Stochastic Optimization for Feasibility Determination:

An Application to Water Pump Operation in Water Distribution Network

by

Yi-An Tsai

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2018 by the
Graduate Supervisory Committee:

Giulia Pedrielli, Chair
Pitu Mirchandani
Giuseppe Mascaro
Zelda Zabinsky
Antonio Candelieri

ARIZONA STATE UNIVERSITY

May 2018

ABSTRACT

The energy consumption by public drinking water and wastewater utilities represent up to 30%-40% of a municipality energy bill. The largest energy consumption is used to operate motors for pumping. As a result, the engineering and control community develop the Variable Speed Pumps (VSPs) which allow for regulating valves in the network instead of the traditional binary ON/OFF pumps. Potentially, VSPs save up to 90% of annual energy cost compared to the binary pump. The control problem has been tackled in the literature as "Pump Scheduling Optimization" (PSO) with a main focus on the cost minimization. Nonetheless, engineering literature is mostly concerned with the problem of understanding "healthy working conditions" (e.g., leakages, breakages) for a water infrastructure rather than the costs. This is very critical because if we operate a network under stress, it may satisfy the demand at present but will likely hinder network functionality in the future.

This research addresses the problem of analyzing working conditions of large water systems by means of a detailed hydraulic simulation model (e.g., EPANet) to gain insights into feasibility with respect to pressure, tank level, etc. This work presents a new framework called Feasible Set Approximation – Probabilistic Branch and Bound (FSA-PBnB) for the definition and determination of feasible solutions in terms of pumps regulation. We propose the concept of feasibility distance, which is measured as the distance of the current solution from the feasibility frontier to estimate the distribution of the feasibility values across the solution space. Based on this estimate, pruning the infeasible regions and maintaining the feasible regions are proposed to identify the desired feasible solutions. We test the proposed algorithm with both theoretical and real water

networks. The results demonstrate that FSA-PBnB has the capability to identify the feasibility profile in an efficient way. Additionally, with the feasibility distance, we can understand the quality of sub-region in terms of feasibility.

The present work provides a basic feasibility determination framework on the low dimension problems. When FSA-PBnB extends to large scale constraint optimization problems, a more intelligent sampling method may be developed to further reduce the computational effort.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my committee chair, Dr. Giulia Pedrielli, for her wise patience, patience and encouragement over last year. Dr. Pedrielli helps me grow from nothing to mature. She gave me many opportunities to present my research. It was a pleasure to work under her supervision.

In addition, I would like to extend my thanks to other professors and students who are involved in our research team: Dr. Zelda Zabinsky, Dr. Antonio Candelieri, Dr. Hao Huang, and Riccardo Perego.

I would like to sincerely thanks to all my committee members, Dr. Pitu Mirchandani and Dr. Giuseppe Mascaro, Dr. Zelda Zabinsky and Dr. Antonio Candelieri for their time, suggestions, and encouragement.

I would like to thank to Chia-Yu Hsu, his support and encouragement make me motivated.

Finally, I would like to thank my family for their love and being my biggest supporters.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

1 CHAPTER 1: INTRODUCTION

This study addresses the problem of efficiently and effectively assessing the quality and safety of working conditions of complex water distribution networks. In view of this goal, we propose an innovative feasibility determination approach which defines metrics and exploits them in order to identify acceptable operational conditions for pumps. In perspective, this characterization can be used as input information to identify cost-optimal operational conditions.

In this chapter, we present the background (section 1.1), an overview of the problem of analyzing and controlling complex water distribution networks. A brief introduction to the proposed methodology is provided in section 1.2. Finally, section 1.3 presents the main structure of the thesis.

1.1 Problem Relevance

An example of real network that refers to the urban area of Milan is presented in Figure 1-1 [1]. The large size of the network consists 118,950 pipelines, 26 different pump stations with 95 pumps, and 33 storage tanks. The Company Metropolitana Milanese (MM), managing the WDN, faces approximately 16,000,000 euros of energy costs, 45% due to pumping operations in the distribution network [2].

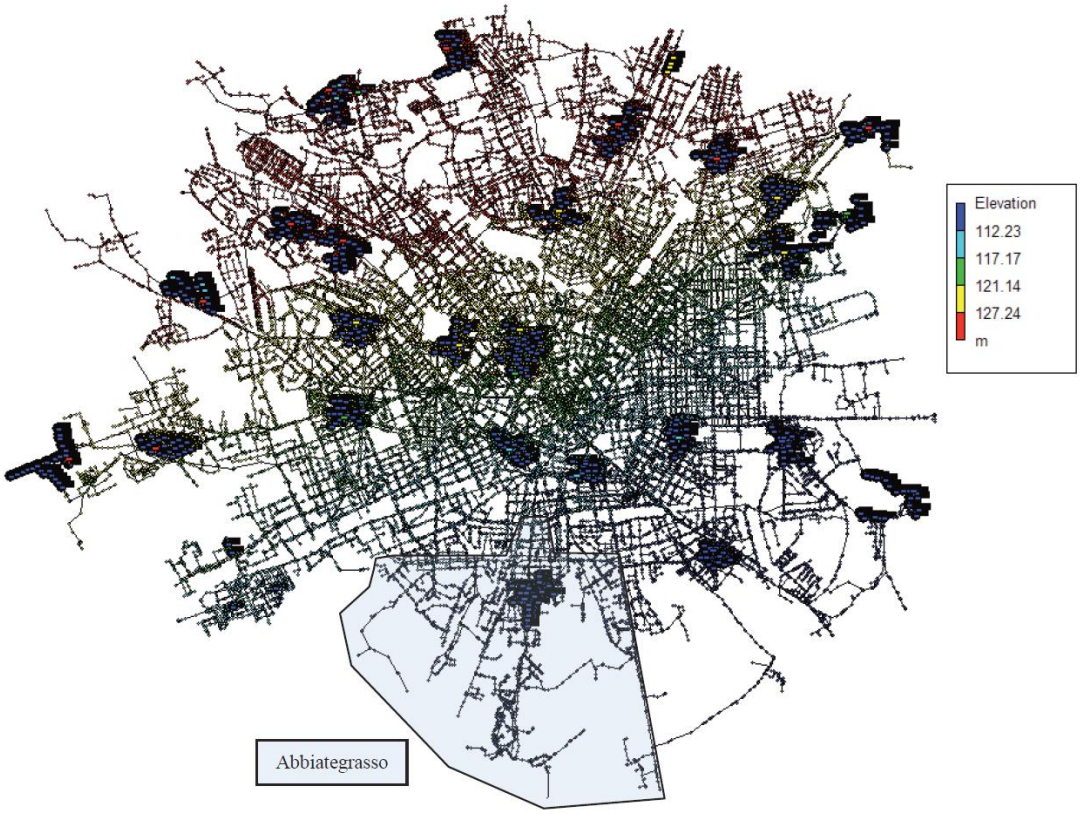

Figure 1-1 Milano WDN Model [1]

A recent report by Copeland and Carter [3], reveals that energy consumption by public drinking water and wastewater utilities, which are primarily owned, and operated, by local governments, can represent 30%-40% of a municipality's energy bill. How to manage energy costs in water distribution systems is an important and increasingly pressing challenge. How can we optimize the behavior of water networks? Looking closer into the main sources of consumption, pumping is responsible for 80% of the overall cost. Hence, controlling pumps in WDNs becomes one of the most crucial aspects to ensure the satisfaction of supply service in terms of quality and quantity of water and achieve performance goal [4].

In this thesis, we face the problem of hourly regulating the pumps in the distribution and transmission network. Such regulation has to be performed in a way that yields a flow able to respond the customer demand, while guaranteeing safe working conditions over the WDN. This problem has been deeply investigated by the research and engineering communities under three main perspectives: (1) Energy Cost Reduction, (2) Pump Scheduling Optimization, (3) Network Resiliency. While the proposed approaches in each of these areas provide contribution in the direction of maximizing the service level of the network while controlling the energy costs, fewer works look into the problem from the perspective of feasibility/safety of the operational conditions of the system. But how do we define "safe working conditions", and why are they important? In this regard, in a complex hydraulic system, there are several physical constraints to consider when operating water pumps. These constraints define the working conditions, feasibility, of the system and they are typically too complicated to be formulated in closed form. Instead, simulation is required to obtain point estimates of such measures (several simulators have been proposed in the literature such as EPANet, Finesse, H2Onet, and Water CAD [7, 8, 9, 10]). The importance of feasibility analysis comes from the fact that minimum cost solutions tend to be "stressful for the network": when the cost-optimal solution is close to the boundary of the feasible set, the optimal pump operation may lead to network malfunction (e.g., breakages, leakages [4]). For example, if the pressure at a pump is close to its upper bound of feasibility, while this will result in satisfactory customer service, it may also lead to leakages and breakages. We argue that having more insights into feasibility of network working conditions will improve and unify the analysis and control of WDNs to achieve

performance goals in terms of energy consumption and efficient pump scheduling in the networks.

In light of this, we bridge the gap between design, scheduling, and control of WDNs, by proposing a new feasibility driven perspective, coupled with simulation-optimization. This approach can provide insights to practitioners regarding energy costs and the working conditions under the specific pump settings. In order to do so, we decompose the Pump Scheduling Optimization (PSO) problem into two stages: (1) the first stage is to determine the feasible set of pump speed settings, and the (2) second is the energy cost optimization and pump scheduling control. This work works in providing methods to tackle problem in (1).

1.2 Solution Approach

Feasibility determination is the main issue we would like to address in the thesis. The whole process to find a safe and optimal pump scheduling setting is described in Figure 1-2.

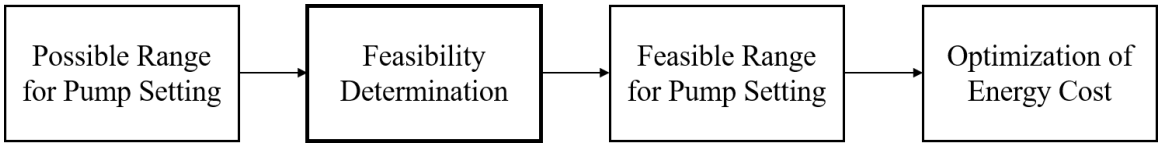

Figure 1-2 Process of Pump Scheduling Optimization

As previously mentioned, there are two main different types of pumps, one is the single speed ON/OFF binary pump and the other is the variable speed pump (VSP). The pump speed can be characterized by its pump curve (the combination of heads and flows that the pump can produce), and any VSP can be programmed to run at many different rotational speed settings, which shifts the position and shape of the pump curve. Specifically, it is possible to regulate the pump speed by controlling the relative speed parameter, e.g. if running the pump at half speed, the relative setting is 0.5 [7].

This work considers networks with VSP pumps and, to approach the feasibility determination, the approach uses the outputs from WDN computational models such as: pressure, load, tank level, and demand. These measures are time series characterizing the several locations of the simulated system.

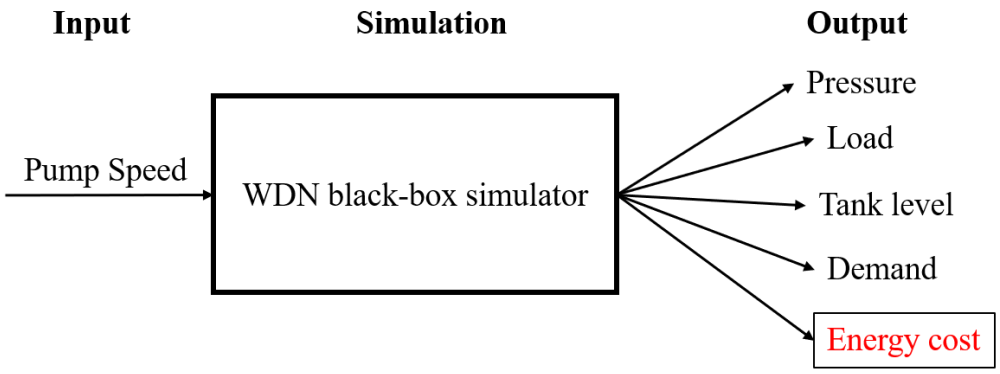

Figure 1-3 WDN Simulation Process

Figure 1-3 depicts a schematic of the WDN simulation system. The simulator requires, as input, the set of proposed pump speeds for each installed pump in the network for each hour of the daily schedule, and it returns, as output, the measures we would like to consider in feasibility determination and energy cost optimization: the pressure at each junction in the system, the load at each pipe, the tank level, the supply at each demand node and the energy cost. As previously mentioned, the energy cost is the metric to be minimized, which is used only during the optimality phase, while the pressure, load, tank level, and demand are the measures characterizing the feasibility of the proposed solution in terms of pump regulation, i.e., they describe the working conditions of the water network for that energy cost.

1.3 Thesis Structure

In chapter 2, simulation of water networks as well as several optimization techniques dealing with Pump Scheduling Optimization and the related results are presented, also, the Probabilistic Branch and Bound algorithm, at the basis of the method proposed in this thesis is investigated along with its challenges when applied to our case [12]. Considering the lack of the current literature, in chapter 0, we propose our innovative algorithm. In

particular, this work results into a completely novel “feasibility phase” in order to apply partitioning to feasibility determination. The new proposed algorithm is called Feasible Set Approximation Probabilistic Branch and Bound (FSA-PBnB). The algorithm is analyzed through a series of preliminary experiments with theoretical nonlinear functions and different numbers of black box constraints. In chapter 4, we apply FSA-PBnB over two different WDNs modeled using EPANet and the result of the approximate feasible set is discussed. Finally, Chapter 5 draws the main achievements along with the open questions for future research.

2 CHAPTER 2: LITERATURE REVIEW

The literature related to WDNs is vast and we focused onto three main areas: (1) Simulation of WDNs, (2) Pump Scheduling Optimization, (3) Probabilistic Branch and Bound algorithms. In section 2.1, we introduce in detail an open-source hydraulic simulation software, EPANet. Several algorithms which have been proposed to address the Pump Scheduling Optimization are presented in section 2.2. In section 2.3, we review a stochastic optimization algorithm, Probabilistic Branch and Bound, which is at the basis of our new proposed feasibility determination approach.

2.1 EPANet

In this work, we adopted a state-of-the-art hydraulic simulator, EPANet 2.0 [7], an open-source freeware that is widely adopted in the literature, to describe the detailed workings of a complex hydraulic system. The main network components considered within EPANet are pipes, nodes (junctions), pumps, valves and storage tanks or reservoirs.

More specifically, the (minimum) set of data needed to create a model of a WDN consists of: (a) coordinates and altimetry for each node; (b) demand profile (i.e., a "pattern") over the simulation time horizon of the water request for each consumption node in the network; (c) size and shape of each tank, with an initial level; (d) connectivity of the WDN (links connecting nodes); (e) length, diameter and roughness of each pipe; (f) efficiency curve of each pump (which can be on/off or VSPs); and (g) energy tariff over the simulation period. A simple example of a WDN model (named Net1) is presented in Figure 2-1. This network has one variable speed pump, one reservoir, one storage tank, and nine nodes. Examples of the input to EPANet for hourly demand pattern and pump curve are illustrated in Figure 2-2 and Figure 2-3, respectively.

