Université de Montréal

Development of New Scenario Decomposition Techniques for Linear and Nonlinear Stochastic Programming

par
Shohre Zehtabian

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de Maître ès sciences (M.Sc.)
en computer science

August, 2016

RÉSUMÉ

Une approche classique pour traiter les problèmes d'optimisation avec incertitude à deux- et multi-étapes est d'utiliser l'analyse par scénario. Pour ce faire, l'incertitude de certaines données du problème est modélisée par vecteurs aléatoires avec des supports finis spécifiques aux étapes. Chacune de ces réalisations représente un scénario. En utilisant des scénarios, il est possible d'étudier des versions plus simples (sous-problèmes) du problème original. Comme technique de décomposition par scénario, l'algorithme de recouvrement progressif est une des méthodes les plus populaires pour résoudre les problèmes de programmation stochastique multi-étapes. Malgré la décomposition complète par scénario, l'efficacité de la méthode du recouvrement progressif est très sensible à certains aspects pratiques, tels que le choix du paramètre de pénalisation et la manipulation du terme quadratique dans la fonction objectif du lagrangien augmenté. Pour le choix du paramètre de pénalisation, nous examinons quelques-unes des méthodes populaires, et nous proposons une nouvelle stratégie adaptive qui vise à mieux suivre le processus de l'algorithme. Des expériences numériques sur des exemples de problèmes stochastiques linéaires multi-étapes suggèrent que la plupart des techniques existantes peuvent présenter une convergence prématurée à une solution sous-optimale ou converger vers la solution optimale, mais avec un taux très lent. En revanche, la nouvelle stratégie paraît robuste et efficace. Elle a convergé vers l'optimalité dans toutes nos expériences et a été la plus rapide dans la plupart des cas. Pour la question de la manipulation du terme quadratique, nous faisons une revue des techniques existantes et nous proposons l'idée de remplacer le terme quadratique par un terme linéaire. Bien que qu'il nous reste encore à tester notre méthode, nous avons l'intuition qu'elle réduira certaines difficultés numériques et théoriques de la méthode de recouvrement progressif.

Mots clés: programmation stochastique, programmation multi-étapes, lagrangien augmenté, méthodes proximales, paramètre de pénalisation, terme quadratique, programmation élastique.

ABSTRACT

In the literature of optimization problems under uncertainty a common approach of dealing with two- and multi-stage problems is to use scenario analysis. To do so, the uncertainty of some data in the problem is modeled by stage specific random vectors with finite supports. Each realization is called a scenario. By using scenarios, it is possible to study smaller versions (subproblems) of the underlying problem. As a scenario decomposition technique, the progressive hedging algorithm is one of the most popular methods in multi-stage stochastic programming problems. In spite of full decomposition over scenarios, progressive hedging efficiency is greatly sensitive to some practical aspects, such as the choice of the penalty parameter and handling the quadratic term in the augmented Lagrangian objective function. For the choice of the penalty parameter, we review some of the popular methods, and design a novel adaptive strategy that aims to better follow the algorithm process. Numerical experiments on linear multi-stage stochastic test problems suggest that most of the existing techniques may exhibit premature convergence to a sub-optimal solution or converge to the optimal solution, but at a very slow rate. In contrast, the new strategy appears to be robust and efficient, converging to optimality in all our experiments and being the fastest in most of them. For the question of handling the quadratic term, we review some existing techniques and we suggest to replace the quadratic term with a linear one. Although this method has yet to be tested, we have the intuition that it will reduce some numerical and theoretical difficulties of progressive hedging in linear problems.

Keywords: stochastic programming, multi-stage programming, scenario analysis, augmented Lagrangian, proximal methods, penalty parameter, quadratic term, elastic programming.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

a.s.	almost surely
API	Application Programming Interface
COIN-OR	Computational Infrastructure for Operation Research
LP	Linear Programming
MIP	Mixed Integer Programming
MPS	Mathematical Programming System
NA	NonAnticipativity
OR	Operations Research
OSI	Open Source Interface
PHA	Progressive Hedging Algorithm
s.t.	such that
SMI	Stochastic Modeling Interfaces
SMPS	Stochastic Mathematical Programming System
SP	Stochastic Programming

NOTATION

Symbol	Definition
$'$	Transpose operator
$\hat{}$	Indicator of average value
$\cup$	Set union symbol
$\langle\cdot\rangle_{\mathcal{D}}$	Inner product on space $\mathcal{D}$
$\langle\cdot\rangle$	Scalar product
$\|.\|_{\mathcal{D}}$	Norm defined on space $\mathcal{D}$
$\|.\|$	Norm (Euclidean, L1, L∞)
$\sum$	Summation
$\max$	Maximum value
$\min$	Minimum value
B	Random event
c	Objective function coefficient (often indexed by stage, c_t)
$\mathcal{D}$	Space of all mappings over the scenario set $\mathcal{S}$
e	Vector of ones
$\mathcal{E}$	Equality constraints index set
$E_{\boldsymbol{\xi}}$	Mathematical expectation corresponding to $\boldsymbol{\xi}$
$E_{\boldsymbol{\xi}\|\dot{\boldsymbol{\xi}}}$	Mathematical conditional expectation corresponding to $\boldsymbol{\xi}$ given $\dot{\boldsymbol{\xi}}$
f	Objective function
$\mathcal{F}$	σ-field of subsets of Ξ
F	Functional
g_i	ith constraint (often indexed by stage, $g_{t,i}$)
G_t	Coefficient matrix of problem structure up to stage (not including) t
i	Constraint index
$\mathcal{I}$	Inequality constraints index set
j	Stage index up to stage (not including) t ($j < t$)
J	Aggregation operator
H	Coefficient matrix of problem structure (often indexed by stage, H_t)

k	Indicator of algorithm iteration
K	Orthogonal operator
L	Lagrangian function
L_A	Augmented Lagrangian function
$\mathcal{L}$	Space of admissible policies
m	Number of constraints (often indexed by stage, m_t)
$\overline{m}$	Number of equality constraints (often indexed by stage, $\overline{m}_t$)
n	Number of variables (often indexed by stage, t)
$\mathcal{N}$	Space of implementable policies
$\mathcal{M}$	Complementary space to $\mathcal{N}$
p	Probability corresponding to a given scenario (often indexed by scenario, p_s)
P	Probability distribution/measure ($P_{\boldsymbol{\xi}}$ corresponding to $\boldsymbol{\xi}$)
$\mathcal{P}_s$	Scenario subproblem
Q	Second-stage (multi-stage) value function
$\mathcal{Q}$	Second-stage (multi-stage) expected (recourse) function
$\mathbb{R}$	Real numbers set
$\mathbb{R}^n$	n-dimensional space of real numbers ($\mathbb{R}^n_+$, space of positive real numbers)
s	Scenario
(s)	Indicator of a specific scenario
S	Number of scenarios
$\mathcal{S}$	Set of all scenarios
$\mathcal{S}_t^{(s)}$	Bundle of scenarios
t	Stage index for multi-stage programs
T	Number of stages
W	Information price system (often indexed by stage, W_t)
x	Decision vector for first stage or multi stages (often indexed by stage, x_t)
$\bar{\boldsymbol{x}}$	Decision process

$\overline{x}_t$	Decision history up to and including stage t
X	Subset of $\mathbb{R}^n$
$\mathcal{X}$	Feasible set of first- or multi-stage decision vectors ($x \in \mathcal{X}$)
y	Second-stage decision vector
ω	Random experiment ($\omega \in \Omega$)
Ω	Set of all random experiments
Ξ	Support of the random vector $\boldsymbol{\xi}$
$\xi = \xi(\omega)$	Realization for random vector after random experiment ω (often indexed by stage, ξ_t)
$\bar{\boldsymbol{\xi}}$	History of random process (often indexed by stage, $\bar{\boldsymbol{\xi}}_t$)
$\bar{\xi}_t$	History of realization for random process up to and including stage t
λ	Lagrange (dual) multiplier vector
$\boldsymbol{\xi}$	Random vector/process (often indexed by stage, $\boldsymbol{\xi}_t$)
ρ	Penalty parameter
ζ	Scalar in penalty parameter initialization
ϵ	Convergence threshold
ϵ_d	Convergence threshold in dual space
τ_ρ	Coefficient in dynamic penalty parameter update
μ	Power in original dynamic penalty parameter update
δ	Coefficient in controlling penalty parameter update
Δ	Elasticity bound
$\gamma_1, \gamma_2, \gamma_3$	Scalars in adaptive penalty parameter update
α, ν, σ	Scalars in adaptive penalty parameter update
θ, β, η	Scalars in adaptive penalty parameter update
Θ	Elastic penalty parameter

There are two kinds of forecasters: those who don't know, and those who don't know they don't know.

John Kenneth Galbraith

CHAPTER 1

INTRODUCTION

Stochastic programming (SP) appeared in early 1950's [16], following fundamental achievements in linear and nonlinear programming. After it was realized that the presence of uncertainty in optimization models creates a need for new problem formulations, many years passed until the formulation and analysis of basic SP models [18]. Optimization problems involving stochastic models occur in almost all areas of science and engineering, from telecommunication and medicine to finance. This stimulates interest of formulating, analyzing and solving such problems. Today, SP offers a variety of models to address the presence of random data in optimization problems: for example, two- and multi-stage models, chance-constrained models, models involving risk measures (see e.g. [4, 35, 54]).

The most widely applied and studied SP models are *two-stage* linear programs. Here the decision maker takes some action in the first stage, after which a random event occurs affecting the outcome of the first-stage decision. A recourse decision can then be made in the second stage that compensates for the effects that might have been experienced as a result of the first-stage decision. One natural generalization of the two-stage model extends it to more stages (multi-stage). In *multi-stage* problems each stage consists of a decision followed by a set of observations of the uncertain parameters which are gradually revealed over time (in this thesis, we will focus on two- and multi-stage problems.)

Although two- and multi-stage stochastic programs are often regarded as the classical SP modeling paradigm, the discipline of SP has grown and broadened to cover a wide range of models and solution approaches. An alternative modeling approach uses so-called *chance-constraints*. Chance-constrained approach does not require that our decisions are feasible for (almost) every outcome of the random parameters, but requires feasibility with at least some specified probability. Despite the usefulness of such techniques, we will not consider them in our work, but will focus on recourse formulations.

A way of modeling the uncertain data is to consider some discrete realizations from a given probability space over time for those data. By using such a method, called *scenario analysis*, it is possible to solve and analyze two- and multi-stage SP problems. Due to the very large size and specific structure of the resulting problems, it is often advisable to employ decomposition techniques. The main idea is to break large problems into manageable pieces corresponding to individual scenarios (a scenario is a sequence of random variables' realizations), solve them and then come up with a good combined solution to the underlying large problem. One such method proposed by Rockafellar and Wets [51] is the *progressive hedging algorithm* (PHA) which is a proximal point based algorithm with splitting ability. This method relaxes the *nonanticipativity* (NA) constraints by introducing a quadratic penalty term with a nonnegative parameter in an augmented Lagrangian function. The NA constraints express that at each stage the decision should depend only on information available at the time of decision making and not on future information.

