized local information and shown to have a convergent property that is rigourously proven. Hence, it will be good if we can propose a search algorithm with proven convergent property and shown to be superior than the current MOEA family.

Meanwhile, we are fortunate as many indicators for Pareto set comparison are available in literature. So, in designing new search algorithms, we can use the indicators to measure their fitness into a multi-objective benchmark problem. The indicators are applied in different perspective (Radziukyniene and Zilinskas, 2008).

For example, the generational distance (GD) suggested by Nebro et al. (2008) shows how far is the approximated Pareto set from the true one. It is calculated by taking the distances of all solutions in the approximated Pareto set to the nearest solution in the true set, and then get the averaged value. Whereas, the inverted generational distance (IGD) (Veldhuizen and Lamont, 1998) indicates how far the true Pareto set is from the approximation, that is calculated in a reverse manner. Although similar, these two indicators are not identical, because in an extreme case where the approximation is exactly a proper subset of the true Pareto set, GD equals to 0 but IGD could be large indicating that the true Pareto set is not completely covered by the approximation.

Other than the distance metrics, Spread indicator (Nebro et al., 2008) is a diversity metric that measures the extent of spread achieved among obtained solutions, and the running performance metrics (Deb and Jain, 2002; Zeng, 2010) is used to dynamically assess the diversity performance of the generated solution sets.

One of the most popular indices is the DHV (Bradstreet et al., 2008; Nebro et al., 2008; Zitzler et al., 2003). There are many research works on how the DHV value can be evaluated in the most efficient way. While et al. (2006) propose a method called HSO that adopts dimension-sweep approach developed by Preparata and Shamos (1985). The HSO is a recursive method, which divides a p dimension problem into several $p-1$ dimension sub-problems which can be further divided until $p=1$ where hyper-volume is no more than the length of a line segment. It is proven that the algorithm runs in time complexity of $O\left(n^{p-1}\right)$ for $p>2$. Upon it, Fonseca et al. (2006) presents an improved algorithm with pruning techniques, which achieves $O\left(n^{p-2}\log n\right)$ time and linear space complexity in the worst case. In our research, HSO is adopted as the way for DHV calculation due to its simplicity in implementation, although it is not the fastest algorithm.

As there is a lack of multi-objective algorithms other than MOEAs, this thesis is aiming to fill in the research gaps, not only in the generic framework, but also a specific search algorithms that has rigorous proof for its convergent property, together with a algorithm that utilizes gradient information into a stochastic search so as to gain an explicit control on balancing solution quality and search efficiency. The methods developed are also inspiring for the development of new MO search algorithms that fitting the different industrial requirement.

Chapter 3

The Multi-Objective Convergent Optimization via Most-Promising-Area Stochastic Search

In this chapter, we simplify the MSOP by assuming the solution space to be discrete. It is a reasonable assumption, as this category of problems has gained application in various industries such as manufacturing, logistics and services, and is attracting more research interests as well. A simple reason is that the performance of these systems largely depends on integer settings like staffing or inventory level, number of equipment, products or customers. Besides, we note that in some circumstances continuous decision variables should also be considered in a discrete sense, for instance, the manufacturing time is usually calculated in number of shifts.

As mentioned in Chapter 1, the Convergent Optimization via Most-Promising-Area Stochastic Search (COMPASS) (Hong and Nelson, 2006) was originally proposed for efficiently solving single-objective SOP, or more precisely the discrete optimization via simulation (DOvS). With this method, solutions are sampled stochastically within the most-promising-area, in which all solutions have shorter Euclidian distance to the current optima than the distance to any current non-optima. The solutions are to be evaluated according to certain simulated allocation rule (SAR) and used to construct the next most-promising-area. It has been proven that the searching converges to the local optima regardless of the searching space being constrained.

Since COMPASS works well for single-objective DOvS, we follow its idea together with the fundamentals of solving MO problems. We propose a MO-COMPASS algorithm that adapts to multi-objective circumstances and illustrate numerical examples to show its ability in achieving the desired efficiency.

3.1 Review on COMPASS

For a fully constrained problem, consider the searching space is Θ and for each $\vec{x} \in \Theta$ the expected single performance measurement is $g(\vec{x})$, which is estimated by sample average $\bar{G}(\vec{x})$ from simulation results. Without any preliminary knowledge, the most-promising-area $\mathcal{C}$ is initially set to Θ.

Let $\mathcal{V}$ be the set of all visited solutions. Every iteration, stochastically select m solutions from $\mathcal{C}$ to be included in $\mathcal{V}$ and apply SAR on it to find the solution with the minimum sample average, and use it to refine the most-promising-area, i.e.,

$$\hat{\vec{x}}^* = \arg\min_{\vec{x}\in\mathcal{V}} \bar{G}(\vec{x}),$$

$$\mathcal{C} = \{\vec{x} \in \Theta \,|\, \forall \vec{y} \in \Theta, \vec{y} \neq \hat{\vec{x}}^* \Rightarrow \|\vec{x} - \hat{\vec{x}}^*\| \leq \|\vec{x} - \vec{y}\|\}.$$

The process can be illustrated by Figure 3.1. From the figure we can see that, in every iteration the most-promising-area (MPA) is constructed in the decision space around the best observed point, while the boundaries the MPA are formed at middle lines between the best and each of the inferior points. Then, in the next iteration, new samples are only generated in the MPA. In such a manner we can ensure that all the new samples have shortest distance to the best known point among all points which have been evaluated.

We should take note that the selection of SAR affects the efficiency of search algorithms, as the search algorithm determines which solutions to visit while the SAR decides how much simulation budget to spend on each visit. According to Hong and Nelson (2006), a valid SAR for COMPASS should satisfy two conditions: (1) the simulation budget allocated to newly visited solution should not be zero; (2) as total budget approaches infinity, the budget allocation to each visited solution should approach infinity as well.

In Xu et al. (2010), it is stated that fixed schedules or optimal computing budget allocation (OCBA) ideas can be adopted for COMPASS. Moreover, some SARs are integrated with search algorithm. For example, He et al.

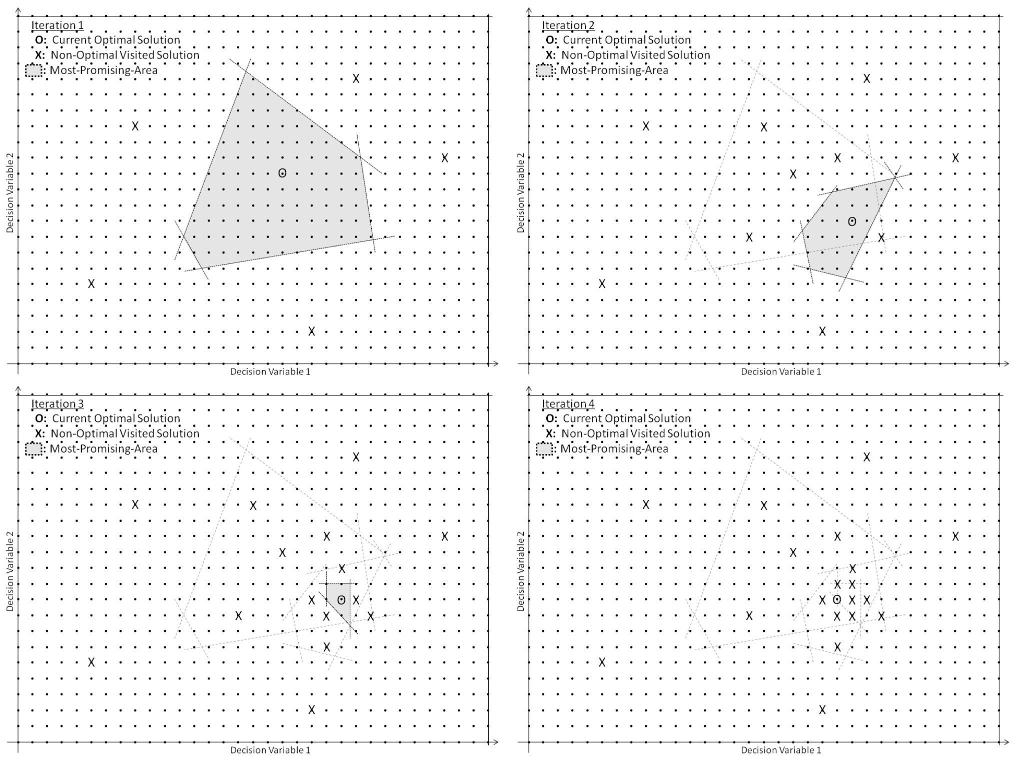


Figure 3.1: Illustration for single-objective COMPASS

(2010) propose an integrated cross-entropy method with OCBA.

It is proven that repeating the process will lead the estimate to converge to the local optima (Hong and Nelson, 2007). The searching process can be terminated either when simulation budget is exhausted or all neighboring solutions of are visited.

For a partially or unconstrained problem, searching starts from a constrained subset $\mathcal{B}$ and follows the similar steps as for fully constrained problems, but the boundaries of the subset are revised accordingly for each iteration so as to reserve certain buffer in each direction from any visited solution whenever it is available. A stronger condition for SAR is required for the case, so as to ensure that the estimation error converges to 0 with sufficiently high rate to compensate the negative effect brought by the increasing size of candidate solutions.

3.2 Fundamentals

3.2.1 Pareto Optimality

To compare two multi-objective solutions, besides looking at the weighted sum of various measurements as a single compromising solution (Butler et al., 2001; Swisher et al., 2003), an alternative approach is to compare measurement for individual objective (Lee et al., 2006, 2010). A solution is claimed to dominate another if and only if all its objective measurements are superior to the others. Thus, in optimization, instead of looking for a single best solution among Θ, we are more interested in finding a set of solutions that are not dominated by the others. We claimed those solutions as best among Θ and the set is referred as the Pareto set

$$\Pi \equiv \{\vec{x} \in \Theta \,|\, \nexists \vec{y} \in \Theta,\, \vec{g}(\vec{y}) \prec \vec{g}(\vec{x})\}.$$

where $\vec{g}(\vec{y}) \prec \vec{g}(\vec{x})$ if and only if $\forall l \in \{1, ..., H\}, g^{(l)}(\vec{y}) \leq g^{(l)}(\vec{x})$ and $\exists l \in \{1, ..., H\}, g^{(l)}(\vec{y}) < g^{(l)}(\vec{x})$.

Similar to single-objective problems, it is difficult to obtain a full Pareto set especially in a huge solution space, since most of the time we are not able to deny the possibility that the current Pareto solution being dominated by some unvisited solutions. Whereas, according to following definition, we can claim a local Pareto optimality without visiting all feasible solutions.

The definition of local Pareto set for a continuous problem has been proposed by Deb (1999), according to which, we can define local Pareto optimality for a multi-objective discrete problem by reconstructing the neighborhood.

Definition 3.1. *For a discrete problem defined on Θ, a solution set $\mathcal{P} \subseteq \Theta$ is claimed as* LPS *if and only if it is a Pareto set on $\mathcal{N}(\mathcal{P}) \equiv \{\vec{z} \in \Theta \,|\, \exists \vec{x} \in \mathcal{P},\, \|\vec{x} - \vec{z}\| \leq 1\}$.*

Or in mathematical form, $\mathcal{P} \subset \Theta$ such that $\forall \vec{x} \in \mathcal{P}$, $\nexists \vec{y} \in \mathcal{N}(\mathcal{P})$ satisfying $\vec{g}(\vec{y}) \prec \vec{g}(\vec{x})$. Note that $\mathcal{N}(\mathcal{P})$ is a neighborhood of $\mathcal{P}$ that contains $\mathcal{P}$ itself.

An illustration is shown by the 4th part of Figure 3.2 (i.e., Iteration K). As all the circled solutions are incomparable with each other, and all their un-circled neighbors are visited and observed to be dominated (simply because the circled ones are interim Pareto solutions among all visited). Then, the set of circled solutions can be claimed as an LPS.

Aiming to identity a local Pareto set for a MDOvS with high efficiency, we propose the MO-COMPASS algorithm in Section 3.3.1 and 3.3.2.

3.2.2 Probability of Correct Selection

Since in an MDOvS, we can only observe the sample averages $\bar{\vec{G}}(\vec{x})$ from simulation results, it is always possible that the Pareto set we selected is not accurate in term of the true performance measure $\vec{g}(\vec{x})$. To measure the selection quality of the Pareto set, a commonly adopted approach is the probability of correct selection, i.e., $\Pr\left\{\mathrm{CS}(\hat{\Pi}, \Theta)\right\}$, defined as the probability of the event that the observed $\hat{\Pi}$ is truly the Pareto set among solution set Θ.

In literature, Lee et al. (2010) developed an upper bound for $\Pr\left\{\mathrm{CS}(\hat{\Pi}, \Theta)\right\}$ based on the bounding of Type I and II error, which is useful when the candidate solution set Θ is given as deterministic.

However, when a stochastic search algorithm for DOvS is to be designed, we need to consider a subset $\mathcal{V} \subset \Theta$ that is randomly selected, on which a Pareto set $\hat{\Pi}_{\mathcal{V}}$ is to be observed based on the sample averages. In such a case, we cannot directly apply Lee et al. (2010)'s method for $\Pr\left\{\mathrm{CS}(\hat{\Pi}_{\mathcal{V}}, \mathcal{V})\right\}$ concerning that $\mathcal{V}$ is stochastic, unless with conditioning on $\mathcal{V} = A$ for all $A \subset \Theta$, which will apparently increase the complexity of the problem since $|\{A \subset \Theta\}| = |\Theta| \cdot 2^{|\Theta|-1}$.

In this paper, we adopt another way of bounding for the probability of CS($\hat{\Pi}_{\mathcal{V}}, \mathcal{V}$), or its complementary event, i.e., false selection denoted as FS($\hat{\Pi}_{\mathcal{V}}, \mathcal{V}$), by considering the fact that false selection occurs only when some pairwise comparison is wrong due to inaccurate evaluation, i.e., $\bar{G}^{(l)}(\vec{x}) \leq \bar{G}^{(l)}(\vec{y})$ for some $\vec{x}, \vec{y} \in \mathcal{V}$ is observed to be true, but in fact $g^{(l)}(\vec{x}) \geq g^{(l)}(\vec{y})$. This idea will be applied in the proof for Theorem 3.1 and 3.2.

3.3 The Multi-Objective COMPASS

3.3.1 For Fully Constrained MDOvS

We first consider an MDOvS where H objectives, i.e., $g^{(l)}(\vec{x})$ for $l \in \{1, \ldots, H\}$, are to be minimized with feasible solution space Θ which is fully constrained or bounded. Note that in this paper, H is assumed to be finite as well.

While structuring the MO-COMPASS algorithm, some basic principle of single-objective COMPASS is carried on, meaning that the most-promising-area is constructed according to Euclidean distances to both "good" and "bad" solutions.

However, instead of a single current best solution, in multi-objective problem, "good" solutions refer to those contained in intermediate Pareto set. Since they are incomparable among each other, the most-promising-areas are constructed for each Pareto solution and are treated indifferently in terms of the chances of sampling new solutions. To be more specific, assuming $\hat{\Pi}_k$ is the observed Pareto set at iteration k and $\mathcal{V}_k$ is the set of all visited solutions, the most-promising-area $\mathcal{C}_k$ is defined as:

$$\mathcal{C}_k \equiv \bigcup_{\vec{z} \in \hat{\Pi}_k} \Big\{ \vec{x} \in \Theta \mid \forall \vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq \|\vec{x} - \vec{y}\| \Big\}, \forall k \geq 1. \tag{3.1}$$

Then the algorithm is described as follows:

Algorithm 3.1: MO-COMPASS for fully constrained DOvS

```
1  Let iteration count $k = 0$, $\mathcal{C}_0 = \Theta$ and $\mathcal{V}_0 = \emptyset$ ;
2  while not terminating do
3      $k \leftarrow k + 1$ ;
4      sample a set solutions $X_k \leftarrow \{\vec{x}_1, \ldots, \vec{x}_m\}$ from $\mathcal{C}_{k-1}$ ;
5      $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup X_k$ ;
6      forall the $\vec{x} \in \mathcal{V}_k$ do
7          apply SAR to determine $a_k(\vec{x})$ and thus $N_k(\vec{x})$;
8          collect $\bar{\vec{G}}_k$ based on simulation observations ;
9      identify $\hat{\Pi}_k$ as the observed Pareto set on $\mathcal{V}_k$ ;
10     construct $\mathcal{C}_k$ based on $\hat{\Pi}_k$ and $\mathcal{V}_k$ ;
```

Note that the H-dimension vector $\bar{\vec{G}}_k$ is the sample average of observed performance measures by iteration k. To be specific, let $a_k(\vec{x})$ be the number of simulation replications assigned to solution $\vec{x}$ according to the SAR, and $\vec{G}_{(i)}$ be the values of performance evaluation from the i^{th} replication, then we have

$$\bar{\vec{G}}_k = \frac{1}{N_k(\vec{x})} \sum_{i=1}^{N_k(\vec{x})} \vec{G}_{(i)} \text{ in which } N_k(\vec{x}) = \sum_{j=1}^{k} a_k(\vec{x}).$$

Practically, the iteration can be repeated until simulation budget is exhausted or we are confident that $\hat{\Pi}_k$ is indeed an LPS (Figure 3.2), for example, all the neighbors of $\hat{\Pi}_k$ are visited and $\hat{\Pi}_k$ remains unchanged

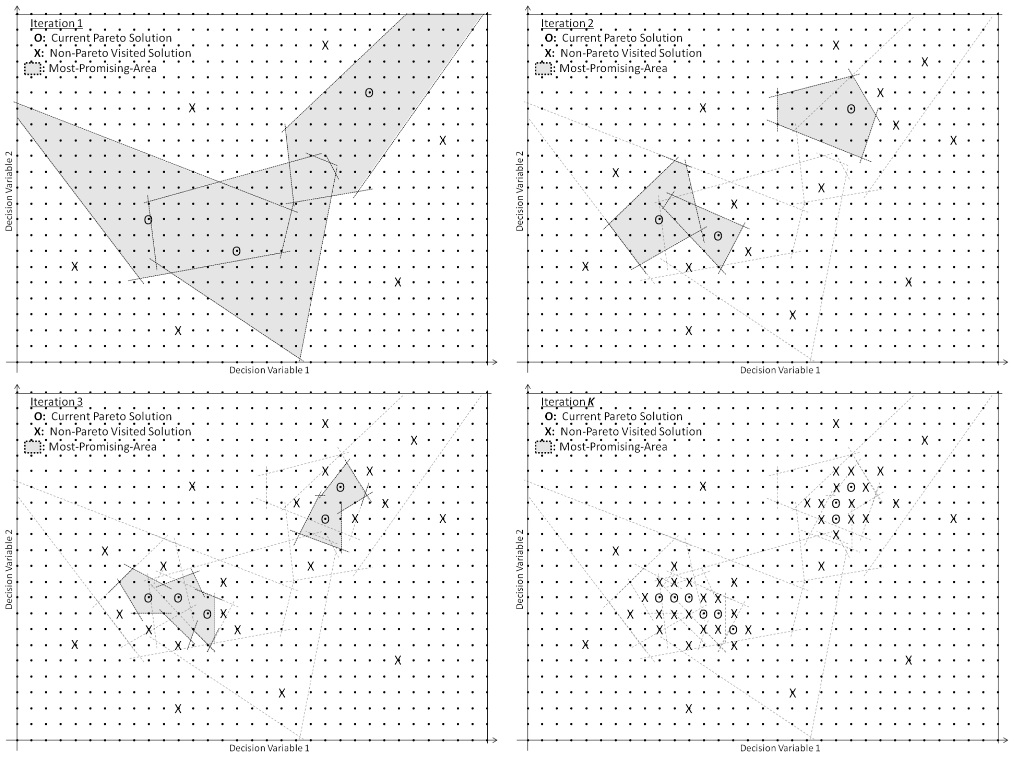


Figure 3.2: Illustration for MO-COMPASS

for a number of iterations. When conditions apply, Theorem 3.2 shows the strong convergence of $\hat{\Pi}_k$ to an LPS. Moreover, in the way to prove it, Theorem 3.1 also tells that the strong convergence property of $\hat{\Pi}_k$ also holds with respect to the Pareto set among all visited solutions.

First, we need to exclude the case that two solutions perform exactly the same for certain objective measure, for which we have Assumption 3.1. Consider a simple example where two solutions $\vec{x}_1$, $\vec{x}_2$ with $g^{(1)}(\vec{x}_1) = g^{(1)}(\vec{x}_2)$ and $g^{(2)}(\vec{x}_1) < g^{(2)}(\vec{x}_2)$. When noise occurs in the objective evaluation, we can never eliminate the possibility that $\bar{G}^{(1)}(\vec{x}_1) > \bar{G}^{(1)}(\vec{x}_2)$, thus the wrong conclusion could be made that two solutions are incomparable. Besides, we also need Assumption 3.2 to ensure that the simulation estimator $\bar{\vec{G}}(\vec{x})$ is consistent.

Assumption 3.1. *There exists an* $\epsilon_0 > 0$ *such that for all* $\vec{x}, \vec{y} \in \Theta$ *and* $l \in \{1, \ldots, H\}$,

$$\vec{x} \neq \vec{y} \Rightarrow |g^{(l)}(\vec{x}) - g^{(l)}(\vec{y})| \geq \epsilon_0.$$

Assumption 3.2. *For every* $\vec{x} \in \Theta$ *and* $l \in \{1, \ldots, H\}$,

$$\Pr\left\{\lim_{r\to\infty} \frac{1}{r} \sum_{i=1}^{r} G^{(l)}(\vec{x}, i) = g^{(l)}(\vec{x})\right\} = 1.$$

We note that Assumption 3.1 can easily holds as for DOvS the solution space is non-continuous. However, in some scenarios it can be violated. In such situation, we can either the apply indifference-zone method (Teng et al., 2010), or modify the performance measure by not changing its original purposes so as to make the assumption satisfied.

Moreover, we need to provide a guideline for selection SAR, which is stated in Condition 3.1.

Condition 3.1. *The SAR guarantees that* $a_k(\vec{x}) \geq 1$ *if* $\vec{x}$ *is a newly visited solution at iteration* k*, i.e.,* $\vec{x} \in \mathcal{V}_k \setminus \mathcal{V}_{k-1}$*, and* $\lim_{k\to\infty} N_k(\vec{x}) = +\infty$ *for all visited solutions, i.e.,* $\vec{x} \in \bigcup_{k=0}^{\infty} \mathcal{V}_k$.

Then we introduce Lemma 3.1, 3.2 and 3.3 which are proven in the Appendix A.1, noted that the neighbourhood of $\mathcal{V}_k$ is defined as

$$\mathcal{N}_k \equiv \left\{\vec{x} \in \Theta \,\middle|\, \exists \vec{z} \in \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq 1\right\}.$$

Based on the lemmas, we can establish the two convergence properties as in Theorem 3.1 and 3.2.

Lemma 3.1. *If Assumption 3.2 and Condition 3.1 are fulfilled, the sequence* $\{\mathcal{V}_1, \mathcal{V}_2, \dots\}$ *generated by Algorithm 3.1 satisfies*

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \ i.o., \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} = 0$$

for any ϵ *such that* $0 < \epsilon < \epsilon_0$.

Theorem 3.1. *If Assumption 3.1, 3.2 and Condition 3.1 are satisfied, the infinite sequence* $\left\{\hat{\Pi}_1, \hat{\Pi}_2, \dots\right\}$ *generated by Algorithm 3.1 converges with probability 1 to the Pareto set among all visited solutions in the sense that*

$$\Pr\left\{\mathrm{FS}\left(\hat{\Pi}_k, \mathcal{V}_k\right) \ i.o.\right\} = 0. \tag{3.2}$$

Proof. Let Π_k^* denote the true Pareto set on $\mathcal{V}_k$. Then, the definition of Pareto set and Assumption 3.1 tells that,

$$\Pi_k^* \equiv \left\{\vec{x} \in \mathcal{V}_k \mid \nexists \vec{y} \in \mathcal{V}_k, \forall l \in \{1, \dots, H\}, g^{(l)}(\vec{x}) > g^{(l)}(\vec{y})\right\}.$$

Meanwhile, $\hat{\Pi}_k$ is based on the sample average $\bar{\vec{G}}_k$, i.e.,

$$\hat{\Pi}_k \equiv \left\{\vec{x} \in \mathcal{V}_k \,\middle|\, \nexists \vec{y} \in \mathcal{V}_k, \forall l \in \{1, \dots, H\}, \bar{G}_k^{(l)}(\vec{x}) > \bar{G}_k^{(l)}(\vec{y})\right\}.$$

So, it is clear that the event $\mathrm{FS}\left(\hat{\Pi}_k, \mathcal{V}_k\right)$, i.e., $\Pi_k^* \neq \hat{\Pi}_k$ happens only if $\bar{G}_k^{(l)}\vec{x} \leq \bar{G}_k^{(l)}(\vec{y})$ for some $\vec{x}, \vec{y} \in \mathcal{V}_k$ and $l \in \{1, \dots, H\}$ such that $g^{(l)}(\vec{x}) >$

$g^{(l)}(\vec{y})$. Thus, in order to prove (3.2), it is sufficient to show

$$\Pr\left\{\begin{array}{r}\bar{G}_k^{(l)}(\vec{x}) \le \bar{G}_k^{(l)}(\vec{y}) \text{ i.o.,} \\ \exists \vec{x}, \vec{y} \in \mathcal{V}_k, l \in \{1, \dots, H\} \text{ s.t. } g^{(l)}(\vec{x}) > g^{(l)}(\vec{y})\end{array}\right\} = 0. \tag{3.3}$$

Besides, Assumption 3.1 tells that $g^{(l)}(\vec{x}) > g^{(l)}(\vec{y})$ implies $g^{(l)}(\vec{x}) - g^{(l)}(\vec{y}) \ge \epsilon_0$. Hence, $\bar{G}_k^{(l)}(\vec{x}) \le \bar{G}_k^{(l)}(\vec{y})$ implies either $\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \frac{\epsilon_0}{2}$ or $\left|\bar{G}_k^{(l)}(\vec{y}) - g^{(l)}(\vec{y})\right| > \frac{\epsilon_0}{2}$, or both.