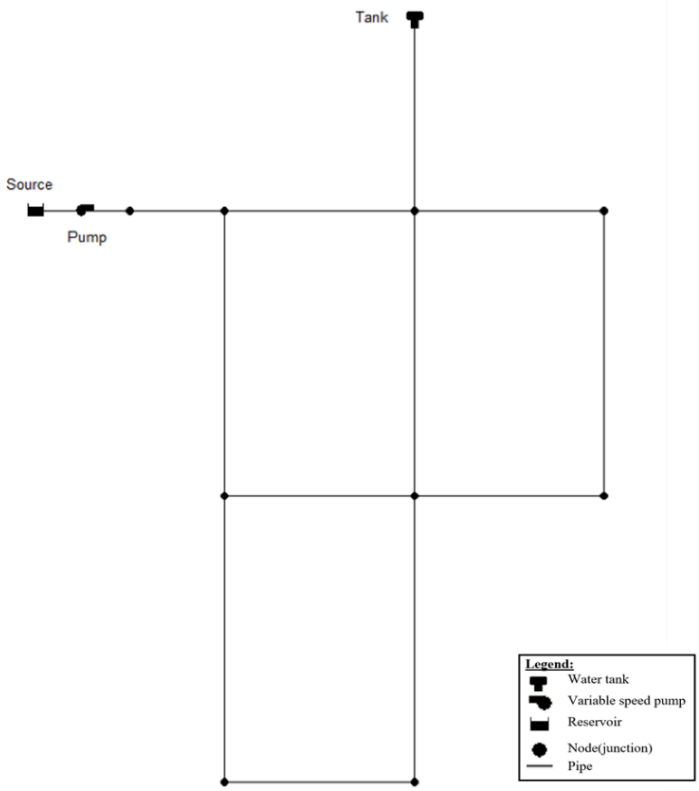

Figure 2-1 Simple WDN Example as Represented in EPANet Interface

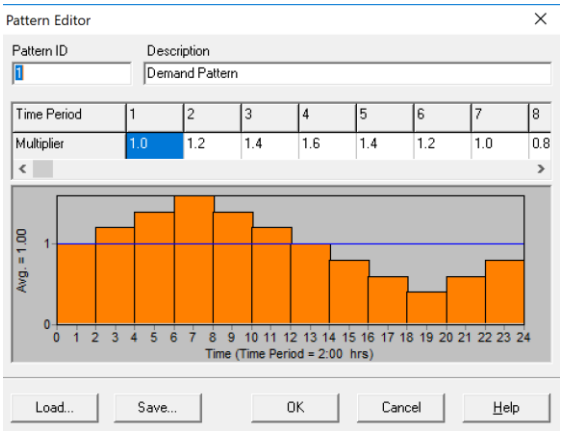

Figure 2-2 An EPANet Hourly Demand Pattern

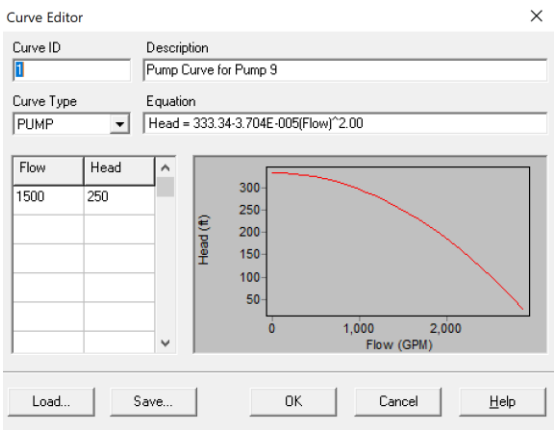

Figure 2-3 An EPANet Pump Efficiency Curve

Concerning the output from the simulator, as mentioned in section 1.2, we focus on four main metrics output from EPANet: pressure, load, supply and tank level. The characteristics of these four measures are defined in the following, along with the way we can estimate them through EPANet [4].

Pressure: Pressure is a key measure for guaranteeing that the water network is working in safe conditions. Indeed, a large pressure may lead to leakages and breakages hindering the functionality of the network, on the other hand, a lower pressure may not satisfy service level requirements of an end consumption points. This measure is easy to derive from the simulator in the form of time series (Figure 2-4 illustrates the pressure at two junctions in the network (red and green curve) over a time period of 24 hours). The granularity of the time series for each node of the network can be set at a predefined sampling frequency. Since optimization often pushes solutions to the boundary of the feasible region, identifying the boundary is an important information for WDN managers who can then

take considerations not modeled in the simulator (such as leakage due to high pressure) into account.

Load: While pressure is typically important for the analysis of junctions distress, load is relevant for performing similar analyses on the network links. While a low load may be desirable to preserve the health of the piping system, it will not guarantee the desired service level. Since EPANet can be run for extended simulation periods, every measure is provided in the output as a time series. We need to investigate ways to derive time series charactering the load form the simulation output.

Tank Level: A WDN not only supplies water to consumption points through pipes, it also utilizes storage devices such as tanks to prevent shortage in presence of demand spikes or small network failures. While a minimum tank level is required to reduce the effect of variance in the demand on the network stress, the tank level must not exceed a predefined upper limit to avoid engineering issues. Storage levels are also easily derived from the output of the simulator for each node at a specified sampling frequency. Figure 2-5 shows the tank level as a time series for the Net1 example.

Supply: Water networks need to provide water to demand nodes. Therefore, it is important to guarantee a satisfactory service level in terms of supply per time unit against request. Especially, since an accurate demand forecast can lead pump scheduling optimization more efficient and effective, WDN managers need to reliably estimate the water demand in the short-term [5]. While an aggregate measure of the daily service level is easy to derive from the simulation output, a dynamic measure is difficult to gather from EPANet. In particular, a first idea is to use the measured network inflow delivered by the pump and the water flow

gathered/delivered by the tank against time. The difference between these two flows is a proxy of the pumped water used for demand satisfaction.

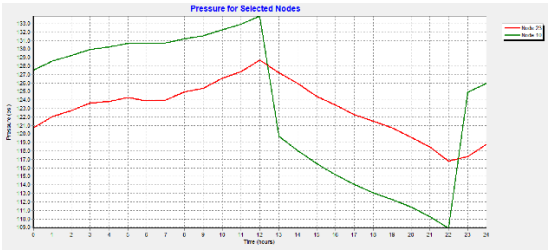

Figure 2-4 Pressure Measured at Two Consumption Nodes of Net1

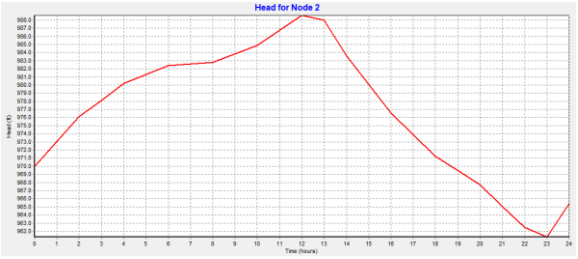

Figure 2-5 Tank Level

2.2 Pump Scheduling Optimization

A recent review on Pump Scheduling Optimization (PSO) and control in WDNs can be found in [6], which reports several classes of existing approaches, including linear programming [13], nonlinear programming [14], and dynamic programming [15]. Mathematical programming-based approaches try to formalize the problem by linearizing/convexifying the equations regulating the flow, thus greatly simplifying the complex water distribution system [4]. As a result, most of the applications are limited to solve the optimization problem only on simple water distribution networks.

Meta-heuristic algorithms, such as Genetic Algorithm [16], Simulated Annealing [18], and Harmony-Search Algorithm [19] have also been proposed. Most of the PSO approaches do not consider the presence of Variable Speed Pumps (VSPs). As a result, the problem is reduced to the decision variables are the pump statuses (0 = pump off, 1=pump on) during a time interval Δt [6].

In general, most of the literature focuses on energy cost minimization, while fewer contributions look at the problem from a feasibility perspective. In fact, most of the

contributions consider the feasibility of pump operation settings by means of a penalty function, added to the objective function [20]. The main idea of penalization is to try to minimize the true objective while also driving the penalty to 0, thus leading to the identification of a feasible, optimal, solution. Lagrangian relaxation is among the most popular techniques used in this area [21]. Although Lagrangian relaxation [20] can guarantee the identification of the optimal solution, it fails to provide insights into feasibility and it is not appropriate in black box settings. This motivates us in the direction of creating algorithms for efficient black-box feasibility determination. Aiming to provide insights to practitioners regarding to feasibility perspective, we develop a feasibility determination algorithm whose output is a set of sub-regions that constitute a controllably accurate approximation of the unknown feasible region.

2.3 Probabilistic Branch and Bound

Our main reference for the proposed new method is the Probabilistic Branch and Bound (PBnB) algorithm [22], that was designed to provide, at each iteration, an approximation of the level set for black box functions.

Specifically, the Probabilistic Branch and Bound (PBnB)[12] algorithm is a partitioning-based random search simulation optimization approach, which was designed for optimizing noisy as well as deterministic black box functions over a potentially mixed continuous integer solution space. Aimed at approximating a user-defined target level set, under a statistically guarantee, PBnB iteratively maintains, prunes and branches subregions. While many simulation optimization algorithms have been proposed that find a single local or global optimal solution, PBnB provides a set of solutions that captures the target level

set, allowing decision makers to make trade-offs between optimal solution and other potential issues.

Before providing the details of the algorithm, Table 2-1 contains the main notation used for the description of the original PBnB algorithm. We will use the same notation in the contributed algroithm.

Table 2-1 Notation in PBnB

Notation	**Description**
α	Confidence level of the estimation of target level set
δ	Define the δ-quantile for the target level set
ϵ	The volume of solutions that can be tolerated to incorrectly prune or maintain
S	Initial solution space
$y(\delta, S)$	Target threshold
k	Iteration index
B	Partition one region into number of disjoint sub-regions
$\tilde{\Sigma}_k^U$	All undecided region in iteration k
$\tilde{\Sigma}_k^M$	All maintained region in iteration k
$\tilde{\Sigma}_1^P$	All pruned region in iteration k
$\sigma_{i,k}$	i^{th} sub-region in $\tilde{\Sigma}_k^U$

A first version of PBnB was proposed in [12] and it is summarized in Figure 2-6. As mentioned at the beginning, PBnB does not return a single optimal solution, but a subset of solutions that approximate a level set chosen by the user to be arbitrarily close to the true (unknown) optimal value of the function. In particular, the user can decide the closeness through the parameter $0 < \delta < 1$. As an example, if the user is interested in the set of top 10% solutions, he will set $\delta = 0.1$. The associated unknown, target threshold (i.e., the level specifying the desired target set) is referred to as $y(\delta, S)$. The parameter α, $0 < \alpha < 1$, is used to control the quality of the $y(\delta, S)$ approximation. With the user-defined parameters, the PBnB algorithm starts to partition the solution space S into B disjoint sub-regions. Based on the new generated sub-regions, PBnB uniformly samples N_k points in each sub-region and ranks the regions by comparing sampled objective evaluations (step 2 in Figure 2-6). In particular, the ranking step in [12] suggests the sub-regions are ranked by the best-found-so-far sampled point. That is, in terms of a minimization problem, the best region $\sigma_{i^*,k}$is the one which contains a minimum sampled objective realization.

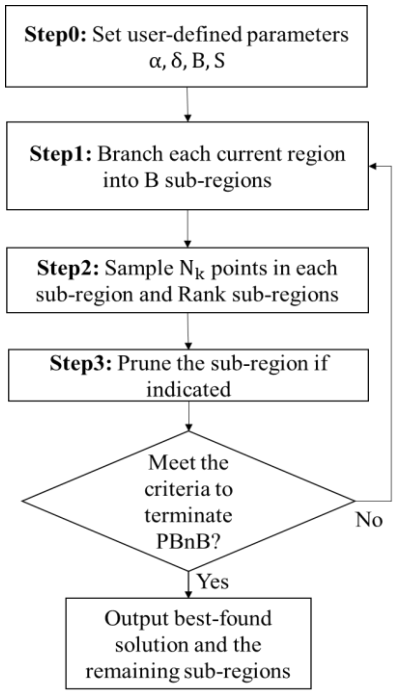

Figure 2-6 Flow chart of Probabilistic Branch and Bound as in [12]

Subsequently, the algorithm starts to prune the undesired sub-region when it is statistically valid to do so. The pruning step states that a sub-region $\sigma_{i,k}$ is to be pruned if the best-found realization in $\sigma_{i,k}$ is worse than the best-found realization in the best sub-region $\sigma_{i^*,k}$ (step 3 in Figure 2-6).

Finally, after PBnB terminates, the remaining sub-regions are the portions of the desired solution set. More, recently [22] proposed a confidence interval driven implementation of the algorithm, which is diplayed in Figure 2-7.

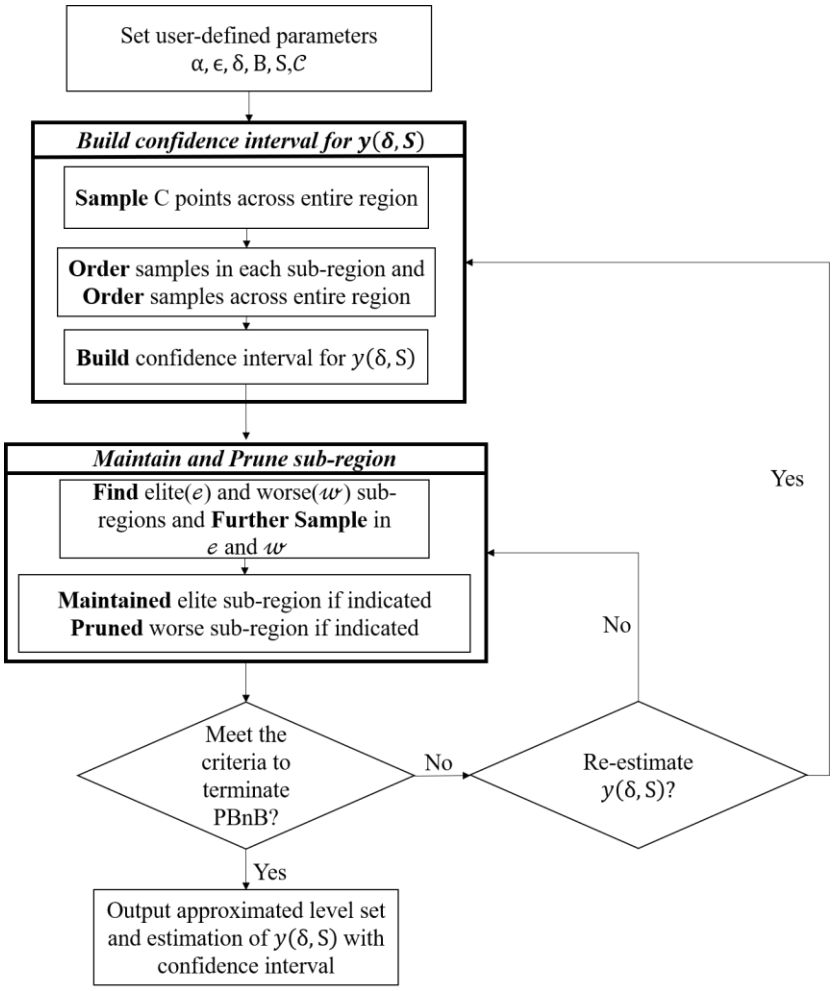

Figure 2-7 Flow Chart of Probabilistic Branch and Bound as in [22]

The lastest version of PBnB in [22] estimates the confidence interval for the unknown threshold $y(\delta, S)$. A new parameter ϵ is introduced to quantify the tolerance of the incorrectly pruning or maintaining. In the recent version, PBnB includes two types of sub-region: pruned and maintained sub-region. Pruned sub-region contains the identified

undesired solutions, on the contary, maintained sub-region has high quality solutions. The rest of sub-regions are referred to as undecided (current, incumbent).