While 25 years old, PHA has been widely used in stochastic optimization problems (see e.g. [14, 26, 29, 42, 61]). Although PHA achieves a full separation of the scenario problems at each iteration, some practical issues, such as how to handle the quadratic term numerically/theoretically and also the choice of the penalty parameter, have always been major challenges for PHA users. In our research, we focus on these issues and try to suggest some solutions for them. We also aim to modify a stochastic open source interface, Stochastic Modeling Interface (SMI), in order to have access to the existing stochastic test problems in a special format (called SMPS format [21]). This way we can study the effect of our contributions to PHA on those test problems and create a stochastic benchmark tool for scenario decomposition techniques (for further details refer to Appendix I).

This thesis is organized as follows. We provide a literature review on two- and multi-stage SP problems, scenario analysis, scenario decomposition with an introduction to PHA and its theoretical and numerical history in Chapter 2. In this chapter, we also discuss our research motivations related to the choice of the penalty parameter and the introduction of a different penalty term. In Chapter 3, we introduce a novel adaptive approach [67]. Finally, we end the thesis with a conclusion and some suggestions for further research in Chapter 4.

CHAPTER 2

LITERATURE REVIEW AND RESEARCH MOTIVATIONS

In this chapter, we introduce SP problems as well as two- and multi-stage formulations of such problems. We then present the concept of scenario analysis for modeling the uncertain data in stochastic problems. Following this, progressive hedging algorithm as a scenario analysis technique is explained in details. Finally, we discuss motivations and directions to be taken in this research in the last section of this chapter.

2.1 Mathematical programming

A *mathematical program* is an optimization problem aiming to maximize or minimize an objective function possibly subject to some constraints. Unconstrained optimization problems can be introduced by the following standard form:

$$\min_{x \in \mathbb{R}^n} f(x),$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function. On the other hand, a constrained optimization problem can be written as

$$\min_{x \in X} f(x) \tag{2.1}$$

$$\text{s.t. } g_i(x) = 0, \; i \in \mathcal{E} \tag{2.2}$$

$$g_i(x) \leq 0, \; i \in \mathcal{I} \tag{2.3}$$

where $g_i : \mathbb{R}^n \to \mathbb{R}$ $(i \in \mathcal{I} \cup \mathcal{E})$, $\mathcal{I}$ and $\mathcal{E}$ are (disjoint) index sets for inequality and equality constraints, respectively. In addition, X is a subset of $\mathbb{R}^n$ and belongs to the domain of the functions f and g_i $(i \in \mathcal{I} \cup \mathcal{E})$. X is usually a subset of $\mathbb{R}^n_+$ or $\mathbb{R}^n$. The relations $g_i(x) = 0$ $(i \in \mathcal{E})$ and $g_i(x) \leq 0$ $(i \in \mathcal{I})$ are called equality and inequality constraints, respectively.

2.2 Stochastic problems

In many optimization problems, there are some data that may be considered uncertain. We here assume that data uncertainty can be represented as a random vector $\boldsymbol{\xi}$ which is defined on a probability space $(\Xi, \Omega, P_{\boldsymbol{\xi}})$. Here, Ξ is the support of $\boldsymbol{\xi}$, Ω is a σ-field [1] of subsets of Ξ, and $P_{\boldsymbol{\xi}}$ is the associated probability distribution/measure.

Considering the general mathematical program (2.1)–(2.3), the functions f and g_i, for $i = 1, \dots, m$ with $m = |\mathcal{E} \cup \mathcal{I}|$ where $|\mathcal{E} \cup \mathcal{I}|$ represents the cardinality of $\mathcal{E} \cup \mathcal{I}$, are not known very accurately, therefore we have

$$\text{“}\min_{x \in X}\text{”} f(x, \boldsymbol{\xi}) \tag{2.4}$$

$$\text{s.t. } g_i(x, \boldsymbol{\xi}) = 0, \ i \in \mathcal{E}, \tag{2.5}$$

$$g_i(x, \boldsymbol{\xi}) \leq 0, \ i \in \mathcal{I}. \tag{2.6}$$

If a vector $\xi = \xi(\omega)$ is the random vector realization known after the random experiment, with the vectors ω in Ω, then the above problem can be reformulated as

$$\text{“}\min_{x \in X}\text{”} f(x, \xi(\omega)) \tag{2.7}$$

$$\text{s.t. } g_i(x, \xi(\omega)) = 0, \ i \in \mathcal{E}, \tag{2.8}$$

$$g_i(x, \xi(\omega)) \leq 0, \ i \in \mathcal{I}. \tag{2.9}$$

where $X \subseteq \mathbb{R}^n$. An accurate probabilistic description of the random variables should be available, under the form of probability distributions, densities or, more generally, probability measures. Here, we assume that the probability distribution/measure P is given and independent of x, and for all x, $f(x, .) : \Xi \to \mathbb{R}$ and $g_i(x, .) : \Xi \to \mathbb{R}$ ($i = 1, \dots, m$) are random variables too.

Ideally, we look for some x that is feasible while minimizing the objective for all or for almost all possible values of ω in Ω. So the definition of feasibility depends on

1. A σ-algebra or σ-field over a set X is a nonempty collection $\dot{X}$ of subsets of X (including X itself) that is closed under the complement and countable unions of its members.

the problem at hand, in particular whether or not we are able to obtain some information about the value of $\xi(\omega)$, before choosing x. Similarly, optimality depends on the uncertainty involved and also on the performance of the objective in (almost) all cases. Therefore, we cannot solve (2.7)–(2.9) by finding the optimal solution for every possible value of ω in Ω and we should clarify the meanings of "$\min$" as well as of the constraints [2, 5].

One possibility is to consider the expectation of the objective function [5], such that (2.7)–(2.9) can be reformulated as:

$$\min_{x \in X} E_{\boldsymbol{\xi}}[f(x,\xi)] \tag{2.10}$$

$$\text{s.t. } g_i(x,\xi) = 0, \text{ a.s., } i \in \mathcal{E}, \tag{2.11}$$

$$g_i(x,\xi) \leq 0, \text{ a.s., } i \in \mathcal{I}. \tag{2.12}$$

where constraints are satisfied for almost all values of ξ. In probability theory, an event is said to happen almost surely (a.s.), if it happens with probability one, i.e. if $(\Xi, \mathcal{F}, P)$ is a probability space, one can say that an event B in $\mathcal{F}$ happens almost surely, if $P(B) = 1$.

2.2.1 Two-stage stochastic program with recourse

One important problem class in SP consists of recourse programs. In this section, we introduce the two-stage recourse problems, for which the set of decisions is divided into two groups:

1. A number of decisions that have to be taken before the experiment. All these decisions are called *first-stage* decisions and the period when these decisions are taken is called the *first stage*.
2. A number of decisions that can be taken after the experiment. They are called *second-stage* decisions. The corresponding period is called *second stage*.

First-stage decisions are represented by vector x_1, while second-stage decisions are represented by vector $x_2(\xi)$ or $x_2(\xi, x_1)$. Then the sequence of events and decisions can be summarized as

$$x_1 \to \xi \to x_2 \text{ or } x_2(\xi, x_1)$$

Therefore, the two-stage problem will be

$$\begin{aligned}
&\min_{x_1 \in X} f_1(x_1) + \mathcal{Q}(x_1) \\
&\text{s.t. } g_{1,i}(x_1) = 0,\ i = 1, \ldots, \overline{m}_1, \\
&\qquad g_{1,i}(x_1) \leq 0,\ i = \overline{m}_1 + 1, \ldots, m_1,
\end{aligned} \tag{2.13}$$

with the recourse $\mathcal{Q}(x_1) = E_{\boldsymbol{\xi}}[Q(x_1, \xi)]$ and

$$\begin{aligned}
Q(x_1, \xi) = &\min_{x_2} f_2(\xi, x_2(\xi)) \\
&\text{s.t. } g_{2,i}(\xi, x_1, x_2(\xi)) = 0,\ i = 1, \ldots, \overline{m}_2, \\
&\qquad g_{2,i}(\xi, x_1, x_2(\xi)) \leq 0,\ i = \overline{m}_2 + 1, \ldots, m_2,
\end{aligned} \tag{2.14}$$

where $\overline{m}_t$ is the number of equality constraints at stage t and m_t is the total number of constraints at stage t.

2.2.2 Multi-stage stochastic programming

Two-stage SP is a special case of a more general SP formulation, called multi-stage programming. In multi-stage setting, the uncertain data $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T$ is revealed gradually overtime, in T periods (it is usually assumed that the data in first stage is deterministic, i.e., $\boldsymbol{\xi}_1 = \xi_1$). The decision process can be shown as:

$$\text{decision}(x_1) \to \text{observation } (\boldsymbol{\xi}_2) \to \text{decision } (x_2) \to \\ \cdots \to \text{observation } (\boldsymbol{\xi}_T) \to \text{decision}(x_T).$$

To introduce the program, consider the random parameter as a data process:

$$\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T).$$

Then we define

$$\bar{\boldsymbol{\xi}}_t = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t\},$$

as the history of the random process up to and including stage t, for $t = 1, \ldots, T$. By

$$\bar{\xi}_t = \{\xi_1, \ldots, \xi_t\}.$$

we present the history of realization for $\bar{\boldsymbol{\xi}}_t$ up to and including stage t. ξ_t is the realization of random data $\boldsymbol{\xi}_t$ at stage t. If x_t is the decision at stage t, then the decision history up to and including stage t can be shown as

$$\overline{x}_t = \{x_1, \ldots, x_t\}.$$

for $t = 1, \ldots, T$. At stage t, we know $\bar{\xi}_t$ as well as $\overline{x}_{t-1}$. So the multi-stage program can be expressed as follows:

$$\begin{aligned} \min_{x_1} f_1(x_1) + E_{\boldsymbol{\xi}_2}[\min_{x_2} f_2(\overline{\xi}_2, x_1, x_2(\xi_1)) + E_{\boldsymbol{\xi}_3|\boldsymbol{\xi}_2}[\min_{x_3}\ f_3(\overline{\xi}_3, \overline{x}_2, x_3(\xi_3)) + \cdots + \\ E_{\boldsymbol{\xi}_T|\boldsymbol{\xi}_2,\ldots,\boldsymbol{\xi}_{T-1}}[\min_{x_T} f_T(\overline{\xi}_T, \overline{x}_{T-1}, x_T(\xi_T))]]] \end{aligned} \tag{2.15}$$

such that

$$\begin{aligned} & g_{1,i}(x_1) = 0, & & i = 1, \ldots, \overline{m}_1, \\ & g_{1,i}(x_1) \leq 0, & & i = \overline{m}_1 + 1, \ldots, m_1, \\ & g_{t,i}(\bar{\xi}_t, \bar{x}_{t-1}(\bar{\xi}_{t-1}), x_t(\xi_t)) = 0, \text{ a.s.}, & & i = 1, \ldots, \overline{m}_t,\ t = 2, \ldots, T, \\ & g_{t,i}(\bar{\xi}_t, \bar{x}_{t-1}(\bar{\xi}_{t-1}), x_t(\xi_t)) \leq 0, \text{ a.s.}, & & i = \overline{m}_t + 1, \ldots, m_t,\ t = 2, \ldots, T, \\ & x_t(\xi_t) \in X_t, & & t = 1, \ldots, T. \end{aligned} \tag{2.16}$$

where $x_1(\xi_1) = x_1$, and $X_t \subseteq \mathbb{R}^{n_t}$ such that $X = \{X_1, \ldots, X_T\}$ and $n_1 + \cdots + n_T = n$.