Therefore, by Lemma 3.1

$$\begin{aligned}\Pr\left\{\bar{G}_k^{(l)}(\vec{x}) \le \bar{G}_k^{(l)}(\vec{y}) \text{ i.o., } \exists \vec{x}, \vec{y} \in \mathcal{V}_k, l \in \{1, \dots, H\} \text{ s.t. } g^{(l)}(\vec{x}) > g^{(l)}(\vec{y})\right\} \le \\ \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \frac{\epsilon_0}{2} \text{ i.o., } \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} + \\ \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{y}) - g^{(l)}(\vec{y})\right| > \frac{\epsilon_0}{2} \text{ i.o., } \exists \vec{y} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} = 0,\end{aligned}$$

which proves (3.3).

□

Lemma 3.2. *For the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \dots\}$ generated by Algorithm 3.1, it holds for all $k \ge 0$ that* $\Pr\{\vec{x} \in \mathcal{V}_{k+1} \,|\, \vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k\} > 0$.

Lemma 3.3. *For the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \dots\}$ generated by Algorithm 3.1,*

$$\Pr\{\mathcal{V}_k \neq \mathcal{V}_{k+1} \textit{ i.o.}\} = 0.$$

Theorem 3.2. *If Assumption 3.1,3.2 and Condition 3.1 are satisfied, the infinite sequence $\left\{\hat{\Pi}_1, \hat{\Pi}_2, \dots\right\}$ generated by Algorithm 3.1 converges with* *probability 1 to a LPS in the sense that*

$$\Pr\left\{\mathrm{FS}\left(\hat{\Pi}_k, \mathcal{N}_k\right) \ \ i.o.\right\} = 0. \tag{3.4}$$

Proof. Consider the condition on whether $\mathcal{N}_k \subset \mathcal{V}_k$, we then have the following

$$\Pr\left\{\mathrm{FS}(\hat{\Pi}_k, \mathcal{N}_k) \text{ i.o.}\right\} \leq \Pr\left\{\mathrm{FS}(\hat{\Pi}_k, \mathcal{V}_k) \text{ i.o.}\right\} + \Pr\left\{\mathcal{N}_k \not\subseteq \mathcal{V}_k \text{ i.o.}\right\}.$$

As $\Pr\left\{\mathrm{FS}(\hat{\Pi}_k, \mathcal{V}_k) \text{ i.o.}\right\} = 0$ is shown by Theorem 3.1, here we only need to prove

$$\Pr\left\{\mathcal{N}_k \not\subseteq \mathcal{V}_k \text{ i.o.}\right\} = 0,$$

or equivalently

$$\Pr\{\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k \text{ i.o., } \exists \vec{x} \in \Theta\} = 0. \tag{3.5}$$

To prove (3.5), we assume the opposite, i.e.,

$$\Pr\{\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k \text{ i.o., } \exists \vec{x} \in \Theta\} > 0. \tag{3.6}$$

Since Lemma 3.2 tells $\Pr\{\vec{x} \in \mathcal{V}_{k+1} \,|\, \vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k\} > 0$ for all $k \geq 0$, together with (3.6) it implies

$$\Pr\{\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k \text{ and } \vec{x} \in \mathcal{V}_{k+1} \text{ i.o., } \exists \vec{x} \in \Theta\} > 0.$$

In addition, as $\exists \vec{x} \in \Theta$ such that $\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k$ and $\vec{x} \in \mathcal{V}_{k+1}$ derives

$\mathcal{V}_k \neq \mathcal{V}_{k+1}$, we have

$$\Pr\{\mathcal{V}_k \neq \mathcal{V}_{k+1} \text{ i.o.}\} \geq \Pr\{\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k \text{ and } \vec{x} \in \mathcal{V}_{k+1} \text{ i.o.}, \exists \vec{x} \in \Theta\} > 0, \tag{3.7}$$

Obviously, (3.7) contradicts with Lemma 3.3. Thus, (3.6) cannot hold, which proves (3.5).

□

3.3.2 For Partially Constrained or Unconstrained MDOvS

The MO-COMPASS is also able to solve partially constrained or unconstrained MDOvS, by searching within a hyper-rectangular subset $\mathcal{B}_k$ at each iteration k and updating the boundaries of $\mathcal{B}_k$ persistently so as to reserve a positive buffer distance $\Delta^{(i)}$ from each visited solution at i^{th} dimension when the space is available. Specifically,

$$\mathcal{B}_k \equiv \bigcap_{i=1}^{p} \left[\underline{b}_k^{(i)}, \bar{b}_k^{(i)}\right] \tag{3.8}$$

in which

$$\underline{b}_k^{(i)} = \min\left\{\underline{b}_{k-1}^{(i)}, \min_{\vec{x}\in\mathcal{V}_k} x^{(i)} - \Delta^{(i)}\right\}$$

and

$$\bar{b}_k^{(i)} = \max\left\{\bar{b}_{k-1}^{(i)}, \max_{\vec{x}\in\mathcal{V}_k} x^{(i)} + \Delta^{(i)}\right\}$$

for all $k \geq 1$. Then, the most-promising-area is constructed correspondingly as

$$\mathcal{C}_k \equiv \bigcup_{\vec{z} \in \hat{\Pi}_k} \left\{ \vec{x} \in \mathcal{B}_k \,\middle|\, \forall \vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq \|\vec{x} - \vec{y}\| \right\}, \forall k \geq 1. \tag{3.9}$$

Given a starting feasible solution $\vec{x}_0$, the MO-COMPASS procedure is described by Algorithm 3.2.

Algorithm 3.2: MO-COMPASS for partially constrained or unconstrained MDOvS

```
1  Initialize iteration count $k = 0$ ;
2  set $\underline{b}_0^{(i)}$ and $\bar{b}_0^{(i)}$ such that $\underline{b}_0^{(i)} < x_0^{(i)} < \bar{b}_0^{(i)}, \forall i \in \{1, \dots, d\}$ ;
3  construct $\mathcal{B}_0$ according to (3.8) ;
4  let $\mathcal{C}_0 = \Theta \cap \mathcal{B}_0$ and $\mathcal{V}_0 = \emptyset$ ;
5  while not terminating do
6      $k \leftarrow k + 1$ ;
7      sample a set solutions $X_k \leftarrow \{\vec{x}_1, \dots, \vec{x}_m\}$ from $\mathcal{C}_{k-1}$ ;
8      $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup X_k$ ;
9      forall the $\vec{x} \in \mathcal{V}_k$ do
10         apply SAR to determine $a_k(\vec{x})$ and thus $N_k(\vec{x})$;
11         collect $\vec{\bar{G}}_k$ based on simulation observations ;
12     identify $\hat{\Pi}_k$ as the observed Pareto set on $\mathcal{V}_k$ ;
13     construct $\mathcal{B}_k$ and thus $\mathcal{C}_k$ based on $\hat{\Pi}_k$ and $\mathcal{V}_k$ ;
```

Be aware that one of the differences from Algorithm 3.1 is that the procedure must to be initialized by a feasible solution $\vec{x}_0$. In order for $\hat{\Pi}_k$ to converge to an LPS, $\vec{x}_0$ must satisfy condition as stated in Assumption 3.3, which is often true in practice when $\vec{x}_0$ is obtained by heuristics or expert opinions.

Assumption 3.3. *There exists a compact set Ω such that $\vec{x}_0 \in \Theta \cap \Omega$ and $\vec{g}(\vec{x}_0) \prec \vec{g}(\vec{x})$ for all $\vec{x} \in \Theta \setminus \Omega$.*

In addition, we also assume that $\vec{\bar{G}}$ is consistent as in Assumption 3.4 and a guideline for SAR is provided in Condition 3.2. So, we can deduce the convergence property for partially constrained or unconstrained cases.

Assumption 3.4. *For every $\vec{x} \in \Theta$ and $l \in \{1, \dots, H\}$, there exists an $r^* > 0$ such that for all $r \geq r^*$ and $\epsilon \in (0, \epsilon_0]$ (note ϵ_0 as in Assumption 3.1), $\Pr\left\{\left|\frac{1}{r}\sum_{i=1}^{r} G^{(l)}(\vec{x}, i) - g^{(l)}(\vec{x})\right| > \epsilon\right\} \leq \lambda(r, \epsilon)$, where $\lambda(r, \epsilon)$ is a strictly decreasing function of r and $\lambda(r, \epsilon) \to 0$ as $r \to \infty$.*

Condition 3.2. *The SAR guarantees that $\min_{\vec{x} \in \mathcal{V}_k} N_k(\vec{x}) \geq r_k$ where $r_0 \geq 1$, $r_{k+1} \geq r_k$ for all $k \geq 0$, $r_k \to \infty$ as $k \to \infty$, and $\lim_{k \to \infty} k^{d+1} \lambda(r_k, \epsilon) = 0$ for all $\epsilon \in (0, \epsilon_0]$, where ϵ_0 is defined in Assumption 3.1.*

Then, similar to the fully constrained cases, the two convergence properties of $\hat{\Pi}_k$ are stated in Theorem 3.3 and 3.4 respectively. To establish the theorems, we need Lemma 3.4, 3.5 and 3.6 which are proven in Appendix A.2.

Lemma 3.4. *If Assumption 3.4 and Condition 3.2 are fulfilled, the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \dots\}$ generated by Algorithm 3.2 satisfies*

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \ i.o., \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} = 0$$

for any ϵ such that $0 < \epsilon < \epsilon_0$.

Theorem 3.3. *If Assumption 3.1, 3.4 and Condition 3.2 are satisfied, the infinite sequence $\left\{\hat{\Pi}_1, \hat{\Pi}_2, \dots\right\}$ generated by Algorithm 3.2 converges with* *probability 1 to the Pareto set among all visited solutions in the sense that*

$$\Pr\left\{\mathrm{FS}(\hat{\Pi}_k, \mathcal{V}_k)\ i.o.\right\} = 0.$$

Proof. Theorem 3.3 can be proven in the same way as for Theorem 3.1, by applying Lemma 3.4 instead of Lemma 3.1.

$\square$

Lemma 3.5. *For the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \ldots\}$ generated by Algorithm 3.2, it holds for all $k \geq 0$ that $\Pr\{\vec{x} \in \mathcal{V}_{k+1} \,|\, \vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k\} > 0$, provided Assumption 3.1, 3.3, 3.4 and Condition 3.2.*

Lemma 3.6. *For the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \ldots\}$ generated by Algorithm 3.2, $\Pr\{\mathcal{V}_k \neq \mathcal{V}_{k+1}\ i.o.\} = 0$, provided Assumption 3.1, 3.3, 3.4 and Condition 3.2.*

Theorem 3.4. *If Assumption 3.1, 3.3, 3.4 and Condition 3.2 are satisfied, the infinite sequence $\left\{\hat{\Pi}_1, \hat{\Pi}_2, \ldots\right\}$ generated by Algorithm 3.2 converges with probability 1 to an LPS in the sense that*

$$\Pr\left\{\mathrm{FS}(\hat{\Pi}_k, \mathcal{N}_k)\ i.o.\right\} = 0.$$

Proof. Theorem 3.4 can be proven in the same way as for Theorem 3.2, by replacing Lemma 3.2 and 3.3 by Lemma 3.5 and 3.6 respectively. $\square$

3.4 Sampling Schemes

For random selection from $\mathcal{C}_k$, a default setting is that that every solution in $\mathcal{C}_k$ has equal probability to be sampled. However, it is difficult to implement in practice. Thus, Hong and Nelson (2006) adopts a RMD method.

In RMD method, we refer the best known point in current iteration as $\tilde{\vec{x}}$; a straight line is randomly selected in the space as long as it passes $\tilde{\vec{x}}$ and coincides with any of the dimensions. The line is then truncated at both ends by the boundaries of the MPA. A new point is uniformly selected on the line segment and it serves as the new $\tilde{\vec{x}}$. The procedure should be repeated by K times before $\tilde{\vec{x}}$ is reported as the new sample, where K is some integer that can be tuned. It is argued that, as K becomes larger, the new sample is uniformly distributed in the MPA.

And later, Hong et al. (2010) suggest the CS scheme which speed-up the convergence for solving high-dimension DOvS. Basically, CS is a special RMD with K set to 1. The sampling procedure can be further improved by reducing redundant linear constraints that form the convex set (Xu et al., 2010), or utilizing gradient information in a stochastic manner (e.g., Section 4.5.2).

We note that both sampling methods are applicable only to problems with convex solution set. Since in MO-COMPASS, $\mathcal{C}_k$ can be non-convex, we divide the selection into two steps: (1) uniform selection of a Pareto solution $\vec{z} \in \hat{\Pi}_k$, followed by (2) sampling on the subset about $\vec{z}$, i.e.,

$$\mathcal{D}_k(\vec{z}) \equiv \left\{ \vec{x} \in \Theta \,\middle|\, \forall \vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq \|\vec{x} - \vec{y}\| \right\}$$

for fully constrained problems, or taking the intersection with $\mathcal{B}_k$ for partially constrained or unconstrained problems. Since it can be observed that in both cases $\mathcal{D}_k(\vec{z})$ is convex, either RMD or CS sampling can be applied. By doing so, although each $\mathcal{D}_k(\vec{z})$ has equal probability to be selected, solution lying in overlapping areas tends to have larger chances. Intuitively, it is consistent with our design principle as approaching to multiple Pareto solutions may imply higher probability of it also being a Pareto solution.

Moreover, we note that the two steps sampling does not affect the unvisited neighbors of $\hat{\Pi}_k$ to be sampled with positive probability provided that $|\mathcal{C}_k| < \infty$. Thus, the local convergence property as been discussed in Section 3.3.1 and 3.3.2 remains.

3.5 Numerical Results

3.5.1 Convergence Test

The algorithm can be tested by constructing a multi-objective mathematical problem defined on $\mathbb{Z}_n^p \equiv \{1, \ldots, n\}^p \to \mathbb{R}^H$ where each objective is a quadratic function formulated as

$$g^{(l)}(\vec{x}) = \sum_{j=1}^{p} (x^{(j)} - x^{*(j)}_{l})^2 \text{ for } 1 \leq l \leq H,$$

in which $\vec{x}_l^* \in \mathbb{Z}_n^p$ is preset as the true optimum for the l^{th} objective.

Without knowing the formulation but only the returned objective values based on a given solution, we apply the MO-COMPASS on $\mathbb{Z}_n^p$ with RMD sampling. In order to show its ability to converge to an LPS, in this experiment we assume there is no noise in the evaluation. Thus each solution is only evaluated once and the algorithm terminates when all solutions in $\mathcal{N}_k$ have been visited and $\hat{\Pi}_k$ has been found to be an LPS. With this setting, the number of evaluations is equivalent to the number of solutions visited.

Besides, according to the problem we constructed, it is easy to observe that even when the number of objectives H remains the same, the size of a possible LPS increases geometrically as the dimension p becomes higher. Thus in the experiment, for easy test and comparison, we control the LPS size as 2 by setting $H = 2$ and selecting adjacent $\vec{x}_1^*$ and $\vec{x}_2^*$ that satisfy $\|\vec{x}_1^* - \vec{x}_2^*\| = 1$, thus we have the unique LPS $\mathcal{P} = \{\vec{x}_1^*, \vec{x}_2^*\}$, so as to eliminate any ambiguity caused by multiple local optima.

Varying the dimension p and scalar n, we test 30 independent applications of MO-COMPASS by initializing with different random seeds. The average number of visits before reaching the LPS is shown in Figures 3.3,3.4 and Table 3.1.

	$n = 10$	$n = 20$	$n = 30$	$n = 40$	$n = 50$
$p = 1$	62.7	39.1	28.9	22.7	19.5
$p = 2$	16.8	4.85	2.51	1.47	1.02
$p = 3$	2.97	4.84×10^{-1}	1.57×10^{-1}	7.24×10^{-2}	3.99×10^{-2}
$p = 4$	6.02×10^{-1}	4.37×10^{-2}	9.25×10^{-3}	3.17×10^{-3}	1.38×10^{-3}
$p = 5$	9.47×10^{-2}	3.47×10^{-3}	5.25×10^{-4}	1.37×10^{-4}	4.76×10^{-5}
$p = 6$	1.50×10^{-2}	2.89×10^{-4}	2.84×10^{-5}	5.45×10^{-6}	1.50×10^{-6}
$p = 7$	2.06×10^{-3}	2.12×10^{-5}	1.44×10^{-6}	2.12×10^{-7}	4.85×10^{-8}
$p = 8$	3.56×10^{-4}	1.67×10^{-6}	7.60×10^{-8}	8.39×10^{-9}	1.47×10^{-9}
$p = 9$	4.56×10^{-7}	1.11×10^{-7}	3.28×10^{-9}	2.73×10^{-10}	4.13×10^{-11}
$p = 10$	6.37×10^{-8}	8.00×10^{-9}	1.58×10^{-10}	9.90×10^{-12}	1.23×10^{-12}

Table 3.1: Proportion of solutions visited before reaching LPS, as p varies (%)

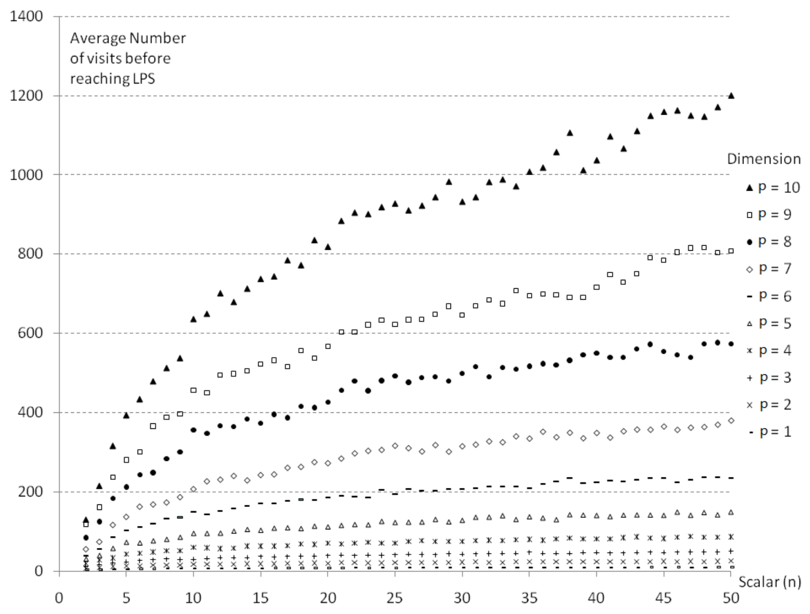


Figure 3.3: Average number of visits before reaching LPS, as n varies

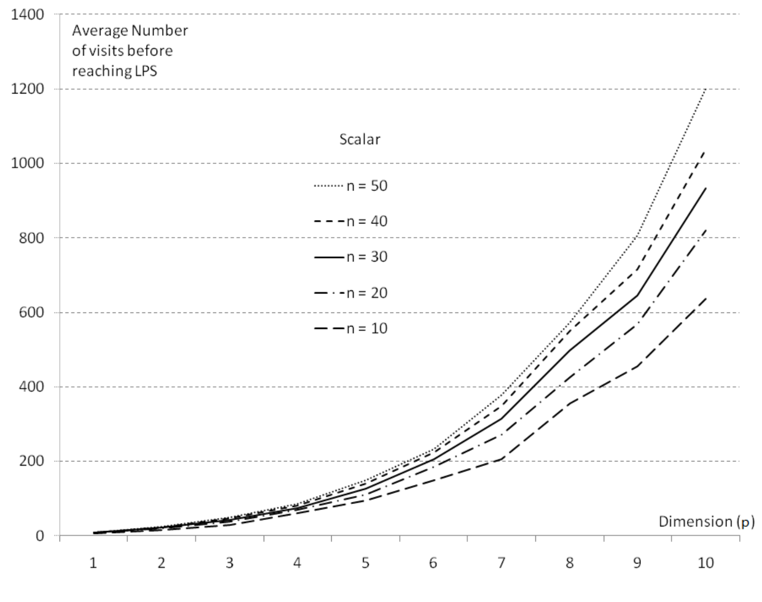


Figure 3.4: Average number of visits before reaching LPS, as p varies

Note that the Pareto solutions $\vec{x}_1^*$ and $\vec{x}_2^*$ are preset with a random selection for each p and n, but remain the same across independent MO-COMASS trials. Thus, the overall trend is not biased by the location of LPS. In addition, the result also indicates that the effect of differing LPS locations is smaller, as the trend versus p and n can be clearly identified.

From Figure 3.3 we observe that when dimension remains the same, the number of visits before reaching LPS increases with the scalar at a rate that is slower than linear. Meanwhile, if the scalar is kept unchanged, the increasing rate becomes steeper as the dimension is higher (Figure 3.4). Also, when the size of the solution space is taken into consideration, the proportion of visited solutions approaches zero (Table 3.1).

Besides, all the results from our numerical settings have shown that, in a noise-free case MO-COMPASS is able to terminate in an LPS with finite iterations.

3.5.2 Benchmark Comparison

This example is to show the convergence property and demonstrate the efficiency of MO-COMPASS. We compare it with an advanced multi-objective genetic algorithm NSGA-II (Deb et al., 2002) in solving a set of testing functions suggested by Zitzler et al. (2000), which are conventionally referred as ZDTs (Table 3.2). Since the comparison of algorithms is in term of the number of visited solutions, again we assume the evaluation is noise-free.

As MO-COMPASS aims at solving MDOvS, we modify the domain of $\vec{x}$ so that only discretized solutions can be chosen, i.e., $\vec{x} \in \left\{0, \frac{1}{L}, \dots, 1\right\}$

where L is the discretization level and p is the dimension of the search space. In our testing, $L = 20$ and $p = 30$ for all ZDTs.

We use CS for MO-COMPASS and set the batch size as 10; while for NSGA-II as the benchmark we set the population size as 50 and mutation rate as 0.01. For both algorithms, number of visits as the budget is capped as $3,000$ on ZDT1-4 and $1,500$ on ZDT6.

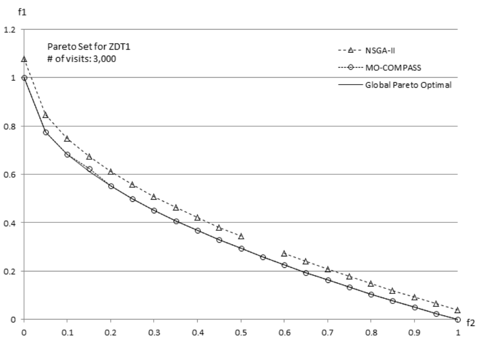
For each ZDT function we have 30 test runs for both algorithms initialized with different random seeds. After that, we aggregate the observed Pareto set from each run and find the Pareto set over all as shown in Figure 3.5. It implies that, for all ZDTs, with the same budget, the aggregated Pareto set obtained by MO-COMPASS dominates the set obtained by NSGA-II, which is an evidence to claim that the former is more efficient.

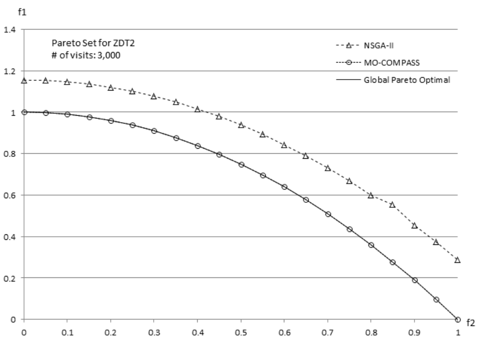
Moreover, noted that the solid lines in Figure 3.5 sketch the true global Pareto set for each function, MO-COMPASS is able to reach near the full set with given budget. To further test its speed of convergence, we run the algorithm until an LPS is reached. Figure 3.5 also shows the average number of visits for MO-COMPASS to terminate across 30 runs.