Each algroithm iteration has two stages: (1) the “inner loop”, and (2) the “outer loop”.

(1) The inner loop classifies the sub-regions as maintained or pruned. For a minimization problem, the “maintained region” indicates the union of all the sub-regions that contain points with sampled objective evaluations less than the confidence interval lower bound of $y(\delta, \mathrm{S})$, on the contrary, the “pruned region” is the union of the sub-regions with sampled valus all larger than the confidence interval upper bound of $y(\delta, \mathrm{S})$. The remaining sub-regions are reflected as undecided (incumbent).

(2) The outer loop stage is responsible for the update of the confidence interval for $y(\delta, \mathrm{S})$. Specifically, additional $\mathcal{C}$ points are sampled at this stage across the entire remaining region and the samples are used to update the estimation of the quantiles for $y(\delta, \mathrm{S})$.

While PBnB is our inspiration, we try to address the challenges in applying the algorithm to our feasibility determination problem. In fact, PBnB was not designed to handle the presence of black box constraints.

3 CHAPTER 3: METHODOLOGY

As mentioned in Chapter 1, feasibility, as a proxy of quality of working conditions, plays a very important role in water distribution networks and can define the network state much better than the only energy cost. In fact, as highlighted in [6], the cost functions used in the reviewed literature do not fully capture the network working conditions. This is particularly critical because, if a cost-optimal solution is near to the boundary of the feasible set, the "optimal" pump operation may lead to network malfunctioning (i.e., leakages, breakages). Due to the interest in the identification of feasible regions, we propose a novel partitioning algorithm to tackle the issue of identifying safe working conditions for highly complex water networks that can only be evaluated by means of simulation.

3.1 FSA-PBnB

In this thesis, we propose, for the first time, a feasibility determination algorithm called Feasibility Set Approximation – PBnB (FSA-PBnB) to find the feasible set in presence of multiple black box constraints for which we can only have a point estimate at specific locations of the solution space by running a simulation model.

Most of partitioning methods developed in the literature make use of penalty functions in order to estimate the feasibility region for a generally non-linearly constrained problem. However, penalty functions provide a poor estimation of the feasibility in that we only associate a binary value to a point and we fail to analyze the impact of each configuration component into the violation of each constraint.

To "explicitly" consider feasibility information produced by the black-box simulation in a natural and efficient way, we propose to look into the details of the distance from the feasible frontier instead of the traditional penalty. We use this measure as the basis for a

new partitioning scheme, which adopts a newly proposed Bayesian posterior elimination probability to select the partitioning plane. More specifically, we choose the plane, which is most likely to separate feasible and infeasible solutions. By doing so, the new generated sub-regions in each iteration will be characterized by higher probability to be eliminated, positively impacting the efficiency of the approach.

Section 3.1.1 describes the details of new partition scheme, while section 3.1.2 presents two different criteria to characterize whether a sub-region generated by the procedure is feasible or not.

3.1.1 Dynamic Partitioning Scheme - Probability of Elimination

In order to reduce the sampling effort, we propose a *Dynamic Partitioning Scheme*, which allows us to intelligently and adaptively choose the partitioning plane at each iteration rather than partitioning the axis recursively as proposed in [12]. In order to achieve such an "intelligent" partitioning, we introduce a metric, the Probability of Elimination, to estimate the probability of a sub-region being "eliminated". The "eliminated" sub-region will not be sent to the next iteration for further partitioning and sampling, instead, it will be left out of investigation.

But when does a region get eliminated? In the feasibility determination phase, there are two possible scenarios leading to a sub-region being "eliminated": the sub-region is *highly likely to be feasible*, or the sub-region is *highly likely to be infeasible*. The former will lead the sub-region to be maintained, and the latter will lead to pruning. In both cases, the sub-region is *eliminated* from further consideration.

The Dynamic Partitioning Scheme enhances the algorithm presented in section 2.3 by adaptively selecting the partitioning direction in a way that the generated sub-regions have maximum Probability of Elimination.

In the following, we characterize the approach to obtain the Probability of Elimination for all existing sub-regions. For each point $\boldsymbol{x}$, we define the following feasibility distance vector as the metric of interest:

$$\boldsymbol{d}(\boldsymbol{x}) = \min\left(\boldsymbol{\vartheta^u} - \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{\vartheta^l}\right) \tag{3.1}$$

Where $\boldsymbol{f}$ is a vector of size C (i.e., $\mathrm{c} = 1,2,\dots,\mathrm{C}$), representing the number of black box constraints, and $\boldsymbol{\vartheta^u}, \boldsymbol{\vartheta^l}$ are the vectors of the upper and lower reference values for *C* constraints, respectively. We can derive the feasibility distance of each point $\boldsymbol{x}$ with respect to each constraint, by using equation (3.1). A point $\boldsymbol{x}$ is infeasible if it violates at least one constraint, namely, $\exists\ \mathrm{c}, \mathrm{c} = 1,2,\dots,\mathrm{C}: d_c(\boldsymbol{x}) < 0$.

In the interest of characterizing the feasibility of the sub-regions generated by the partitioning plane, instead of considering the feasibility of each point, we look into the feasibility of a specific sub-region $\sigma_{i,k}$. To achieve this goal, we study the multivariate random variable $\mathbf{D}\left(\sigma_{i,k}\right)$, of dimension C, representing a measure of feasibility distance over each constrains but *across the sub-region* $\sigma_{i,k}$. Namely, if $N_{i,k}$ points have been sampled in the region $\sigma_{i,k}$ at iteration *k*, we have:

$$\mathbf{D}(\sigma_{ik})|\{d(\boldsymbol{x}_{ik})\}, \boldsymbol{x}_{ik} \in \mathbb{S}_{ik} \sim \mathcal{N}(\boldsymbol{\theta}_{ik}) \tag{3.2}$$

where $\{d(\boldsymbol{x}_h)\}$ represents the collection of vector measures in equation (3.1) evaluated for all the points in the sampled set of sub-region *i* at iteration *k*, namely $\mathbb{S}_{i,k}$ such that $\left|\mathbb{S}_{i,k}\right| = N_{i,k}$. These points constitute the input for the evaluation of the parameters of the distribution $\mathcal{N}$, i.e., $\boldsymbol{\theta}(\sigma_{i,k})$. The density derived in equation (3.2) is at the basis of the derivation of the probability of a region to be very infeasible (and pruned by our algorithm) or to be very feasible (and maintained by our algorithm). Since we are looking into a measure of average feasibility distance within a specific sub-region, regardless of the shape of the population distribution given random and independent samples, we assume the distribution of sample means approaches normality as the number of samples increases by the Central Limit Theorem result. Hence, we assume that the feasibility metric (distance) for each constraint in each region follows a normal distribution and the constraints are all independent. As a result, $\boldsymbol{D}(\sigma_{ik})$ is a Multi Variate Normal Distribution and $\boldsymbol{\theta}(\sigma_{i,k})$ in (3.2) represents the $\mathrm{C} \times 1$ mean vector ($\boldsymbol{\mu}$) and $\mathrm{C} \times \mathrm{C}$ variance-covariance matrix ($\boldsymbol{\Sigma}$) with all off-diagonal values equal to 0.

Through the feasibility metrics, the probability of $\sigma_{i,k}$ to be feasible can be derived as the probability of all *C* the components of the multivariate random variable $\mathbf{D}(\sigma_{i,k})$ to have positive value (referring to the joint distribution as *P*):

$$\hat{P}_{feasible}(\sigma_{i,k}) = \mathrm{P}\left(\mathbf{D}(\sigma_{i,k}) > \mathbf{0}\right) = \prod_{c=1}^{C}\left[1 - F_{D_c(\sigma_{ik})}(0)\right]$$

Where $F_{D_c(\sigma_{ik})}$ refers to the cumulative density function (CDF) obtained by marginalizing the CDF of $\mathbf{D}(\sigma_{ik})$ with respect to the c-th component (constraint).

Conversely, the probability of $\sigma_{i,k}$ to be infeasible is that at least one of the constraint is violated, Probability of infeasible is defined as:

$$\hat{P}_{infeasible}(\sigma_{i,k}) \;=\; 1 - \;\mathrm{P}(\mathbf{D}(\sigma_{i,k}) \;>\; \mathbf{0})$$

As mentioned at the beginning, the Probability of Elimination is the probability of a sub-region to be maintained (feasible) or pruned (infeasible). Hence, choosing the maximum between $\hat{P}_{feasible}(\sigma_{i,k})$ and $\hat{P}_{infeasible}(\sigma_{i,k})$ helps to identify the possibility for $\sigma_{i,k}$ to be eliminated.

$$\hat{P}_{elimination}(\sigma_{i,k}) \;=\; \mathrm{Max}\,(\hat{P}_{feasible}(\sigma_{i,k})\,,\hat{P}_{infeasible}(\sigma_{i,k})\,)$$

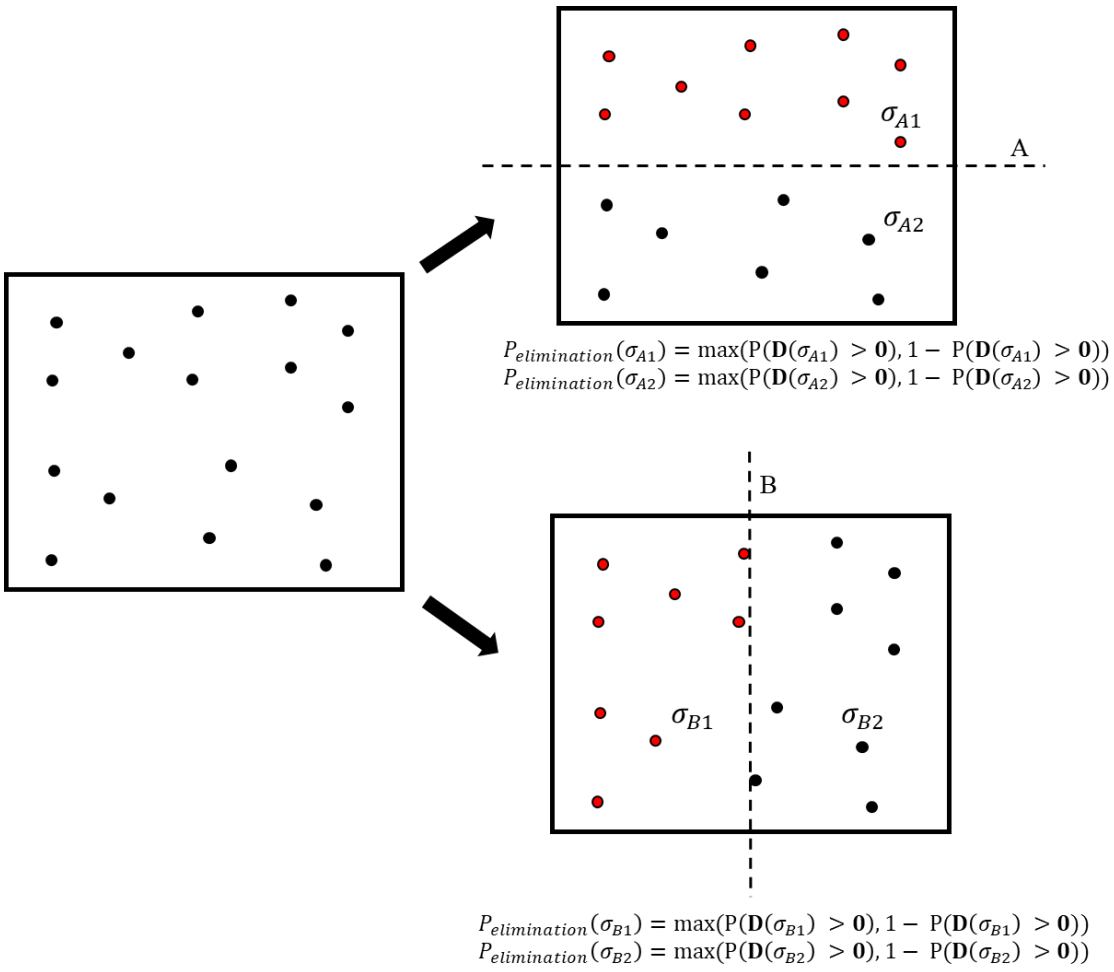

Figure 3-1 Example of Alternative Partitioning Planes

Figure 3-1shows an example of 2-dimensional case with n=16 sampled points. There are two partitioning options, i.e., according to plane A or B, that are compared. For each partitioning option, we compute the posterior elimination probability of each sub-region. Next, the partitioning plane A is characterized by $P_{elimination}(A) = \max(P_{elimination}(\sigma_{A1}), P_{elimination}(\sigma_{A2}))$. Likewise, partitioning plane B is characterized by $P_{elimination}(B) = \max(P_{elimination}(\sigma_{B1}), P_{elimination}(\sigma_{B2}))$. We then select the partitioning plane with the largest associated Probability of Elimination.

The procedures of the Dynamic Partitioning Scheme is depicted in Figure 3-2.

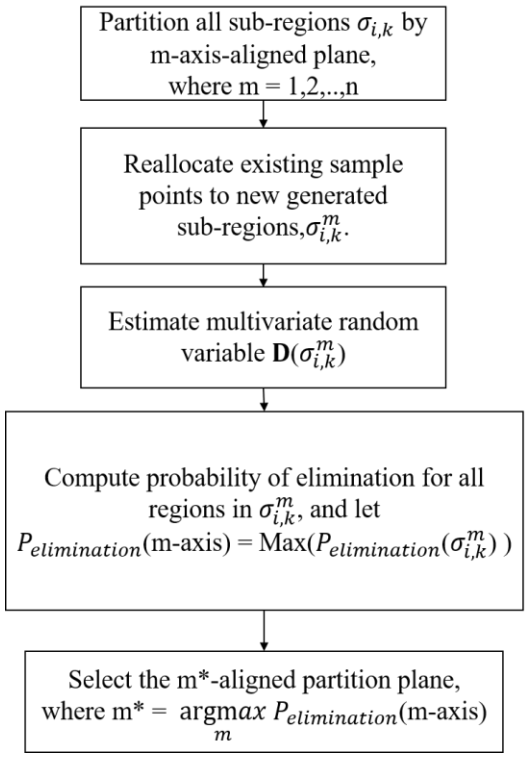

Figure 3-2 Dynamic Partitioning Scheme for n-dimensional Case

To explain what m-axis aligned plane is in the first step, we use the 2-dimensional case (x (horizontal) and y(vertical) axis) in Figure 3-1 as example. At beginning, we define two disjoint sub-regions will be generated by one partitioning plane. The partitioning plane candidates are x-axis aligned plane(s) and y-axis aligned plane(s). If x-axis is selected, the x-axis aligned plane(s) is perpendicular to y-axis and evenly partition the solution space into 2 sub-regions, therefore, we can derive plane A in Table 3-1. Likewise, if y-axis is selected, the y-axis aligned plane(s), plane B, is perpendicular to x-axis and evenly partition the solution space into 2 sub-regions.