We rewrite (2.15)–(2.16) as

$$\min_{x_1 \in \mathcal{X}_1} f_1(x_1) + \sum_{t=2}^{T} E_{\boldsymbol{\xi}_2,\ldots,\boldsymbol{\xi}_t}[Q_t(\overline{\xi}_t, \overline{x}_{t-1}, x_t(\xi_t))], \tag{2.17}$$

where

$$\mathcal{X}_1 = \{g_{1,i}(x_1) = 0,\ i = 1,\ldots,\overline{m}_1,\ \text{and}\ g_{1,i}(x_1) \le 0,\ i = \overline{m}_1 + 1,\ldots,m_1,\ \text{and}\ x_1 \in X_1\}, \tag{2.18}$$

and

$$\begin{aligned} Q_t(\overline{\xi}_t, \overline{x}_{t-1}, x_t(\xi_t)) = \inf_{x_t(\xi_t) \in X_t} \{ & f_t(\bar{\xi}_t, \bar{x}_{t-1}, x_t(\xi_t)) \mid g_{t,i}(\bar{\xi}_t, \bar{x}_{t-1}, x_t(\xi_t)) = 0, \\ & i = 1,\ldots,\overline{m}_t,\ \text{and}\ g_{t,i}(\bar{\xi}_t, \bar{x}_{t-1}, x_t(\xi_t)) \le 0,\ i = \overline{m}_t + 1,\ldots,m_t\}. \end{aligned} \tag{2.19}$$

fro $t = 2,\ldots,T$ (for more details, we refer to [55]).

2.3 A scenario decomposition technique

In this section, we introduce the concept of scenario analysis as well as all the details of a scenario decomposition technique – the progressive hedging algorithm – and we then present all the structures, including uncertainty, forming the basis of this approach.

2.3.1 Scenario analysis

To control or analyze the systems involving some level of uncertainty, scenario analysis is one of the common practical approaches. In scenario analysis, the uncertainty of parameters or components of the system is modeled by a small number of subproblems derived from an underlying optimization problem. These subproblems correspond to different "scenarios" [51]. Mathematically speaking, a scenario s is sequence of outcomes $\xi_t,\ t = 1,\ldots,T$, denoted by

$$s = \left(\xi_1^{(s)}, \xi_2^{(s)}, \ldots, \xi_T^{(s)}\right)$$

associated to a probability $P_{\boldsymbol{\xi}}[\xi = s] = p_s$. Assume that the support set Ξ is finite, then $\mathcal{S} = \{1, \ldots, S\}$ is the set of scenarios.

A tree of scenarios represents the possible outcomes. It has nodes organized in levels which correspond to stages $1, \ldots, T$. At level $t = 1$, there is only the root of the tree, denoted by *Root* and it corresponds to the first-stage decisions, made before any realization of the random parameters. Each realization of $\boldsymbol{\xi}_{t+1}$, conditional to $\overline{\xi}_t$, is associated with a node (see for instance [2, 55]).

Figure 2.1 shows a tree which has three stages with Root at the first stage. Then from stage one to stage two, there are three realizations which generate nodes 2, 3 and 4. At the last stage, the number of possible realizations depends on the ancestor nodes, and is 3, 1 and 2 respectively, leading to a total of six nodes. These six nodes correspond to six scenarios where each is a path from the Root to a leaf. For example, Scenario 1 $= \{\xi_1^1, \xi_2^2, \xi_3^5\}$ is a path containing nodes $\{1, 2, 5\}$, Scenario 2 $= \{\xi_1^1, \xi_2^2, \xi_3^6\}$ is a path containing nodes $\{1, 2, 6\}$, ..., and Scenario 6 $= \{\xi_1^1, \xi_2^4, \xi_3^{10}\}$ is a path containing nodes $\{1, 4, 10\}$. Here, we show the realization of the random variables at a specific node nd, for $nd = 1, \ldots, 10$, of stage t by ξ_t^{nd}.

Using the concept of scenarios, we can reformulate the two-stage stochastic problem (2.13)-(2.14) as the following:

$$\min_{(x_1^{(1)}, \ldots, x_1^{(S)})} \sum_{s=1}^{S} p_s(f_1(x_1^{(s)}) + f_2(s, x_1^{(s)}, x_2^{(s)})) \tag{2.20}$$

subject to

$$g_{1,i}(x_1^{(s)}) = 0, \qquad i = 1, \ldots, \overline{m}_1,\ s = 1, \ldots, S, \tag{2.21}$$

$$g_{1,i}(x_1^{(s)}) \leq 0, \qquad i = \overline{m}_1 + 1, \ldots, m_1,\ s = 1, \ldots, S, \tag{2.22}$$

$$g_{2,i}(s, x_1^{(s)}, x_2^{(s)}) = 0, \qquad i = 1, \ldots, \overline{m}_2,\ s = 1, \ldots, S, \tag{2.23}$$

$$g_{2,i}(s, x_1^{(s)}, x_2^{(s)}) \leq 0, \qquad i = \overline{m}_2 + 1, \ldots, m_2,\ s = 1, \ldots, S, \tag{2.24}$$

$$x_t^{(s)} \in X_t, \qquad t = 1, 2,\ s = 1, \ldots, S, \tag{2.25}$$

$$x_1^{(s)} \text{ is nonanticipative}, \qquad s = 1, \ldots, S. \tag{2.26}$$

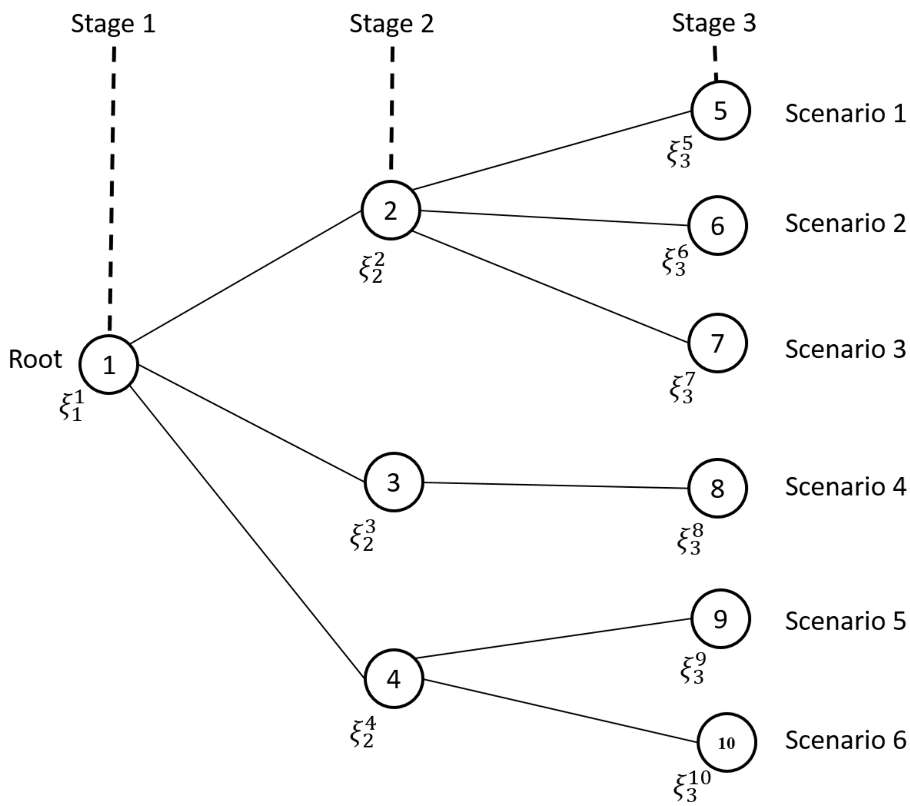

Figure 2.1 – Scenario tree with three stages and six scenarios

Constraints (2.26) are the NA constraints which link the scenarios together. To introduce the NA concept more precisely, we say that as long as two scenarios share a common history we should make the same decisions. Therefore, in Figure 2.1 nodes and arcs can be replicated to show NA. The new graph, shown in Figure 2.2, has the same number of stages and scenarios as in the scenario tree, but more nodes in the first and the second stages. This way we emphasize the fact that the scenarios sharing information at each stage should produce the same decisions. So Figure 2.2 basically shows that:

$$\begin{aligned} &x_1^{(1)} = x_1^{(2)} = x_1^{(3)} = x_1^{(4)} = x_1^{(5)} = x_1^{(6)}, \\ &x_2^{(1)} = x_2^{(2)} = x_2^{(3)},\ x_2^{(5)} = x_2^{(6)}. \end{aligned} \tag{2.27}$$

To introduce a general form of the NA constraints, let $\tau^{(s)}$, $s = 1, \ldots, S-1$, be the stage up to which scenarios s and $s+1$ share the same story. Therefore, the NA constraints can be presented by [2, 55]

$$x_t^{(s)} = x_t^{(s+1)},\ t = 1, \ldots, \tau^{(s)},\ s = 1, \ldots, S-1. \tag{2.28}$$

The constraints (2.28) can be rewritten in a compact form of

$$Nx = 0,$$

where N is the NA matrix, full-row rank and defined by

$$\begin{bmatrix} I^{(1)} & 0 & -I^{(1)} & 0 & & & & & & \\ & & I^{(2)} & 0 & -I^{(2)} & 0 & & & & \\ & & & & & & \ddots & \ddots & & \\ & & & & & & I^{(S-1)} & 0 & -I^{(S-1)} & 0 \end{bmatrix}$$

where I is the identity matrix and

$$x = (x_1^{(1)}, \ldots, x_T^{(1)}, x_1^{(2)}, \ldots, x_T^{(2)}, \ldots, x_1^{(S)}, \ldots, x_T^{(S)}).$$