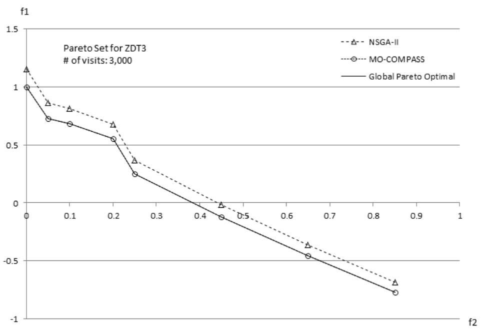
3.5.3 Adaptiveness Test

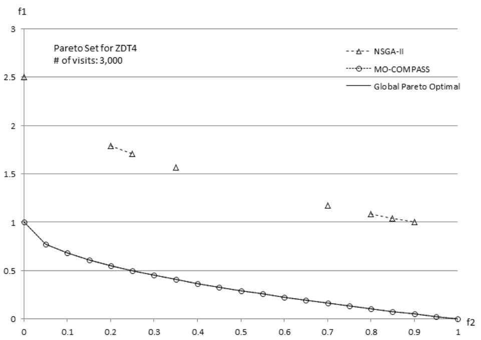
Consider an MDOvS where the simulation evaluation error does occur, this experiment shows the convergence of $\Pr\{\mathrm{CS}(\hat{\Pi}, \Theta)\}$ as iteration goes on. For illustration, the ZDT1 with $L = 20$ and $p = 20$ is used.

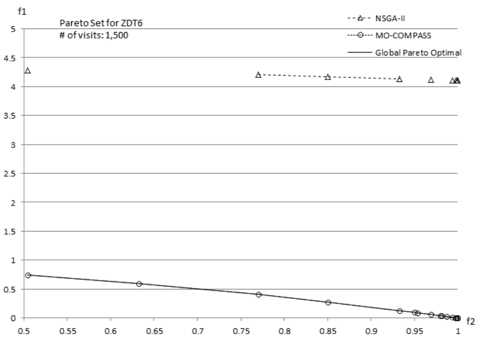
However, we can easily observe that Assumption 3.1 is violated, because for any two solutions $\vec{x}$, $\vec{y}$ that $\vec{x} \neq \vec{y}$ as long as $x_1 = y_1$ we have $g^{(1)}(\vec{x}) =$











Average Number of Visits for
MO-COMPASS to Terminate

Test Function	Average Num. of Visits
ZDT1	4,323
ZDT2	3,067
ZDT3	2,758
ZDT4	3,174
ZDT6	1,870

Figure 3.5: Illustration of MO-COMPASS

$g^{(1)}(\vec{y})$. To avoid the violation, instead of to minimize $g^{(1)}$ and $g^{(2)}$, we modify the ZDT1 as to minimize $g^{(1)} + 0.05g^{(2)}$ and $g^{(2)}$ which does not affect the intended purpose. Then, an error term $\varepsilon \sim \text{Norm}(0, 10^{-3})$ is added to both objective evaluations so that Assumption 3.2 holds. Besides, the SAR has been set as an equal allocation with $N_k(\vec{x}) = \min\{1, \log k\}$ for all $\vec{x} \in \mathcal{V}_k$, which satisfies Condition 3.1.

To observe $\Pr\{\text{CS}(\hat{\Pi}, \Theta)\}$, we conduct the experiment with $1,000$ independent runs, each starting with a different random seed. The number of evaluations is recorded once "correct selection" starts to occur, meaning it is found to be true that the observed Pareto is not only an LPS (as in Theorem 3.2), but also the Pareto set on all visited solutions (as in Theorem 3.1). Be aware that due to random initialization, the resulted LPS can be different among test runs. Also, for computational simplicity, in order to claim that an observed set is an LPS, we only check for its sufficient conditions, i.e. the set is a Pareto set on all visited solutions and all neighbors of the set are visited. In other words, the detection of the occurrence of correct selection is delayed in practice.

Then we count the observed occurrence of correct selection versus the number of evaluations, so as to estimate the convergence of $\Pr\{\text{CS}(\hat{\Pi}, \Theta)\}$ to 1 shown by Figure 3.6. It infers that for MO-COMPASS with dedicated SAR, the probability of correct selection converges to 1 as number of simulation evaluations increases.

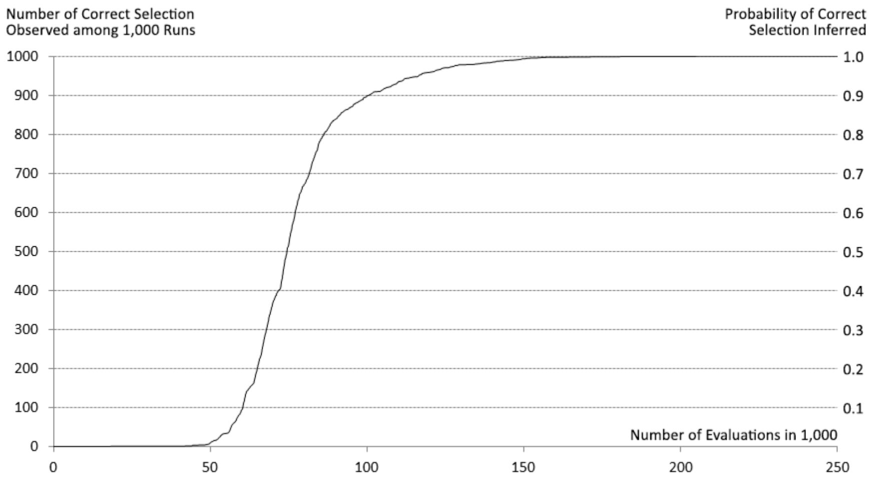


Figure 3.6: Convergence of $\Pr\{\mathrm{CS}(\hat{\Pi}, \Theta)\}$ considering simulation evaluation error.

3.5.4 Industrial Application

As mentioned in Chapter 1, D-SIMSPAIR$^{\mathrm{TM}}$ is a typical industrial application of MDOvS.

In this experiment, we apply the MO-COMPASS in D-SIMSPAIR$^{\mathrm{TM}}$with a flight network contains 69 stock locations and three part numbers with low / medium / high demands respectively are chosen for illustration. S-ince it has been ensured in D-SIMSPAIR$^{\mathrm{TM}}$that each simulation evaluation is given sufficient computational budget to achieve required accuracy, we treat the evaluation as noise-free.

As there are two phases for its optimization procedure, namely "enumeration" which samples and looks for optimal inventory configuration for specified part numbers, and "navigation" which is to find the best combination of inventory configurations across all parts, our testing addresses only the "enumeration" phase and aims to find a Pareto set of inventory configurations in terms of high service level and low cost. The “navigation” phase is not considered in this test.

Obviously, for each part, the number of possible inventory configurations is at least $(X+1)^{69}$, where X is the maximum inventory level allowed in each location. A simple calculation shows that if $X = 1$ and every inventory configuration take simulation 1 second to evaluate, an exhausted search could take more than 10^{13} years to complete, which is impossible to be done in practice. Since then, we can never tell the optimal solutions with 100% confidence. Hence, in this experiment to measure the effectiveness of MO-COMPASS, we refer to a pseudo optimality which is the Pareto set among all solutions that have ever been simulated in our past study (with various algorithms and different random seeds).

Besides, we proposed a simple random search algorithm as a benchmark to show the efficiency of MO-COMPASS. The random search starts with the same initial solution as MO-COMPASS, but at each iteration randomly selects a solution from the history and uniformly vary its value at one coordinate within the feasible region, so as to generate a new sample. It can be referred as CS without any constraint from the MPA.

To plot the improvement history of search, we adopt the concept of dominated hyper-volume (Section 5.1.1) to uniquely indicate the Pareto set quality. It makes more sense where the cost and service level are treated as two objectives, because when the service level is bounded in the range of 0 and 1, the indicator can be intuitively interpreted as the average cost saving along the spectrum of the service level. In our study, set the maximum cost as $\$12,000$ and given $10,000$ simulation evaluation for each, the comparison between MO-COMPASS and the random search for the three parts are illustrated by Figure 3.7.

We can conclude from the comparison that, the MO-COMPASS is much more efficient in terms of the gap to the pseudo optimality when a limited budget is given. For low/medium/high demand parts, the gaps are \$113.13 (1.04%), \$121.63 (1.22%) and \$468.30 (4.34%) respectively while for random search are \$7,568 (69.5%), \$8,455 (84.7%) and \$5,622 (52.5%).

For MO-COMPASS, the gap becomes larger as demand becomes higher because the solution space is increasing while the simulated budget is fixed. And, for high-demand part, the random search seems to be superior at the beginning of the search, basically due to the reason that it has a wider exploration in the large solution space in the early phase while MO-COMPASS could be constrained in its MPA. However, as MPA keeps refining itself, the advantage of MO-COMPASS shows up.

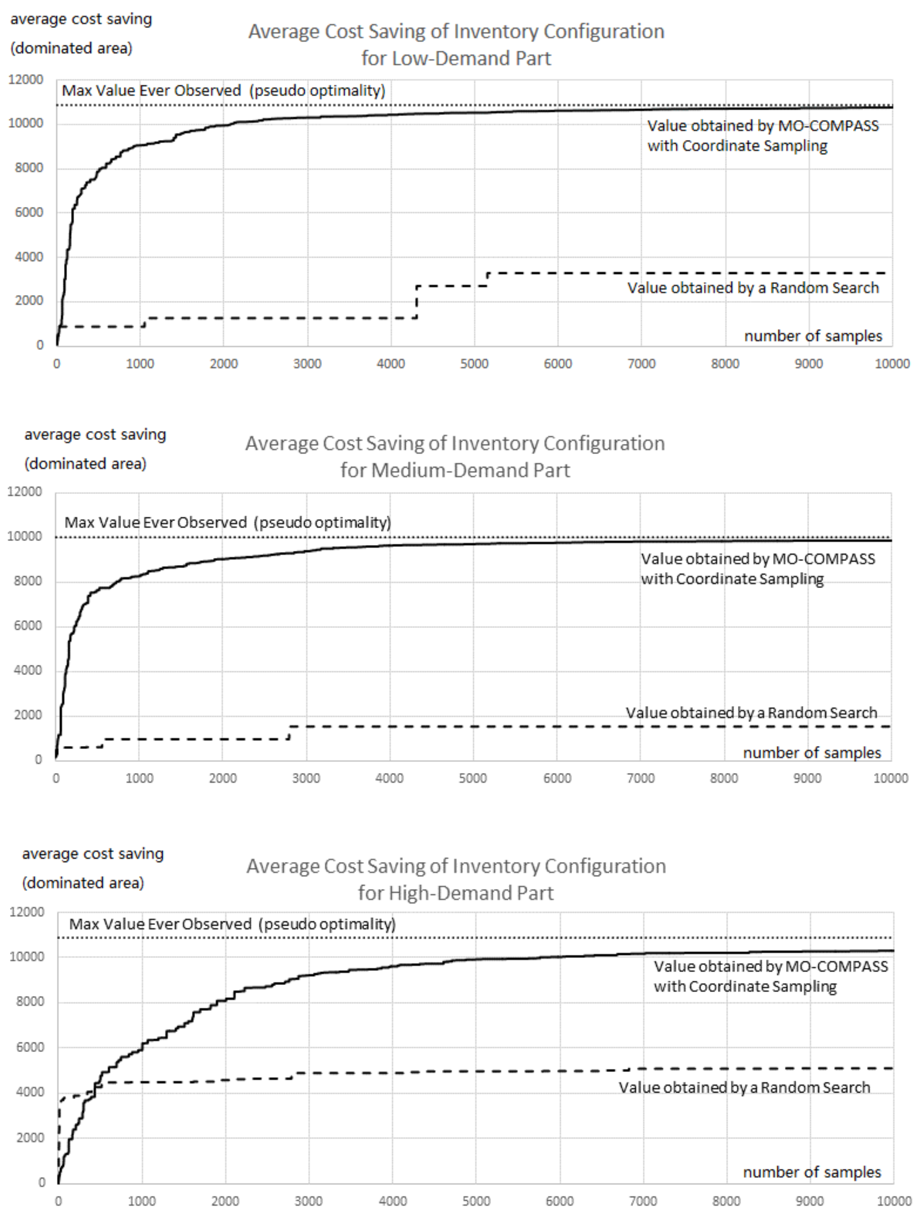


Figure 3.7: Performance of MO-COMPASS in D-SIMSPAIR^{TM} compared to random search.

General Structure	$\min g^{(1)}(\vec{x}), g^{(2)}(\vec{x})$, s.t. $g^{(2)}(\vec{x}) = f(x_2, \ldots, x_d) \cdot h\left(g^{(1)}(\vec{x}), f(x_2, \ldots, x_d)\right)$ where $\vec{x} = (x_1, \ldots, x_d) \in [0,1]^d$
ZDT1	$g^{(1)}(\vec{x}) = x_1$ $f(x_2, \ldots, x_d) = 1 + 9 \cdot \sum_{i=2}^{d} x_i/(d-1)$ $h\left(g^{(1)}, f\right) = 1 - \sqrt{g^{(1)}/f}$ (The function has a convex Pareto-optimal front)
ZDT2	$g^{(1)}(\vec{x}) = x_1$ $f(x_2, \ldots, x_d) = 1 + 9 \cdot \sum_{i=2}^{d} x_i/(d-1)$ $h\left(g^{(1)}, f\right) = 1 - \left(g^{(1)}/f\right)^2$ (The function is the non-convex counterpart to ZDT1)
ZDT3	$g^{(1)}(\vec{x}) = x_1$ $f(x_2, \ldots, x_d) = 1 + 9 \cdot \sum_{i=2}^{d} x_i/(d-1)$ $h\left(g^{(1)}, f\right) = 1 - \sqrt{g^{(1)}/f} - \left(g^{(1)}/f\right) \sin\left(10\pi g^{(1)}\right)$ (The Pareto-optimal front contains several non-continuous convex parts)
ZDT4	$g^{(1)}(\vec{x}) = x_1$ $f(x_2, \ldots, x_d) = 1 + 10(d-1) + \sum_{i=2}^{d} \left(x_i^2 - 10\cos\left(4\pi x_i\right)\right)$ $h\left(g^{(1)}, f\right) = 1 - \sqrt{g^{(1)}/f} - \left(g^{(1)}/f\right) \sin\left(10\pi g^{(1)}\right)$ (The Pareto-optimal front contains several non-continuous convex parts)
ZDT6	$g^{(1)}(\vec{x}) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f(x_2, \ldots, x_d) = 1 + 9 \cdot \left[\left(\sum_{i=2}^{d} x_i\right)/(d-1)\right]^{0.25}$ $h\left(g^{(1)}, f\right) = 1 - \left(g^{(1)}/f\right)^2$ (The function includes difficulties caused by non-uniformity of the search space)

Table 3.2: The ZDT testing functions.

Chapter 4

The Gradient Oriented Polar Random Search

As in the previous chapter, we discussed about the MO-COMPASS that is shown to have local convergent property in solving MSOP. Start from this chapter, we would also like to focus on how the gradient-based techniques can help to improve the search efficiency, especially when a broad exploration is concerned together with effective utilization of the local information.

In this chapter, we first propose a brand new polar coordinate system, and two random distributions are defined based on it, namely the polar uniform distribution and the oriented polar distribution, with which we can easily control the randomness injected to a search direction. Based on the proposed coordinate system, in Section 4.2 we propose a new search algorithm called the GO-POLARS. Subsequently, the local convergence property and numerical examples are to be discussed.

In order to have a wider application, the technique is designed in continuous solution space. We test it on several single-objective problems. Then in the next chapter, with the MO search framework (Algorithm 1.1) and unified gradient concept (Section 5.2), we will apply the technique into MSOP and the effectiveness is shown by numerical examples as well.

4.1 The Polar Framework

4.1.1 Hyper Polar Coordinates

For a p-dimensional optimization problem, a Cartesian coordinate system is usually adopted to uniquely identify a solution point in the domain space. In Cartesian system, all coordinates are orthogonal to each other, and a point is denoted by $\vec{x} = [x_1, \ldots, x_p]$ such that x_i refers to its projected position on the ith coordinate.

Cartesian system is a natural way to represent solutions of optimization problems, because in many cases x_i directly refers a decision parameter. However, we observe that for many adaptive or local search algorithms Cartesian representation may not be the best choice as the search is driven by two key factors, namely the direction and the distance. But neither of them is explicitly expressed in a Cartesian system. Thus, we may think of an alternative way to denote the solution, such as polar coordinates.

It should be well known that, a polar coordinate system can be defined on a two-dimensional space in which every point is denoted by its angle with respect to an axis and distance to the origin (Weisstein, 2009). Besides, the similar idea can be brought into a three-dimensional case so as to form a system called spherical coordinates (Weisstein, 2005) or spherical polar coordinates (Arfken, 1985; Walton, 1967). However, higher dimension cases are seldom discussed in literature. So, as following we propose a hyper polar coordinate representation that can be adopted for any high dimensional cases.

Definition 4.1. *In a p-dimensional polar coordinate system, a point is denoted by $[\,r, \vec{\theta}\,]$, in which $r \in [0, \infty)$ and $\vec{\theta} \in [0, 2\pi) \times [0, \pi]^{p-2}$, if its Euclidean distance from the origin is r (radial coordinate) and $\vec{\theta}$ (angular coordinate) refers its direction in the space in the sense that θ_i denotes its angle with respect to the positive direction of the $i + 1^{th}$ axis towards the hyperplane spanned by the first i axes.*

To be more specific, the conversion from polar to Cartesian coordinates in p-dimensional space can be described by Equation (4.1) and (4.2). The degree of freedom of both coordinates remains the same.

$$x_1 = r \prod_{j=1}^{p-1} \sin\theta_j \;, \tag{4.1}$$

$$x_i = r \cos\theta_{i-1} \prod_{j=i}^{p-1} \sin\theta_j \text{ for } 2 \le i \le p \;. \tag{4.2}$$

However, the conversion above is not invertible, because some points which are uniquely represented by Cartesian coordinates may have multiple representations in a polar system, e.g. for any θ in the domain we have $[0, 0]_{\text{Cart}} \to [0, \theta]_{\text{Polar}}$, and $[0, 0, 1]_{\text{Cart}} \to [1, \theta, 0]_{\text{Polar}}$. To eliminate the ambiguous, by Algorithm 4.1 we provide a sequential way of converting Cartesian to polar coordinates. We notice that, since θ_i for $i \geq 2$ is defined on $[0, \pi]$, we can find it by directly applying arc-cosine function; but for θ_1 that is defined on $[0, 2\pi)$, both sine and cosine functions need to be addressed for a unique identification.

Algorithm 4.1: Conversion from Cartesian to polar coordinates

```
1 $r \leftarrow \sqrt{\sum_{i=1}^{p} x_i^2}$ ;
2 if $r = 0$ then
3     $\vec{\theta} \leftarrow \vec{0}$;
4 else
5     $i \leftarrow p$;
6     while $i > 2$ do
7         $\theta_{i-1} \leftarrow \arccos\left[x_i / \left(r \prod_{j=i}^{p-1} \sin\theta_j\right)\right]$;
8         $i \leftarrow i - 1$ ;
9     Solve $\cos\theta_1 = x_2 / \left(r \prod_{j=2}^{p-1} \sin\theta_j\right)$ and
      $\sin\theta_1 = x_1 / \left(r \prod_{j=2}^{p-1} \sin\theta_j\right)$, so as to get $\theta_1$.
```

An illustration for the hyper polar coordinate representation in the two and three dimensional space can be found in Figure 4.1.

4.1.2 Polar Uniform Distribution

With hyper polar coordinates we are able to denote a point in terms of the direction and distance referring to a given position, which provides an advantage for algorithms to explicitly control their search process. But as mentioned in early part of this chapter where the variation is involved in sampling a direction, random distribution need to be defined before we move to introduce the algorithm. First of all, we look at a uniform case.

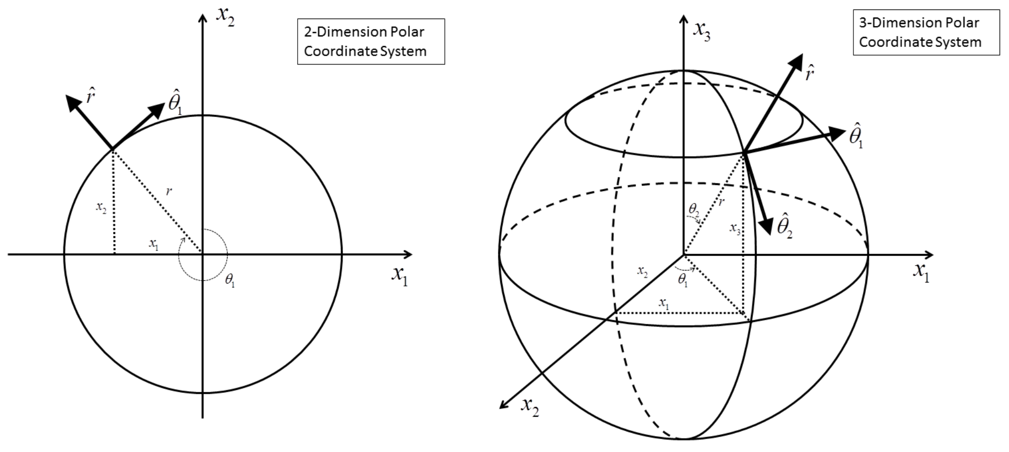


Figure 4.1: An illustration of hyper polar coordinates (in 2D & 3D)

We notice that the uniform sampling is not straightforward as in a p-dimension space using Cartesian coordinates, in which we can simply sample each x_i uniformly within the domain. By using polar coordinates, sampling each θ_j uniformly will cause points unevenly distributed on a unit hypersphere (Figure 4.2). To be more specific, points tend to concentrate around latter axes. Such an effect can also be concluded by analyzing E-quation (4.1) and (4.2), where the product $\prod_{j=1}^{p-1} \sin\theta_j$ takes in more factors for small i with larger p and we know for sure that all $|\sin\theta_j| \leq 1$. As the result, the value on earlier axes tends to have high density around 0. Obviously, this way of sampling does not satisfy the uniformity we desire, since points on certain directions have higher chance to be sampled compared to the rest and the contrast becomes sharper as p increases.