Preliminary Results for Dynamic Partition Scheme

Consider a four-dimensional Sinusoidal function as an example, namely:

$$f_x(\boldsymbol{x}) = -2.5\prod_{i=1}^{4}\sin\left(\frac{\pi x_i}{180}\right) - \prod_{i=1}^{4}\sin\left(\frac{\pi x_i}{36}\right) \quad , \quad 0 \le x_i \le 180, i = 1, \dots, 4$$

Figure 3-3 and Figure 3-4 display the outcome of the distribution of the Probability of Elimination of new sub-regions in each iteration.

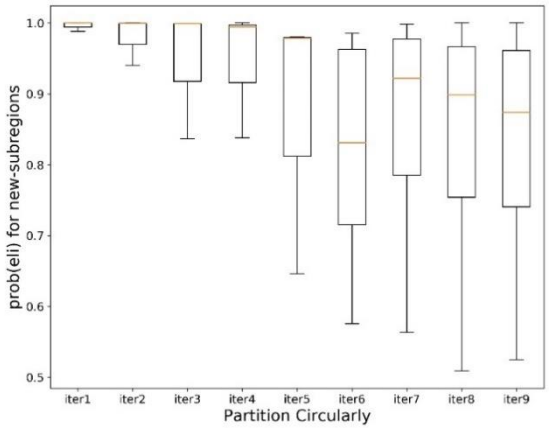

Figure 3-3 Probability of Elimination of New Sub-regions in Each Iteration by Original Partitioning Scheme in [12]

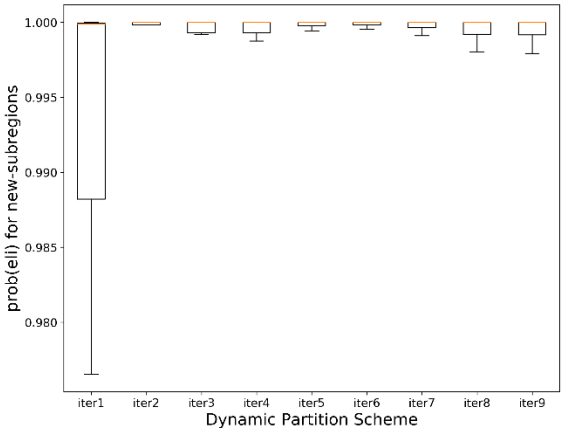

Figure 3-4 Probability of Elimination of New Sub-regions at Each Iteration Resulting from the Dynamic Partitioning Scheme

Figure 3-3 shows the original partitioning scheme resulting from the implementation of the original PBnB algorithm [12]. We can observe that there is large variance of $P_{elimination}(\sigma_{i,k})$ at each iteration, resulting in inefficiencies in separating feasible and non-feasible solutions. This is not surprising since the original algorithm does not look into elimination probability.

On the contrary, the proposed dynamic partition scheme in Figure 3-4 shows the majority of the new sub-regions in each iteration are with high Probability of Elimination. Through the result, we can see the dynamic partitioning scheme leads the algorithm to find a good partitioning direction to separate the feasible and infeasible regions iteratively.

3.1.2 Classification Criteria – Classify Feasible Region and Infeasible Region

Differently, and complementary, to the choice of the separation plane, the classification aims at establishing whether a sub-region is feasible or infeasible, once partitioning has been performed. We propose two classification frameworks to identify the feasible set. Section 3.1.2.1 illustrates the Pointwise Comparison framework, while Section 3.1.2.2 shows the Quantile Comparison framework.

3.1.2.1 Pointwise Comparison

Concerning the identification of the sub-regions based on the feasibility, the criteria for determining whether a specific sub-region needs to be pruned, maintained or branched plays a critical role. While the original implementation of the PBnB algorithm described in section 2.3 has good performance in optimization settings, in terms of the feasibility determination, the rule for sub-regions to be classified leads to quick elimination of feasible regions, which is clearly a drawback for our application. This "extreme" elimination is due to the fact that in [12] sub-regions are ranked according to the most promising point evaluation, that is, translated into our settings, the "most feasible" point within the "most feasible" sub-region leads to the determination of the rank-1 feasible region. The elimination criteria proposed therein compares the "most feasible" point within the "most feasible" sub-region against the "most feasible" point in other regions. Doing so, every sub-region will always be eliminated unless the evaluation of the "most feasible" point in non-best region and the "most feasible" point in "most feasible" region have the same value. To overcome this difficulty, we developed a pointwise comparison alternative which ranks the sub-regions based upon the average of the distance measures with the sub-region. Then we compare the "most infeasible" point in the "most feasible" sub-region against the "most

feasible" point in all other sub-regions. More specifically, we identify the feasible regions through the Euclidean distance measure associated to each point $\boldsymbol{x}$, namely:

$$d_{\mathrm{E}}(\boldsymbol{x}) = \|\boldsymbol{d}^{+}(\boldsymbol{x})\|_2 \tag{3.3}$$

Where $\boldsymbol{d}^{+}(\boldsymbol{x}) = \left[|min\big(0, d_1(\mathbf{x})\big)|, |min\big(0, d_2(\mathbf{x})\big)|, \ldots, |min\big(0, d_C(\mathbf{x})\big)|\right]$ is the vector having as positive components only the infeasible distances derived from the elements in equation (3.1). The subscript E in $d_{\mathrm{E}}(\boldsymbol{x})$ refers to "Euclidean". The new metric introduced in equation (3.3) is the Euclidean distance between the point $\boldsymbol{x}$ and the feasible frontier. If the sample point $\boldsymbol{x}$ is feasible, the metric $d_{\mathrm{E}}(\boldsymbol{x}) = 0$.

Specifically, for each sampled solution $\boldsymbol{x}$, we measure the amount of infeasibility instead of just using a large penalty value (as more traditional in Lagrangian-type approaches). With the Euclidean distance metric, we can provide more insights and better inform the partitioning algorithm.

The FSA-PBnB with Pointwise Comparison algorithm is presented in the following with the detailed steps.

Step0: Initialize the Parameters and sample initial points

Input the user defined parameters $\alpha, \delta, B, S, N_k$. Initialize the maintained, pruned, undecided regions and iteration counter k: $\Sigma_1 = \mathbb{X}$ (where $\mathbb{X}$ is the set considering only box constraints for initial solution space), $\tilde{\Sigma}_1^U = \mathbb{X}$, $\tilde{\Sigma}_1^M = \emptyset$, $\tilde{\Sigma}_1^P = \emptyset$, $\mathrm{k} = 1$, $\alpha_1 = \frac{\alpha}{2}$.

For the initial undecided region $\tilde{\Sigma}_1^U = \mathbb{X}$, uniformly sample N^1 initial points over the entire space.

$$N^1 = \left\lceil \frac{\ln(\alpha_1)}{\ln(1-\delta)} \right\rceil$$

Step1: Dynamic Partitioning Plane Selection

Based on all the sampled points within the current undecided region $\tilde{\Sigma}_k^U$, assess the Probability of Elimination among all proposed partitioning planes, as in the flow chart in Figure 3-2 and perform the partitioning along the selected direction. Denotes $\tilde{\Sigma}_k^{U'}$ as the union of all new generated sub-regions after partitioning.

Step2: Sample Points and Rank the sub-regions

Sample points up to N_i^k for each undecided sub-region $\sigma_{i,k} \in \tilde{\Sigma}_k^{U'}$, where i is the index of sub-region and $N_i^k = \left\lceil \frac{\ln(\alpha_k)}{\ln(1-\delta)} \right\rceil$ [12].

Evaluate the sampled points by the distance function $d_{\mathrm{E}}(\boldsymbol{x_{i,j,k}})$ in equation (3.3), where $\boldsymbol{x_{i,j,k}} \in \sigma_{i,k}$, i =1,…$|\tilde{\Sigma}_k^{U'}|$ and for j = 1,…, N_i^k, and calculate the average distance value for $\sigma_{i,k}$, $\overline{d_{\mathrm{E}}}$, as it follows:

$$\overline{d_{\mathrm{E}}}(\sigma_{i,k}) = \frac{\sum_{j=1}^{N_i^k} d_{\mathrm{E}}(\boldsymbol{x_{i,j,k}})}{N_i^k}$$

Rank the current sub-regions according to $\overline{d_{\mathrm{E}}}(\sigma_{i,k})$ with $\sigma_{(i)}$ denoting the i^{th} best sub-region,

$$\overline{d_{\mathrm{E}}}(\sigma_{(1),k}) \leq \overline{d_{\mathrm{E}}}(\sigma_{(2),k}) \leq \ldots \leq \overline{d_{\mathrm{E}}}\left(\sigma_{(|\tilde{\Sigma}_k^{U'}|),k}\right)$$

Rank all the sample points $\boldsymbol{x}_{(\boldsymbol{i}),\boldsymbol{j},\boldsymbol{k}} \in \sigma_{(i),k}$ according to $d_{\mathrm{E}}(\boldsymbol{x}_{(\boldsymbol{i}),\boldsymbol{j},\boldsymbol{k}})$ with $\sigma_{(i),(j),k}$ denoting the j^{th} best point in the i^{th} best sub-region,

$$d_{\mathrm{E}}(\boldsymbol{x}_{(\boldsymbol{i}),(\boldsymbol{1}),\boldsymbol{k}}) \leq d_{\mathrm{E}}(\boldsymbol{x}_{(\boldsymbol{i}),(\boldsymbol{2}),\boldsymbol{k}}) \leq \ldots \leq d_{\mathrm{E}}\left(\boldsymbol{x}_{(\boldsymbol{i}),(\boldsymbol{N_i^k}),\boldsymbol{k}}\right)$$

From the above ranking methodology, we define $\sigma_{(1),k}$ is the most feasible sub-region and $\boldsymbol{x}_{(\boldsymbol{1}),(\boldsymbol{N_i^k}),\boldsymbol{k}}$ is the most infeasible point in $\sigma_{(1),k}$.

Step3: Maintain the region

Define the indicator functions $\mathcal{I}_{m,k}$ for m = 1,…, $|\tilde{\Sigma}_k^{U'}|$ as:

$$\mathcal{I}_{m,k} = \begin{cases} 1, & if\ \overline{d_{\mathrm{E}}}\left(\sigma_{(m),k}\right) = 0 \\ 0, & otherwise. \end{cases}$$

Where $\mathcal{I}_{m,k} = 1$ indicates that the sub-region $\sigma_{(m),k}$ can be maintained.

Step4: Prune the region

Denote the most feasible sub-region $\sigma_{(1),k}$ and the most infeasible point $\boldsymbol{x}_{(\mathbf{1}),(\boldsymbol{N}_{\boldsymbol{i}}^{\boldsymbol{k}}),\boldsymbol{k}}$ as ${\sigma^*}_k$ and $\boldsymbol{x}_{*(\mathbf{1}),\boldsymbol{k}}$, respectively.

Define the indicator functions $\mathcal{I}_{p,k}$ for p = 2,…, $|\tilde{\Sigma}_k^{U'}|$

$$\mathcal{I}_{p,k} = \begin{cases} 1, & if\ d_{\mathrm{E}}\left(\boldsymbol{x}_{(\boldsymbol{p}),(\mathbf{1}),\boldsymbol{k}}\right) > d_{\mathrm{E}}(\boldsymbol{x}_{*(\mathbf{1}),\boldsymbol{k}}) \\ 0, & otherwise. \end{cases}$$

Where $\mathcal{I}_{p,k} = 1$ indicates that the sub-region $\sigma_{(p),k}$ can be pruned.

Step5: Update undecided Region

Update the undecided sub-regions, which are not be maintained or pruned, at the *k*-th iteration:

$$\tilde{\Sigma}_{k+1}^{U} = \tilde{\Sigma}_k^{U'} \backslash \left(\bigcup_{\{m:\, \mathcal{I}_{m,k}=1\}} \sigma_{(m),k} \right) \backslash \left(\bigcup_{\{p:\, \mathcal{I}_{p,k}=1\}} \sigma_{(p),k} \right)$$

Update the set of the maintained sub-regions

$$\tilde{\Sigma}_{k+1}^{M} = \tilde{\Sigma}_k^{M} \cup \left(\bigcup_{\{m:\mathcal{I}_{m,k}=1\}} \sigma_{(m),k} \right)$$

Step6: Terminate FSA-PBnB

If the criteria to terminate FSA-PBnB is reached, output the $\tilde{\Sigma}_{k+1}^{U}$ and $\tilde{\Sigma}_{k+1}^{M}$. Otherwise, let $\alpha_{k+1} = \frac{\alpha_k}{2}, k \rightarrow k+1$, and go back to step1.

Ranking sub-regions is an important step in PBnB because, through ranking, we can choose the best sub-region among all the contending sub-regions in terms of distance measure and take it as a reference region to perform pruning. In order to evaluate a region, the average of the distance measures allows us to include information from all the samples, returning a more accurate representation of the feasibility of a sub-region than just the minimum distance value. Regarding to the criteria in step 3, the sub-region $\sigma_{(m),k}$ will be maintained if all sampled points in $\sigma_{(m),k}$ are feasible, that is, all the distance measures $d_{\mathrm{E}}(\boldsymbol{x}_{(\boldsymbol{m}),\boldsymbol{j},\boldsymbol{k}})$ within the sub-region are equal to 0.

With respect to the issue of quick elimination of feasible regions, the criteria in Step 4 asserts that the sub-region will be pruned if its "most feasible" point is worse than "the most infeasible" point in the most feasible sub-region. For example, $\sigma_{(p),k}$ is pruned if the smallest distance measure in $\sigma_{(p),k}$ is larger than the largest distance measure in ${\sigma^*}_k$. As a result, if the sampled points in a sub-region $\sigma_{i,k}$ are all infeasible, as long as the smallest distance measure in $\sigma_{i,k}$ is smaller than the largest distance measure in the most feasible region, the $\sigma_{i,k}$ will be kept in the next iteration.

3.1.2.2 Quantile Comparison

In connection with the proposed Dynamic Partitioning Scheme, which characterizes the feasibility of each sub-region by means of the feasibility distance metric $\mathbf{D}(\sigma_{i,k})$, we propose the quantile comparison elimination criteria, which uses the distribution of the feasibility metric and the estimates of the related upper and lower quantiles.