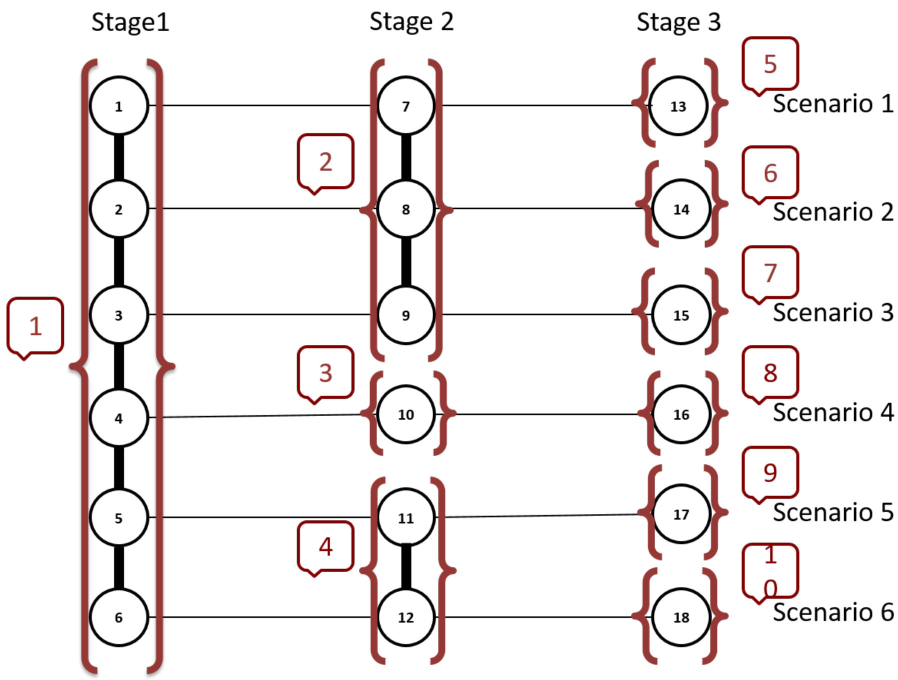

Figure 2.2 – Nonaticipativity graph which shows a structure of the scenario tree including the NA constraints. At each stage thick lines show the links created among the scenarios because of the NA constraints; thin lines show the scenario paths; red squares show the scenarios which share information (nodes); and circles show the nodes. The number of nodes, at each stage, is multiplied by the number of scenarios that share a node in the scenario tree. At stage one, there are six nodes, because all 6 scenarios share the root. At stage two, there are three nodes connecting scenarios 1, 2, and 3, one disconnected node for scenario 4, and two nodes connecting scenarios 5 and 6. At the last stage, all the nodes are disconnected because the scenarios do not share information.

In the next section, we explain a way of formulating the NA constraints which was introduced by Rockafellar and Wets [51].

Before ending this section, we present a scenario format of the multi-stage problem (2.17), (2.18) and (2.19). We denote by $\overline{\xi}_t^{(s)} = \{\xi_1^{(s)}, \xi_2^{(s)}, \ldots, \xi_t^{(s)}\}$ the vector of realizations in successive stages, $x^{(s)} = (x_1^{(s)}, x_2^{(s)}, \ldots, x_T^{(s)})$ the vector of decisions in successive stages and $\overline{x}_t^{(s)} = \{x_1^{(s)}, x_2^{(s)}, \ldots, x_t^{(s)}\}$ the vector of decisions up to and including stage t, associated to the scenario s for $s = 1, \dots, S$. Then we have:

$$\min_{(x^{(1)},\dots,x^{(S)})} \sum_{s=1}^{S} p_s \left(f_1(x_1^{(s)}) + \sum_{t=2}^{T} f_t(s, \overline{x}_{t-1}^{(s)}, x_t^{(s)}) \right) \tag{2.29}$$

subject to

$$\begin{aligned}
& g_{1,i}(x_1^{(s)}) = 0, && i = 1, \dots, \overline{m}_1,\ s = 1, \dots, S, && (2.30)\\
& g_{1,i}(x_1^{(s)}) \leq 0, && i = \overline{m}_1 + 1, \dots, m_1,\ s = 1, \dots, S, && (2.31)\\
& g_{t,i}\left(s, \overline{x}_{t-1}^{(s)}, x_t^{(s)}\right) = 0, && i = 1, \dots, \overline{m}_t,\ t = 2, \dots, T,\ s = 1, \dots, S, && (2.32)\\
& g_{t,i}\left(s, \overline{x}_{t-1}^{(s)}, x_t^{(s)}\right) \leq 0, && i = \overline{m}_t + 1, \dots, m_t,\ t = 2, \dots, T,\ s = 1, \dots, S, && (2.33)\\
& x_t^{(s)} \in X_t, && t = 1, \dots, T,\ s = 1, \dots, S, && (2.34)\\
& x_t^{(s)} \text{ is nonanticipative}, && t = 1, \dots, T,\ s = 1, \dots, S. && (2.35)
\end{aligned}$$

2.3.2 Scenario and solution aggregation

In (2.29), let us define $f^{(s)}(x^{(s)}) = f_1(x_1^{(s)}) + \sum_{t=2}^{T} f_t(s, \overline{x}_{t-1}^{(s)}, x_t^{(s)})$ and $\mathcal{X}^{(s)} = \{x^{(s)} \in \mathbb{R}^n \,|\, \text{satisfying}\, (2.30) - (2.34)\}$, $s = 1, \dots, S$, then we can reformulate the subproblem associated to scenario s as

$$(\mathcal{P}_s) \qquad \min_{x^{(s)} \in \mathcal{X}^{(s)}} f^{(s)}(x^{(s)})$$
$$\text{s.t. } x_t^{(s)} \text{ is nonanticipative},\ t = 1, \dots, T.$$

According to Rockafellar and Wets [51], one way of expressing the NA constraints is to use the concept of *scenario bundling*, i.e. to partition the scenario set $\mathcal{S}$ into finitely many disjoint subsets at each time stage t. A bundle consists of scenarios that are indistinguishable at time t, i.e. $\mathcal{S}_t^{(s)} = \left\{ \dot{s} \,\middle|\, \bar{\xi}_t^{(s)} = \bar{\xi}_t^{(\dot{s})} \right\}$. If we consider Figure 2.2, then at first stage we have one single bundle (shown by square 1) which is shared between all scenarios and also contains all scenarios, i.e.

$$\mathcal{S}_1^{(s)} = \{(1), (2), (3), (4), (5), (6)\},\ \text{for } s = 1, \dots, 6.$$

At the second stage, there are three bundles (shown by squares 2, 3, 4) where each contains different scenarios as follows:

$$\mathcal{S}_2^{(s)} = \{(1), (2), (3)\}, \text{ for } s = 1, 2, 3,$$
$$\mathcal{S}_2^{(s)} = \{(4)\}, \text{ for } s = 4,$$
$$\mathcal{S}_2^{(s)} = \{(5), (6)\}, \text{ for } s = 5, 6.$$

Finally, stage three contains six bundles (shown by squares 5, 6, 7, 8, 9, 10):

$$\mathcal{S}_3^{(s)} = \{(1)\}, \text{ for } s = 1,$$
$$\mathcal{S}_3^{(s)} = \{(2)\}, \text{ for } s = 2,$$
$$\mathcal{S}_3^{(s)} = \{(3)\}, \text{ for } s = 3,$$
$$\mathcal{S}_3^{(s)} = \{(4)\}, \text{ for } s = 4,$$
$$\mathcal{S}_3^{(s)} = \{(5)\}, \text{ for } s = 5,$$
$$\mathcal{S}_3^{(s)} = \{(6)\}, \text{ for } s = 6.$$

Back to the discussion about the modeling of the NA conditions, we can say that if $\dot{s}$ and $\ddot{s}$ are two indistinguishable scenarios from $\mathcal{S}_t^{(s)}$ at time t, then $x_t^{(\dot{s})} = x_t^{(\ddot{s})}$. In other words, $x_t^{(\dot{s})}$ must be constant for all $\dot{s} \in \mathcal{S}_t^{(s)}$.

Thus from the space of all mappings $x : \mathcal{S} \to \mathbb{R}^n$, (denoted by $\mathcal{D}$), with components $x_t : \mathcal{S} \to \mathbb{R}^{n_t}$, $n_1 + \cdots + n_T = n$, a subspace

$$\mathcal{N} = \{x \in \mathcal{D} \mid x_t \text{ is constant on each bundle for } t = 1, \ldots, T\},$$

can be defined by specifying the solutions that meet the NA constraints. Solutions x belonging to $\mathcal{N}$ are called *implementable* solutions. We should note that there is a distinction between implementable solutions and *admissible* solutions, which belong to the set

$$\mathcal{L} = \{x \in \mathcal{D} \mid x_t^{(s)} \in \mathcal{X}^{(s)} \text{ for all } s \in \mathcal{S}\ t = 1, \ldots, T\}.$$

Now consider the collection of scenario subproblems $(\mathcal{P}_s)$ and assume that we can modify their objectives and generate various solutions $x \in \mathcal{D}$. These solutions are called *contingent*, because they are obtained by solving perturbed versions of the scenario subproblems. The goal of modifying the objective functions in subproblems is to find an implementable solution for the underlying stochastic optimization problem.

A contingent solution is at least always admissible: $x \in \mathcal{L}$. A solution which is both admissible and implementable is what it is wished for, i.e. a *feasible* solution. According to Rockafellar and Wets [51], a way of finding a feasible solution is to model the NA constraints first. To do so, they set the value of $x_t^{(s)}$ equal to the conditional expectation over its corresponding bundle at each stage t. In other words,

$$x_t^{(s)} = E\left[x_t^{(\dot{s})} \,\middle|\, \dot{s} \in \mathcal{S}_t^{(s)}\right]. \tag{2.36}$$

The conditional expectation in (2.36) is a "weighted average" of all solutions $x_t^{(\dot{s})}$ for scenarios in the bundle $\mathcal{S}_t^{(s)}$, i.e.

$$E\left[x_t^{(\dot{s})} \,\middle|\, \dot{s} \in \mathcal{S}_t^{(s)}\right] = \frac{\sum_{\dot{s}\in\mathcal{S}_t^{(s)}} p_{\dot{s}} x_t^{(\dot{s})}}{\sum_{\dot{s}\in\mathcal{S}_t^{(s)}} p_{\dot{s}}} = \hat{x}_t^{(s)}, \tag{2.37}$$

where we denote the weighted average by $\hat{x}_t^{(s)}$, for $t = 1, \ldots, T$. For example in Figure 2.2, we can define $\hat{x}_1^{(s)}$ as

$$\hat{x}_1^{(s)} = \frac{p_1 x_1^{(1)} + p_2 x_1^{(2)} + p_3 x_1^{(3)} + p_4 x_1^{(4)} + p_5 x_1^{(5)} + p_6 x_1^{(6)}}{p_1 + p_2 + p_3 + p_4 + p_5 + p_6},$$

for $s = 1, \ldots, 6$; $\hat{x}_2^{(s)}$ as

$$\begin{aligned}
\hat{x}_2^{(s)} &= \frac{p_1 x_2^{(1)} + p_2 x_2^{(2)} + p_3 x_2^{(3)}}{p_1 + p_2 + p_3},\ s = 1, 2, 3, \\
\hat{x}_2^{(s)} &= \frac{p_4 x_2^{(4)}}{p_4},\ s = 4, \\
\hat{x}_2^{(s)} &= \frac{p_5 x_2^{(5)} + p_6 x_2^{(6)}}{p_5 + p_6},\ s = 5, 6,
\end{aligned}$$

and $\hat{x}_3^{(s)}$ as

$$\hat{x}_3^{(s)} = \frac{p_1 x_3^{(1)}}{p_1},\ s = 1,$$
$$\hat{x}_3^{(s)} = \frac{p_2 x_3^{(2)}}{p_2},\ s = 2,$$
$$\hat{x}_3^{(s)} = \frac{p_3 x_3^{(3)}}{p_3},\ s = 3,$$
$$\hat{x}_3^{(s)} = \frac{p_4 x_3^{(4)}}{p_4},\ s = 4,$$
$$\hat{x}_3^{(s)} = \frac{p_5 x_3^{(5)}}{p_5},\ s = 5,$$
$$\hat{x}_6^{(s)} = \frac{p_6 x_3^{(6)}}{p_6},\ s = 6.$$