Hence, we should look at the problem from a different point of view. We can consider a hyper-ball with radius r around the origin, so that for all the points spread in the outermost layer, each of them should have

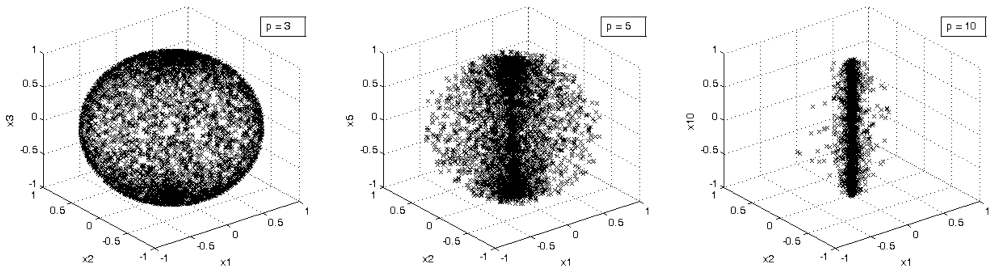


Figure 4.2: Biased polar uniform distribution with $r = 1$ and $p = 3, 5, 10$.

equal opportunity to be sampled as the direction. From mathematical point of view, let $f(r, \vec{\theta})$ be the probability density function, then within an infinitesimal space around the point $[r, \vec{\theta}]$, the probability for points to be sampled is

$$f(r, \vec{\theta}) \cdot \partial(r, \theta_1, \ldots, \theta_{p-1}).$$

By consensus of uniformity, this probability should be proportional to the volume of the infinitesimal space $\partial V = \partial(x_1, \ldots, x_p)$, meaning there exists a function $c(r) \geq 0$ depends only on r such that

$$\frac{f(r, \vec{\theta}) \cdot \partial(r, \theta_1, \ldots, \theta_{p-1})}{\partial(x_1, \ldots, x_p)} = c(r).$$

Further notice that the Jacobian determinant (Kaplan, 1991) is the ratio of the hyper-volumes mapping between different coordinate systems. For conversion from p-dimensional polar to Cartesian coordinates, we denote it as $|J_p|$ and it can be shown that

$$|J_p| = \left| \frac{\partial(x_1, \ldots, x_p)}{\partial(r, \theta_1, \ldots, \theta_{p-1})} \right| = (-r)^{p-1} \prod_{j=1}^{p-1} \sin^{j-1} \theta_j. \tag{4.3}$$

Proof. We prove (4.3) by induction. Consider the base case, i.e., $p = 2$. We have

$$|J_2| = \left|\frac{\partial(x_1, x_2)}{\partial(r, \theta_1)}\right| = \begin{vmatrix} \partial x_1/\partial r & \partial x_1/\partial\theta_1 \\ \partial x_2/\partial r & \partial x_2/\partial\theta_1 \end{vmatrix} = \begin{vmatrix} \sin\theta_1 & r\cos\theta_1 \\ \cos\theta_1 & -r\sin\theta_1 \end{vmatrix} = -r,$$

which satisfies (4.3). Then we only need to prove that for all $p \geq 2$,

$$|J_{p+1}| = (-r\sin^{p-1}\theta_p)\,|J_p|\,. \tag{4.4}$$

For any $p \geq 2$, from (4.1) and (4.2) we can derive a general form of Jacobian matrix $|J_{p+1}|$ in terms of $|J_p|$, i.e.,

$$J_{p+1} = \begin{bmatrix} J_p^{(1)} \cdot \sin\theta_p & J_p^{(2,\dots,p)} \cdot \sin\theta_p & J_p^{(1)} \cdot r\cos\theta_p \\ \cos\theta_p & \vec{0} & -r\sin\theta_p \end{bmatrix}.$$

Note that $J_p^{(1)}$ is the 1st column of matrix J_p, and $J_p^{(2,\dots,p)}$ is the $(p-1)\times p$ matrix consists of the 2nd to pth columns of J_p. So it follows that

$$\begin{aligned} |J_{p+1}| = (-1)^p\cos\theta_p & \begin{vmatrix} J_p^{(2,\dots,p)} \cdot \sin\theta_p & J_p^{(1)} \cdot r\cos\theta_p \end{vmatrix} \\ -r\sin\theta_p & \begin{vmatrix} J_p^{(1)} \cdot \sin\theta_p & J_p^{(2,\dots,p)} \cdot \sin\theta_p \end{vmatrix}, \end{aligned} \tag{4.5}$$

in which the matrix $\begin{bmatrix} J_p^{(2,\dots,p)} \cdot \sin\theta_p & J_p^{(1)} \cdot r\cos\theta_p \end{bmatrix}$ can be obtained from J_p by interchanging $(p-1)$ pairs of columns and multiplying $(p-1)$ columns by $\sin\theta_p$ and 1 column by $r\cos\theta_p$, while the matrix $\begin{bmatrix} J_p^{(1)} \cdot \sin\theta_p & J_p^{(2,\dots,p)} \cdot \sin\theta_p \end{bmatrix}$

is equivalent to $\sin\theta_p \cdot J_p$. The matrix operations above imply that,

$$\left| \begin{matrix} J_p^{(2,\ldots,p)} \cdot \sin\theta_p & J_p^{(1)} \cdot r\cos\theta_p \end{matrix} \right| = (-1)^{p-1} \cdot \sin^{p-1}\theta_p \cdot (r\cos\theta_p) \, |J_p|$$

and

$$\left| \begin{matrix} J_p^{(1)} \cdot \sin\theta_p & J_p^{(2,\ldots,p)} \cdot \sin\theta_p \end{matrix} \right| = \sin^p\theta_p \, |J_p| \, .$$

Thus, combining with (4.5) we have

$$\begin{aligned} |J_{p+1}| &= -r\cos^2\theta_p \sin^{p-1}\theta_p \, |J_p| - r\sin\theta_p \sin^p\theta_p \, |J_p| \\ &= \left(-r\sin^{p-1}\theta_p\right) |J_p| \, , \end{aligned} \tag{4.6}$$

which proves (4.4). □

Then, we derive the probability density function for a polar uniform distribution as in (4.7) and thus have Definition 4.2. Note that $c(r)$ has to ensure that the integral of $f(\,r, \vec{\theta}\,)$ on the domain equals to 1.

$$f(\,r, \vec{\theta}\,) = c(r) \cdot r^{p-1} \prod_{j=1}^{p-1} \sin^{j-1}\theta_j. \tag{4.7}$$

Definition 4.2. *A random point $[\,r, \vec{\theta}\,]$ is said to be from a p-dimensional polar uniform distribution, denoted as U^p_{polar}, if its probability density function is given as in* (4.7).

When enforce $r = 1$, i.e., let $c(r) = 0$ if $r \neq 1$, the angular coordinate $\vec{\theta}$ can be sampled uniformly in the sense that $[\,1, \vec{\theta}\,] \sim \mathrm{U}^p_{\mathrm{polar}}$, which is illustrated by Figure 4.3. Since r is fixed and θ_j is independent from each

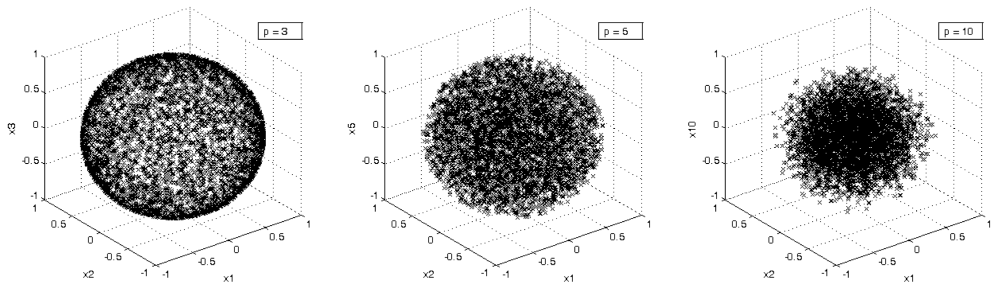


Figure 4.3: Polar uniform distribution with $r = 1$ and $p = 3, 5, 10$.

other, we can decompose the probability density function for each j as

$$f_j(\theta_j) = c_j \sin^{j-1}\theta_j, \text{ for } j = 1, \dots, p-1, \tag{4.8}$$

where $c_1 = \frac{1}{2\pi}$ and $c_j = \int_0^\pi \sin^{j-1}\theta d\theta$ for $j \geq 2$. As there is no close form for c_j, one of the sampling method is to apply numerical approaches, such as acceptance-rejection method or Alias method (Schwarz, 2011; Vose, 1991) after discretization into small intervals, so that the constant term can be ignored.

In practice, some good properties can be observed. Since (4.8) is independent with dimension p, a point $[1, \vec{\theta}] \sim \mathrm{U}^p_{\text{polar}}$ can be easily extended to $\mathrm{U}^{p+1}_{\text{polar}}$ by adding element θ_p sampled from distribution with density $f_p(\theta_p) = c_p \sin^{p-1}\theta_p$.

Moreover, considering (4.1) and (4.2), we notice that the newly added θ_p does not affect the relative values of previous x_is since all of them are simply scaled by $\sin\theta_p$, whereas the density in (4.8) only ensures that the new comer x_{p+1} plays harmoniously with the early ones by maintaining the uniformity into the higher dimension. This property is important when we need to control the distribution to be concentrated, which is to be discussed later in Section 4.1.3.

We also observe that, a multivariate normal distribution $\mathrm{N}(\vec{0}, I\sigma^2)$ with any σ is a special case for polar uniform distribution when it is converted into polar coordinates, as stated in Theorem 4.1.

Theorem 4.1. *If a vector $\vec{x} = [x_1, \ldots, x_p]$ is from a multivariate normal distribution $N(\vec{0}, I\sigma^2)$, its corresponding polar coordinates $[r, \vec{\theta}] \sim U^p_{polar}$.*

Proof. Since $[x_1, \ldots, x_p] \sim \mathrm{N}(\vec{0}, I\sigma^2)$, we have

$$f(x_1, \ldots, x_p) = \prod_{i=1}^{p} \phi(x_i) = (2\pi\sigma^2)^{-\frac{p}{2}} \exp\left(-\frac{p}{2}\sum_{i=1}^{p}\frac{x_i^2}{2\sigma^2}\right).$$

Note that with conversion to polar coordinates, $\sum_{i=1}^{p} x_i^2 = r^2$. By applying the Jacobian determinant as in (4.3), we can derive

$$f(r, \vec{\theta}) = f(x_1, \ldots, x_p)\,|J_p| = c(r) \cdot r^{p-1} \prod_{j=1}^{p-1} \sin^{j-1}\theta_j$$

where $c(r) = \left(2\pi\sigma^2 \exp\left(\frac{r^2}{2\sigma^2}\right)\right)^{-p/2}$. According to Definition 4.2, $[r, \vec{\theta}] \sim \mathrm{U}^{p}_{\mathrm{polar}}$. □

We observe the case where $p = 2$, the result of Theorem 4.1 has been applied in a reversed manner for generating normal random variables (Muller, 1959), i.e., sample a vector $[r, \theta_1]$ from a 2-dimension polar uniform distribution with specified $c(r)$ that depends on σ, and then claim the corresponding Cartesian coordinates x_1, x_2 from an independent normal distribution $\mathrm{N}(0, \sigma^2)$ respectively.

Meanwhile, for the case where $p > 2$, Theorem 4.1 can simplify polar uniform generation by using random variables form multivariate normal distribution to sample $\vec{\theta}$.

4.1.3 Oriented Polar Distribution

Firstly, we consider a simple case, where we want the sampled direction to be concentrated around a given direction $\vec{\tilde{d}}$ that coincides with the positive direction of the p^{th} axis, i.e. $\vec{\tilde{d}} = \vec{e}_p$. It means that, under the Cartesian representation only x_p has a priority to choose larger value. Thus, using the property discussed in Section 4.1.2, we may have a point $[1, \vec{\theta}] \sim \mathrm{U}^{p-1}_{\text{polar}}$, and extend it to p-dimension by adding θ_{p-1} where the distribution can be adjusted from (4.8), so that x_p has higher chance to take large value without touching the ratios among the others. A typical way is to take the composite density with concentrating function φ that decreases on $[0, \pi]$, and symmetrically increases on $(\pi, 2\pi)$, i.e.,

$$f_{p-1}(\theta_{p-1}) = c'_{p-1} \sin^{p-2} \theta_{p-1} \cdot \varphi(\theta_{p-1}). \tag{4.9}$$

where $\varphi(\alpha) > \varphi(\beta)$ and $\varphi(\alpha) = \varphi(2\pi - \alpha)$ for any $0 \leq \alpha \leq \beta \leq \pi$. One example of φ is the density function of a normal distribution $\mathrm{N}(0, \sigma^2)$ with reflection at π, i.e.,

$$\begin{aligned} \phi_\sigma(\theta) &= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{\theta^2}{2\sigma^2}\right) \text{ for } \theta \in [0, \pi], \\ &\text{and } \phi_\sigma(\theta) = \phi_\sigma(2\pi - \theta) \text{ for } \theta \in (\pi, 2\pi). \end{aligned} \tag{4.10}$$

Note that (4.9) is defined on $[0, 2\pi)$ for $p = 2$, and $[0, \pi]$ for $p > 2$.

By using (4.10), we have an explicit way to control the degree of concentration, namely the magnitude of σ directly refers to the deviation of the sample from $\vec{\tilde{d}}$. An extreme case can be observed when $\sigma = 0$ so that $\phi_\sigma(\theta_{p-1})$ for all $\theta_{p-1} \neq 0$, meaning that all sampled direction coincide with $\vec{\tilde{d}}$ almost for sure. In another way, if $\sigma = \infty$, we have equal value of $\phi_\sigma(\theta_{p-1})$ at all θ_{p-1}, thus the term is cancelled out from (4.9). In that case, the distribution becomes $\mathrm{U}^p_{\mathrm{polar}}$. Thus, $\sigma < \infty$ is one and the only condition to ensure that $\phi_\sigma(\theta)$ is valid as a concentrating function.

For a general $\varphi(\theta)$, we define the standard polar oriented distribution as in Definition 4.3.

Definition 4.3. *A random point $[\,r, \vec{\theta}\,]$ is said to be from a p-dimensional standard polar oriented distribution, denoted as O^p_{polar}, if $[r, \theta_1, \ldots, \theta_{p-2}] \sim U^{p-1}_{polar}$ and θ_{p-1} has distribution as in* (4.9).

Given any r, the procedure of generating $[\,r, \vec{\theta}\,] \sim \mathrm{O}^p_{\mathrm{polar}}$ is described by Algorithm 4.2. With $\phi_\sigma(\theta)$ set as the concentrating function and choose σ to be different values, we have the illustration of sampled points shown by Figure 4.4.

Algorithm 4.2: Sampling from a standard polar oriented distribution

```
1 $j \leftarrow 1$ ;
2 while $j < p - 1$ do
3     Sample $\theta_j$ from its domain with density as in (4.8) ;
4     $j \leftarrow j + 1$ ;
5 Sample $\theta_{p-1}$ from its domain with density as in (4.9) ;
6 $\vec{\theta} \leftarrow [\theta_1, \ldots, \theta_{p-1}]$.
```

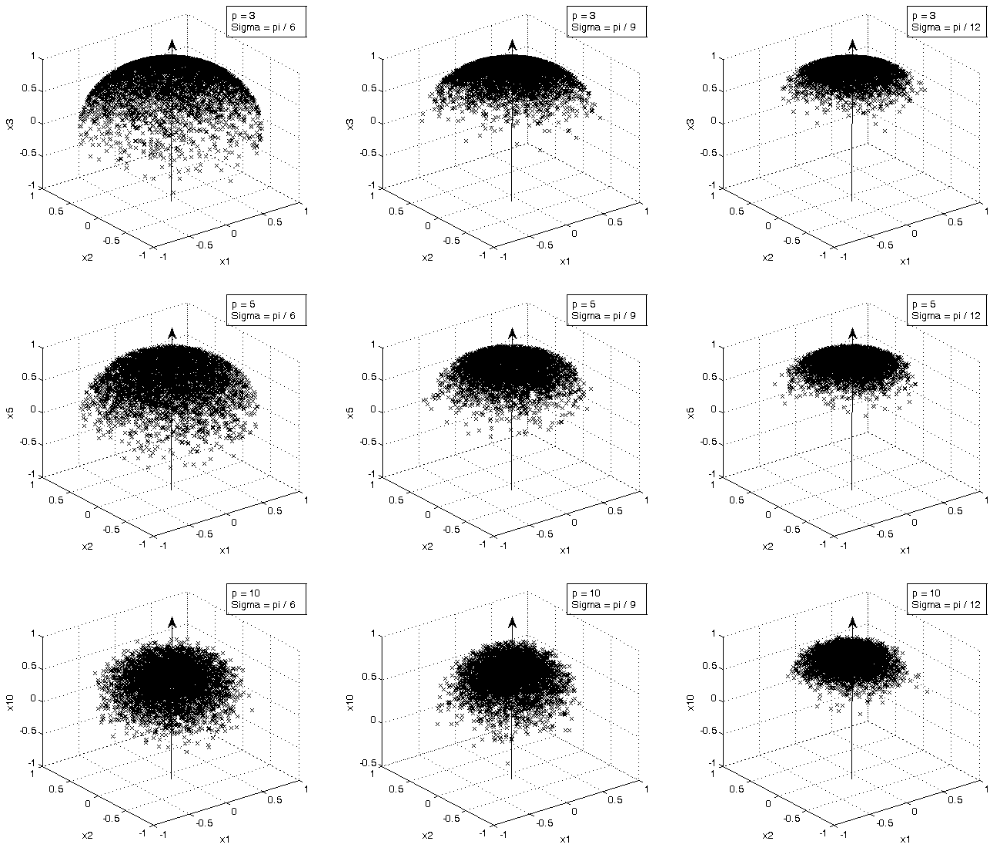


Figure 4.4: Standard polar oriented distribution with $r = 1$, $p = 3, 5, 10$ and $\phi_\sigma(\theta)$ for which $\sigma = \pi/6, \pi/9, \pi/12$.

Denote $\vec{d} = ([1, \vec{\theta}])_{\text{Cart}}$ as the Cartesian conversion of $[1, \vec{\theta}]$, we can analyze the expectation of $\vec{d}$, so as to derive Theorem 4.2. Later we will use its corollary to prove the local convergence property in Section 4.3.

Theorem 4.2. *For a unit vector $\vec{d} \sim O^p_{polar}$, we can always finds a scalar $\gamma \in (0, 1]$ such that $\mathbb{E}\left[\vec{d}\right] = \gamma \cdot \vec{e}_p$.*

Proof. From (4.2), it is straightforward that

$$\mathbb{E}\left[d_p\right] = \int_{\theta_{p-1}} 1 \cdot \cos\theta_{p-1} \cdot f_{p-1}(\theta_{p-1})\, d\theta_{p-1} \leq \int_{\theta_{p-1}} f_{p-1}(\theta_{p-1})\, d\theta_{p-1} = 1. \tag{4.11}$$

At the same time, considering (4.9) we also have

$$\mathbb{E}\left[d_p\right] = \int_{\theta_{p-1}} 1 \cdot \cos\theta_{p-1} \cdot c'_{p-1} \sin^{p-2}\theta_{p-1} \cdot \varphi(\theta_{p-1})\, d\theta_{p-1}. \tag{4.12}$$

For the case when $p = 2$,

$$\begin{aligned}(4.12) &= 2c'_1 \left(\int_{\theta_1=0}^{\theta_1=\pi/2} \varphi(\theta_1)\, d\sin\theta_1 - \int_{\theta_1=\pi}^{\theta_1=\pi/2} \varphi(\theta_1)\, d\sin\theta_1 \right) \\ &= 2c'_1 \cdot \int_0^1 \left(\varphi\left(\arcsin\left(t\right)\right) - \varphi\left(\pi - \arcsin\left(t\right)\right)\right) dt.\end{aligned}$$

Similarly, when $p > 2$,

$$\begin{aligned}(4.12) &= \frac{c'_{p-1}}{p-1} \left(\int_{\theta_{p-1}=0}^{\theta_{p-1}=\pi/2} \varphi(\theta_{p-1})\, d\sin\theta_{p-1} - \int_{\theta_{p-1}=\pi}^{\theta_{p-1}=\pi/2} \varphi(\theta_{p-1})\, d\sin\theta_{p-1} \right) \\ &= \frac{c'_{p-1}}{p-1} \cdot \int_0^1 \left(\varphi\left(\arcsin\left(t^{1/(p-1)}\right)\right) - \varphi\left(\pi - \arcsin\left(t^{1/(p-1)}\right)\right)\right) dt.\end{aligned}$$

We observe that $t \in [0,1] \Rightarrow t^{1/(p-1)} \in [0,1]$ for all $p \geq 2$, thus

$$\arcsin\left(t^{1/(p-1)}\right) \in [0, \pi/2]$$

which implies

$$0 \leq \arcsin\left(t^{1/(p-1)}\right) \leq \pi - \arcsin\left(t^{1/(p-1)}\right) \leq \pi.$$

Because as the concentrating function, φ has to be monotonically decreasing on $[0,\pi]$ which is discussed previously, so we can conclude that

$$\varphi\left(\arcsin\left(t^{1/(p-1)}\right)\right) - \varphi\left(\pi - \arcsin\left(t^{1/(p-1)}\right)\right) \geq 0,$$

and we note that the equality holds only when $t = 1$.

Besides, $c'_{p-1} > 0$ for all $p \geq 2$. Hence, it holds that (4.12) > 0. Together with (4.11), we conclude that

$$0 < \mathbb{E}\left[d_p\right] \leq 1.$$

Moreover, from (4.1), (4.2) and (4.8) we can calculate that $\mathbb{E}\left[d_i\right] = 0$ for $1 \leq i \leq p-1$. Thus, a final conclusion can be made as, for $\vec{d} \sim \mathrm{O}^{p}_{\mathrm{polar}}$

$$\exists\, \gamma \in (0,1], \mathbb{E}\left[\vec{d}\right] = \gamma \cdot \vec{e}_p. \tag{4.13}$$

□

The result can also be visualized as intuitively Algorithm 4.2 is designed

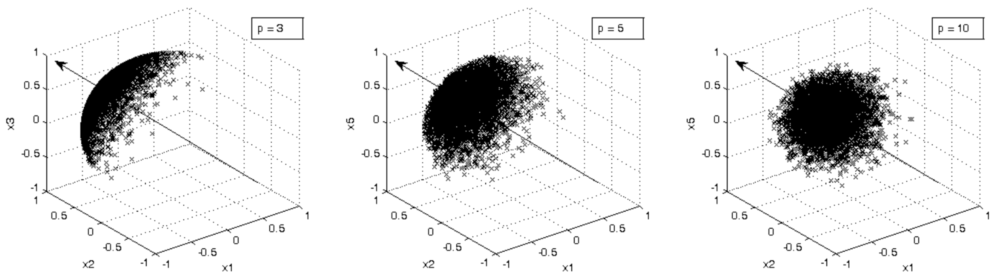


Figure 4.5: Oriented polar distribution with $\vec{\tilde{d}} = \sum_{i=2}^{p} \vec{e}_i - \vec{e}_1$, $\phi_{\pi/9}$ and $p = 3, 5, 10$.

to ensure distribution is centra-symmetric about the p^{th} axis.

Note that the value of γ depends only on p and φ. Specially, when $\varphi = \phi_\sigma$ and p is fixed, γ is a monotonically decreasing function about σ, written as $\gamma(\sigma)$.

For the case where the given $\vec{\tilde{d}}$ is an arbitrary unit vector, a linear transformation can be applied such that every point obtained by Algorithm 4.2 is reflected on a line lies in the middle of $\vec{\tilde{d}}$ and $\vec{e}_p$. In a reverse manner, we have Definition 4.4 for the oriented polar distribution. Figure 4.5 is an illustration.

Definition 4.4. *A random point* $\vec{d} = \left(\left[r, \vec{\theta}\right]\right)_{Cart}$ *is said to be from a* p*-dimensional polar distribution oriented by a unit vector* $\vec{\tilde{d}}$*, denoted as* $O^p_{polar}\left(\vec{\tilde{d}}\right)$*, if* $\left(2\vec{d} \cdot \vec{m}/\|\vec{m}\|^2\right)\vec{m} - \vec{d} \sim O^p_{polar}$ *where* $\vec{m} = \left(\vec{\tilde{d}} + \vec{e}_p\right)/2$*.*

Obviously, as the result of linear transformation, from Theorem 4.2 we have Corollary 4.3.

Corollary 4.3. *For a unit vector* $\vec{d} \sim O^p_{polar}\left(\vec{\tilde{d}}\right)$*, we can always finds a scalar* $\gamma \in (0, 1]$ *such that* $\mathbb{E}\left[\vec{d}\right] = \gamma \cdot \vec{\tilde{d}}$*.*

4.2 The Algorithm

The Gradient Oriented Polar Random Search (GO-POLARS) is designed in an adaptive manner. At each iteration, the optimum estimate moves to a random direction with a step size which is guided by the gradient. Specifically, let $\Theta \subseteq \mathbb{Z}^p$ be the feasible region, the search algorithm can be described as in Algorithm 4.3.

Algorithm 4.3: Gradient oriented polar random search

```
1  Pick an initial guess $\hat{\vec{x}}_0 \in \Theta$, and $k \leftarrow 0$ ;
2  while not terminating do
3      Select a deviation parameter $\sigma_k$ ;
4      Generate a unit $\vec{d}_k$ (as the sampling direction) from
       $\mathrm{O}^p_{\text{polar}}\left(\frac{\hat{\vec{\nabla}}(\hat{\vec{x}}_k)}{\|\hat{\vec{\nabla}}(\hat{\vec{x}}_k)\|}\right)$ with $\varphi = \phi_{\sigma_k}$, in which $\hat{\vec{\nabla}}\left(\hat{\vec{x}}_k\right)$ is the estimated
       gradient at $\hat{\vec{x}}_k$ ;
5      $\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - b_k \left\|\hat{\vec{\nabla}}\left(\hat{\vec{x}}_k\right)\right\| \vec{d}_k$ ;
6      if $\hat{\vec{x}}_{new} \in \Theta$ and $L\left(\hat{\vec{x}}_{new}\right) < L\left(\hat{\vec{x}}_k\right)$ then
7          $\hat{\vec{x}}_{k+1} \leftarrow \hat{\vec{x}}_{\text{new}}$
8      else
9          $\hat{\vec{x}}_{k+1} \leftarrow \hat{\vec{x}}_k$
10     $k \leftarrow k+1$
```

Remark. The search procedure can be tuned by controlling the gain sequence b_k and the direction deviation sequence σ_k. In Section 4.3, we will discuss conditions in terms of b_k and σ_k for the algorithm to converge to a local optimum.

4.3 Local Convergence Property

In literature, local convergence properties of stochastic algorithms are often shown by convergence theory of SA (Spall, 2003), such as in simulated annealing (Gelfand and Mitter, 1993), genetic algorithms (Yin et al., 1995), neutral network back-propagation (Spall and Cristion, 1994), and etc.. We notice that GO-POLARS also shares some similarities with SA, i.e., both have estimates updated adaptively according to the gradient information with certain noise. So in this subsection, we relate GO-POLARS to SA and conclude the convergence conditions for the sequence b_k and σ_k.