Herein, we adopt the feasibility measure in equation (3.1) to construct the feasibility distribution of each constraint and perform the quantile comparison. The resulting, modified, algorithm is provided in the following with the detail of the algorithmic steps. Step 0, and Step1 are identical to the Pointwise comparison (section 3.1.2.1) case and are therefore omitted.

Step2: Sample Points and Rank the sub-regions

Sample points up to N_i^k for each undecided sub-region $\sigma_{i,k}$, where $N_i^k = \left\lceil \frac{\ln(\alpha_k)}{\ln(1-\delta)} \right\rceil$, and estimate the parameters of the multivariate random variable $\mathbf{D}(\sigma_{i,k})$.

Assume $\mathbf{D}(\sigma_{i,k})$ has independent components, and compute the probability of being feasible for sub-region $\sigma_{i,k}$ using:

$$P_{feasible}(\sigma_{i,k}) = \prod_{c=1}^{C} \mathrm{P}(\mathrm{D}_c(\sigma_{i,k}) > 0)$$

Where *C* is the number of constraints.

Rank all the "contending" sub-regions according to $P_{feasible}(\sigma_{(i),k})$ with $\sigma_{(i),k}$ denoting the i^{th} best sub-region:

$$P_{feasible}(\sigma_{(1),k}) \geq P_{feasible}(\sigma_{(2),k}) \geq \ldots \geq P_{feasible}(\sigma_{(|\tilde{\Sigma}_k^{U'}|),k})$$

From the above ranking methodology, the sub-region $\sigma_{(1),k}$ has the largest associated probability to be feasible.

Step3: Find the Upper and Lower Quantile respect to each constraint for all sub-regions

Define $\mathbb{Q}(\sigma_{i,k})$ as a $\mathrm{C} \times 2$ matrix, including *C* rows for the constraints and 2 columns for the upper quantile and lower quantile. Based on the parameters of the multivariate random variable $\mathbf{D}(\sigma_{i,k})$, it is possible to find the ℓ and $\mathcal{u}$ quantiles for the constraint c such that:

$$\mathrm{Q}_c^{\ell}(\sigma_{i,k}) = Z_{\ell} \times \mathrm{s}_{\boldsymbol{c}} + \mu_{\boldsymbol{c}}$$

$$\mathrm{Q}_c^{\mathcal{u}}(\sigma_{i,k}) = Z_{\mathcal{u}} \times \mathrm{s}_{\boldsymbol{c}} + \mu_{\boldsymbol{c}}$$

where, $\mathrm{s}_{\boldsymbol{c}}$ is the standard deviation of the feasibility distance for c^{th} constraint, and Z is the inverse standard normal.

Step4: Maintain the region

Denote $\mathbb{Q}^{\ell}(\sigma_{i,k})$ as the lower quantile vector, extracted from the first column of $\mathbb{Q}(\sigma_{i,k})$

Define the indicator functions $\mathcal{I}_{m,k}$ *for m* = 1, …, $|\tilde{\Sigma}_k^{U'}|$

$$\mathcal{I}_{m,k} = \begin{cases} 1, & if\ \mathbb{Q}^{\ell}(\sigma_{(m),k}) \geq \mathbf{0} \\ 0, & otherwise. \end{cases}$$

Where $\mathcal{I}_{m,k} = 1$ indicates that the sub-region $\sigma_{(m),k}$ can be maintained.

Step5: Prune the region

Denote the most feasible sub-region $\sigma_{(1),k}$ and the corresponding lower quantile vector $\mathbb{Q}^{\ell}(\sigma_{(1),k})$ as ${\sigma^*}_k$ and $\mathbb{Q}^{\ell^*}$, respectively.

Define the indicator functions $\mathcal{I}_{p,k}$ for p = 2,…, $|\tilde{\Sigma}_k^{U'}|$

$$\mathcal{I}_{p,k} = \begin{cases} 1, & \exists\ \mathrm{c} \in \mathrm{C}\colon \mathrm{Q}_c^{\mathcal{u}}(\sigma_{(p),k}) \leq 0 \text{ and } \mathrm{Q}_c^{\mathcal{u}}(\sigma_{(p),k}) \leq \mathrm{Q}_c^{\ell^*} \\ 0, & otherwise. \end{cases}$$

Where $\mathcal{I}_{p,k} = 1$ indicates that the sub-region $\sigma_{(p),k}$ can be pruned.

The remaining steps, are exactly as the same as the step 5 and step 6 in section 3.1.2.1, and are therefore omitted here.

In order to explicitly consider the feasibility information, during step 4, we check all the elements within the lower quantile vector to verify the presence of negative components. If all the lower quantiles (worst feasibility distance) are positive (refer to equation (3.1)), positive distance represents feasibility), we conjecture the solutions in the sub-region satisfy all the constraints and the sub-region can be classified as feasible (maintained) region.

In Step 5, the sub-region will be pruned if at least one upper quantile (i.e., the estimate of the best scenario) for any of the C constraints simultaneously satisfies two conditions: (1) the upper quantile of constraint *c* is negative, and (2) the upper quantile of constraint *c* is worse than the lower quantile (the worst feasibility scenario) of constraint *c* in the most feasible sub-region. Due to the sampling error, the sub-region which contains the feasible set might be characterized as infeasible and be pruned by satisfying the condition (1). To reduce this phenomenon, the condition (2) is added to the pruning criteria. As a result, similar to the pointwise comparison, the upper quantile of the constraint in a sub-region is compared against the lower quantile of the constraint in the best (most feasible) sub-region.

3.2 Numerical experiments on theoratical functions

To showcase the performance of FSA-PBnB over generally constrained optimization problems, we use the Sinusoidal function in different dimensions. This function is frequently used in the global optimization literature, and we added two constraints, defined as follows.

$$f_x(\boldsymbol{x}) = -2.5\prod_{i=1}^{n}\sin\left(\frac{\pi x_i}{180}\right) - \prod_{i=1}^{n}\sin\left(\frac{\pi x_i}{36}\right) \tag{3.4}$$

With constraints

$$f_x(\boldsymbol{x}) \leq -2.3 \tag{3.5}$$

$$g(\boldsymbol{x}) \geq 0 \quad , g(\boldsymbol{x}) = \begin{cases} 5.7 & x_1 \leq 90 \\ -5.7 & x_1 > 90 \end{cases} \tag{3.6}$$

$$0 \leq x_i \leq 180, i = 1, \ldots, n$$

The global optimum is located in $\boldsymbol{x}^* = (90, \ldots, 90)$ with a function value $f(\boldsymbol{x}^*) = -3.5$.

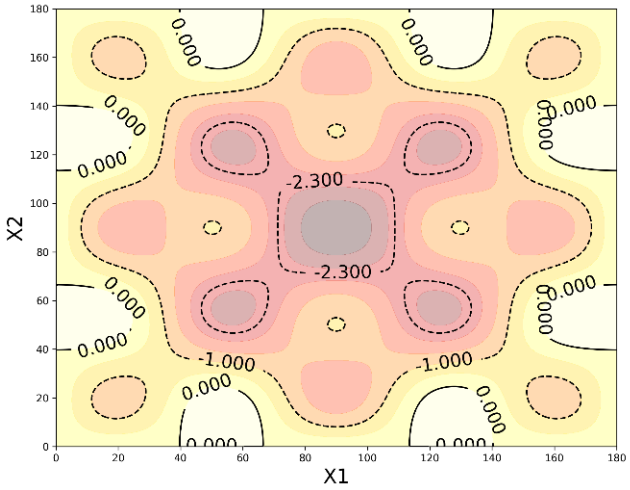

Figure 3-5 Contour Plot of 2D Sinusoidal Function

3.2.1 Parameter Settings for the Numerical experiment

To test the robustness of the proposed algorithms, we evaluated their performance over 2, 3, and 4 dimensional cases on the Sinusoidal Function. For each case, we first tested the function with one constraint, e.g. we only considered equation (3.5) and dismissed equation (3.6) in our experiment, then, we took both constraints into consideration.

The FSA-PBnB algorithm parameters were: $\delta = 0.1, \alpha = 0.25, B = 3$. FSA-PBnB terminated when the number of iterations reached the maximum number set by the user. We used the values in Table 3-1Table 3-1 Criteria to Terminate the Algorithm).

Table 3-1 Criteria to Terminate the Algorithm

Dimension	2	3	4
Iteration (K)	10	13	15

In the interest of limiting the computational time, we set the simulation time limit to 1000 seconds for each macro replication of the entire algorithm. For each experiment, we collected the metrics in Table 3-2.

Table 3-2 Definition of Metric for Experiment Result

Metric	**Definition**
$\boldsymbol{\gamma}$	Empirical probability of the optimal solution to end up in the remaining regions (remaining region = $\tilde{\Sigma}_{k^*}^{U} \cup \tilde{\Sigma}_{k^*}^{M}$) computed out of the algorithmic macro-replications
T_Pts	Total Number of points sampled after PBnB terminates
R(PV)	Ratio of the pruned volume to initial volume (S)
R(UV)	Ratio of the undecided volume to initial volume (S)
R(MV)	Ratio of the maintained volume to initial volume (S)
R(Remaining)	Ratio of the remaining volume to initial volume (S)
R(TV)	Ratio of the true volume against the initial volume (S)

3.2.2 Numerical Experiment Result

In this section, we present the experiment results obtained from the two algorithms: FSA-PBnB with Pointwise-Comparison algorithm and FSA-PBnB with Quantile- Comparison. To evaluate the performance on feasibility determination, the true volume of the feasible region is compared with the numerical experiment results. While the volume of the true feasible region cannot be analytically derived for most of the tested nonlinear functions, we applied a grid search concept to approximate the true feasible volume. We divided the whole solution space into *n* equally-sized hyper-cubes and the feasibility of the central

point in the hyper-cube stands for the feasibility of the hyper-cube. The approximated true feasible volume is computed by summing up the volume of feasible hyper-cubes. The feasible volume of single constraint and multiple constraints cases are illustrated in Table 3-3.

Table 3-3 Ratio of the True Feasible Volume R(TV) of Feasible Set

	Dimension		
	2	3	4
Constraint (3.5)	8.77%	1.92%	0.3%
Constraint (3.5) and (3.6)	4.5%	1.0%	0.17%

3.2.2.1 Pointwise-Comparison FSA-PBnB

For the 2-dimensional Sinusoidal function with constraint(s), Figure 3-6 and Figure 3-7 show the undecided regions (in orange) and the maintained regions (in blue) over the contours resulting from 10 iterations of FSA-PBnB. The inner space of the black dash line is true level set of $f_x(\boldsymbol{x}) \leq -2.3$ as from equation (3.5). The red cross symbol in the middle is the global optima $f_x(\boldsymbol{x}) = -3.5$.

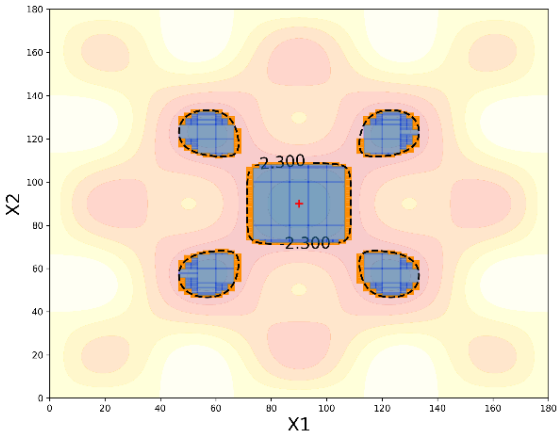

Figure 3-6 Result of Pointwise-Comparison for Single Constraint on 2D Sinusoidal Function

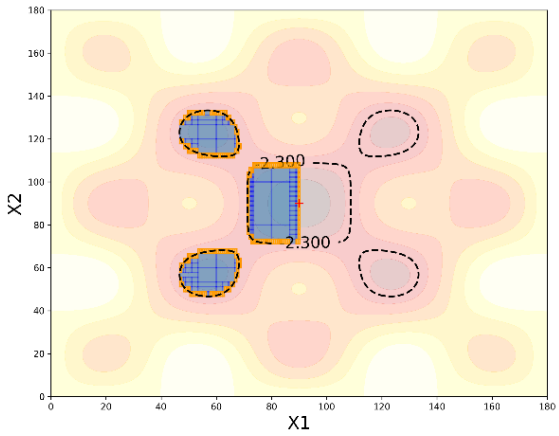

Figure 3-7 Result of Pointwise-Comparison for Multiple Constraints on 2D Sinusoidal Function

Figure 3-6 presents the result for the sinusoidal function with the single constraint in equation (3.5), while Figure 3-7 shows the results obtained considering both constraints. We can observe that the more inside the feasible boundary, the larger the blue rectangles are, that is, the regions are "maintained" earlier. Also, it is possible to observe how the undecided regions are mostly located at the boundary of the feasible set, hinting that the regions around the boundary require more iterations as more partitioning and sampling are needed to classify the regions.

The Pointwise-Comparison algorithm shows very good performance on capturing the true feasible region for the single and multiple constraints for the 2-dimensional case. Particularly, for the multiple constraints case, even if the global optimal is located on the boundary of the feasible set, it is kept within the remaining region ($\tilde{\Sigma}_{k^*}^{U} \cup \tilde{\Sigma}_{k^*}^{M}$), which means that, after feasibility determination, we will not lose the chance to find the global optimal solution in optimality phase.

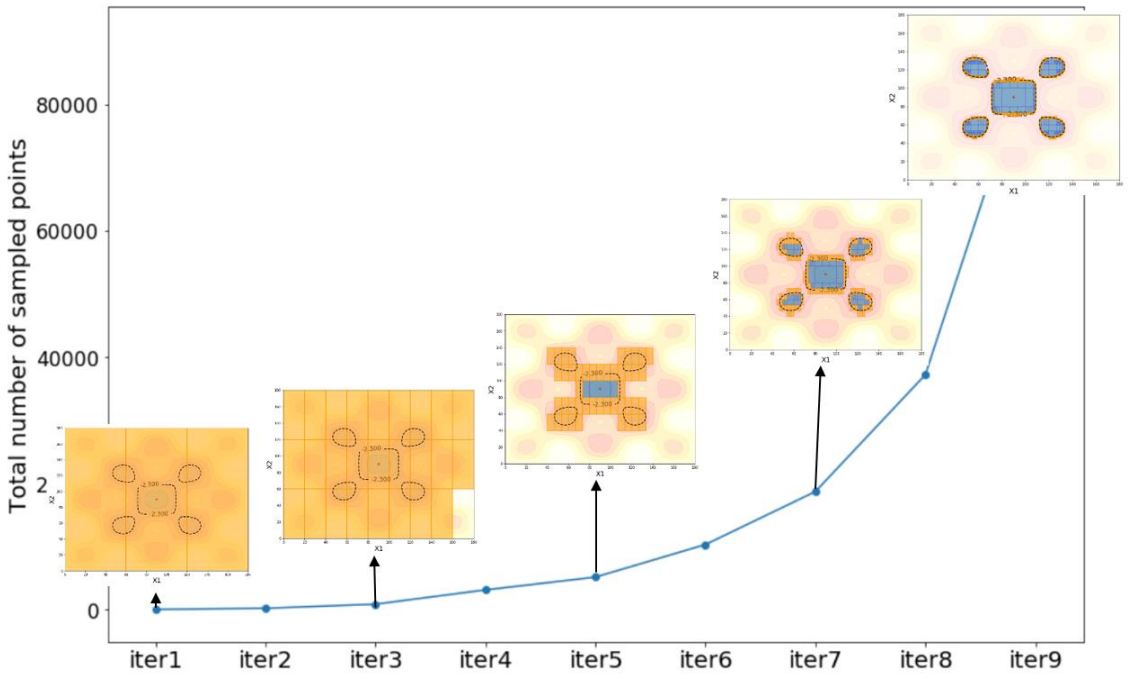

Figure 3-8 Detail of FSA-PBnB on 2-D Sinusoidal Function

Figure 3-8 shows the detail of the cumulated number of sampled points in each iteration. We can observe that the number of sampled points grows exponentially with the iterations. At iteration 5, FSA-PBnB captures small parts of feasible region and, at iteration 7, the

profile of the feasibility set is identified. From iteration 7 to 9, FSA-PBnB is mainly refining the feasibility boarder.