Considering (2.36) and (2.37), the NA constraints can be reformulated as

$$x_t^{(s)} = \hat{x}_t^{(s)},\ t = 1, \ldots, T. \tag{2.38}$$

Clearly $\hat{x}$ is implementable as it satisfies the NA constraints, i.e. $\hat{x} \in \mathcal{N}$. In order to explain more the logic behind this definition of NA, Rockafellar and Wets [51] introduced two transformations (operators) which we discuss briefly here. The first one is a transformation

$$J : x \to \hat{x} \text{ defined by } (2.36) - (2.37),$$

which is linear and satisfies $J^2 = J$ (see [51]). J is a projection from $\mathcal{D}$ (recall from page 14 that $\mathcal{D}$ is the space of all the solutions defined on the scenario set $\mathcal{S}$) onto $\mathcal{N}$ which depends on the information structure (scenario tree) and weights p_s. This operator is called *aggregation operator* or *conditional expectation operator*.

The second operator is

$$K = I - J, \text{ such that for } x \in \mathcal{D},\ Kx = x - \hat{x}, \tag{2.39}$$

which is an orthogonal projection on the subspace of $\mathcal{D}$ complementary to $\mathcal{N}$. The new subspace can be denoted by $\mathcal{M}$:

$$\begin{aligned}\mathcal{M} = \mathcal{N}^{\perp} &= \{\lambda \in \mathcal{D} \mid J\lambda = 0\} \\ &= \{\lambda \in \mathcal{D} \mid E\left[\lambda_t^{(s)} | \mathcal{S}_t^{(s)}\right] = 0,\ t = 1, \ldots, T\}.\end{aligned} \tag{2.40}$$

Therefore, the NA constraints can be represented by the linear constraint $Kx = 0$ and the subspace $\mathcal{N}$ as

$$\mathcal{N} = \{x \in \mathcal{D} \mid Kx = 0\} = \{x \in \mathcal{D} \mid x = \hat{x}\}. \tag{2.41}$$

Now through these notations and definitions, if we rewrite the objective function in (2.29) as

$$F(x) = \sum_{s \in \mathcal{S}} p_s f^{(s)}(x^{(s)}), \tag{2.42}$$

then the problem (2.29)–(2.35) is equivalent to finding a solution to the problem

$$\min F(x) \text{ over all } x \in \mathcal{L} \cap \mathcal{N}. \tag{2.43}$$

If the functional F in (2.42) is written as

$$F(x) = E[f^{(s)}(x^{(s)})], \tag{2.44}$$

then the problem to be solved has the form

$$(\mathcal{P}) \qquad \min_x E[f^{(s)}(x^{(s)})] \text{ subject to } x \in \mathcal{L},\ Kx = 0. \tag{2.45}$$

As said before, it is possible to decompose and solve problem $(\mathcal{P})$, if there are no the NA constraints ($Kx = 0$). To get rid of the NA constraints, Rockafellar and Wets [51] first used a Lagrangian method on problem $(\mathcal{P})$. If $y \in \mathcal{D}$ stands for Lagrange multipliers, then the Lagrangian of objective function in (2.45) will look like

$$F(x) + \langle Kx, y\rangle_{\mathcal{D}} \text{ for } x \in \mathcal{L}, y \in \mathcal{D}, \tag{2.46}$$

where $\langle , \rangle_{\mathcal{D}}$ is an inner product on $\mathcal{D}$ and defined by

$$\langle x, y\rangle_{\mathcal{D}} := E\{\langle x^{(s)} \cdot y^{(s)}\rangle\} = \sum_{s\in\mathcal{S}} p^{(s)}\langle x^{(s)} \cdot y^{(s)}\rangle,$$

and $\langle\cdot\rangle$ is the scalar product on $\mathbb{R}^n$. However, since K is an orthogonal projection then

$$\langle Kx, y\rangle_{\mathcal{D}} = \langle x, Ky\rangle_{\mathcal{D}} = \langle Kx, Ky\rangle_{\mathcal{D}}.$$

If in (2.46), we replace $\langle Kx, y\rangle_{\mathcal{D}}$ with $\langle x, Ky\rangle_{\mathcal{D}}$ and then denote $Ky \in \mathcal{M}$ by λ, then the Lagrangian can be written as

$$L(x, \lambda) = F(x) + \langle x, \lambda\rangle_{\mathcal{D}} \text{ for } x \in \mathcal{L}, \lambda \in \mathcal{M}. \tag{2.47}$$

The multipliers λ are called *information price system*. Due to the numerical limitations of Lagrangian method, Rockafellar and Wets introduced an *augmented Lagrangian*, with respect to the NA constraints, to the ordinary Lagrangian (2.47) as

$$L_A(x, \lambda, \rho) = F(x) + \langle x, \lambda\rangle_{\mathcal{D}} + \frac{1}{2}\rho\|Kx\|^2_{\mathcal{D}}, \text{ for } x \in \mathcal{L}, \lambda \in \mathcal{M},\ \rho > 0, \tag{2.48}$$

where the norm is defined as $\|x\|_{\mathcal{D}} = [E\{\|x^{(s)}\|^2\}]^{1/2}$, with $\|\cdot\|$ being the ordinary Euclidean norm on $\mathbb{R}^n$, and ρ being a penalty parameter. Therefore, problem $(\mathcal{P})$ to be solved becomes

$$\min_x L_A(x, \lambda, \rho) \text{ subject to } x \in \mathcal{L}. \tag{2.49}$$

We note that the augmented Lagrangian function (2.48) cannot be useful directly, because the presence of the term $\|Kx\|^2_{\mathcal{D}}$ makes it impossible to decompose $(\mathcal{P})$ into subproblems. However, Rockafellar and Wets [51] suggested to replace Kx by $x - \hat{x}$ in

(2.48), and then by fixing $\hat{x}$, repeatedly solve the program by updating the Lagrange multipliers vector and the value of $\hat{x}$ between consecutive solutions. The resulting algorithm is known as progressive hedging algorithm (PHA) and is summarized in the next section.

2.3.3 Progressive hedging algorithm

By rewriting the objective function in (2.49) as

$$L_A(x, \lambda, \rho) = E\left[f^{(s)}(x^{(s)}) + \sum_{t=1}^{T}\left(\lambda_t^{(s)\prime} x_t^{(s)} + \frac{\rho}{2}\left\|x_t^{(s)} - \hat{x}_t^{(s)}\right\|^2\right)\right],$$

it is possible to decompose $L_A(x, \lambda, \rho)$ into objective functions per scenario. This way we can present PHA as following.

Step 0. Set $\hat{x}^{(s),0} = \left(\hat{x}_1^{(s),0}, \ldots, \hat{x}_T^{(s),0}\right)$ and $k = 0$. Choose $\lambda^{(s),0} = \mathbf{0}$, $\rho^0 > 0$.

Step 1. Compute $x^{(s),k+1} = (x_1^{s,k+1}, \ldots, x_T^{s,k+1})$, $s = 1, \ldots, S$, by solving each scenario subproblem

$$\begin{aligned} &\min_{x^{(s)}} f^{(s)}(x^{(s)}) + \sum_{t=1}^{T}\left(\lambda_t^{(s)\prime} x_t^{(s)} + \frac{\rho^k}{2}\left\|x_t^{(s)} - \hat{x}_t^{(s),k}\right\|^2\right) \\ &\text{s.t. } x^{(s)} \in \mathcal{X}^{(s)}. \end{aligned} \tag{2.50}$$

Step 2. For $s = 1, \ldots, S$, $t = 1, \ldots, T$, set

$$\hat{x}_t^{(s),k+1} = \frac{\sum_{\dot{s}\in\mathcal{S}_t^{(s)}} p_{\dot{s}} x_t^{(\dot{s}),k+1}}{\sum_{\dot{s}\in\mathcal{S}_t^{(s)}} p_{\dot{s}}}.$$

Step 3. Set ρ^{k+1} and

$$\lambda_t^{(s),k+1} = \lambda_t^{(s),k} + \rho^k\left(x_t^{(s),k+1} - \hat{x}_t^{(s),k+1}\right),\ t = 1, \ldots, T,\ s \in \mathcal{S}.$$

Step 4. Stop if convergence is achieved. Otherwise, set $k \leftarrow k + 1$ and return to Step 1.

Several practical issues arise. A first question is the choice of the stopping criteria in Step 4. Rockafellar and Wets [51] proposed to stop if

$$\sqrt{\sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2 + \sum_{s\in\mathcal{S}} p_s \|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2} \leq \varepsilon,$$

or equivalently if

$$\sqrt{\sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k}\|^2} \leq \varepsilon, \tag{2.51}$$

as it can be easily proved (see Appendix II.1) that

$$\sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k}\|^2 = \sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2 + \sum_{s\in\mathcal{S}} p_s \|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2. \tag{2.52}$$

A second question concerns the initialization of the primal variables. As discussed in Chiche [11], Chapter 6, various strategies have been considered but the most popular is to set $x^{(s),0}$, $s = 1, \ldots, S$ as the solution of the subproblem associated to s, without the NA constraints:

$$\begin{aligned} \min_{x^{(s)}} \; & f^{(s)}(x^{(s)}) \\ \text{s.t. } & x^{(s)} \in \mathcal{X}^{(s)}, \end{aligned}$$

and $\hat{x}^{(s),0}$ is computed using (2.37). dos Santos et al. [59] compared this procedure to other initializations, but did not find significant improvements.