We start with rewriting Step 5 in Algorithm 4.3 as an SA type, i.e., $\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - a_k \vec{Y}_k\left(\hat{\vec{x}}_k\right)$ where

$$a_k = \gamma\left(\sigma_k\right) \cdot b_k, \tag{4.14}$$

$$\text{and } \vec{Y}_k\left(\hat{\vec{x}}_k\right) = \frac{\left\|\hat{\vec{\nabla}}\left(\hat{\vec{x}}_k\right)\right\|}{\gamma\left(\sigma_k\right)} \cdot \vec{d}_k. \tag{4.15}$$

Note that in a typical SA procedure, a_k is the gain sequence and $\vec{Y}_k\left(\hat{\vec{x}}_k\right) = \vec{\nabla}\left(\hat{\vec{x}}_k\right) + \vec{\varepsilon}_k\left(\hat{\vec{x}}_k\right)$ is an approximation of gradient $\vec{\nabla}$ with error term $\vec{\varepsilon}_k$. The "statistics" conditions for strong convergence can be specified as in (4.16) - (4.19) (Blum, 1954a,b; Nevel'son and Khas'inskiĭ, 1973).

$$a_k > 0, a_k \to 0, \sum_{k=0}^{\infty} a_k = \infty, \text{ and } \sum_{k=0}^{\infty} a_k^2 < \infty, \tag{4.16}$$

$$\inf_{\eta < \|\vec{x} - \vec{x}^*\| < 1/\eta} \left(\vec{x} - \vec{x}^*\right)^{\mathrm{T}} B \vec{\nabla}\left(\vec{x}\right) > 0 \text{ for all } 0 < \eta < 1, \tag{4.17}$$

$$\mathbb{E}\left[\vec{Y}_k\left(\vec{x}\right) - \vec{\nabla}\left(\vec{x}\right)\right] = \vec{0} \text{ for all } \vec{x} \text{ and } k, \tag{4.18}$$

$$\mathbb{E}\left[\left\|\vec{Y}_k\left(\hat{\vec{x}}_k\right)\right\|^2\right] \leq c\left(1 + \|\vec{x}\|^2\right) \text{ for all } \vec{x}, k \text{ and some } c > 0, \tag{4.19}$$

where B is some symmetric, positive definite matrix.

We observe that (4.17) is the condition on the problem nature which is independent of algorithm parameters. So if we can provide necessary conditions on GO-POLARS for (4.16), (4.18) and (4.19), the local convergence property can be derived in Theorem 4.4.

Theorem 4.4. *Given that conditions in* (4.17), (4.20), (4.21) *and* (4.22) *are satisfied, the search iterate* $\hat{\vec{x}}_k$ *generated by Algorithm 4.3 converges to a local optimum almost surely.*

$$\exists \sigma^* < \infty \text{ such that } \forall k, \sigma_k \leq \sigma^*, \tag{4.20}$$

$$b_k > 0, b_k \to 0, \sum_{k=0}^{\infty} b_k = \infty, \text{ and } \sum_{k=0}^{\infty} b_k^2 < \infty, \tag{4.21}$$

$$\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|^2 \leq c\left(1 + \|\vec{x}\|^2\right) \text{ for all } \vec{x}, k \text{ and some } c > 0. \tag{4.22}$$

Proof. As discussed in Section 4.1.3, $\gamma\left(\sigma\right)$ is monotonically decreasing about σ on $(0, 1]$. Hence, (4.20) implies $0 < \gamma\left(\sigma^*\right) \leq \gamma\left(\sigma_k\right) \leq 1$. Together with (4.14), we can derive from (4.21) that

$$b_k > 0 \Rightarrow a_k = \gamma\left(\sigma_k\right) \cdot b_k > 0,$$

$$b_k \to 0 \Rightarrow a_k = \gamma\left(\sigma_k\right) \cdot b_k \leq b_k \to 0,$$

$$\sum_{k=0}^{\infty} b_k = \infty \Rightarrow \sum_{k=0}^{\infty} a_k = \sum_{k=0}^{\infty} \gamma\left(\sigma_k\right) \cdot b_k \geq \gamma\left(\sigma^*\right) \sum_{k=0}^{\infty} b_k = \infty, \text{ and}$$

$$\sum_{k=0}^{\infty} b_k^2 < \infty \Rightarrow \sum_{k=0}^{\infty} a_k^2 = \sum_{k=0}^{\infty} \gamma^2\left(\sigma_k\right) \cdot b_k^2 \leq \gamma^2\left(\sigma^*\right) \sum_{k=0}^{\infty} b_k^2 < \sum_{k=0}^{\infty} b_k^2 < \infty.$$

The above shows that (4.16) is satisfied.

Then we consider the expectation of the error term,

$$\mathbb{E}\left[\vec{\varepsilon}_k\left(\vec{x}\right)\right] = \mathbb{E}\left[\vec{Y}_k\left(\hat{\vec{x}}_k\right) - \vec{\nabla}\left(\hat{\vec{x}}_k\right)\right].$$

From (4.15) and Corollary 4.3, we know that for all $\vec{x}$ and k,

$$\mathbb{E}\left[\vec{Y}_k\left(\hat{\vec{x}}_k\right)\right] = \frac{\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|}{\gamma\left(\sigma_k\right)} \mathbb{E}\left[\vec{d}_k\right] = \frac{\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|}{\gamma\left(\sigma_k\right)} \cdot \gamma\left(\sigma_k\right) \frac{\vec{\nabla}\left(\hat{\vec{x}}_k\right)}{\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|} = \vec{\nabla}\left(\hat{\vec{x}}_k\right),$$

implying $\mathbb{E}\left[\vec{\varepsilon}_k\left(\vec{x}\right)\right] = \vec{0}$. Thus, (4.18) holds.

Similarly, to prove (4.19), we observe

$$\mathbb{E}\left[\left\|\vec{Y}_k\left(\hat{\vec{x}}_k\right)\right\|^2\right] = \left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\| \cdot \mathbb{E}\left[\left\|\vec{d}_k\right\|^2\right] / \gamma^2\left(\sigma_k\right).$$

As $\vec{d}_k$ is a unit vector, $\left\|\vec{d}_k\right\| = 1$ with no doubt. Then (4.22) is sufficient to meet requirement of (4.19). Again, (4.22) is a condition that solely depends on $\vec{g}$ due to the problem nature.

Hence, Theorem 4.4 is proven. $\square$

4.4 Mechanism of Local-optimum Breakout

For a multi-modal optimization problem, to prevent $\hat{\vec{x}}_k$ to be trapped in certain local region, we can apply a breakout mechanism in addition to Algorithm 4.3.

The mechanism can be stimulated when $\left\| g\left(\hat{\vec{x}}_k\right) \right\|$ is observed to be smaller than a threshold τ indicating that a local optimum has been reached. Then without using any gradient information we let $\vec{d} \sim \mathrm{U}^p_{\text{polar}}$ and

$$\hat{\vec{x}}_{k+1} = \hat{\vec{x}}_k + b_{\text{Jump}} \cdot \vec{d}$$

where b_{Jump} is a jumping distance that we believe to get rid the local region. Then re-initialize the b_k sequence by replacing b_{k+1} by b_0. Notice that, parameter τ and b_{Jump} can be tuned in the sense that small τ has better exploitation within the local region while MLB is effective only when b_{Jump} is large enough.

4.5 Numerical Experiments

4.5.1 A Benchmark Comparison

In this section, we compare GO-POLARS with several benchmark search algorithms including gradient-based search and metaheuristics local search.

The Goldstein-Prices function is a two-dimensional global optimization test function as defined in (4.23). Note that the global minimum occurs at $\vec{x}^* = [0, -1]$ with $g(\vec{x}^*) = 3$, and several local minima occur as well. Set the search domain $\Theta = \mathbb{R}^2$ and assume that the gradient can be calculated at every $\vec{x} \in \Theta$, we used the function to compare the performance of GO-POLARS with SD, SAN and RPG as described in Table 4.1.

$$\begin{aligned} g(\vec{x}) = &\left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \\ &\cdot \left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right] \end{aligned} \tag{4.23}$$

For fair comparison, we adopt a neighborhood structure setting in SAN that is similar to the GO-POLARS iterate. But instead of choosing direction from a polar normal distribution oriented by the gradient, we let it be generated by a multivariate normal distribution that does not involve gradient. However, for comparison consistency, the magnitude of gradient is used in determining the sample distance. In the experiment, we set $a_k = 0.001/k$ and $\sigma_k = (1 - k/500)\pi$, so that the deviation of sampled direction gradually decreases from π to 0, which ensures a better exploration at the beginning of the search while obtained a better convergence when it approaches to the end. Besides, for SAN, the temperature T_k is set to be $t(500-k)$. As the experiment does not show significant difference when t is tuned to be any positive value, we set $t = 1$ for illustration. While for RPG, we select the perturbation variable $\vec{d}_k$ from a standard multivariate normal

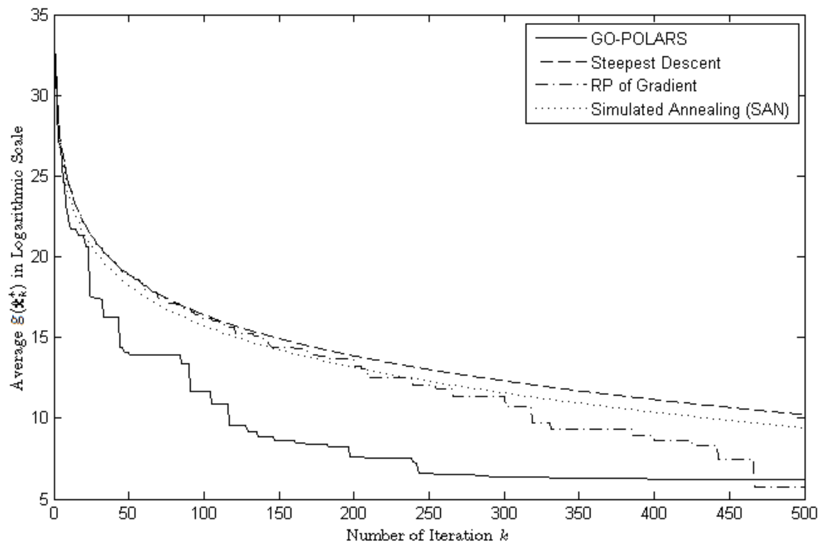


Figure 4.6: Average $g\left(\hat{\vec{x}}_k^*\right)$ by different search algorithms.

distribution, and λ_k is tuned as well so as to achieve a better performance at $\sqrt{100/\log(k+1)}$.

The four algorithms can be correlated by starting with a same initial solution $\vec{x}_0$ that is randomly selected from $[-2,2]^2$, and run the algorithms until $k = 500$. Repeat the process for 50 replications, we then present the average $g(\vec{x}_k^*)$ in Figure 4.6. Note that in each replication, $\vec{x}_k^*$ denotes the best solution visited upon iteration k. It is obvious that the average performance of RPG and GO-POLARS across replication is superior than both SD and SAN.

To analyze the reason, we notice that in SD only single direction is allowed to be sampled. As the problem has multiple local optima, it incurs a larger probability of being trapped in one of them. But GO-POLARS ensures that all directions have a positive chance to be selected when $\sigma \neq 0$, which enlarges the pool of solutions that can be explored.

SAN also allows solutions to be sampled on every direction. However, it only rejects inferior samples after the solution is evaluated, according to an artificial temperature parameter T_k. While in GO-POLARS, solutions on different directions can be filtered without any evaluation by the gradient-oriented polar distribution.

Besides, the performance of GO-POLARS and RPG is similar at the end of the search, although GO-POLARS has an obvious advantage at the early stage. The reason being that both algorithms integrate the advantage from random search and utilizing gradient information. Whereas GO-POLARS perturbs the direction instead of the point, so that the exploration effect is multiplied by the stepsize a_k which is large at the beginning. For RPG, as the perturbation term is controlled separately, its effect remains significant throughout the search, but relatively small compared with the effect from the initial stepsize.

4.5.2 Application in Stochastic Search

As stated in Chapter 2, almost all stochastic search algorithms involve random sampling within a specified neighborhood, where it is assumed that gradient information is not available. However, in the cases when gradient can be observed or estimated, we can apply GO-POLARS to help in sampling good solutions more efficiently. On the other hand, if GO-POLARS alone could not obtain desired efficiency, to integrate it with an advanced stochastic search will probably make the achievement.

Assume solutions are to be sampled from a convex set Θ in which $\hat{\vec{x}}^*$ is the best known up-to-date. We may sample

$$\hat{\vec{x}}_{\text{new}} = \hat{\vec{x}}^* - r \cdot \vec{d}, \text{ where } \vec{d} \sim \mathrm{O}^p_{\text{polar}} \left(\frac{\vec{\nabla}\left(\hat{\vec{x}}_k\right)}{\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|} \right) \text{ and } r \sim \mathrm{U}(0, R] \tag{4.24}$$

in which R is the maximum value of r that ensures $\hat{\vec{x}}_{\text{new}} \in \Theta$.

We illustrate the concept using COMPASS (Hong and Nelson, 2006), which is initially proposed for solving discrete optimization problems, but has been observed performing well also for continuous cases. The main idea of the algorithm is to construct a most-promising-area after evaluation of all historical samples and in a new iteration retake samples within the area according to a given sampling scheme. For instance, Hong and Nelson (2006) suggest the RMD method aiming to generate samples almost uniformly. But later it is identified to be less efficient in solving high-dimensional problems, for which the CS is proposed instead (Hong et al., 2010).

We apply the COMPASS on a high-dimension continuous test function as in (4.25). The function is initially proposed by Rosenbrock (1960) with $p = 2$ and extended by Moré et al. (1981) to higher dimension. Here, we use the setting $p = 10$. Note that it has a unique optimum $g(\vec{x}^*)$ occurring at $\vec{x}^* = [1, 1, \ldots, 1]$.

$$g(\vec{x}) = \sum_{i=1}^{p/2} \left[100 \left(x_{2i} - x_{2i-1}^2 \right)^2 + \left(1 - x_{2i-1} \right)^2 \right] \text{ with } \Theta = [-4, 4]^p \tag{4.25}$$

Two sampling schemes are compared in the test, namely the CS and

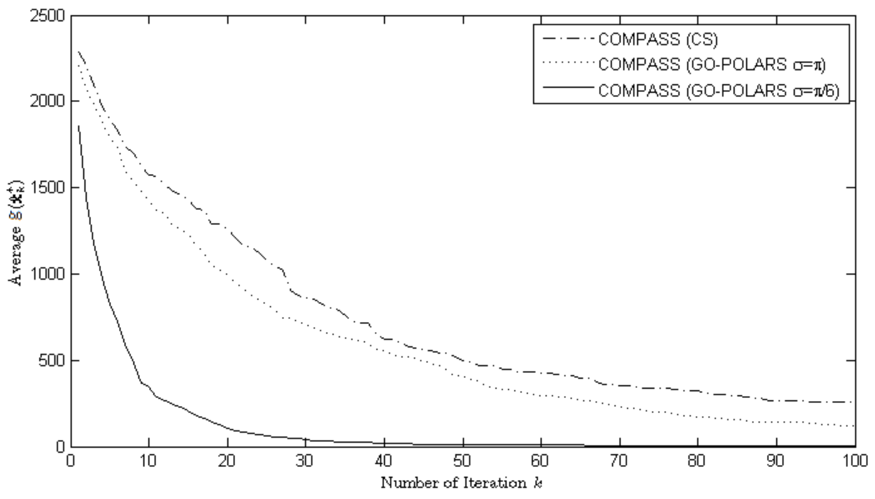


Figure 4.7: Average $g\left(\hat{\vec{x}}_k^*\right)$ by COMPASS with different sampling schemes.

GO-POLARS sampling as in (4.24), for which ϕ_σ is adopted as the concentrating function and σ is set to π and $\pi/6$ respectively. Besides, the batch size of COMPASS, i.e., the number of solutions to be sampled in each iteration, is set to 1.

From the average $g\left(\vec{x}_k^*\right)$ drawn from 50 replications (Figure 4.7), we conclude that compared with CS, the hybridized GO-POLARS provides a higher convergent rate and the rate increases as the sampling concentrates to the gradient direction (denoted by smaller σ). However, how to select σ so as to achieve the highest convergent rate remains as an open issue for future study.

In addition, by a long run study we found it almost impossible for CS converge to the unique optimum, simply due to the reason that CS is designed intently for discrete problems while in continuous cases the search could be trapped in the region where solution cannot be improved on any coordinate directions. Thus, for COMPASS to be applied in solving continuous problems, GO-POLARS is one of the only choices.

Algorithm	Search Iterate (Neighborhood Structure)	Condition for Accepting $\hat{\vec{x}}_{\text{new}}$
GO-POLARS	$\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - a_k \left\| \vec{\nabla}\left(\hat{\vec{x}}_k\right) \right\| \vec{d}_k$ where $\vec{d}_k \sim \mathrm{O}^{p}_{\text{polar}}\left(\frac{\vec{\nabla}\left(\hat{\vec{x}}_k\right)}{\left\|\vec{\nabla}\left(\hat{\vec{x}}_k\right)\right\|}\right)$	$g\left(\hat{\vec{x}}_{\text{new}}\right) < g\left(\hat{\vec{x}}_k\right)$
Steepest descent (SD)	$\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - a_k \cdot \vec{\nabla}\left(\hat{\vec{x}}_k\right)$	$g\left(\hat{\vec{x}}_{\text{new}}\right) < g\left(\hat{\vec{x}}_k\right)$
Random Perturbation of Gradient	$\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - a_k \cdot \vec{\nabla}\left(\hat{\vec{x}}_k\right) + \lambda_k \vec{d}_k$ where $\vec{d}_k \sim \mathrm{N}\left(\vec{0}, I_p\right)$, $\lambda_k = \sqrt{\frac{100}{\log(k+1)}}$	$g\left(\hat{\vec{x}}_{\text{new}}\right) < g\left(\hat{\vec{x}}_k\right)$
Simulated Annealing (SAN)	$\hat{\vec{x}}_{\text{new}} \leftarrow \hat{\vec{x}}_k - a_k \left\| \vec{\nabla}\left(\hat{\vec{x}}_k\right) \right\| \vec{d}_k$ where $\vec{d}_k \sim \mathrm{U}^{p}_{\text{polar}}$	$g\left(\hat{\vec{x}}_{\text{new}}\right) < g\left(\hat{\vec{x}}_k\right)$ or $z < \exp\left(\frac{g\left(\hat{\vec{x}}_k\right) - g\left(\hat{\vec{x}}_{\text{new}}\right)}{T_k}\right)$ where $z \sim \mathrm{U}(0,1)$

Table 4.1: The overview of settings for testing algorithms.

Chapter 5

The Multi-Objective GO-POLARS and Gradient-Based Techniques for MSOP

In this chapter, we extend the GO-POLARS into the application for multi-objective problem. The main challenge we have encountered is that it is difficult to have a unified gradient as multiple objectives need to be considered.

Back to the motivation for designing gradient-based techniques, our main concern is to improvement the search efficiency. So instead of directly looking into the unified gradient, we need to clarify the indicator for the search improvement in a multi-objective circumstance. Then, with the unified gradient developed from the indicator, we are able to apply various gradient-based techniques including GO-POLARS.

5.1 Indicators for Pareto Set Improvement

As we are aiming to develop fast search algorithm that is able to identify a good Pareto set, it is important to find an indicator for the quality of the set. It is not a trivial problem, as there could be many ways to compare two different sets, especially when their elements are different from each others and both sets contain many non-dominated solutions even when elements in the other set are considered.

Although there are many types of metrics widely adopted in research problems (Chapter 2), in practice shortcoming occurs due to the fact that we cannot use a single value to uniquely identify the quality of the Pareto set, in terms of both distance and diversity, so as to track the improvement history. Besides, most of the metrics require that the real Pareto set is known which may not be true in many scenarios. Therefore, the DHV indicator (Bradstreet et al., 2008; Nebro et al., 2008; Zitzler et al., 2003) will become a better choice.

5.1.1 The Dominated Hyper-Volume (DHV)

The hyper-volume is a scalar metric that indicates how much solution space is dominated by a specific Pareto set, provided with an arbitrary worst case scenario serving as a finite boundary. Therefore, a large hyper-volume value implies good quality of Pareto set.

We can have an illustration in bi-objective case where the hyper-volume is the area dominated by Pareto solutions (Figure 5.1). Comparing the chart in quadrant (a) and (b), we clearly see that since solution 2 & 3 are dominated by solution 2' & 3' respectively, the Pareto set in (b) is superior than (a) which can be reflected by the larger DHV when the worst case scenario reference keeps consistent. Meanwhile, if we compare (a) and (c) it shows that the short coverage, i.e., missing of solution 3 & 4, can also be reflected by a decrement in DHV value. Other than these, (a) and (d) illustrate that with the same range of coverage, a dense Pareto set has larger DHV quantity as well.

Another advantage of DHV indicator is that it is not necessary to know the real Pareto set. Instead, we only need an arbitrary worst case point to bound the finite region. It is sufficient when we compare different algorithms in term of their improvement rate and relative achievements, i.e., DHV serves as a first order indicator rather than for the absolute quality. For example, in previous chapter, Figure 3.7 compares MO-COMPASS with random search using DHV indicator which can be interpreted as average cost savings under the specific scenario. In that case, we do not know the exact optimal Pareto set, but it is sufficient to compare the relative improvements between algorithms, in contrast with a worst case scenario at cost $12,000 and service level 0.0.

5.1.2 Hyper-Volume Calculation

The calculation of hyper-volume has addressed a lot of research interest. As mentioned in Chapter 2, HSO (While et al., 2006) and dimension-sweep

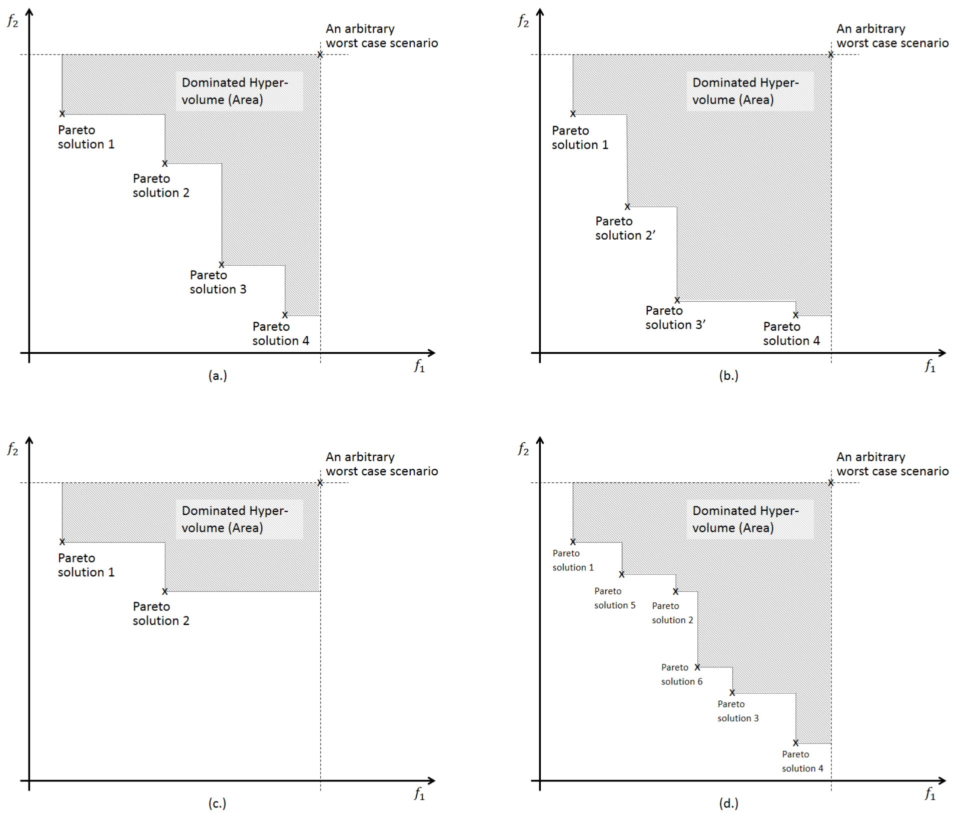


Figure 5.1: Illustration of DHV indicator ($\mathcal{H}$) for bi-objective Pareto sets

approach (Preparata and Shamos, 1985) provides a recursive method that derive a p dimension problem into several $p-1$ dimension sub-problems and eventually to the problem with $p=1$ where hyper-volume is just a line segment.

More importantly, the approach of dimension-sweep is helpful in our analysis of search efficiency where DHV is considered as the improvement indicator. Details will be discussed in Section 5.2.1.

5.1.3 Issues on Search Efficiency

With the concept introduced previously, we can quantify the search efficiency of a multi-objective problem as the marginal increment of hyper-volume noted as $\Delta\mathcal{H}$ in a unit time or search iteration, in which $\mathcal{H}$ refers to the dominated hyper-volume in contrast to a given worst scenario point.

Further notice that, in the framework proposed in Algorithm 1.1, pivot solution is selected one (or, two in some cases, e.g., NSGA-II) at a time from the Pareto set. Instead of considering $\Delta\mathcal{H}$ on the whole Pareto front, it is reasonable and more practical to analyze the partial-DHV due to individual Pareto $\vec{x}$, noted as $\mathcal{H}_{\vec{x}}$, and the marginal increment due to its improvement (or in other words its offspring). We note it as $\Delta\mathcal{H}_{\vec{x}}$, $\forall\vec{x}\in\Pi_k$.