The algorithm was run 100 times under each test case for all the dimensions. The detailed performance is presented in Table 3-4 and Table 3-5. In each cell, except for the metric γ, the upper value is the mean of all replications and the value in the parenthesis is the coefficient of variation.

Table 3-4 Result of Pointwise-Comparison for Single Constraint

Dimension	T_Pts	γ	R(PV)	R(UV)	R(MV)	R(Remaining)/R(TV)
2	71.475×10^3 (14.38%)	1	90.54% (1.13%)	1.88% (3.43%)	7.58% (1.13%)	9.46% / 8.77%
3	12.591×10^5 (27.48%)	1	97.43% (0.15%)	1.33% (16.83%)	1.25% (6.92%)	2.58% / 1.92%
4	47.607×10^5 (20.63%)	1	99.39 (0.08%)	0.5% (19.57%)	0.1% (24.17%)	0.6% / 0.3%

From the numerical results on the sinusoidal function, as the dimension of the problem grows, $\gamma = 1$ and the true optimal solution is always included in the remaining region (union of the undecided and maintained sub-regions). Observing the metric R(PV), it is clear that the algorithm succeeded in pruning a large portion of the region in all the dimensional cases. Therefore, the results demonstrate Pointwise-Comparison FSA-PBnB algorithm helps to delete many undesired regions (infeasible region), while maintaining the true optimal solution.

The performance of the FSA-PBnB can be observed by comparing the metrics R(Remaining) against the ratio of the true feasible volume, R(TV). From Table 3-4, we observe that a larger portion of volume is returned with respect to the true volume as the dimension of the problem grows. The result demonstrates that FSA-PBnB is a conservative

algorithm. We also recall that R(remaining) consists of R(UV) and R(MV). By looking into the portion of R(UV) and R(MV) in R(remaining), the 2-dimensional case shows that R(MV) makes up 80% and R(UV) makes up 20% of R(Remaining) at iteration 10. In the 3-dimensional case, at iteration 13, R(MV) makes up 52% and R(UV) makes up 48% of R(Remaining). These ratios suggest that the majority of the feasible regions are captured in the maintained regions. However, in the 4-dimensional case, R(MV) makes up only 17% of R(remaining), that is, R(UV) may contain a large portion of infeasible solutions.

According to the results from Figure 3-6 and Figure 3-7, we see that the sub-regions around the boundary require more partitioning and sampling to be accurately classified. Looking at the number of sampled points in Table 3-4, all the cases require a large budget to evaluate the feasible region. This is due to the fact that the greater part of the points are used to distinguish the feasible and infeasible regions from the undecided region. Hence, it is clear how important it is to consider the trade-off between sampling effort and shifting undecided regions to maintained regions.

Table 3-5 Result of Pointwise-Comparison for Multiple Constraints

Dimension	T_Pts	γ	R(PV)	R(UV)	R(MV)	R(Remaining)/R(TV)
2	52.252×10^3 (28.41%)	1	94.89% (0.45%)	1.51% (39.71%)	3.6% (4.94%)	5.11% / 4.5%
3	10.950×10^5 (40.61%)	1	98.38% (0.37%)	1.14% (45.02%)	0.47% (33.42%)	1.61% / 1.0%
4	32.515×10^5 (27.83%)	1	99.62 (0.08%)	0.35% (26.59%)	0.04% (28.27%)	0.39% / 0.17%

Table 3-5 shows the experimental results for the different cases obtained considering both constraints (3.5) and (3.6). We can observe that, regardless of the problem dimension, γ is always 1, which means that the true optimal solution $\boldsymbol{x}^* = (90,\ldots,90)$ is never eliminated by FSA-PBnB. By comparing the results of R(Remaining) to R(TV), we observe that, under the stopping criteria we adopt in this experiment, the R(Remaining) is always larger

than R(TV). The results represent the feasible set are fully captured in dimension 2, 3 and 4, respectively.

3.2.2.2 Quantile-Comparison FSA-PBnB

In this section, the Quantile Comparison version of the algorithm is used to classify feasible regions and infeasible regions. For the 2-dimensional Sinusoidal function with constraint(s), we visualize the undecided regions (in orange) and the maintained regions (in blue) over the contours after FSA-PBnB terminates at iteration 10 in Figure 3-9 and Figure 3-10. The results look similar to the pointwise comparison version. The undecided regions are located around the boundary of feasible set.

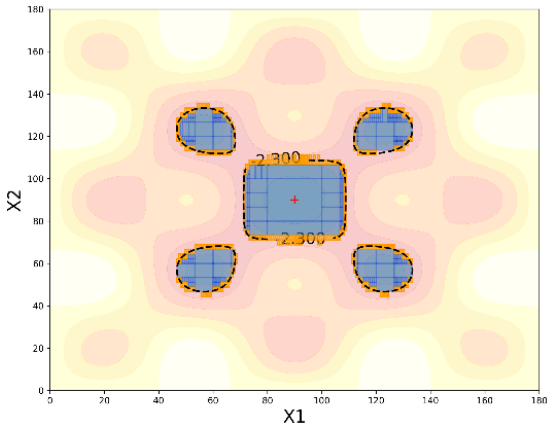

Figure 3-9 Result of Quantile-Comparison for Single Constraint on 2D Sinusoidal Function

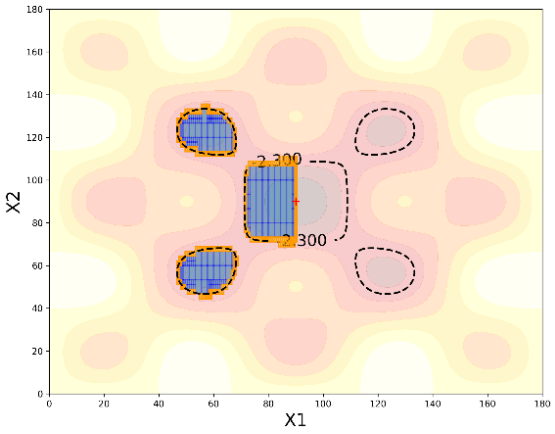

Figure 3-10 Result of Quantile-Comparison for Multiple Constraints on 2D Sinusoidal Function

Figure 3-9 and Figure 3-10 show the results of Quantile Comparison implementation on sinusoidal function with the same experiment parameter setting as Pointwise-Comparison FSA-PBnB. Regarding γ, all the test cases indicate that the remaining region of Quantile-Comparison FSA-PBnB always contain the optimal solution. Looking into the metrics, R(UV), R(MV), and R(remaining), we do not observe statistical difference between the two approaches difference from the result of Table 3-4 and Table 3-5.

Table 3-6 Result of Quantile-Comparison for Single Constraint

Dimension	T_Pts	γ	R(PV)	R(UV)	R(MV)	R(Remaining)/R(TV)
2	79.261×10^3 (14.8%)	1	90.37% (0.14%)	2.05% (3.01%)	7.58% (1.38%)	9.63% / 8.77%
3	13.411×10^5 (39.31%)	1	97.38% (0.22%)	1.35% (20.73%)	1.27% (7.02%)	2.62% / 1.92%
4	39.866×10^5 (28.14%)	1	99.44% (0.1%)	0.45% (26.22%)	0.11% (19.31%)	0.56% / 0.3%

Table 3-7 Result of Quantile-Comparison for Multiple Constraint

Dimension	T_Pts	γ	R(PV)	R(UV)	R(MV)	R(Remaining)/R(TV)
2	58.238×10^3 (26.96%)	1	94.94% (0.41%)	0.04% (0.01%)	3.67% (3.52%)	5.06% / 4.5%
3	10.993×10^5 (44.9%)	1	98.4% (0.43%)	1.12% (49.64%)	0.49% (28.6%)	1.61% / 1.0%
4	31.598×10^5 (42.44%)	1	99.64% (0.1%)	0.33% (31.57%)	0.04% (26.28%)	0.37% / 0.17%

3.3 FSA-PBnB Discussion

We developed two classification criteria to use in FSA-PBnB and we tested the performance of the algorithms for the identification of the feasible set for the Sinusoidal test function. From the experimental results both the Pointwise-Comparison version and Quantile-Comparison version, show satisfactory results in terms of power of pruning and conservativeness.

Pointwise-Comparison uses the aggregated feasibility measure to evaluate the feasibility of each point. While it can identify the feasible set well, it does not make use of the information from each constraint in the sub-regions.

On the other hand, Quantile-Comparison evaluates the feasibility for each constraint and we can interpret the sub-regions by analyzing the feasibility metric (feasibility distance) and find the critical or the binding constraint in the optimization problem. As mentioned in chapter 1, the feasibility problem plays an important role in Pump Scheduling Optimization

(PSO). Indeed, our algorithm can provide the insights on the safety of the working condition under several pump speed settings to the engineers. In chapter 4, we apply the Quantile-Comparison FSA-PBnB on the Water Distribution Network and show how to analyze and provide the insights from the output of the approximated feasible set.

4 CHAPTER 4: NUMERICAL RESULTS ON WATER DISTRIBUTION NETWORK

In this section, we illustrate the performance of the Quantile-Comparison FSA-PBnB on two different Water Distribution Networks (WDNs) examples. The hydraulic networks are modeled by using the EPANet simulator (refer to Chapter 2, section 2.1 for details on the simulation tool). We have engaged in collaboration with Professor Antonio Candelieri and a PhD student Riccardo Perego from Università Milano Bicocca to work on the project. Our collaborators developed the WDN models in EPANet and exposed the simulator as a webservice. From our side, we created the connection between the simulator and the algorithm which was developed in Python3.6.

Section 4.1 presents the preliminary results obtained for the simple network case known as Net1 (refer to Figure 2-1), while in section 4.2, we consider a real network "Abbiategrasso pilot", located in the southern part of Milan (as shown Figure 1-1)

4.1 Simple Network - Net 1

The Net1 example is provided within the EPANet package. The main hydraulic components are shown in Figure 2-1, this network has 1 variable speed pump, 9 nodes, 1 storage tank and 1 reservoir. The simulation horizon was set to 2hrs with two time slots of 1 hours each, i.e., the pump speed can take one value for each hour. The electricity tariff is set to 0.0244 [$/kWh].

4.1.1 Preliminary Simulation Study on Net1 Example

For the Net1 Example feasibility is entirely determined by the pressure at each node of the network, i.e., considering the EPANet output (refer to section 2.1 for more details). To understand the distribution of pressures in the Net1 case, we ran EPANet over 10,000

configurations in terms regulation of the variable speed pump (uniformly sampled in the interval $(0,1) \times (0,1)$ for the two time-slots referring to the regulations during the first and second hour, respectively). The output from EPANet returns 2 values of pressure (one for each time slot) at each node, as shown in Table 4-1, reporting an example of the output when the input speed is [0.5,0.8].

Table 4-1 Output of Pressure Matrix from Simulator

	1st time slot	**2nd time slot**
Node1	71.329369	76.501038
Node2	74.332375	74.479042
Node3	73.606232	73.572853
Node4	76.424744	76.473328
Node5	74.84594	74.734673
Node6	69.352013	68.95034
Node7	74.816154	86.274994
Node8	72.2006	73.062622
Node9	1.767739	0.080292

Since the simulator returns a matrix of pressures, we compute an aggregated measure of pressure. We consider the pressure reached at each time slot to be constrained to be larger than a lower bound of pressure $\vartheta_c^{\ell} = 0$. Hence, we have c=1,2; wher one constraint is needed for each time slot and we only have a lower bound, i.e., $\vartheta_c^{u} = \infty, \mathrm{c} = 1,2$. For each time slot, we only look at the junction achieving the minimum pressure as representative of the corresponding time step. The feasibility distance vector for each constraint (as described in equation (3.1) can be defined as follows:

$$d(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) - \vartheta_c^{\ell} \\ f_2(\boldsymbol{x}) - \vartheta_c^{\ell} \end{bmatrix}$$

Where $f_{\cdot}(x) = \min_{n=1,\dots,N} \mathscr{p}_{t.,n}$, where N is the number of the nodes in the network and $\mathscr{p}_{t.,n}$ is the pressure at the n-th node during time slot t..

The aggregate measure of pressure is the Euclidean distance from a lower bound of $\vartheta_c^{\ell} = 0$ to the minimum pressure of each time slot, i.e.,

$$\mathrm{d}_E(\boldsymbol{x}) = \sqrt{\sum_{c=1}^{2} \left((f_c(\boldsymbol{x}) - \vartheta_c^{\ell})^{-} \right)^2} \tag{4-1}$$

When this aggregated pressure measure equals 0, the corresponding pump speed is considered to be infeasible working condition.

Results from the simulation study are shown in Figure 4-1, where each control value set for the pump in two time-slot $\mathbf{x} \in (0,1) \times (0,1)$ ("x_time_slot1" and "x_time_slot2") and "Pressure_distance" represent the aggregated pressure measure.

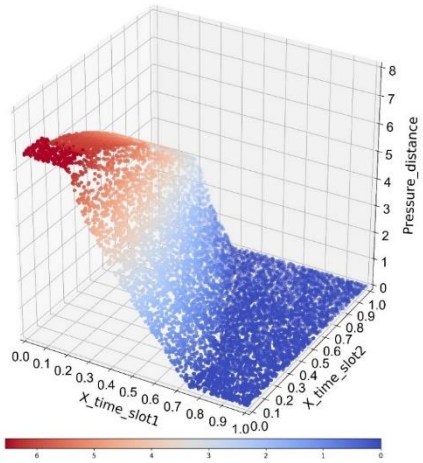

Figure 4-1 Aggregate Pressure Measure for Net1

4.1.2 Implementation of Quantile Comparison FSA-PBnB to Net1 Example

As mentioned multiple times in this thesis, the EPANet simulator is a black-box. By connecting to the simulator through the webservice, we can embed EPANet within the FSA-PBnB algorithm with Python3.6.