2.4 Theoretical and numerical history of the progressive hedging algorithm

PHA was first developed for convex stochastic problems with continuous variables. Rockafellar and Wets [51] provided a proof of convergence to an optimal solution in convex case, no matter if the subproblems are solved exactly or approximately. They showed that an optimal solution is reached after a finite number of iterations, or an infinite sequence of iterations is produced and all limit points are solutions of the original problem. They also proved the linear rate of convergence to a global solution in the case of linear-quadratic stochastic problems. In the nonconvex case, they proved that under mild conditions, if the algorithm converges, the limit solution is a locally optimal solution of the original problem. Later, in the case of stochastic mixed integer programming (MIP) problems, Løkketangen and Woodruff [42] introduced a new convergence criterion: integer convergence. By this new concept, if the integer components of a solution are found, they will be fixed and the rest of components are calculated by solving a deterministic equivalent problem (DEP) of the stochastic problem. However they did not provide any proof of convergence. Following the structure of Chiche's PhD thesis [11], there are studies which focused on solving the scenario subproblems approximately, such as the works of Barro and Canestreli [1], Helgason and Wallace [27]. Other studies such as the works of Fan and Liu [19], dos Santos et al. [59] introduced some strategies to initialize the primal variables, i.e. $\hat{x}$. The effect of increasing the number of scenarios on the number of iterations was studied in different applications by Berland and Haugen [3], Chun and Robinson [12], Mulvey and Vladimirou [45]. These researchers argued about the effect of going from linear to nonlinear subproblems on the number of iterations. Mulvey and Vladimirou [45] suggested to set the dual variables, i.e. λ, to a value other than zero in their specific problem. In order to accelerate the convergence of PHA in MIP applications, Watson et al. [62] introduced some fixing techniques for integer variables. In order to decrease the running time of PHA, Carpentier et al. [9], Crainic et al. [15] proposed some scenario clustering techniques. Because of the special characteristic of PHA, which is solving the subproblems independently at each iteration, it is possible to use parallelization techniques to speed up the algorithm. Somervell [56] suggested six ways of parallelizing PHA. Wets [64] discussed the effect of changing the probabilities of scenarios on the solution returned by PHA. In stochastic MIP problems there is a risk of non-convergence while using PHA. To deal with this issue, Watson et al. [62] suggested some heuristics to detect the cyclic behavior and enforce the convergence.

By looking at the literature on PHA, we can see that it has been used as a decomposition technique in a wide range of applications for which we provide a list in Table 2.I. However, the list is not exhaustive.

Many of these studies have also focused on different aspects of PHA, such as: choices of the penalty parameter; handling the NA constraints and quadratic term in the objec-

Application	References
Energy production/operation planning	Carpentier et al. [8], Chiche [11], Gonçalves et al. [23, 24], Helseth [28], Parriani et al. [48], Reis et al. [50], Ryan et al. [52], dos Santos et al. [53], Takriti et al. [58], dos Santos et al. [59], Wu et al. [65], Zéphyr et al. [68]
Logistics and transportation planning/management	Fan and Liu [19], Hvattum and Løkketangen [29], Listes and Dekker [40], Perboli et al. [49], Watson et al. [62]
Networks planning	Carvalho et al. [10], Crainic et al. [14], Jönsson et al. [31], Mulvey and Vladimirou [43, 44, 45, 46], Watson and Woodruff [61]
Production/Operation planning	Gul [25], Haugen et al. [26], Jonsbråten [30], Jörnsten and Bjorndal [32], Jörnsten and Leisten [33], Klimeš et al. [37], Lamghari and Dimitrakopoulos [38], Veliz et al. [60], Zanjani et al. [66]
Financial planning	Barro and Canestreli [1], Fulga [20]

Table 2.I – Applications of progressive hedging algorithm

tive function; solving subproblems approximately; initialization of primal and dual variables, i.e. $\hat{x}$ and λ respectively; handling integer variables by variable fixing techniques; regrouping/clustering scenarios; parallelization of scenario subproblems; change of scenarios' probabilities; detecting cycles; etc. In addition, most of these researches have suggested enhancements which are usually problem-dependent.

In this research, we focus on the choice of the penalty parameter and handling the quadratic term in PHA objective function, thus we provide a comprehensive literature review on those points. For the rest of challenges associated with PHA, we refer the interested readers to Chiche [11]. In her PhD thesis, she applied PHA to a large-scale energy production problem and for that she provided a broad review on almost all the aspects of PHA.

2.5 Research motivations

In this section, we present the directions of this research as well as our motivations.

2.5.1 Penalty parameter choices in progressive hedging algorithm

Rockafellar and Wets [51] analyzed PHA and established its convergence with a constant penalty parameter over the iterations. However, many authors have observed that in practice, the choice of penalty parameter value will greatly impact the numerical behavior of the algorithm. Considering the stopping criterion (2.51), Helgason and Wallace [27] commented that the penalty parameter should be as small as possible but large enough to guarantee the convergence, more specifically to produce a monotone decrease in the criteria. Mulvey and Vladimirou [44, 45] showed that the overall convergence rate of PHA is particularly sensitive to the choice of ρ. Small values of the penalty parameter tend to produce a fast initial progress in primal sequence $\{\hat{x}^k\}$ with a slower progress in dual space, i.e. the sequence $\{\lambda^k\}$, while large values lead to an opposite behavior. They first suggested to consider a larger penalty parameter when the NA constraints are more restrictive, and introduced the idea to dynamically update the parameter, increasing the value over PHA iterations. However, they also noticed that a sudden increase in penalty parameter can drive the algorithm toward ill-conditioning or a suboptimal solution, suggesting to increase the penalty parameter smoothly. They also proposed to implement a sudden reduction to improve the convergence in primal space, if the dual convergence is already achieved. Chun and Robinson [12] used two predefined values: they initialized the parameter with a large value and then changed it to a small value, if there was enough improvement in the dual sequence. Jonsbråten [30] decided to maintain the penalty parameter to zero, and defined a dynamically updated step size to compute the Lagrange multipliers. Some authors considered the possibility to use different penalty parameters, depending on the affected variables. In particular, Somervell [56] suggested to use predefined fixed bundle-stage wise values, while Watson et al. [62] proposed to set penalty parameters proportionally to the cost coefficient in the objective function, when dealing with linear functions. Fan and Liu [19], inspired by Chun and Robinson [12] and Watson et al. [62], explored the use of two fixed values and cost-proportional values.

Following the idea of dynamic update, Reis et al. [50] decreased the penalty over the iterations, while other authors, as Carpentier et al. [8], Crainic et al. [14], chose to increase the penalty parameter. Hvattum and Løkketangen [29] suggested a controlling approach based on criteria (2.51) for updating the penalty parameter. They reduced ρ if $\sum_{s\in\mathcal{S}} p_s\|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2$ does not decrease, and they increased ρ if $\sum_{s\in\mathcal{S}} p_s\|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2$ does not decrease. They also considered a node-cost proportional update. Inspired by them, Gul [25] suggested to dynamically update the penalty parameter, increasing the parameter if no progress is observed for the dual variables, and decreasing it if no progress is observed for the primal variables. Gonçalves et al. [23] used an increase factor proportional to the NA violation. They also insisted that the initial penalty parameter should be chosen small enough, for instance with a value of 10^{-4}. Zéphyr et al. [68] dynamically updated the penalty parameter using coefficients based on the optimality and NA indicators, allowing to increase or decrease the parameter.

Many strategies for updating the penalty parameter have been considered, highlighting the sensitivity of PHA to its value, but no clear consensus exists to date. We summarize the main approaches in Table 2.II. More recently, Chiche [11] highlighted that a dynamic update strategy can even lead to a complete failure of PHA. She provided an example where an apparently genuine choice of the penalty parameter results in a cyclic behavior of PHA between two solutions, none of them satisfying the NA constraints. In her example, the penalty parameter oscillates between two inverse values. Therefore, while the use of a fixed penalty parameter is usually associated with a slow convergence, care must be exercised when designing a dynamic update strategy if we want to improve PHA performance. We propose a novel approach that aims to learn from the algorithm process, while remaining simple and independent of the application. The idea and its numerical results are presented in Chapter 3.

2.5.2 Handling the quadratic penalty term in progressive hedging algorithm

Another big challenge in using PHA is how to handle the quadratic penalty term in the objective function. Depending on the application in which PHA has been used, different researchers suggested different strategies. We present some of these methods in two major categories based on the nature of the applications: applications with convex

Strategy	Penalty parameter update
Fixed value [51]	$\rho = a^1 > 0$
Dynamic increase with possibly a sudden reduction [45]	$\rho^{k+1} = (\tau_\rho \rho^k)^\mu$, $\tau_\rho \geq 1, 0 < \mu \leq 1$, $\rho^{k+1} = \rho_\varepsilon$, $\rho_\varepsilon > 0$, small
Fixed to two predefined positive values with reduction [12]	$\rho = a^2 > 0$, if the convergence in dual space is not achieved, otherwise $\rho = a^3 > 0$, with $a^3 < a^2$
Bundle-stage wise value [56]	$\rho_t^{(\dot{s})}$, for $\dot{s} \in \mathcal{S}_t^{(s)}$ and $t = 1, \cdots, T$
Cost-proportional value [62]	$\rho(i) = \frac{c(i)}{\max_s\{x^{(s),0} - \min_s x^{(s),0} + 1\}}$ or $\rho(i) = \frac{c(i)}{\max\{\sum_s p_s \lvert x^{(s),0} - \hat{x}^0\rvert, 1\}}$
Decreasing values [50]	$\rho^k = \frac{1}{a^4 + a^5 k}$, $a^4, a^5 \in (0, 1)$
Increasing values [14]	$\rho^{k+1} = \tau_\rho \rho^k$, $\tau_\rho \geq 1$
Dynamic update, convergence- and cost-proportional node-wise values [29]	$\rho_v = c_v \delta(v) \rho^{k+1}$ for node $v \in \mathcal{V}$, where $\rho^{k+1} = \tau_{inc} \rho^k$ with $\tau_{inc} > 1$, if the progress toward dual convergence is negative, $\rho^{k+1} = \tau_{dec} \rho^k$ with $1 > \tau_{dec} > 0$, if the progress toward primal convergence is negative, for some discount factor $\delta(v)$
Dynamic update, convergence-proportional values by predefined multipliers [25]	$\rho^{k+1} = a^6 \rho^k$, if the progress toward dual convergence is negative, $\rho^{k+1} = \frac{1}{a^6} \rho^k$, if the progress toward primal convergence is negative, with $a^6 > 1$
Increasing values, with a NA-proportional multiplier [23]	$\rho^{k+1} = \rho^k \left\{ a^7 E\left[\sum_t \left(\frac{\lVert x_t^{(s),k} - \hat{x}^k \rVert^2}{x_{t,\max}^{(s),k} - x_{t,\min}^{(s),k} + 1} \right) \right] + 1 \right\}$, $a^7 > 1$,
Dynamic update, bounded values based on the optimality and NA indicators [68]	$\rho^{k+1} = \max\{0.01, \min\{100, q^{k+1}\rho^k\}\}$, where $q^{k+1} = \left(\max\{l^{k+1}, h^{k+1}\}\right)^{\frac{1}{1+0.01(k-1)}}$, where l^{k+1} and h^{k+1} are the optimality and NA indicators, respectively

Table 2.II – Penalty parameter updates

stochastic models and applications with nonconvex stochastic models. Then we suggest a new technique to deal with the quadratic term.