An illustration is in Figure 5.2, where (b), (c) and (d) respectively shows three categories of offsprings, namely dominated, incomparable and dominating in term of its relation to the pivot solution. They are in contrast to the original case shown in (a). The dotted region indicates $\mathcal{H}_{\vec{x}}$ the darken rectangular indicates the $\Delta\mathcal{H}_{\vec{x}}$ (equivalent to $\Delta\mathcal{H}$) in each scenario.

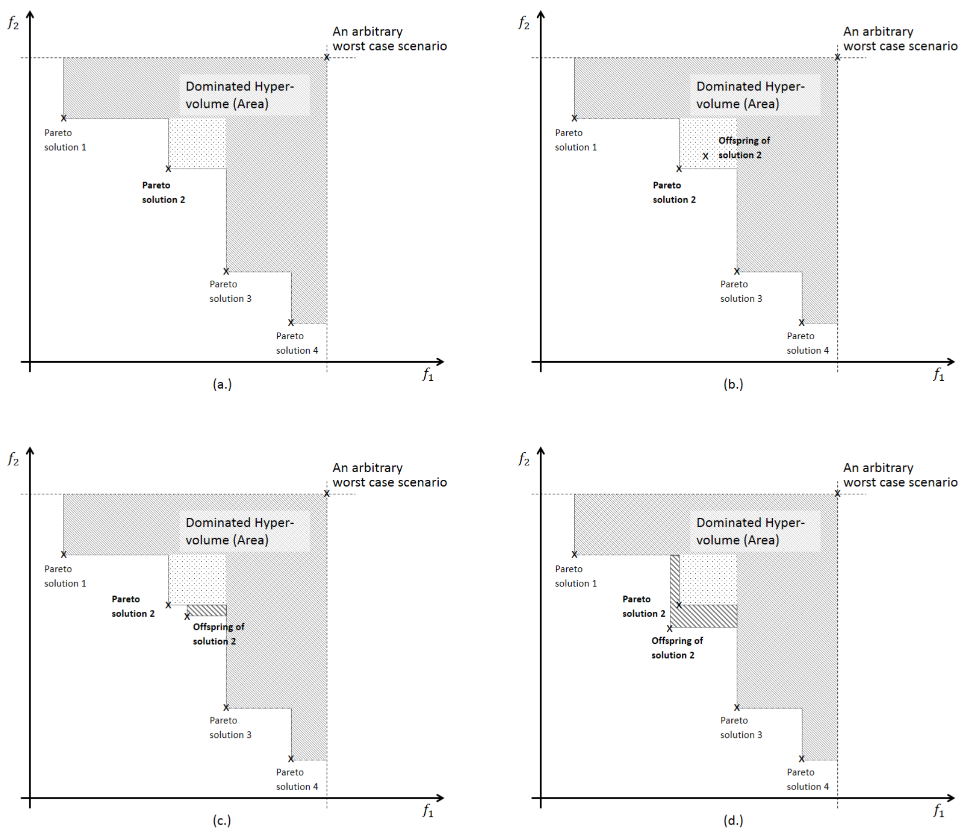


Figure 5.2: Illustration of Marginal-DHV ($\Delta\mathcal{H}$) due to offspring of pivot solution in a bi-objective Pareto set

Therefore, as equivalent to maximizing the search efficiency, our objective can be describe as in each iteration k,

$$\max_{\vec{x}\in\Pi_k} \mathbb{E}\left[\Delta\mathcal{H}_{\vec{x}}\right]. \tag{5.1}$$

As being discussed in Chapter 1, under the framework proposed we could approach the goal in (5.1) by adjusting the scheme of pivot selection for $\vec{x}$ and sampling in its neighborhood $\mathbb{N}(\vec{x})$.

5.1.4 Issues on Solution Spread and Diversity

Beside the issue of efficiency, in conducting a MO search we are also concerning about whether the obtained Pareto set is well spread over the solution space, as in that way the decision maker will have a more diverse candidate pool to select from. Usually, to judge a well-spread Pareto set, there are two criteria need to be taken care, namely whether the Pareto set covers the whole range of the objective domain, and whether they are evenly distributed on the range.

Although in literature there are a few metrics dealing with this issue, e.g., Spread indicator (Nebro et al., 2008) and the running performance metrics (Deb and Jain, 2002; Zeng, 2010), here we are more interested in identifying its relationship with hyper-volume indicator, so that we are able to find a singular metric for both search efficiency and solution spread and diversity.

The Figure 5.3 shows several different scenarios of the solutions spread in a two-dimension objective space. A well distributed Pareto front is illustrated in Quadrant (a), where we can see that all the partial-DHV $\mathcal{H}_{\vec{x}}$ are similar with each other. Intuitively, if we would like to further improve the Pareto set, each solution on the frontier should be treated equally so as to move them to the left bottom corner as a whole.

A different scenario is displayed by Quadrant (b), where Pareto solutions are not evenly distributed, since they are not covering the whole range of objective domain. It is obvious that all of them squeeze towards the upper boundary of g_1, or in other words, the lower end of g_2, while we are lack of solutions with larger value on g_2 but small value on g_1. In that case, the straightforward way to improve the Pareto could be sampling more solutions in the lacking space, thus the solution 1 should be given more attention than the others as it is the most frontier solution to the area. It can be reflected by the partial-DHV because $\mathcal{H}_{\vec{x}_1}$ is certainly the largest among all.

The similar situation can be found in Quadrant (d). In the scenario, the objective range is not well covered as the solutions are concentrating to the centre. Then if we look at the area of partial-DHV, the conclusion can be made that we should focus more on solution 1 and 4 so as to make the Pareto front complete.

A counter party scenario is shown by Quadrant (c) in which Pareto solutions have high dense at both ends. We can see that although they cover the whole objective range, the middle part is missing, i.e., there are few solutions with medium g_1 and medium g_2. In such a case, probably we can sample more solutions around solution 1 and solution 2 as they are the closest to the missing area. It also coincides with the conclusion we make

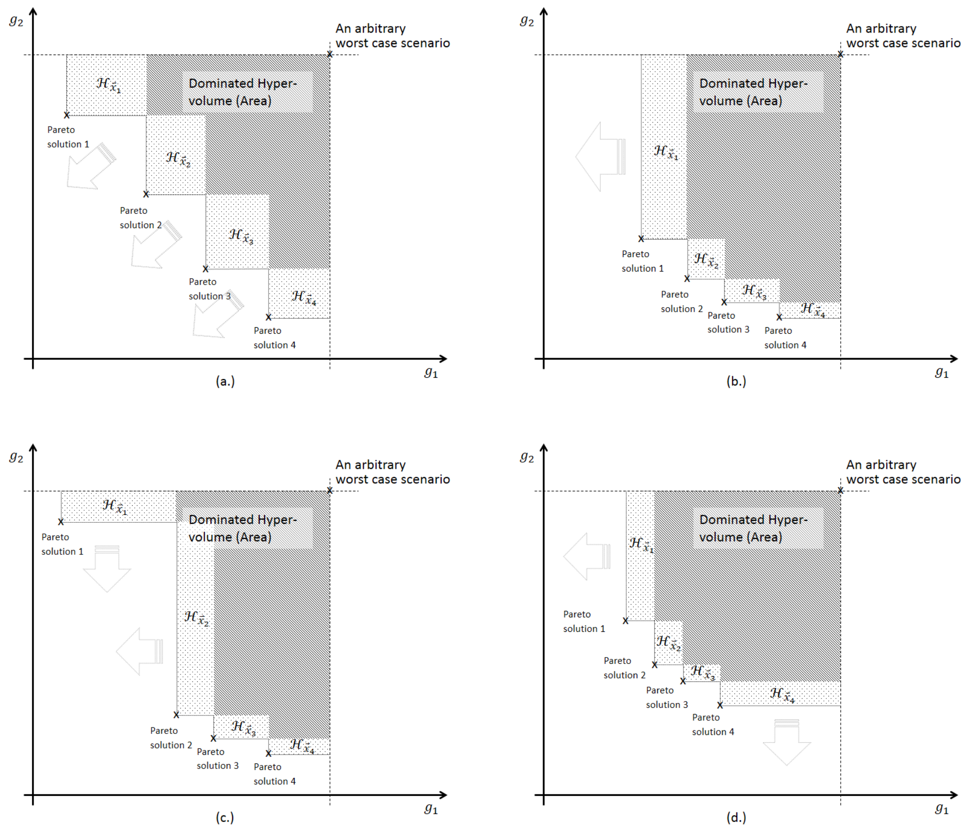

g_2
g_1
An arbitrary worst case scenario
Dominated Hyper-volume (Area)
$\mathcal{H}_{\vec{x}_1}$
$\mathcal{H}_{\vec{x}_2}$
$\mathcal{H}_{\vec{x}_3}$
$\mathcal{H}_{\vec{x}_4}$
Pareto solution 1
Pareto solution 2
Pareto solution 3
Pareto solution 4
(a.)
(b.)
(c.)
(d.)

Figure 5.3: Illustration of different Solution-Spread and the indication by $\mathcal{H}_{\vec{x}}$

according to the partial-DHV as $\mathcal{H}_{\vec{x}_1}$ and $\mathcal{H}_{\vec{x}_2}$ are higher than the others.

From the examples above we can conclude that the partial-DHV associating to any Pareto solution x reflects the density around x, as a high value implies low density and vice versa. For example, in part (a), all the four points have similar area of $\mathcal{H}_{\vec{x}}$, and it is obvious that the points are evenly distributed on the Pareto front; whereas in part (b), (c) and (d), the solution density is relatively low around points which have larger area of $\mathcal{H}_{\vec{x}}$, and in such cases solutions in the Pareto set are less diverse.

Hence, in order to obtain a well covered and evenly distributed Pareto set, it is equivalent to

$$\min \max_{\vec{x} \in \Pi_k} \mathcal{H}_{\vec{x}}.$$

We found that the concept can also be extend to any high dimensional objective space, the detailed technique is mentioned in Section 5.2.1. Applying the concept in the random search framework we proposed in Algorithm 1.1, we could sample more around solutions with higher value of $\mathcal{H}_{\vec{x}}$ rather than the ones with lower value.

Remark. The worst case scenario point becomes more importance in the context. Although arbitrary, it has to be representative enough to indicate the worst value of each objective. Otherwise, it limits the room of spread for the Pareto set.

5.1.5 Discussion on Terminating Conditions

Another advantage of the DHV indicator is that it makes easy for us to track the search history and obtain a singular sign of when shall we terminate the search because the result is satisfying or no more improvement can be expected.

A straightforward approach is to draw a relationship between the DHV value and the time, or in other form it can be number of visits or number of simulation evaluations. We can then apply regression method on the relationship so as to predict the DHV growing in the future. If the improvement is not worthy for spending additional computing budget, we shall terminate the search.

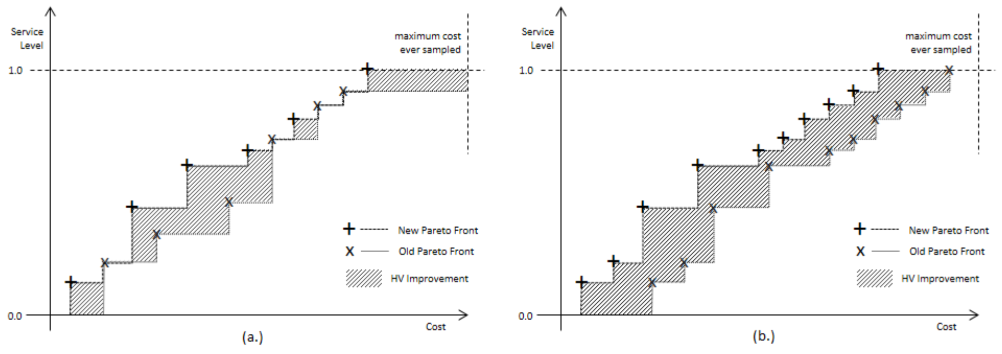


Figure 5.4: Improvement in DHV for a D-SIMSPAIR^{TM}scenario

An industrial application is implemented for D-SIMSPAIR^{TM}as in Section 3.5.4. In the application, the two objectives we are maximizing the service level and minimizing the total cost, which provides an even more valuable insight for the DHV indicator. As we can see from Part (b) of Figure 5.4, that showing a very special case where all Pareto solutions have same amount of cost saving but with service level unchanged, it is not difficult to be concluded that the change in the DHV value exactly measures the cost saving cost at each point as the service level is bounded between 0 and 1. To be more general, if we consider any arbitrary scenario such as in Part (a), the change in DHV can be intuitively interpreted as the average cost saving of Pareto solutions throughout the service level spectrum.

For illustration, the Figure 5.5 shows the improvement of non-dominated area in a D-SIMSPAIR^{TM}search run, i.e., the complement of DHV value mentioned previously, because we can interpret it as the averaged cost we have achieved so as to avoid the maximum cost we set arbitrarily.

The solid line in the figure records the reduction of the non-dominated area, i.e., the averaged cost. And with regression method, we can fit it into a power function described as

$$y = y_0 + \alpha \cdot x^{-\beta}$$

and identify the parameters as $y_0 = 24,261.77$, $\alpha = 241,824$ and $\beta = 0.57$. We can see that the R-Square value for the regression model is 98.31%, which indicates that the observations fit the model quite well.

Two important conclusions can be made from the model. Firstly, the y_0 set an asymptotic line to which the average cost can approach. In an idea situation, we may treat it as the lowest cost, or the optimal solution we are targeting at. Another indicator is

$$\Delta y = \alpha \left[x_0^{-\beta} - (x_0 + \Delta x)^{-\beta} \right]$$

that denotes the marginal cost reduction with extra computing budget Δx in addition to existing x_0. For example, if we would like to terminate the search once the the expected averaged cost reduction is less than \$100 in the next 1 hour, provided that each solution visit takes 1 second, we should terminate it at $16,673$ visits.

We should note that the regression parameters can be changed dynamically as more observations are collected. So, the prediction would be inaccurate when only a few observations is obtained, e.g., at the beginning of the search, or when there is a big jump just occurs. Thus, in practice we may also look at other criteria concerned by the problem owner.

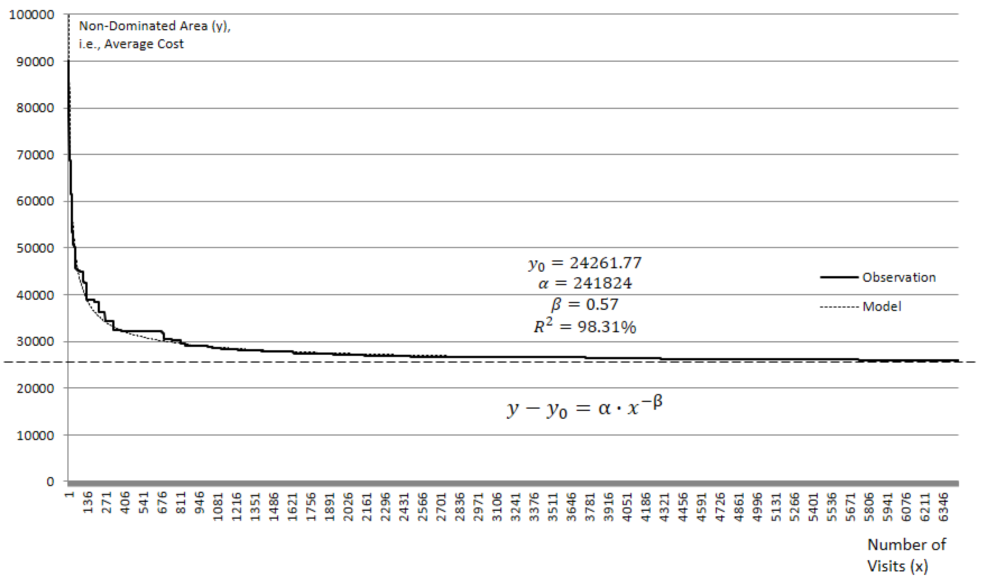


Figure 5.5: Reduction of averaged cost for the D-SIMSPAIR$^{\text{TM}}$scenario

5.2 A Unified Gradient for MSOP

Concerning the search efficiency and solutions spread and diversity, one typical approach is to consider the gradient of $\mathcal{H}_{\vec{x}}$ in term of $\vec{x}$, i.e.,

$$\vec{\nabla}_{\mathcal{H}_{\vec{x}}} = \frac{\partial \mathcal{H}_{\vec{x}}}{\partial \vec{x}}. \tag{5.2}$$

The direction of the gradient suggests one of the fastest ways to increase $\mathcal{H}_{\vec{x}}$ and at the same time the magnitude of the gradient relatively reflects the magnitude of $\mathcal{H}_{\vec{x}}$. Based on what we discussed in Section 5.1.3 and 5.1.4, the search algorithm for MSOP can be designed based on this singular gradient information.

However, $\vec{\nabla}_{\mathcal{H}_{\vec{x}}}$ can hardly be measured straightforward, as $\mathcal{H}_{\vec{x}}$ is a function defined on objective values $\vec{g}\,(\vec{x})$, but $\vec{x}$ is the parameter given in the decision space. Thus we need to consider a composite function on $\vec{x}$, i.e.,

$$\mathcal{H}_{\vec{x}} = \mathcal{H}_{\vec{g}} \circ \vec{g}\left(\vec{x}\right). \tag{5.3}$$

in which $\mathcal{H}_{\vec{g}}$ is the function of DHV in the objective space. Then, applying the chain rule, its gradient can be expressed as

$$\vec{\nabla}_{\mathcal{H}_{\vec{x}}} = \frac{\partial \mathcal{H}_{\vec{g}}}{\partial \vec{g}} \times \frac{\partial \vec{g}\left(\vec{x}\right)}{\partial \vec{x}}, \tag{5.4}$$

in which

$$\frac{\partial \vec{g}\left(\vec{x}\right)}{\partial \vec{x}} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \cdots & \frac{\partial g_1}{\partial x_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_h}{\partial x_1} & \cdots & \frac{\partial g_h}{\partial x_p} \end{bmatrix} \tag{5.5}$$

is a matrix and its element values could be measured or approximated from simulation evaluation using techniques such as FDSA (Blum, 1954b; Kiefer and Wolfowitz, 1952) or SPSA (Spall, 1999, 2003). Thus, we let

$$\vec{\lambda}_{\vec{g}} = \frac{\partial \mathcal{H}_{\vec{g}}}{\partial \vec{g}}, \tag{5.6}$$

and name it as the weighing vector as it balances gradients for different objectives. We find that for an MSOP, $\vec{\lambda}_{\vec{g}}$ becomes the key for obtaining a unified gradient $\vec{\nabla}_{\mathcal{H}_{\vec{x}}}$ according to (5.4).

5.2.1 The Objective Weighing Vector

The Quadrant (d) of Figure 5.2 provides an good insight for identifying $\vec{\lambda}_{\vec{g}}$ in a bi-objective case. Based on Equation (5.6), $\vec{\lambda}_{\vec{g}}$ is no more than a vector consists of elements showing the area increment of the dotted rectangular when the pivot solution (left-bottom point) is moved by an infinitesimal distance in each coordinate.

We demonstrate how $\vec{\lambda}_{\vec{g}}$ can be identified in bi-objective case by Figure 5.6. In the bi-objective case, it is obviously that the magnitude of two elements λ_1, λ_2 are respectively the width and height of rectangular, or more specifically

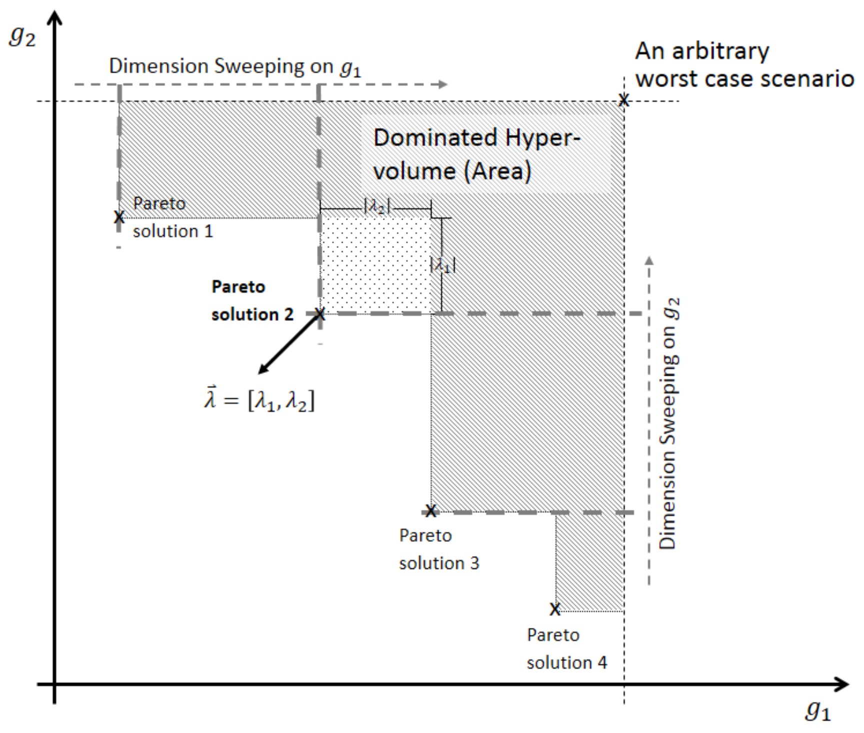
$$\lambda_1 = g_2^{(1)} - g_2^{(2)}, \lambda_2 = g_1^{(3)} - g_1^{(2)},$$

noted that $g_j^{(i)}$ is the g_j value for i^{th} solution.

While to have a more general elaboration, we need to apply the dimension sweeping technique, so as to extend it for any h-dimension scenario. In order to get λ_i we sweep a perpendicular hyper-plane on coordinate of g_i for all Pareto solutions from lower to higher end, and keep observing the intersected hyper-area (HA) with the dominated region.

The original dimension sweeping technique proposes that, the DHV can be calculated by integrating HA on the sweeping depth (Preparata and Shamos, 1985; While et al., 2006). But here, since we need to know the infinitesimal increment of DHV at specific point $\vec{g}$ by moving it on dimension i, we take the amount of HA increment when the sweeping hyper-plane just reach $\vec{g}$, i.e., the difference of HA at $\vec{g}$ compared with the HA at the previous point, and record it as λ_i. We do it for all $i \in \{1, \ldots, h\}$, so as to obtain $\vec{\lambda}_{\vec{g}}$.

It is worthy to notice that While et al. (2006) also provides the way to calculated HA at each point. Basically, it is part of the recursive process



g_2
Dimension Sweeping on g_1
An arbitrary worst case scenario
Dominated Hyper-volume (Area)
Pareto solution 1
$|\lambda_2|$
$|\lambda_1|$
Pareto solution 2
$\vec{\lambda} = [\lambda_1, \lambda_2]$
Dimension Sweeping on g_2
Pareto solution 3
Pareto solution 4
g_1

Figure 5.6: Identify $\vec{\lambda}_{\vec{g}}$ in a bi-objective Pareto set

for calculating DHV, as a h-dimensional hyper-area is indeed a $(h-1)$-dimensional hyper-volume.

5.3 Gradient-Based Techniques

With the unified gradient proposed through Section 5.2, we can apply gradient related techniques in designing search algorithms or improving existing ones. According to the general framework in Algorithm 1.1, the techniques can be applied in two critical stages, namely, the pivot selection and sampling of new solutions.

5.3.1 Weighed Pivot Selection

For pivot selection, a simple approach is to rank the importance of each Pareto solution according to the magnitude of its unified gradient because it indicates the potential of occurring better solution nearby. Then we can make the selection based on the probability proportional to it, we refer the method as weighed pivot selection (WPS).

5.3.2 Gradient-Oriented Coordinate Sampling

For sampling of new solutions, a straight forward example could be following the idea of steepest descend method, meaning that in every iteration sample a solution on the gradient direction of a pivot solution with a predetermined step-size. However, we should notice that MSOP is different from a single objective optimization problem as we are not aiming to find a unique optimal solution but a complete set of non-dominated solutions. So, in order to diversify the sampled population, we would like to inject certain noise into the sampling.

Based on the MO-COMPASS proposed in Chapter 3 that is suitable for discrete scenarios, below of this section illustrates an enhanced version by introducing the gradient-oriented coordinate sampling (GOCS) as the sampling scheme.