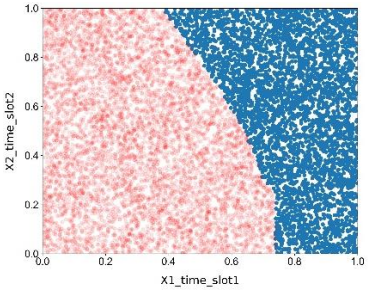

Figure 4-2 Projection of Pressure Distance

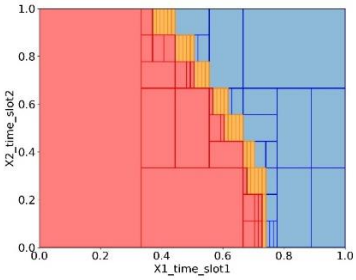

Figure 4-3 Approximation of Feasible Region(Iteration=6)

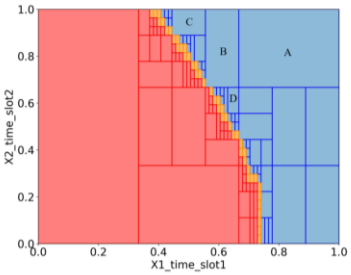

Figure 4-4 Approximation of Feasible Region(Iteration=7)

Figure 4-2 shows the projection of pressure distance in Figure 4-1 onto X1_time_slot1and X2_time_slot2, the feasible points ($\mathrm{d}_E(\boldsymbol{x}) = \mathbf{0}$) are in blue and infeasible points ($\mathrm{d}_E(\boldsymbol{x}) > \mathbf{0}$) are in red.

Figure 4 3 and Figure 4 4 present the results obtained by an application of the FSA-PBnB with a different number of iterations K. The maintained sub-regions are in blue, pruned in red, and undecided in orange. The approximation of the feasible region (in blue) in Figure 4 3 is close to the area in Figure 4 2. Figure 4 3 shows that, at iteration 6, after 3707 simulation runs, the algorithm can better refine the region profile. The feasible region is small compared to the overall search space and the undecided regions form an “orange belt” which can be interpreted as the feasibility boundary in terms of the pressure. After the 7th iteration, the number of simulation runs grows up to 8025, but only a small part of the undecided sub-regions shifts to the maintained sub-regions. As a result, the exponentially growing effort in simulation does not appear to be justified by the improvement. Hence, users have to trade the sampling effort off against the accuracy level of the approximated feasible region.

According to the graph, it can be observed that the leftmost red rectangle is pruned at the 1st iteration, which indicates the searching space reduces to $\frac{2}{3}$ at the very beginning and save the sampling effort to the next iteration.

As mentioned in section 3.1.2.2, with Quantile Comparison FSA-PBnB, we estimate the random feasibility distance parameters with equation (3.2) and the algorithm finds the lower quantile to decide the maintained regions (also called feasible region). Table 4-2 shows the lower quantile of the feasibility measure at each time step (constraint) of four randomly chosen maintained regions (as indicated in Figure 4-4). Details for all the maintained regions are reported in the appendix A.

Table 4-2 Feasibility Measures for Selected Regions

Sub-Region	*A*	*B*	*C*	*D*
Feasible Speed Range [X1_time_slot1] [X2_time_slot2]	 [0.667, 1] [0.667, 1]	 [0.556, 0.667] [0.667, 1]	 [0.444, 0.556] [0.889, 1]	 [0.63, 0.667] [0.556, 0.667]
$Q^{\ell}_{c\_1st\ hour}$	3.29	2.18	1.18	2.93
$Q^{\ell}_{c\_2nd\ hour}$	1.32	0.29	0.16	0.24

We can analyze the safety of the regions by means of $Q^{\ell}_{c\_1st\ hour}$ and $Q^{\ell}_{c\_2st\ hour}$. Assume the larger the value is, the safer the working condition is, then we can observe that region *A* might be the safest range to operate the pump for the Net1; however, regarding to the corresponding speed range, the higher speed might yield more energy cost.

In short, the distance measure does provide some information to engineers to understand the water distribution system and they can choose the desired regions to run the optimality phase.

4.2 Abbiategrasso Pilot Network – 2 Pumps with 2 Time-Slots

The second WDN test case is the urban area of Milan (Figure 1-1). We consider the "Abbiategrasso pilot", located in the southern part of the city. The model consists of 7546 nodes, 6073 pipes, 1961 valves, 1 reservoir and two variable speed pumps. The model is built in EPANet and the hydraulic components are shown in Figure 4-5. The electricity tariff with 0.0244 $/kWh for 00:00-07:00 and 0.1194 $/kWh for 08:00-24:00 is considered for simulations.

Figure 4-5 Abbiategrasso Pilot Model as Presented in the EPANet Interface

The pump operation of variable speed pumps (VSPs) is proposed by relative speed within interval $(0,1)$ in two time-slots (referring to the regulations during the first and second half of the day). The input of the simulator is the speed of 2 pumps with 2 time-slots, which is arranged as:

$$[\text{pump1\_timeslot1}, \text{pump1\_timeslot2}, \text{pump2\_timeslot1}, \text{pump2\_timeslot2}]$$

The simulation duration is 24hrs with two time slots. The first time slot controls 00:00-07:00 hours and the second time slot controls 07:00-24:00 hours. i.e., the pump speed can take two values during the day.

Since the simulation duration is based on 24hrs simulation, the EPANet run returns 24 values of pressure at the pump (one for every hour).

The output of the pressure matrix is illustrated below:

$$\text{Pressure} = \begin{bmatrix} p_{n=1,t1} & \cdots & p_{n=1,t24} \\ \vdots & \ddots & \vdots \\ p_{n=7546,t1} & \cdots & p_{n=7546,t24} \end{bmatrix}$$

According to the data provided from the company Metropolitana Milanese S.p.A, (MM) [2], the lower bound of the pressure in "Abbiategrasso pilot" is 20 [Pa] and no upper bound is provided.

For the real case, Abbiategrasso pilot model, in our experiment, again, feasibility is entirely determined by the pressure at each node of the network. We consider the pressure reached at each time step (1hour) to be constrained to be larger than the lower bound of the pressure. As a result, the number of constraints in the experiment is 24. The experiment setting is the same in section 4.1.2. We find the most infeasible pressure node (junction) (the minimum pressure) in each column to stand for the pressure of the corresponding time step.

Experiment Result

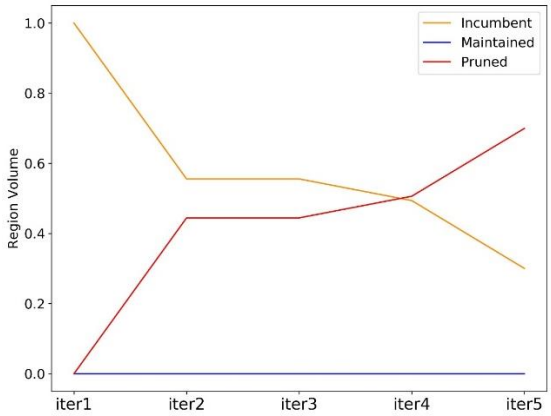

Figure 4-6 Volume Change in Pruned/Maintained/Undecided region

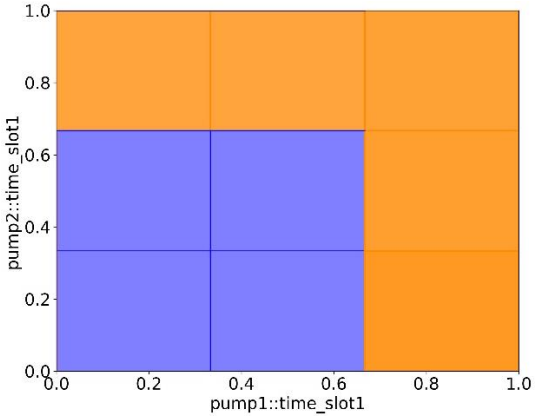

Figure 4-7 Region Result on Pump1::time_slot1 and Pump2_time_slot1

After FSA-PBnB terminated at iteration 5 with 5948 simulation runs, there is up to 70% of total volume being pruned. However, there is no region being classified as maintained region (feasible region) at this stage. All the remaining regions belong to the undecided regions. Figure 4-6 shows that the ratio of undecided volume and the ratio of volume pruned keep changing by iterations, but the ratio of volume maintained is always 0. We presume that the real-world problem is supposed to be more complicated than the theoretical case. In this experiment, running 5 iterations is not enough to shape the feasibility profile. Therefore, it may require more partitioning and sampling to identify the feasible regions from the undecided regions.

We display the obtained result with two dimensions (pump::timeslot2 × pump2::timeslot2) of four-dimensional sub-regions in Figure 4-7. The pruned regions are in blue and the undecided regions are in orange. Regarding the pruned regions, we can observe that, regardless of the pump speed ranges for pump1::timeslot2 and pump2::timeslot2, as long

as both of the speed of pump1::timeslot1 and pump2::timeslot2 are less than 0.67 simultaneously, the pump speed configuration is infeasible.

Furthermore, as mentioned in 3.1.2.2, Quantile Comparison FSA-PBnB returns the remaining (sum of maintained and incumbent) regions and the feasibility distance of each constraint within the sub-regions. In this regard, Figure 4-8 and Figure 4-9 demonstrate the feasibility distance metric in two undecided sub-regions, respectively.

In Figure 4-8, the distance metric defined in equation (3.1) is reported for all the points sampled up to the fifth iteration of the FSA-PBnB algorithm for the undecided sub-region defined by the four vertexes (x^l_{1,t_1}=0.66, x^u_{1,t_1}=71),(x^l_{1,t_2}=0, x^u_{1,t_2} 1), (x^l_{2,t_1}=0.66, x^u_{2,t_1} 1), and (x^l_{2,t_2}=0.15, x^u_{2,t_2}=0.19). The x-axis represents each hour of the day for which EPANet returns the desired values of pressure. It is important to highlight that we can use only two speeds. One is in the interval [00:00AM-7:00AM] and the second set up for the rest of the day. As a result, we notice a clear difference in the pressure distribution when we compare the first hours in the day against the later timings. Referring to the green line as the feasibility reference, it is possible to observe that the second time slot is more likely to be infeasible, while the first time slot does not show particular problems. However, the presence of both feasible and infeasible samples leads to the inability to maintain/prune the region.

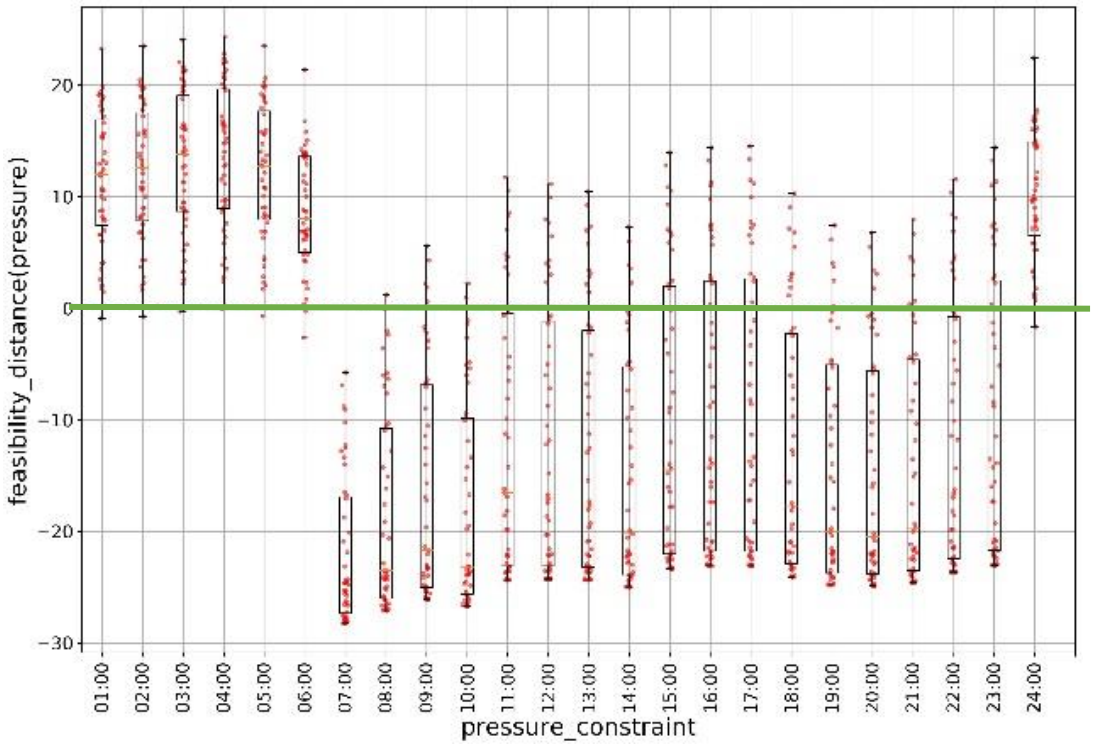

Figure 4-8 Distribution of Feasibility Distance(pressure) in Sub-region [0.66-71]×[0-1] × [0.66-1] × [0.15-0.19]

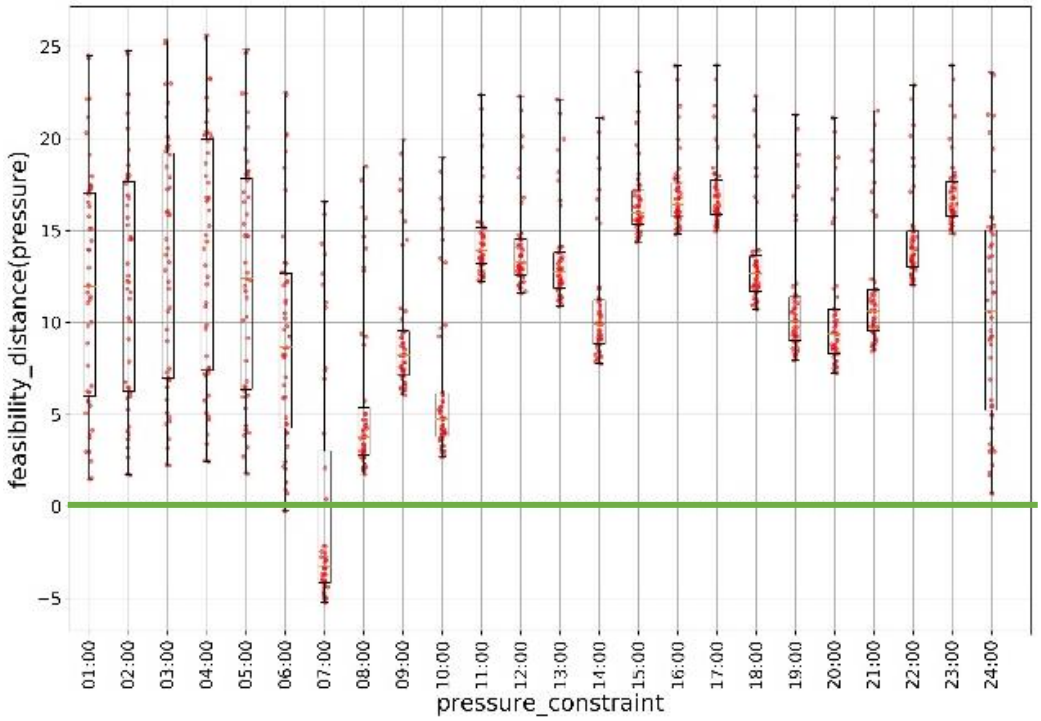

Figure 4-9 Distribution of Feasibility Distance(pressure) in Sub-region [0.66-71] × [0-1] × [0.66-1] × [0.96-1]

5 CHAPTER 5: CONCLUSIONS AND FUTURE RESEARCH

In this thesis, a new feasibility determination approach is proposed to analyze the quality of working conditions in complex water distribution networks (WDNs). Most of the literature focuses on cost minimization, fewer contributions look at the problem from a feasibility perspective. Nonetheless, feasibility is a critical issue in WDNs because if the network is set up to work in "extreme" conditions, it has higher possibility to leak and or break. In this thesis, we analyze the feasibility of the WDNs operations. Due to the complexity of the water supply system, it is difficult to assess the operational feasibility in closed form and a simulator needs to be used instead. We adopt a hydraulic solver EPANet 2.0 to simulate the response of the water distribution network with varying pump operation. We propose a novel algorithm called Feasible Set Approximation – PBnB (FSA-PBnB) to tackle the feasibility determination problem. The algorithm is tested to one theoretical function, Sinusoidal function, with 2,3, and 4 dimensional cases. The obtained results show that PBnB (FSA-PBnB) successfully prunes large amount of undesired regions and reaches satisfactory approximations of the feasible set. Results also put the light on the trade-off between the sampling effort and the marginal gain in terms of accuracy of the identification on feasible region.