2.5.2.1 Current methods for convex stochastic programming problems

Instead of using PHA, Chun and Robinson [12] decided to relax the NA constraints with standard Lagrangian relaxation in linear multi-stage stochastic military force planing and production planing problems that are loosely coupled. To solve the new problem, they used a bundle/trust-region method. To solve a linear multi-stage mid-term operation problem in hydrothermal systems, Gonçalves et al. [23] used PHA. However, in order to reduce the size of the problem, they defined the NA constraints on the variables which define the rest of variables. Therefore, they dropped the quadratic term on the dependent variables. In another study, Helseth [28] applied PHA to a linear multi-stage stochastic energy production scheduling problem and to get rid of the quadratic penalty term, they approximated the quadratic term with a dynamic piece-wise linearization, iteratively.

To solve a nonlinear multi-stage stochastic problem, Liu et al. [41] used the PHA objective function with an extra penalty term. In other words, after relaxing the NA constraints as in PHA, they also relaxed other constraints with a logarithmic barrier function. This way they have an unconstrained problem which is solved approximately with Newton direction method. In fact, the original idea of using a log-barrier penalty function was first introduced by Zhao [69] who suggested to relax the NA constraints with a Lagrangian relaxation and then to use a logarithmic barrier on other constraints.

2.5.2.2 Current methods for mixed-integer stochastic programming problems

Mixed-integer problems having a quadratic term in the objective function can be quite troublesome. For example, Jørnsten [34] used the logic of scenario aggregation to a mixed integer $\{0, 1\}$ investment problem in oil and gas industry. However, they relaxed the NA constraints by a Lagrangian relaxation and solved subproblems with a subgradient method. In a similar setting, Jönsson et al. [31] dealt with a two-stage stochastic inventory allocation problem with integer variables in both stages. They also followed the idea of scenario aggregation with Lagrangian relaxation of a perturbed representation of the NA constraints. Later, Jörnsten and Bjorndal [32] applied PHA to a multi-stage stochastic dynamic location problem where there are continuous and binary variables. To construct the PHA objective function, they included NA on continuous variables and dropped it on the binary ones, because the binary variables are dependent on continuous variables in their problem. The logic of PHA is used in a mixed integer multi-stage stochastic unit commitment problem but without a quadratic penalty term. In this way, Takriti et al. [58] relaxed the NA constraints with a Lagrangian relaxation and decompose each scenario subproblem to single-generator subproblems. The single-generator subproblems are solved by dynamic programming. In the context of mixed integer binary stochastic problems, Løkketangen and Woodruff [42] introduced a new convergence criteria: integer convergence. With this new concept, if the integer components of a solution are found, they will be fixed and the rest of the components are calculated by solving a DEP. Løkketangen and Woodruff solved the subproblems with a tabu search method. For another example in oil industry, Jonsbråten [30] restricted the quadratic penalty term only to continuous variables of a mixed integer multi-stage stochastic problem. However, the quadratic term in PHA is also dropped and the interaction between continuous and integer variables is taken into account for the Lagrangian steps. In another example, a multi-stage stochastic energy unit commitment problem with both continuous and binary variables is studied by Takriti and Birge [57]. They use PHA to relax the NA constraints involving binary variables. However, in the computations they drop the quadratic penalty term. After the introduction of integer convergence, Haugen et al. [26] applied PHA to a mixed integer multi-stage stochastic lot-sizing problem and used a dynamic programming algorithm to solve the subproblems. Then after achieving the integer convergence, they followed the fixation strategy, as in [42]. In a mixed integer two-stage stochastic fleet-composition problem with integer variables at first stage and integer and binary variables at second stage, Listes and Dekker [40] applied PHA to a linear relaxation of the stochastic program and executed a rounding procedure to get integer values for the first stage variables involved in NA. For a two-stage stochastic network design problem with mixed integer binary variables, Crainic et al. [14] took advantage of having the first stage variables with binary values. They reduced the original objective function of PHA to an equivalent linear function and solved the subproblems by tabu search, as in [42]. In a Python tool for multi-stage stochastic problems (PySP), Watson et al. [63] linearized the quadratic term of PHA statically. Their idea was an inspiration for Helseth [28] who decided to dynamically linearize the quadratic term in the PHA objective function. In an energy production problem, Parriani et al. [48] also dealt with a two-stage stochastic unit commitment problem, but with integer variables in both stages. To get rid of the quadratic term, they replaced it with the L1-norm of the NA constraints in the PHA objective function.

2.5.2.3 New method: an elastic progressive hedging algorithm

As this research focuses on linear stochastic problems, we look for a way to keep the objective function linear. We propose to make the NA constraints elastic and add a linear penalty on the violation of constraints in the original objective function. To be more precise, by adding the quadratic penalty function over the NA constraints at iteration k, the linearity in subproblem

$$\begin{aligned} \min_{x^{(s)}} \ & c^{(s)\prime} x^{(s)} \\ \text{s.t. } & x^{(s)} \in \mathcal{X}^{(s)}, \\ & x^{(s)} - \hat{x}^{(s),k-1} = 0, \end{aligned} \tag{2.53}$$

is destroyed. Recall from Section 2.3 that $c^{(s)}$ represents the cost coefficients, $\mathcal{X}^{(s)}$ is the set of scenario-dependent constraints, $\hat{x}^{(s),k-1}$ represents the average solution from iteration $k-1$, and $x^{(s)}$ is the solution to be found for each scenario s. To keep the problem linear while forcing NA, we got inspired by the idea of *elastic programming* [7] in the history of linear programming. In elastic programming, the constraints are modified by adding some artificial nonnegative variables such that the violation of each constraint is allowed at some cost. So the constraints become goals and the question of if or how much the constraints are to be violated can be answered by the model. Such models are called elastic models. In MINOS software, Murtagh and Saunders [47] used a composite objective function of the original objective plus a penalty term on constraints violations. The SNOPT software [22] follows the same idea than MINOS. Gill et al. [22] applied the idea of elastic programming to a sequential quadratic programming (SQP) method for nonlinear problems. In SNOPT, they elasticize the nonlinear constraints and add an L1-norm on the constraints violation to the objective function.

Considering these remarks, our idea is to let the NA constraints be elastic as well and at the same time penalize the violation in the objective function, for each scenario subproblem. This way, we believe that we can still use the scenario aggregation logic behind PHA and also create a linear objective function instead of the quadratic function.

To adopt this technique, we propose to add two nonnegative artificial variables v^s and w^s to the NA constraints (for further details on how to elasticize equality and inequality constraints, we refer to [6]) and we penalize this violation of constraints. Therefore, we have

$$\begin{aligned}\min_{x^{(s)},v^{(s)},w^{(s)}} \quad & {c^{(s)}}'x^{(s)} + \Theta e'(v^{(s)} + w^{(s)}) \\ \text{s.t. } & x^{(s)} \in \mathcal{X}^{(s)}, \\ & x^{(s)} - \hat{x}^{(s),k-1} + v^{(s)} - w^{(s)} = 0, \\ & v^{(s)} \geq 0, \\ & w^{(s)} \geq 0.\end{aligned} \tag{2.54}$$

where $\Theta > 0$ is a fixed scalar and is called the *elastic penalty parameter*. For the proof of convergence, Boman [6] suggests to force the L1-norm of the elastic term to be bounded in the subproblem:

$$\begin{aligned}\min_{x^{(s)},v^{(s)},w^{(s)}} \quad & {c^{(s)}}'x^{(s)} + \Theta e'(v^{(s)} + w^{(s)}) \\ \text{s.t. } & x^{s} \in \mathcal{X}^{(s)}, \\ & x^{(s)} - \hat{x}^{(s),k-1} + v^{(s)} - w^{(s)} = 0, \\ & \|v^{(s)} - w^{(s)}\|_1 \leq \Delta \\ & v^{(s)} \geq 0, \\ & w^{(s)} \geq 0.\end{aligned} \tag{2.55}$$

where $\Delta \geq 0$ is the required bound, called the *elasticity bound*. This idea still needs to be tested numerically and the proof of convergence should be provided. Some further questions should be answered as well, such as whether we consider a trust-region [13] context (like what has been done in [39] as a decomposition algorithm for stochastic programming) for this new elastic PHA subproblem and how to choose the values of Θ and Δ.

CHAPTER 3

PENALTY PARAMETER UPDATE STRATEGIES IN THE PROGRESSIVE HEDGING ALGORITHM

In this chapter we introduce our new penalty parameter update as well as its numerical analysis.

3.1 Adaptive penalty parameter update

As pointed out by Takriti and Birge [57], PHA is a proximal point method producing the contraction of a sequence of primal-dual pairs $\{(\hat{x}^{(s),k}, \lambda^{(s),k})\}$ around an optimal saddle point. The primal convergence can be monitored by considering the expectation of the changes between consecutive NA solutions $\sum_{s\in\mathcal{S}} p_s \|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|$, from Step 3 of PHA presented in Chapter 2, while $\sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|$ gives the order of the changes in dual variables. Recall that from (2.52),

$$\sum_{s\in\mathcal{S}} p^s \|x^{(s),k+1} - \hat{x}^{(s),k}\|^2 = \sum_{s\in\mathcal{S}} p^s \|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2 + \sum_{s\in\mathcal{S}} p^s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2,$$

i.e. PHA aggregates the primal and dual changes. In contrast to many papers that monitor the values of $\sum_{s\in\mathcal{S}} p^s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2$ to decide to increase or decrease the penalty parameter ρ, we first check the changes in $\{\hat{x}^k\}$, i.e. $\sum_{s\in\mathcal{S}} p^s \|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2$. The main motivation is to avoid to enforce the NA constraints when we have not yet identified the correct NA solution, or in other terms, when the approximation of NA is not accurate enough. If we enforce the NA constraints when they are not approximated accurately, we could converge to a suboptimal solution, as also noticed by Chun and Robinson [12]. Therefore, if the primal variables significantly change, we avoid to increase the penalty parameter. Simultaneously, we aim to keep a balance between the Lagrangian function and the quadratic penalty. If the NA solution seems to stabilize, but we observe larger the NA constraints violation, we slightly increase the penalty parameter if the new violations are significantly more important. Otherwise, we keep the penalty parameter fixed. We do not expect this case to often happen, but if it occurs, we try to stabilize the process. Finally, if none of the previous situations occur, we deduce that we have achieved convergence in the primal space, so we force convergence in the dual space by increasing the penalty parameter value. The procedure is presented in more detail below.

Step 0. Set $\gamma_1,\ \gamma_2,\ \gamma_3 \in (0,1)$, $\alpha,\ \nu,\ \sigma \in (0,1)$, $1 < \theta < \beta < \eta$.