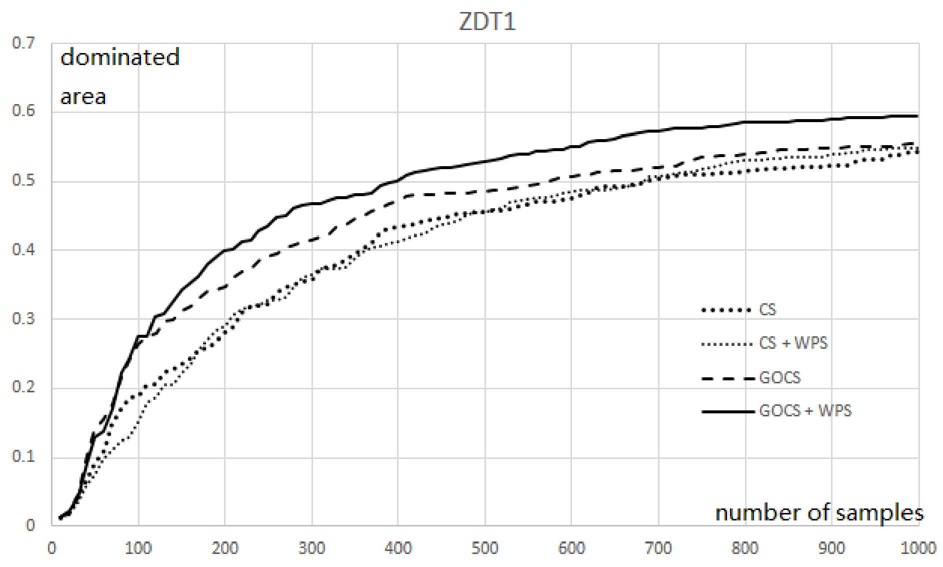
We introduce GOCS by comparing it with the CS as in Section 3.4. Generally, CS has two steps, namely (1) uniformly select a dimension, (2) find the range of values on the dimension in the most-promising-area and uniformly sample a point. The GOCS modifies CS in the sense that the dimension i is selected with probabilities proportional to the magnitude of gradient element, i.e., $|g_i|$; and the range is truncated at the pivot solution and only the part coincides with the sign of g_i is taken to be sampled. Because of the truncation, if the same candidate pool as for CS is to be maintained, we can simple make use of CS with a small portion of chance in addition to the GOCS procedure shown above.

Then we test the two techniques, namely WPS and GOCS, by applying them in MO-COMPASS for solving ZDT1 and ZDT2 as in Section 3.5.2. For both problems, we set the discretization levels to $10,000$ and dimensions to 30 so as to make the solution space relatively large. While for the search algorithm, batch size is set to 10 and 50 search runs are applied for observing the average dominated hyper volume versus number of solutions sampled.

From Figure 5.7, it is not difficult to conclude that both WPS and GOCS are able to make the search algorithm faster and the combination of the two techniques achieved the best performance among all. Given limited budget at $1,000$ samples, we have 9.56% DHV improvement with WPS + GOCS for ZDT1 and 7.08% for ZDT2.

However, we should note that the advantage of the gradient based technique is highly based on the problem nature. Usually, it works well for problem with a small number of local optima or when the global optimum is much superior than the others. In practice, the advantage will become more obvious when the problem scale is larger.

The trend can be shown by the D-SIMSPAIR$^{\text{TM}}$ application mentioned in 3.5.4, where we also tested WPS + GOCS for all the three parts with low,



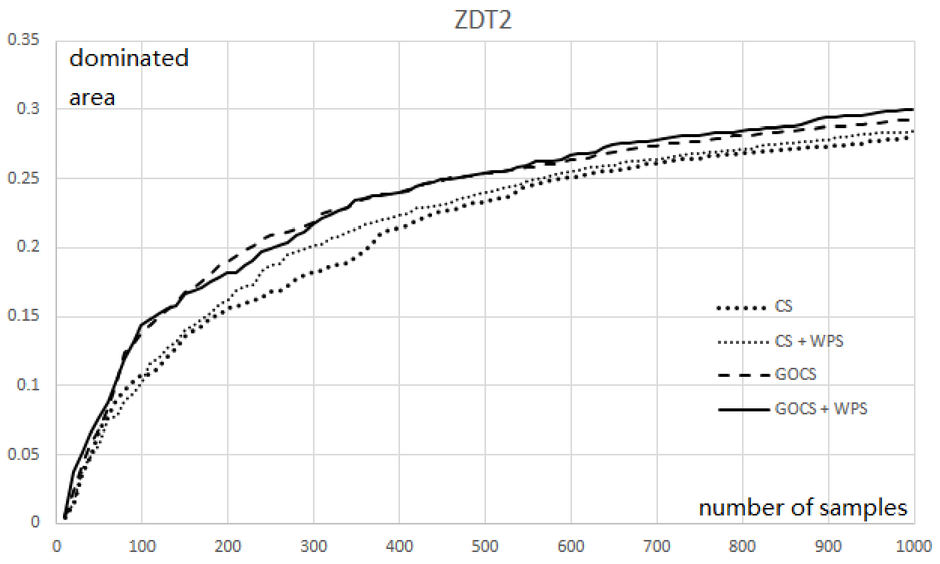


Figure 5.7: Improvement in Search Efficiency by WPS and GOCS.

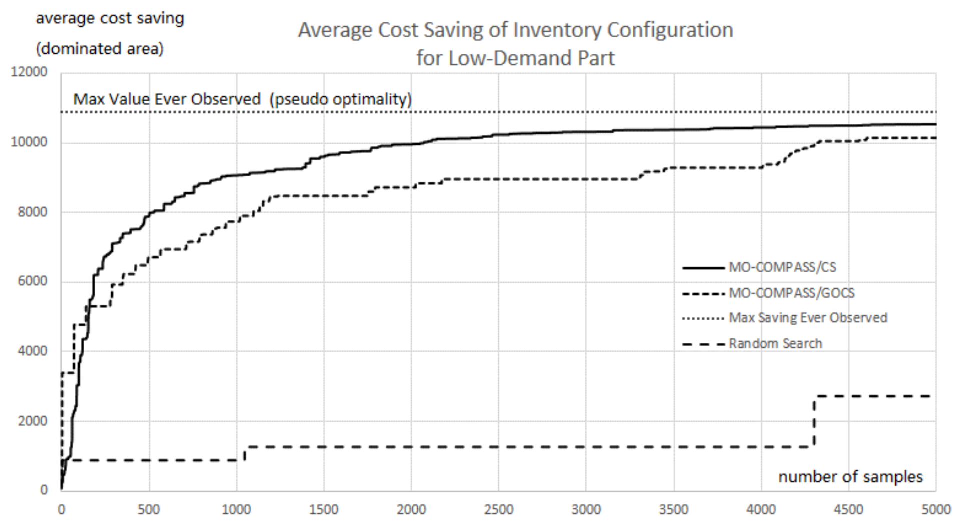


Figure 5.8: Gradient-Based Efficiency Improvement for D-SIMSPAIR^{TM} Low-Demand Case.

medium and high demand. The result is shown in Figure 5.8 to 5.10. For low demand part the improvement is negative as the search into gradient direction loses opportunity to browse more potential solution area; while for medium and high demand part, the improvement becomes more obvious as the problem scale increases.

Nevertheless, the GOCS is helpful in efficiently solving industrial problems as normally those with large scale are the most essential ones.

5.3.3 GO-POLARS Sampling

Alternatively, we can apply WPS with GO-POLARS sampling (GPS) as introduced in Section 4.5.2, in which the unified gradient is used as for the orientation.

The numerical examples are conducted for ZDT1-6 (Zitzler et al., 2000)

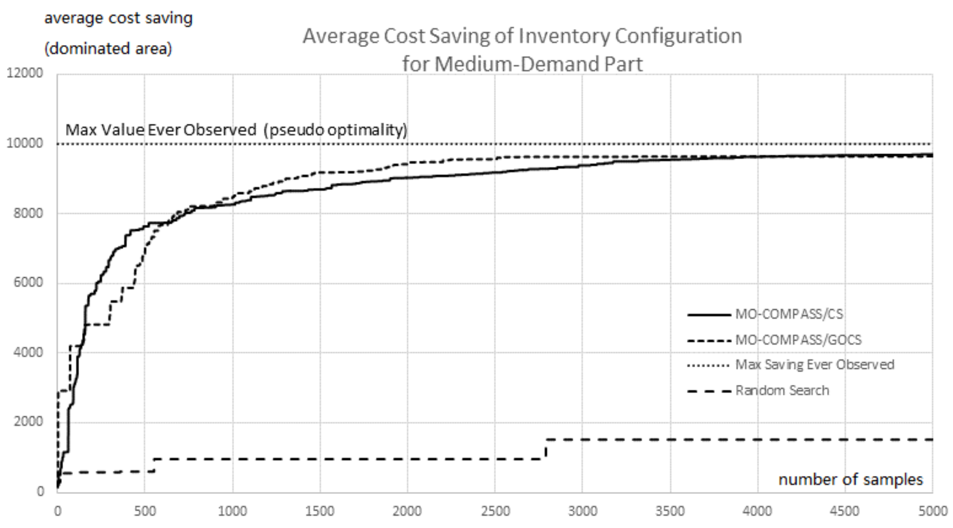


Figure 5.9: Gradient-Based Efficiency Improvement for D-SIMSPAIR^{TM} Medium-Demand Case.

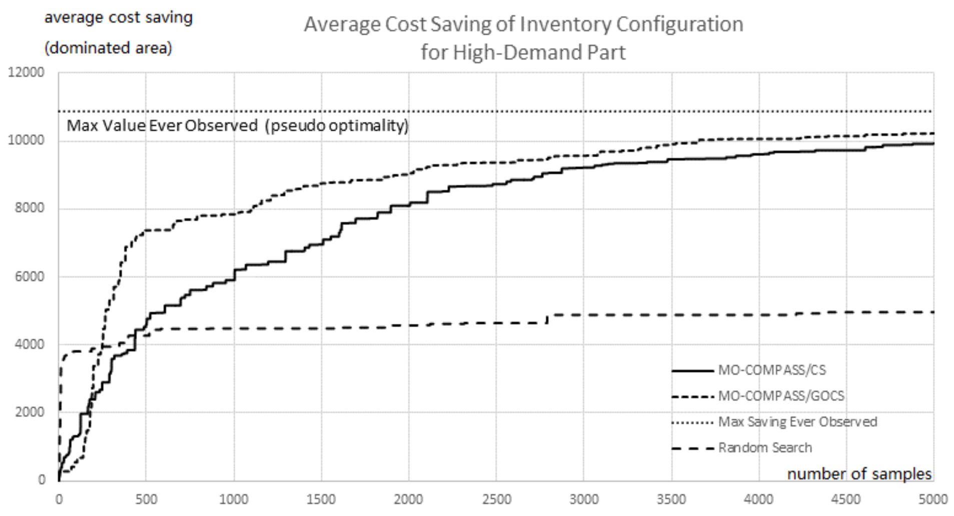


Figure 5.10: Gradient-Based Efficiency Improvement for D-SIMSPAIR^{TM} High-Demand Case.

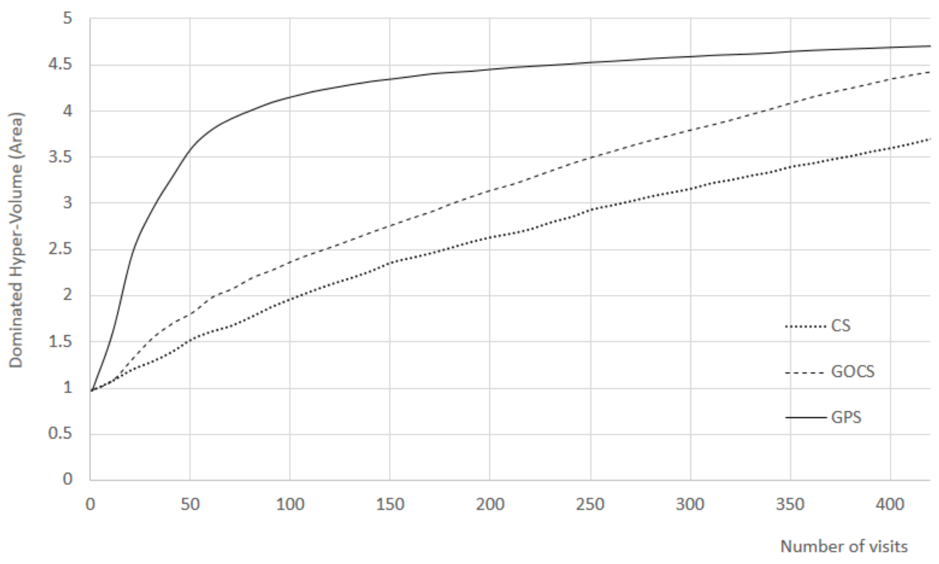


Figure 5.11: GO-POLARS Sampling in MO-COMPASS for ZDT1.

in discrete cases, where we set the dimension to 30, discretization level to 1,000. For the GO-POLARS as the sampling scheme, we let $\sigma = \pi/3$. And in order for GPS to be incorporated in MO-COMPASS, it works in a discrete manner, meaning that the sampled points are always rounded to the nearest integers for evaluation. The algorithm batch size is set to 10, while 50 search runs are observed and the averaged dominated hyper-volume, i.e., area in the bi-objective scenario, is recorded versus number of visits to samples. Given limited number of visits, the results are shown by Figure 5.11 to 5.15.

For ZDT1-3, we compare GPS with CS and GOCS. Whereas for ZDT4 and ZDT6, we only compared with CS as GOCS does not show to work well due to the problems nature.

It can be observed that, with the limited budget, the GPS improves

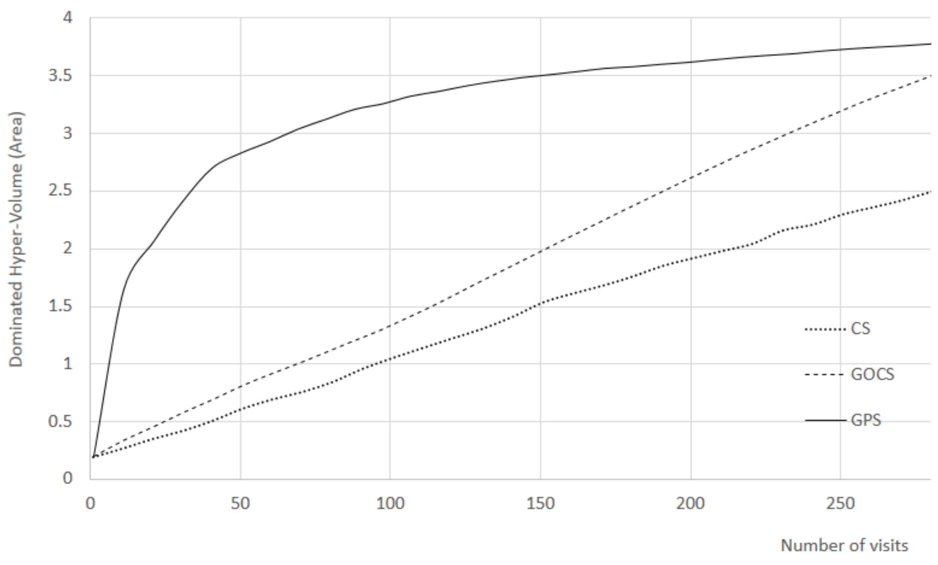


Figure 5.12: GO-POLARS Sampling in MO-COMPASS for ZDT2.

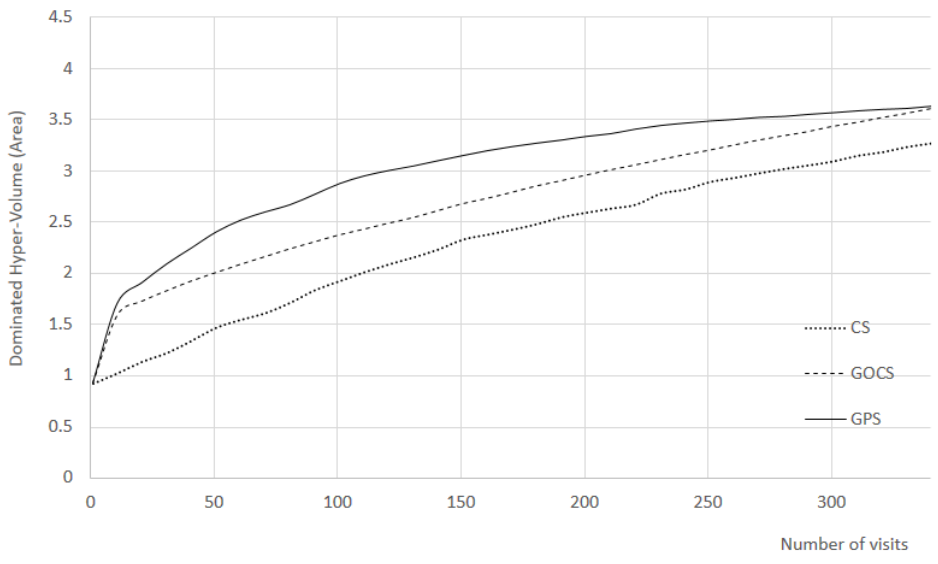


Figure 5.13: GO-POLARS Sampling in MO-COMPASS for ZDT3.

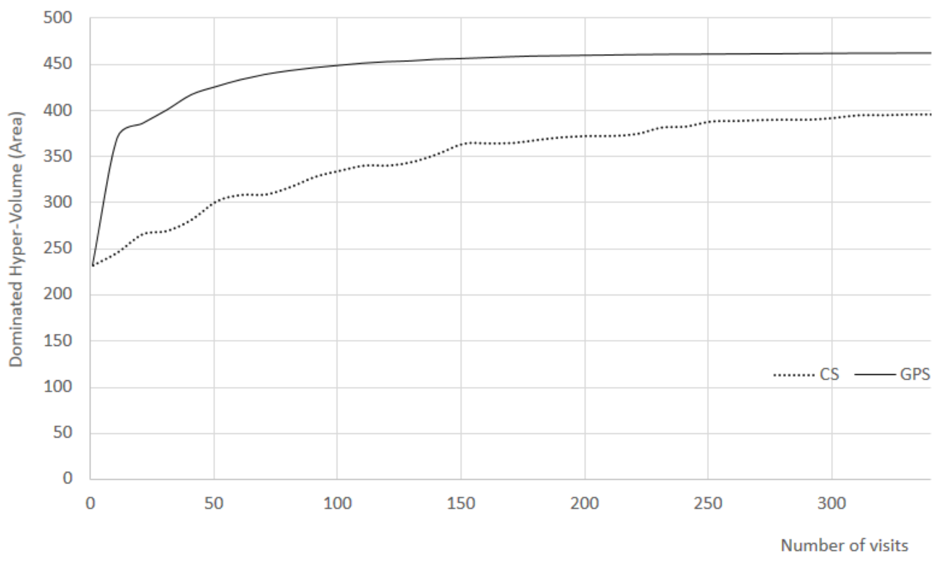


Figure 5.14: GO-POLARS Sampling in MO-COMPASS for ZDT4.

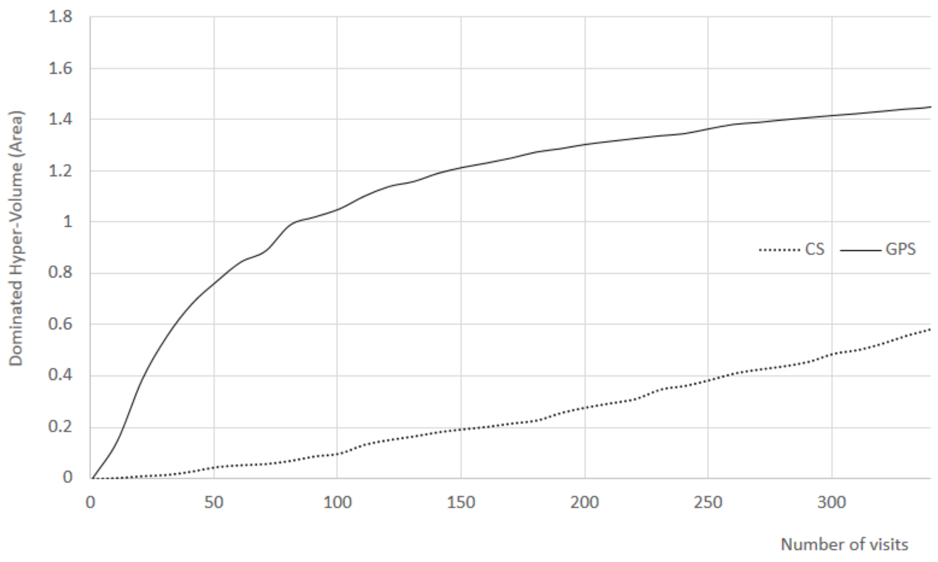


Figure 5.15: GO-POLARS Sampling in MO-COMPASS for ZDT6.

the DHV with the highest rate comparing with others. The result can be expected since with GO-POLARS we utilized more local information.

However, we should also notice that GPS may not have the best performance when more budget is assigned to the search. Two possible reasons can be addressed.

Firstly, in the numerical test we keep the same σ throughout the search, by which the advantages of GO-POLARS have not been fully taken as a small σ helps to increase the improvement rate but easy to be trapped in local optimum as search going on. So, in future we may try to vary the σ in different phase of the search, although the rule for doing so should be carefully designed.

Secondly, we are only testing the algorithm in discrete cases. Hence, the bias exists when we round the decision points to their nearest integers. The impact becomes more significant when the search going on because the MPA shrinks at the same time. In future study, we may relax the problem settings to continuous, or apply an independent GO-POLARS to an MSOP so as to get rid of the shrinking MPA.

Chapter 6

Conclusion

In this thesis, we proposed a generic framework for designing various types of MO search algorithms. Based on the framework, an advanced stochastic search algorithm MO-COMPASS for solving MDOvS was firstly developed with solid theoretical foundations and clearly specified SARs.

With rigorous mathematical proofs, we are confident that MO-COMPASS is able to strongly converge to an LPS with finite iterations. The results are further confirmed by numerical examples from both mathematical formulation and industry application. Compared with academic and industrial benchmarks, MO-COMPASS also demonstrates a more competitive capability of solving large scale problems with a high efficiency. It can be applied to various industrial problems and enhance the system performance effectively and efficiently.

To strengthen the search efficiency, especially to make it suitable for continuous problems and easy to control exploration of the search space, we continued to work on a brand new search algorithm concept, namely the GO-POLARS, after we reviewed two categories of search algorithms for optimization problems and suggested that incorporating randomness in utilizing gradient information will improve both gradient-based search and metaheuristics local search.

The GO-POLARS algorithm was based on a newly proposed hyper polar coordinate representation and associated random distributions. It has been shown to have the strong local convergence property and works well in numerical examples either independently or hybridizing with sophisticated stochastic algorithms.

In addition, we adopted and further improved the concept of dominated hyper volume that concerning both search efficiency and solutions spread and diversity, and proposed a method based on it to identify a unified gradient for MSOP. With this contribution, we could incorporate several gradient-based techniques, including GO-POLARS, into the MO search framework.

In future research, we could analyze the way to incorporate with more advanced SARs in the MO search algorithm, e.g., MO-COMPASS and etc., such as multi-objective optimal computing budget allocation (MOCBA) with indifference-zone (Teng et al., 2010). Additionally, we may study the global convergence property of MO-COMPASS and GO-POLARS so as to strengthen them for industrial use, as what has been achieved for single-objective case (Xu et al., 2010). Also, the behavior and restriction of those algorithms in practice can be further investigated.

We may also address the adjustment of σ for GO-POLARS and analyze how it affects the solutions quality versus search efficiency for different applications. The possibility to fit GO-POLARS into MO search framework so as to design more advanced algorithms can also be discussed. Besides, instead of gradient, other directional information based on the nature of respective problems can also be used to orient the polar random distribution. Then a large number of search and sampling algorithms can be developed based on the concept.

Overall, with the promising numerical results and the broad derivatives, we have plenty of reason to believe that we are opening a new era of stochastic search for multi-objective simulation optimization.

Appendix A

Proofs of Lemmas for MO-COMPASS

A.1 For Fully Constrained MDOvS

Proof of Lemma 3.1. Consider a modified version of Algorithm 3.1 as Algorithm A.1, in which we have $\vec{x} \in \Theta$ instead of $\vec{x} \in \mathcal{V}_k$ in Line 6. Correspondingly, for the SAR, Condition A.1 is applied in addition to Condition 3.1.

Condition A.1. *The SAR ensures that for each $\vec{y} \in \Theta \setminus \mathcal{V}_k$ at iteration k, new simulation evaluations are obtained with amount $N_k(\vec{y}) = \min_{\vec{x}\in\mathcal{V}_k} N_k(\vec{x})$, for calculating $\bar{\vec{G}}_k(\vec{y})$, which is to be discarded at the beginning of iteration $k+1$.*

Since the modification does not affect the way of generating $\mathcal{V}_k$, or

Algorithm A.1: MO-COMPASS for fully constrained DOvS (Modified)

```
1  Let iteration count $k = 0$, $\mathcal{C}_0 = \Theta$ and $\mathcal{V}_0 = \emptyset$ ;
2  while not terminating do
3      $k \leftarrow k + 1$ ;
4      sample a set solutions $X_k \leftarrow \{\vec{x}_1, \dots, \vec{x}_m\}$ from $\mathcal{C}_{k-1}$ ;
5      $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup X_k$ ;
6      forall the $\vec{x} \in \Theta$ do
7          apply SAR to determine $a_k(\vec{x})$ and thus $N_k(\vec{x})$;
8          collect $\vec{\bar{G}}_k$ based on simulation observations ;
9      identify $\hat{\Pi}_k$ as the observed Pareto set on $\mathcal{V}_k$ ;
10     construct $\mathcal{C}_k$ based on $\hat{\Pi}_k$ and $\mathcal{V}_k$ ;
```

allocating simulation to any $\vec{x} \in \mathcal{V}_k$, if we can prove

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \text{ i.o., } \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} = 0, \forall \epsilon \text{ s.t. } 0 < \epsilon < \epsilon_0 \tag{A.1}$$

for Algorithm A.1 with Assumption 3.2, Condition 3.1 and A.1, the same result follows for Algorithm 3.1 when only Assumption 3.2 and Condition 3.1 apply.