The algorithm was applied to two different Water Distribution Networks. First, we considered a simple network Net1 with one variable speed pump (VSP) and a two-hour simulation duration. The result shows the ratio of approximated feasible region volume (maintained region) is 37.8%, and ratio of infeasible region (pruned region) is around 60%, which leads to a significant reduction of the solution space.

To the real network, Abbiategrasso pilot example, after running 5 iterations, while there is no feasible region being identify, FSA-PBnB pruned 70% of initial solution space volume. To achieve long-term planning as well as short-term (hourly) control of networks, it is necessary to do further studies in feasibility measures like Load and Demand. Additionally, the dependency between constraints is required to do further investigation to describe the working condition of networks more realistic.

Applying FSA-PBnB with these feasibility measures helps to better understand the working conditions in the network with respect to different pump operating conditions. From the engineering perspective, results can be integrated with domain knowledge to better characterize and design a safe water distribution system.

REFERENCES

[1] Gama, M. C., Lanfranchi, E. A., Pan, Q., & Jonoski, A. (2015). Water distribution network model building, case study: Milano, Italy. Procedia Engineering, 119, 573-582.

[2] Castro-Gama, M., Pan, Q., Lanfranchi, E. A., Jonoski, A., & Solomatine, D. P. (2017). Pump scheduling for a large water distribution network. Milan, Italy. Procedia Engineering, 186, 436-443.

[3] Copeland, C., and N. Carter. 2017. "Energy-Water Nexus: The Water Sector's Energy Use". Congressional Research Service Report R43200.

[4] G. Pedrielli, and Z. B. Zabinsky. Collaborative Research: Advancing Analysis and Control of Complex Water Distribution Network.

[5] Candelieri, A., Soldi, D., & Archetti, F. (2015). Short-term forecasting of hourly water consumption by using automatic metering readers data. Procedia Engineering, 119, 844-853.

[6] Mala-Jetmarova, H., Sultanova, N., & Savic, D. (2017). Lost in optimisation of water distribution systems? A literature review of system operation. Environmental Modelling & Software, 93, 209-254.

[7] Rossman, L. A. (2000). EPANet 2 users manual. water supply and water resources division, national risk management research laboratory. Cincinnati, United States Environmental Protection Agency.

[8] Balut, A., & Urbaniak, A. (2011, May). Management of water pipeline networks supported by hydraulic models and information systems. In Carpathian Control Conference (ICCC), 2011 12th International (pp. 16-21). IEEE.

[9] Paluszczyszyn, D., Skworcow, P., & Ulanicki, B. (2015). Modelling and simulation of water distribution systems with quantised state system methods. Procedia Engineering, 119, 554-563.

[10] Schmid, R. (2002). Review of modeling software for piped distribution networks. Skat_foundation. St. Gallen.

[11] Wang, H., Wang, H., & Zhu, T. (2017). A new hydraulic regulation method on district heating system with distributed variable-speed pumps. Energy Conversion and Management, 147, 174-189.

[12] Zabinsky, Z. B., Wang, W., Prasetio, Y., Ghate, A., & Yen, J. W. (2011, December). Adaptive probabilistic branch and bound for level set approximation. In Proceedings of the Winter Simulation Conference (pp. 4151-4162). Winter Simulation Conference.

[13] Pasha, M. F. K., & Lansey, K. (2009). Optimal pump scheduling by linear programming. In World Environmental and Water Resources Congress 2009: Great Rivers (pp. 1-10).

[14] Chase, D. V., & Ormsbee, L. E. (1989). Optimal pump operation of water distribution systems with multiple storage tanks. In Water Resources Planning and Management (pp. 733-736). ASCE.

[15] Sterling, M. J. H., & Coulbeck, B. (1975). A dynamic programming solution to optimization of pumping costs. Proceedings of the Institution of Civil Engineers, 59(4), 813-818.

[16] Nicklow, J., Reed, P., Savic, D., Dessalegne, T., Harrell, L., Chan-Hilton, A., ... & Zechman, E. (2009). State of the art for genetic algorithms and beyond in water resources planning and management. Journal of Water Resources Planning and Management, 136(4), 412-432.

[17] Savic, D. A., Walters, G. A., & Schwab, M. (1997, April). Multiobjective genetic algorithms for pump scheduling in water supply. In AISB international workshop on evolutionary computing (pp. 227-235). Springer, Berlin, Heidelberg.

[18] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. science, 220(4598), 671-680.

[19] De Paola, F., Fontana, N., Giugni, M., Marini, G., & Pugliese, F. (2016). An application of the Harmony-Search Multi-Objective (HSMO) optimization algorithm for the solution of pump scheduling problem. Procedia engineering, 162, 494-502.

[20] Biles, W. E., Kleijnen, J. P., van Beers, W., & Van Nieuwenhuyse, I. (2007, December). Kriging metamodeling in constrained simulation optimization: an explorative study. In Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come (pp. 355-362). IEEE Press.

[21] Bertsekas, D. P. (2014). Constrained optimization and Lagrange multiplier methods. Academic press.

[22] Huang, H., and Z. B. Zabinsky. 2017. “Partition-based Approach to Level Set Approximation for Simulation Optimization”. Paper Submitted for Publication to INFORMS Journal of Computing.

APPENDIX A

MAINTAINED REGIONS OF NET1

Region indices	X1_time slot1	X1_time slot2	Lower quantile of pressure_distance (con1)	Lower quantile of pressure_distance (con2)
1.	[0.66666, 0.99999]	[0.66666, 0.99999]	3.29	1.32
2.	[0.55555, 0.66666]	[0.66666, 0.99999]	2.18	0.29
3.	[0.77777, 0.88888]	[0. , 0.33333]	4.14	0.2
4.	[0.88888, 0.99999]	[0. , 0.33333]	5.36	1.37
5.	[0.77777, 0.88888]	[0.33333, 0.66666]	4.29	0.92
6.	[0.88888, 0.99999]	[0.33333, 0.66666]	5.48	1.86
7.	[0.44444, 0.55555]	[0.88888, 0.99999]	1.18	0.16
8.	[0.66666, 0.77777]	[0.44444, 0.55555]	3.29	0.14
9.	[0.66666, 0.77777]	[0.55555, 0.66666]	3.34	0.68
10.	[0.62963, 0.66667]	[0.55555, 0.66666]	2.93	0.24
11.	[0.51852, 0.55556]	[0.77777, 0.88888]	1.9	0.19
12.	[0.74074, 0.77778]	[0.11111, 0.22222]	4.04	0.02
13.	[0.74074, 0.77778]	[0.22222, 0.33333]	4.03	0.04
14.	[0.7037 , 0.74074]	[0.33333, 0.44444]	3.67	0.16
15.	[0.74074, 0.77778]	[0.33333, 0.44444]	4.04	0.51
16.	[0.61729, 0.62964]	[0.55555, 0.66666]	2.84	0.08
17.	[0.50618, 0.51853]	[0.77777, 0.88888]	1.81	0.02
18.	[0.74074, 0.75309]	[0. , 0.11111]	4.04	0.02
19.	[0.75309, 0.76544]	[0. , 0.11111]	4.16	0.13
20.	[0.76544, 0.77779]	[0. , 0.11111]	4.27	0.25
21.	[0.62963, 0.64198]	[0.51852, 0.55556]	2.96	0.02
22.	[0.64198, 0.65433]	[0.48148, 0.51852]	3.07	0.01
23.	[0.64198, 0.65433]	[0.51852, 0.55556]	3.07	0.15
24.	[0.65433, 0.66668]	[0.48148, 0.51852]	3.19	0.12
25.	[0.65433, 0.66668]	[0.51852, 0.55556]	3.18	0.25
26.	[0.58025, 0.5926]	[0.62963, 0.66667]	2.49	0.02
27.	[0.59259, 0.60494]	[0.62963, 0.66667]	2.61	0.14
28.	[0.60494, 0.61729]	[0.59259, 0.62963]	2.72	0.09
29.	[0.60494, 0.61729]	[0.62963, 0.66667]	2.71	0.24
30.	[0.40741, 0.41976]	[0.96296, 1.]	0.95	0.12
31.	[0.41976, 0.43211]	[0.92592, 0.96296]	1.05	0.02
32.	[0.41976, 0.43211]	[0.96296, 1.]	1.06	0.24
33.	[0.43211, 0.44446]	[0.92592, 0.96296]	1.16	0.18
34.	[0.43211, 0.44446]	[0.96296, 1.]	1.16	0.41
35.	[0.53087, 0.54322]	[0.74074, 0.77778]	2.04	0.11
36.	[0.54322, 0.55557]	[0.7037 , 0.74074]	2.14	0.
37.	[0.54322, 0.55557]	[0.74074, 0.77778]	2.16	0.22
38.	[0.46914, 0.48149]	[0.85185, 0.88889]	1.48	0.1
39.	[0.48148, 0.49383]	[0.85185, 0.88889]	1.6	0.23
40.	[0.49383, 0.50618]	[0.81481, 0.85185]	1.71	0.12
41.	[0.49383, 0.50618]	[0.85185, 0.88889]	1.72	0.36
42.	[0.7037 , 0.71605]	[0.2963 , 0.33334]	3.67	0.07
43.	[0.71605, 0.7284]	[0.2963 , 0.33334]	3.79	0.19
44.	[0.7284 , 0.74075]	[0.25926, 0.2963]	3.9	0.03
45.	[0.7284 , 0.74075]	[0.2963 , 0.33334]	3.91	0.31

46.	[0.67901, 0.69136]	[0.40741, 0.44445]	3.42	0.07
47.	[0.69136, 0.70371]	[0.33333, 0.37037]	3.54	0.01
48.	[0.69136, 0.70371]	[0.37037, 0.40741]	3.55	0.1
49.	[0.69136, 0.70371]	[0.40741, 0.44445]	3.54	0.2

APPENDIX B

UNDECIDED REGIONS OF ABBIATEGRASSO PILOT NETWORK

Region indices	Pump1_time slot1	Pump1_time slot2	Pump2_time slot1	Pump2_time slot2
1.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.03704, 0.07408]
2.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.22222, 0.25926]
3.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.51852, 0.55556]
4.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.62963, 0.66667]
5.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.66666, 0.7037]
6.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.7037 , 0.74074]
7.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.74074, 0.77778]
8.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.77777, 0.81481]
9.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.81481, 0.85185]
10.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.85185, 0.88889]
11.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.88888, 0.92592]
12.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.92592, 0.96296]
13.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.96296, 1.]
14.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.22222, 0.25926]
15.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.25926, 0.2963]
16.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.44444, 0.48148]
17.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.48148, 0.51852]
18.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.55555, 0.59259]
19.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.7037 , 0.74074]
20.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.74074, 0.77778]
21.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.77777, 0.81481]
22.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.81481, 0.85185]
23.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.85185, 0.88889]
24.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.88888, 0.92592]
25.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.92592, 0.96296]
26.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.96296, 1.]
27.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0. , 0.03704]
28.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.11111, 0.14815]
29.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.18519, 0.22223]
30.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.22222, 0.25926]
31.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.2963 , 0.33334]
32.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.37037, 0.40741]
33.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.40741, 0.44445]
34.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.59259, 0.62963]
35.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.62963, 0.66667]
36.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.66666, 0.7037]
37.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.7037 , 0.74074]
38.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.74074, 0.77778]
39.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.77777, 0.81481]
40.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.81481, 0.85185]
41.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.85185, 0.88889]
42.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.88888, 0.92592]
43.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.92592, 0.96296]
44.	[0.66666, 0.99999]	[0., 1.]	[0. , 0.33333]	[0.96296, 1.]
45.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.03704, 0.07408]
46.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.07408, 0.11112]

47.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.11111, 0.14815]
48.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.25926, 0.2963]
49.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.33333, 0.37037]
50.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.40741, 0.44445]
51.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.66666, 0.7037]
52.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.7037 , 0.74074]
53.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.74074, 0.77778]
54.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.77777, 0.81481]
55.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.81481, 0.85185]
56.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.85185, 0.88889]
57.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.88888, 0.92592]
58.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.92592, 0.96296]
59.	[0.66666, 0.99999]	[0., 1.]	[0.33333, 0.66666]	[0.96296, 1.]
60.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.14815, 0.18519]
61.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.2963 , 0.33334]
62.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.33333, 0.37037]
63.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.59259, 0.62963]
64.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.62963, 0.66667]
65.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.66666, 0.7037]
66.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.7037 , 0.74074]
67.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.74074, 0.77778]
68.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.77777, 0.81481]
69.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.81481, 0.85185]
70.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.85185, 0.88889]
71.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.88888, 0.92592]
72.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.92592, 0.96296]
73.	[0.66666, 0.99999]	[0., 1.]	[0.66666, 0.99999]	[0.96296, 1.]
74.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.03704, 0.07408]
75.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.22222, 0.25926]
76.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.51852, 0.55556]
77.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.62963, 0.66667]
78.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.66666, 0.7037]
79.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.7037 , 0.74074]
80.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.74074, 0.77778]
81.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.77777, 0.81481]
82.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.81481, 0.85185]
83.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.85185, 0.88889]
84.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.88888, 0.92592]
85.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.92592, 0.96296]
86.	[0. , 0.33333]	[0., 1.]	[0.66666, 0.99999]	[0.96296, 1.]
87.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.22222, 0.25926]
88.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.25926, 0.2963]
89.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.44444, 0.48148]
90.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.48148, 0.51852]
91.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.55555, 0.59259]
92.	[0.33333, 0.66666]	[0., 1.]	[0.66666, 0.99999]	[0.7037 , 0.74074]