According to Algorithm A.1, $\mathcal{V}_k \subseteq \mathcal{V}_{k+1} \subseteq \Theta$. Since $|\Theta| < \infty$, we conclude that $\mathcal{V}_\infty = \bigcup_{k=0}^{\infty} \mathcal{V}_k$ which depends on the sequence $\{\mathcal{V}_1, \mathcal{V}_2, \dots\}$ must exists.

Therefore, for all $\vec{x} \in \mathcal{V}_\infty$, Condition 3.1 provides

$$\lim_{k \to \infty} N_k(\vec{x}) = +\infty. \tag{A.2}$$

Meanwhile, Condition A.1 ensures that (A.2) holds for all $\vec{x} \in \Theta \setminus \mathcal{V}_\infty$. It implies that the same result remains for all $\vec{x} \in \Theta$.

As $\forall k \geq 0, \mathcal{V}_k \subseteq \Theta$, for all ϵ such that $0 < \epsilon < \epsilon_0$,

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \text{ i.o., } \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \ldots, H\}\right\}$$

$$\leq \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \text{ i.o., } \exists \vec{x} \in \Theta, l \in \{1, \ldots, H\}\right\}. \qquad \text{(A.3)}$$

By the strong law of large numbers, Assumption 3.2 and (A.2) imply that

$$\text{(A.3)} \leq \sum_{\vec{x} \in \Theta, l \in \{1, \ldots, H\}} \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \text{ i.o.}\right\} = 0, \qquad \text{(A.4)}$$

which proves (A.1).

□

Proof of Lemma 3.2. For all $k \geq 0$, given $\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k$, the definition of $\mathcal{N}_k$ implies

$$\exists \vec{z} \in \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq 1. \qquad \text{(A.5)}$$

Besides, for all $\vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k$ we have $\vec{x} \neq \vec{y}$ since $\vec{x} \notin \mathcal{V}_k$ but $\vec{y} \in \mathcal{V}_k$. Thus, $\|\vec{x} - \vec{y}\| \geq 1$. Consider (A.5) as well, we then have

$$\exists \vec{z} \in \hat{\Pi}_k, \forall \vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq \|\vec{x} - \vec{y}\|. \qquad \text{(A.6)}$$

Because of definition of $\mathcal{C}_k$ as in (3.1), (A.6) implies $\vec{x} \in \mathcal{C}_k$. As the result, Algorithm 3.1 guarantees

$$\Pr\{\vec{x} \in \mathcal{V}_{k+1} \,|\, \vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k\} \geq \frac{1}{|\mathcal{C}_k|} \geq \frac{1}{|\Theta|} > 0.$$

$\square$

A.2 For Partially Constrained or Unconstrained MDOvS

Proof of Lemma 3.3. According to Algorithm 3.1, $\mathcal{V}_k \subseteq \mathcal{V}_{k+1} \subseteq \Theta$, $\forall k \geq 0$. Besides, $|\Theta| < \infty$ as the problem is fully constrained. Therefore, it must be true that $\Pr\{\mathcal{V}_k \neq \mathcal{V}_{k+1} \text{ i.o.}\} = 0$.

$\square$

Proof of Lemma 3.4. Define

$$\mathbb{B}_k \equiv \bigcap_{i=1}^{d} \left[\underline{b}_0^{(i)} - k\Delta^{(i)}, \bar{b}_0^{(i)} + k\Delta^{(i)}\right]. \tag{A.7}$$

So for all $k \geq 0$, we know that $\mathbb{B}_k$ is finite and deterministic, and $\mathcal{B}_k \subseteq \mathbb{B}_k \subset \Theta$. Then, we consider a modified version of Algorithm 3.2 as Algorithm A.2, in which we have $\vec{x} \in \mathbb{B}_k$ instead of $\vec{x} \in \mathcal{V}_k$ in Line 9. For the SAR, Condition A.2 is enforced beside Condition 3.2.

Condition A.2. *The SAR ensures that for each $\vec{y} \in \mathbb{B}_k \setminus \mathcal{V}_k$ at iteration k, new simulation evaluations are obtained with amount $N_k(\vec{y}) = r_k$ as in Condition 3.2, for calculating $\vec{\bar{G}}_k(\vec{y})$, which is to be discarded at the beginning of iteration $k+1$.*

Similar to the proof for Lemma 3.1, as the modification does not affect the way of generating $\mathcal{V}_k$, or allocating simulation to any $\vec{x} \in \mathcal{V}_k$, if we can

Algorithm A.2: MO-COMPASS for partially constrained or unconstrained MDOvS (Modified)

```
1  Initialize iteration count $k = 0$ ;
2  set $\underline{b}_0^{(i)}$ and $\bar{b}_0^{(i)}$ such that $\underline{b}_0^{(i)} < x_0^{(i)} < \bar{b}_0^{(i)}, \forall i \in \{1, \dots, d\}$ ;
3  construct $\mathcal{B}_0$ according to (3.8) ;
4  let $\mathcal{C}_0 = \Theta \cap \mathcal{B}_0$ and $\mathcal{V}_0 = \emptyset$ ;
5  while not terminating do
6      $k \leftarrow k + 1$ ;
7      sample a set solutions $X_k \leftarrow \{\vec{x}_1, \dots, \vec{x}_m\}$ from $\mathcal{C}_{k-1}$ ;
8      $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup X_k$ ;
9      forall the $\vec{x} \in \mathbb{B}_k$ do
10         apply SAR to determine $a_k(\vec{x})$ and thus $N_k(\vec{x})$;
11         collect $\bar{\vec{G}}_k$ based on simulation observations ;
12     identify $\hat{\Pi}_k$ as the observed Pareto set on $\mathcal{V}_k$ ;
13     construct $\mathcal{B}_k$ and thus $\mathcal{C}_k$ based on $\hat{\Pi}_k$ and $\mathcal{V}_k$ ;
```

prove

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon \text{ i.o., } \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} = 0, \forall \epsilon \text{ s.t. } 0 < \epsilon < \epsilon_0 \tag{A.8}$$

for Algorithm A.2 with Assumption 3.4, Condition 3.2 and A.2, the same result follows for Algorithm 3.2 when only Assumption 3.4 and Condition 3.2 apply.

$\forall k \geq 0$, as $\mathcal{V}_k \subseteq \mathbb{B}_k$, given any ϵ such that $0 < \epsilon < \epsilon_0$, we have

$$\begin{aligned}
&\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon, \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \dots, H\}\right\} \\
\leq\ &\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon, \exists \vec{x} \in \mathbb{B}_k, l \in \{1, \dots, H\}\right\} \\
\leq\ &\sum_{\vec{x} \in \mathbb{B}_k, l \in \{1, \dots, H\}} \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon\right\}.
\end{aligned} \tag{A.9}$$

By Condition 3.2 we know that there exists $K > 0$ such that for all $k \geq K$, we have $r_k \geq r^*$. In addition, Condition 3.2 and Condition A.2 ensure that for all $\vec{x}$ from either $\mathcal{V}_k$ or $\mathbb{B}_k \setminus \mathcal{V}_k$, it always holds that $N_k(\vec{x}) \geq r_k$. Therefore, according to Assumption 3.4, for all $k \geq K$ and all ϵ such that $0 < \epsilon < \epsilon_0$,

$$\Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon\right\} \leq \lambda(r_k, \epsilon), \forall \vec{x} \in \mathbb{B}_k, l \in \{1, \ldots, H\}, \quad \text{(A.10)}$$

and thus,

$$\text{(A.9)} \leq H|\mathbb{B}_k|\lambda(r_k, \epsilon) \leq H(b + \Delta k)^d \lambda(r_k, \epsilon) \quad \text{(A.11)}$$

because $|\mathbb{B}_k| \leq (b + \Delta k)^d$ where

$$b = \max_{1 \leq i \leq d} \left[\bar{b}_0^{(i)} - \underline{b}_0^{(i)} + 1\right]$$

and

$$\Delta = \max_{1 \leq i \leq d} \left[2\Delta^{(i)}\right].$$

Then, since Condition 3.2 also provides

$$\lim_{k \to \infty} k^{d+1} \lambda(r_k, \epsilon) = 0,$$

from (A.11) we have

$$\sum_{k=0}^{\infty} \Pr\left\{\left|\bar{G}_k^{(l)}(\vec{x}) - g^{(l)}(\vec{x})\right| > \epsilon, \exists \vec{x} \in \mathcal{V}_k, l \in \{1, \ldots, H\}\right\}$$

$$\leq K + H \sum_{k=K}^{\infty} (b + \Delta k)^d \lambda(r_k, \epsilon) < \infty, \tag{A.12}$$

which proves (A.8) by the first Borel-Cantelli lemma (Billingsley, 1995).

$\square$

Proof of Lemma 3.5. For all $k \geq 0$, given $\vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k$, by the same way as in the proof of Lemma 3.2, we have $\vec{x} \in \mathcal{C}_k$, noted that the definition for $\mathcal{C}_k$ is given in (3.9), and $\mathcal{N}_k \subseteq \mathcal{B}_k$ when $\Delta^{(i)} > 0$ for all i. Hence, according to the Algorithm 3.2

$$\Pr\{\vec{x} \in \mathcal{V}_{k+1} \,|\, \vec{x} \in \mathcal{N}_k \setminus \mathcal{V}_k\} \geq \frac{1}{|\mathcal{C}_k|}. \tag{A.13}$$

To prove (A.13) > 0 for all $k \geq 0$, we only need to show

$$|\mathcal{C}_k| < \infty, \forall k \geq 0, \text{ w.p.1.} \tag{A.14}$$

Thus, we define for all $\vec{z} \in \Theta$,

$$\mathcal{D}_k(\vec{z}) \equiv \left\{ \vec{x} \in \Theta \,\middle|\, \forall \vec{y} \in \mathcal{V}_k \setminus \hat{\Pi}_k, \|\vec{x} - \vec{z}\| \leq \|\vec{x} - \vec{y}\| \right\}.$$

Then, instead of (3.9), an alternative way to define $\mathcal{C}_k$ is $\mathcal{C}_k \equiv \mathcal{B}_k \cap \bigcup_{\vec{z} \in \hat{\Pi}_k} \mathcal{D}_k(\vec{z})$. It implies

$$\mathcal{C}_k \subseteq \mathcal{B}_k \subseteq \mathbb{B}_k \tag{A.15}$$

in which $\mathbb{B}_k$ is defined in (A.7), and

$$\mathcal{C}_k \subseteq \bigcup_{\vec{z}\in\hat{\Pi}_k} \mathcal{D}_k(\vec{z}). \tag{A.16}$$

In addition, Assumption 3.3 tells that for all k the true Pareto set on $\mathcal{V}_k$, that is Π_k^*, must be contained in Ω which is finite, i.e., $\Pi_k^* \subseteq \Omega$ and $|\Omega| < \infty$. Besides, Theorem 3.3 ensures $\Pr\left\{\hat{\Pi}_k \neq \Pi_k^* \text{ i.o.}\right\} = 0$, thus

$$\Pr\left\{\hat{\Pi}_k \not\subseteq \Omega \text{ i.o.}\right\} = 0. \tag{A.17}$$

So, w.p.1 there exists a set $A \subseteq \Omega$ such that $A = \mathcal{W}$ in which

$$\mathcal{W} \equiv \left\{\vec{z} \in \Theta \;\middle|\; \vec{z} \in \hat{\Pi}_k \text{ i.o.}\right\} \tag{A.18}$$

that depends on the sequence of $\mathcal{V}_k$, namely

$$\sum_{A\subseteq\Omega} \Pr\left\{A = \mathcal{W}\right\} = 1. \tag{A.19}$$

Condition on $A = \mathcal{W}$ according to (A.19), so we have

$$\begin{aligned} &\Pr\left\{\left|\bigcup_{\vec{z}\in\hat{\Pi}_k} \mathcal{D}_k(\vec{z})\right| = \infty \text{ i.o.}\right\} \leq \\ &\sum_{A\subseteq\Omega} \Pr\left\{\sum_{\vec{z}\in A} \left|\mathcal{D}_k(\vec{z})\right| = \infty \text{ i.o.} \;\middle|\; A = \mathcal{W}\right\}. \end{aligned} \tag{A.20}$$

Moreover, we notice that according to Algorithm 3.2, the procedure of sampling in and construct $\mathcal{C}_k$ can be decomposed into sub-procedures of sampling and updating $\mathcal{D}_k(\vec{z})$ for all $\vec{z} \in \hat{\Pi}_k$. As the sub-procedure does not make any difference compared to single-objective COMPASS, the same result follows Hong and Nelson (2006) that

$$\Pr\left\{|\mathcal{D}_k(\vec{z})| = \infty \text{ i.o.} \;\middle|\; \vec{z} \in \hat{\Pi}_k \text{ i.o.}\right\} = 0, \tag{A.21}$$

which implies (A.20) $= 0$. Therefore, consider (A.16) we then have

$$\Pr\{\ |\mathcal{C}_k| = \infty \text{ i.o.}\} = 0. \tag{A.22}$$

The result in (A.22) means that w.p.1 there exists $K > 0$ (depending on the sequence of $\mathcal{V}_k$) such that for all $k \geq K$, $|\mathcal{C}_k| < \infty$. Besides, according to (A.15), when $k < K$ we also have $|\mathcal{C}_k| < |\mathbb{B}_K| < \infty$. Thus, (A.14) is proven.

$\square$

Proof of Lemma 3.6. According to Algorithm 3.2, $\mathcal{V}_k \neq \mathcal{V}_{k+1}$ only if $|\mathcal{C}_k \setminus \mathcal{V}_k| > 0$, for which a necessary condition is $\left|\bigcup_{\vec{z}\in\hat{\Pi}_k} \mathcal{D}_k(\vec{z}) \setminus \mathcal{V}_k\right| > 0$ because of (A.16). So, we have

$$\Pr\{\mathcal{V}_k \neq \mathcal{V}_{k+1} \text{ i.o.}\} \leq \Pr\left\{\left|\bigcup_{\vec{z}\in\hat{\Pi}_k} \mathcal{D}_k(\vec{z}) \setminus \mathcal{V}_k\right| > 0 \text{ i.o.}\right\}. \tag{A.23}$$

If $\hat{\Pi}_k$ can be bounded by the set Ω after a finite number of iterations, i.e., when $\hat{\Pi}_k \not\subseteq \Omega$ i.o. is not true, we can condition (A.23) on the event

$\mathcal{W} = A$. Hence,

$$\begin{aligned}(\text{A.23}) &\leq \\ \sum_{A\subseteq\Omega} \Pr\left\{ \left|\bigcup_{\vec{z}\in A} \mathcal{D}_k(\vec{z}) \setminus \mathcal{V}_k\right| > 0 \text{ i.o.} \;\middle|\; \mathcal{W} = A \right\} &+ \Pr\left\{\hat{\Pi}_k \not\subseteq \Omega \text{ i.o.}\right\} \qquad (\text{A.24}) \\ &\leq \sum_{A\subseteq\Omega} \Pr\left\{ \sum_{\vec{z}\in A} \left|\mathcal{D}_k(\vec{z}) \setminus \mathcal{V}_k\right| > 0 \text{ i.o.} \;\middle|\; \mathcal{W} = A \right\}\end{aligned}$$

since (A.17) tells $\Pr\left\{\hat{\Pi}_k \not\subseteq \Omega \text{ i.o.}\right\} = 0$.

Consider the sub-procedure of sampling and updating $\mathcal{D}_k(\vec{z})$ for all $\vec{z} \in \hat{\Pi}_k$. Hong and Nelson (2006) shows that w.p.1 there exists $K > 0$ such that $|\mathcal{D}_k(\vec{z})| < \infty$ for all $k \geq K$. And according to Algorithm 3.2, all $\vec{x}$ sampled from $\mathcal{D}_k(\vec{z})$ are to be included in $\mathcal{V}_{k+1}$, so we have

$$\Pr\left\{ \left|\mathcal{D}_k(\vec{z}) \setminus \mathcal{V}_k\right| > 0 \text{ i.o.} \;\middle|\; \vec{z} \in \hat{\Pi}_k \text{ i.o.} \right\} = 0$$

which implies (A.24) $= 0$.

$\square$

Bibliography

Arfken, G. (1985). *Spherical Polar Coordinates.* Academic Press, Orlando, FL, 3 edition.

Billingsley, P. (1995). *Probability and Measure.* Wiley-Interscience, 3 edition.

Blum, J. R. (1954a). Approximation methods which converge with probability one. *Annals of Mathematical Statistics*, 25:382 – 386.

Blum, J. R. (1954b). Multidimensional stochastic approximation methods. *Annals of Mathematical Statistics*, 25:737 – 744.

Bradstreet, L., While, L., and Barone, L. (2008). A fast incremental hypervolume algorithm. *Evolutionary Computation, IEEE Transactions on*, 12(6):714–723.

Butler, J., Morrice, D. J., and Mullarkey, P. W. (2001). A multiple attribute utility theory approach to ranking and selection. *Management Science*, 47(6):800 – 816.

Czyzak, P. and Jaskiewicz, A. (1998). Pareto simulated annealing a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7:34 – 47.

Deb, K. (1999). Multi-objective genetic algorithms: problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205 – 230.

Deb, K. and Jain, S. (2002). Running performance metrics for evolutionary multi-objective optimizations. In *Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02), (Singapore)*, pages 13 – 20.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182 – 97.

Debye, P. (1909). Nherungsformeln fr die zylinderfunktionen fr groe werte des arguments und unbeschrnkt vernderliche werte des index. *Mathematische Annalen*, 67(4):535 – 558.

Fleischer, M. (2002). The measure of pareto optima: Applications to multi-objective metaheuristics. Technical report, Institute for Systems Research, University of Maryland.

Fonseca, C., Paquete, L., and Lopez-Ibanez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 1157 – 1163.

Gelfand, S. and Mitter, S. (1993). Metropolis-type annealing algorithms for global optimization in rd. *SIAM Journal on Control and Optimization*, 31(1):111 – 31.

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

He, D., Lee, L. H., Chen, C.-H., Fu, M. C., and Wasserkrug, S. (2010). Simulation optimization using the cross-entropy method with optimal computing budget allocation. *ACM Transactions on Modeling and Computer Simulation*, 20(1).

Hong, L. J. and Nelson, B. L. (2006). Discrete optimization via simulation using compass. *Operations Research*, 54(1):115 – 129.

Hong, L. J. and Nelson, B. L. (2007). A framework for locally convergent random-search algorithms for discrete optimization via simulation. *ACM Transactions on Modeling and Computer Simulation*, 17(4).

Hong, L. J., Nelson, B. L., and Xu, J. (2010). Speeding up compass for high-dimensional discrete optimization via simulation. *Operations Research Letters*, 38(6):550 – 555.

Kaplan, W. (1991). *Advanced Calculus.* Addison-Wesley, Redwood City, CA, 4 edition.

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

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671 – 680.

Knowles, J. and Corne, D. (1999). The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. In *Proceedings of the 1999 Congress on Evolutionary Computation*, page 98 105, Piscataway, NJ. IEEE Press.

Lee, L. H., Chew, E. P., and Teng, S. (2006). Integration of statistical selection with search mechanism for solving multi-objective simulation-optimization problems. In *Proceedings of the 38th Conference on Winter Simulation*, WSC '06, pages 294–303. Winter Simulation Conference.

Lee, L. H., Chew, E. P., Teng, S., and Chen, Y. (2008). Multi-objective simulation-based evolutionary algorithm for an aircraft spare parts allocation problem. *European Journal of Operational Research*, 189(2):476 – 491.

Lee, L. H., Chew, E. P., Teng, S., and Goldsman, D. (2010). Finding the non-dominated pareto set for multi-objective simulation models. *IIE Transactions*, 42(9):656 – 674.

Lendermann, P., Gan, B. P., Julka, N., Schirrmann, A., and Fromm, G. (2010). Unlocking value from component exchange contracts in aviation using simulation-based optimisation. In *Simulation Conference (WSC), Proceedings of the 2010 Winter*, pages 2034–2045.

Marler, R. T. and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26.

Maryak, J. and Chin, D. (2008). Global random optimization by simultaneous perturbation stochastic approximation. *Automatic Control, IEEE Transactions on*, 53(3):780 – 783.

Moré, J. J., Garbow, B. S., and Hillstrom, K. E. (1981). Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, 7(1):17 – 41.

Muller, M. E. (1959). A comparison of methods for generating normal deviates on digital computers. *J. ACM*, 6(3):376 – 383.

Nebro, A., Luna, F., Alba, E., Dorronsoro, B., Durillo, J., and Beham, A. (2008). Abyss: Adapting scatter search to multiobjective optimization. *Evolutionary Computation, IEEE Transactions on*, 12(4):439 – 457.

Nevel'son, M. and Khas'inskiĭ, R. (1973). *Stochastic Aproximation and Recursive Estimation*. Translations of Mathematical Monographs. American Mathematical Society, Providence, RI.

Pogu, M. and Souza De Cursi, J. (1994). Global optimization by random perturbation of the gradient method with a fixed parameter. *Journal of Global Optimization*, 5:159 – 180.

Preparata, F. P. and Shamos, M. I. (1985). *Computational Geometry*. Texts & Monographs in Computer Science. Springer-Verlag.

Radziukyniene, I. and Zilinskas, A. (2008). Evolutionary methods for multiobjective portfolio optimization. In *Proceedings of the World Congress on Engineering 2008*, volume 2, London, U.K.

Rosenbrock, H. H. (1960). An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175 – 184.

Schwarz, K. (2011). Darts, dice, and coins: Sampling from a discrete distribution. KeithSchwarz.com: `http://www.keithschwarz.com/darts-dice-coins/`. [Retrieved 28-March-2012].

Shi, L. and Olafsson, S. (2000). Nested partitions method for global optimization. *Operations Research*, 48(3):390 – 407.

Spall, J. C. (1999). Overview of the simultaneous perturbation method for efficient optimization. *American Society of Civil Engineers - Task Committee Reports*, pages 141–154.

Spall, J. C. (2003). *Introduction to stochastic search and optimization: estimation, simulation, and control.* John Wiley & Sons, Inc.

Spall, J. C. and Cristion, J. A. (1994). Nonlinear adaptive control using neural networks: estimation with a smoothed form of simultaneous perturbation gradient approximation. volume 3, pages 2560 – 2564, Baltimore, MD, USA.

Srinivas, N. and Deb, K. (1995). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2:221 – 248.

Swisher, J., Jacobson, S., and Yucesan, E. (2003). Discrete-event simulation optimization using ranking, selection, and multiple comparison procedures: a survey. *ACM Transactions on Modeling and Computer Simulation*, 13(2):134 – 54.

Teng, S., Lee, L. H., and Chew, E. P. (2010). Integration of indifference-zone with multi-objective computing budget allocation. *European Journal of Operational Research*, 203(2):419 – 429.

Veldhuizen, D. A. V. and Lamont, G. B. (1998). Multiobjective evolutionary algorithm research: A history and analysis. Technical report, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AF-B, OH 45433-7765.

Vose, M. (1991). A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering*, 17(9):972 – 975.

Walton, J. (1967). Tensor calculations on computer: appendix. *Communications of the ACM*, 10(3):183 – 185.

Weisstein, E. W. (2005). Spherical coordinates. MathWorld: http://mathworld.wolfram.com/SphericalCoordinates.html. [Retrieved 14-March-2012].

Weisstein, E. W. (2009). Polar coordinates. MathWorld: http://mathworld.wolfram.com/PolarCoordinates.html. [Retrieved 14-March-2012].

While, L., Hingston, P., Barone, L., and Huband, S. (2006). A faster algorithm for calculating hypervolume. *Evolutionary Computation, IEEE Transactions on*, 10(1):29 – 38.

Xu, J., Nelson, B., and Hong, L. (2010). Industrial strength compass: A comprehensive algorithm and software for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation*, 20(1):3 (29 pp.).

Yin, G., Rudolph, G., and Schwefel, H. P. (1995). Analyzing the (1, λ) evolution strategy via stochastic approximation methods. *Evolutionary Computation*, 3(4):473 – 489.

Zeng, Fanchao ; Low, M. Y. H. D. J. Z. S. C. W. (2010). Self-adaptive mechanism for multi-objective evolutionary algorithms. In *Proceedings of the International MultiConference of Engineers and Computer Scientists 2010, IMECS 2010*, pages 7 – 12, Kowloon, Hong kong.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

Zitzler, E., Deb, K., and Thiele, L. (Summer 2000). Comparison of multiobjective evolutionary algorithms: empirical results. *Evolutionary Computation*, 8(2):173 – 95.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *Evolutionary Computation, IEEE Transactions on*, 7(2):117–132.