Step 1. If the change in $\{\hat{x}^k\}$ is large enough, i.e. if

$$\frac{E\left[\left\|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\right\|^2\right]}{\max\left\{E\left[\|\hat{x}^{(s),k+1}\|^2\right], E\left[\|\hat{x}^{(s),k}\|^2\right]\right\}} \geq \gamma_1,$$

or if the quadratic penalty term is important compared to the Lagrangian function, i.e. if

$$\rho^k E\left[\left\|x^{(s),k+1} - \hat{x}^{(s),k+1}\right\|^2\right] \\ \geq \sigma E\left[\left|f^{(s)}(x^{(s),k+1}) + \lambda^{(s),k'}\left(x^{(s),k+1} - \hat{x}^{(s),k}\right)\right|\right],$$

then

a) if the change in $\{\hat{x}^k\}$ is dominating the change in $\{\lambda^k\}$ such that

$$\frac{E\left[\left\|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\right\|^2\right] - E\left[\left\|x^{(s),k+1} - \hat{x}^{(s),k+1}\right\|^2\right]}{\max\left\{1, E\left[\|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2\right]\right\}} > \gamma_2,$$

then decrease the penalty parameter by setting $\rho^{k+1} = \alpha\rho^k$,

b) else if the change in $\{\lambda^k\}$ is dominating the change in $\{\hat{x}^k\}$ such that

$$\frac{E\left[\left\|x^{(s),k+1} - \hat{x}^{(s),k+1}\right\|^2\right] - E\left[\left\|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\right\|^2\right]}{\max\left\{1, E\left[\|\hat{x}^{(s),k+1} - \hat{x}^{(s),k}\|^2\right]\right\}} > \gamma_3,$$

then increase the penalty parameter by setting $\rho^{k+1} = \theta\rho^k$,

c) otherwise, keep the penalty parameter fixed by setting $\rho^{k+1} = \rho^k$,

otherwise go to Step 2.

Step 2. If there is no significant change in $\{\hat{x}^k\}$ but the change in the dual sequence $\{\lambda^k\}$ is getting larger, i.e. the NA violation increases over the iterations:

$$E\left[\left\|x^{(s),k+1} - \hat{x}^{(s),k+1}\right\|^2\right] > E\left[\left\|x^{(s),k} - \hat{x}^{(s),k}\right\|^2\right],$$

then

a) if the increase is large, i.e.

$$\frac{E\left[\left\|x^{(s),k+1} - \hat{x}^{(s),k+1}\right\|^2\right] - E\left[\left\|x^{(s),k} - \hat{x}^{(s),k}\right\|^2\right]}{E\left[\left\|x^{(s),k} - \hat{x}^{(s),k}\right\|^2\right]} > \nu,$$

then increase the penalty parameter by setting $\rho^{k+1} = \beta\rho^k$,

b) else keep the penalty parameter fixed by setting $\rho^{k+1} = \rho^k$.

Otherwise, go to Step 3.

Step 3. Increase the penalty parameter by setting $\rho^{k+1} = \eta\rho^k$.

A last point to discuss is the choice of the initial penalty parameter ρ^0. As the Lagrange multipliers vector λ^0 is set to zero, the initial augmented Lagrangian is

$$E\left[f^{(s)}(x^{(s),0})\right] + \frac{\rho^0}{2}E\left[\left\|x^{(s),0} - \hat{x}^{(s),0}\right\|^2\right].$$

This suggests to balance the two terms, leading to

$$\rho^0 = \frac{\max\left\{1, 2\zeta\left|E\left[f^{(s)}(x^{(s),0})\right]\right|\right\}}{\max\left\{1, E\left[\|x^{(s),0} - \hat{x}^{(s),0}\|^2\right]\right\}}, \tag{3.1}$$

with $\zeta > 0$.

3.2 Computational study

In order to numerically validate our approach, we compare it with some of the propositions identified in the literature. We first consider fixed values, dynamic update with/without dropping from Mulvey and Vladimirou [44], a simplified version of convergence-proportional update by Hvattum and Løkketangen [29], excluding problem-dependent aspects, and optimality- and NA-proportional update by Zéphyr et al. [68]. We do not examine cost-proportional penalties as some of the test problems have many variables with null costs.

We also limit ourselves to linear problems of the form

$$\begin{aligned}
\min_x \ & \sum_{s\in\mathcal{S}} p_s \left(\sum_{t=1}^{T} c_t^{(s)\prime} x_t^{(s)} \right) \\
\text{s.t. } & H_1^{(s)} x_1^{(s)} = b_1^{(s)}, \\
& \sum_{j<t} G_j^{(s)} x_j^{(s)} + H_t^{(s)} x_t^{(s)} = b_t^{(s)},\ t = 2,\dots,T,\ s\in\mathcal{S}, \\
& x_t^{(s)} \geq 0,\ t = 1,\dots,T,\ s\in\mathcal{S}, \\
& x_t^{(s)} \text{ is nonanticipative},\ t = 1,\dots,T,\ s\in\mathcal{S}.
\end{aligned}$$

The problems were collected from SMI (`http://www.coin-or.org/projects/Smi.xml`) and SPLIB collection proposed by V. Zverovich (`https://github.com/vitaut/splib`). We also created a modified version of the problem KW3R by adding randomness in the constraints matrix and we denote this version by KW3Rb. The characteristics of the problems are summarized in Table 3.I. We used SMI to parse the SMPS files describing them and we solved their deterministic equivalent formulations using CPLEX 12.5 in order to have reference optimal values. We implemented PHA in C++ and the scenario subproblems are solved with CPLEX. The numerical tests were performed on a cluster of computers with 2.40 GHZ Intel(R) Xeon(R) E5620 CPU (quad-core) with 2 threads each and 98 GB of RAM.

We compare the use of a fixed penalty parameter (referred as Fixed) with several of the identified strategies, namely the dynamic update proposed by Mulvey and Vladimirou

Problem	#stages	#scenarios	Optimal value
KW3R	3	9	2613
KW3Rb	3	9	3204
app0110R	3	9	41.96
SGPF3Y3	3	25	-2967.91
Asset Mgt	4	8	-164.74
SGPF5Y4	4	125	-4031.3
wat10I16	10	16	-2158.75
wat10C32	10	32	-2611.92

Table 3.I – Problems

[45] with reduction (referred as M&VR) and without reduction (referred as M&V), the controlled dynamic update designed by Hvattum and Løkketangen [29] (referred as H&L), the learning update developed by Zéphyr et al. [68] (referred as Z&L&L) and the adaptive update (referred as Adaptive). The initial penalty parameter ρ^0 is set as in (3.1), or by using the recommended values in the original papers, and we compute $x_0^{(s)}$, $s = 1, \ldots, S$, by solving the scenario subproblems without the NA constraints. We tried three different settings corresponding to different values of ζ, as the smaller the value of ζ, the less we enforce the initial NA solution. We test the method with $\zeta = 0.01$ (small), $\zeta = 0.10$ (medium), and $\zeta = 0.50$ (large). Moreover, the parameters associated with each approach are chosen as described below.

For the M&V update, we set $\rho^{k+1} = (\tau_\rho \rho^k)^\mu$, and consider two settings recommended by Mulvey and Vladimirou [45] referring to them by a and b, respectively. In setting a, we have $\tau_\rho = 1.1$, $\mu = 0.80$, $\rho^0 = 0.02$, and $\rho_{min} = 0.05$, while in setting b, we have $\tau_\rho = 1.25$, $\mu = 0.95$, $\rho^0 = 0.05$, and $\rho_{min} = 0.05$. To drop the penalty parameter as they suggested, we check if $\sum_{s \in \mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k+1}\|^2 \leq \varepsilon_d$, with $\varepsilon_d = 10^{-5}$, is satisfied. For H&L, we set $\rho^{k+1} = \delta \rho^k$, if the progress toward dual convergence is negative, and $\rho^{k+1} = \frac{1}{\delta}\rho^k$, if the progress toward primal convergence is negative, with $\delta = 1.8$ and $\rho^0 = 0.3$, We follow Zéphyr et al. [68] recommendations for the implementation of their strategy. Finally, the adaptive strategy parameters are chosen as $\gamma_1 = 10^{-5}$, $\gamma_2 = 0.01$, $\gamma_3 = 0.25$, $\sigma = 10^{-5}$, $\alpha = 0.95$, $\theta = 1.09$, $\nu = 0.1$, $\beta = 1.1$, and $\eta = 1.25$.

The stopping criteria is a normalized version of (2.51). We stop if

$$\sqrt{\frac{\sum_{s\in\mathcal{S}} p_s \|x^{(s),k+1} - \hat{x}^{(s),k}\|^2}{\max\left\{1, \sum_{s\in\mathcal{S}} p_s \|\hat{x}^{(s),k}\|^2\right\}}} \leq \varepsilon, \tag{3.2}$$

with $\varepsilon = 10^{-5}$, and set the iteration limit to 500 and the time limit to 36000 seconds (or 10 hours). We declare convergence if the difference with the optimal value is less than 0.1% within the time and iteration limits. A summary of our main results is given in Tables 3.II–3.IV, where the methods are compared when the same initial penalty parameter is used, while we present detailed results in Appendix II, problem by problem. When we reach the time or iterations limit, we indicate "Limit" if the final solution is within a 0.1% optimality gap, otherwise we mention "Wrong" if the final solution has an optimality gap greater than 0.1%, and "Infeasible" if the NA constraints are not satisfied. If (3.2) is satisfied within the defined limits, we report the number of PHA iterations and the optimality gap in brackets in case we have converged to a suboptimal solution, i.e. the optimality gap is greater than 0.1%.

We also graphically compare the methods by means of the performance profiles [17]. In the figures, P designates the percentage of problems which are solved within a factor τ of the best solver, using the number of iterations as our performance metric. We first compare the methods for a given ζ, and then compare the choices of ζ for fixed and adaptive strategies. Figures 3.1, 3.2, and 3.3 show that existing strategies have difficulties to converge towards the optimal solution, a fixed parameter strategy being surprisingly more efficient when the initial penalty parameter is not chosen small enough. For a small initial penalty parameter ($\zeta = 0.01$), the approach designed by Mulvey and Vladimirou [45] has a slight advantage over the other methods, but it disappears when increasing the initial penalty parameter value. Overall, none of the existing approaches has been found to significantly outperforms the other ones. The adaptive strategy proved to be quite efficient, as it is the fastest approach for most of the problems, whatever the choice of ρ^0, and it is the only one to always deliver the correct solution. It is therefore interesting to explore the influence of the initial penalty parameter value. We draw the performance profile of the fixed and adaptive strategies in Figure 3.4. We again see that the adap-

Method	KW3R	KW3Rb	app0110R	SGPF3Y3
Fixed	30	Infs.	Infs.	9
M&VRa	43	Limit	Wrong	12(1.1%)
M&VRb	48	Limit	Limit	13(0.3%)
M&Va	27	346	105	12(1.1%)
M&Vb	23	244	92	13(0.3%)
H&L	115	Infs.	Infs.	9
Z&L&L	27	261	109	18(1.9%)
Adaptive	25	139	108	10
Method	Asset-Mgt	SGPF5Y4	wat10I16	wat10C32
Fixed	6	Limit	342	Limit
M&VRa	8	20(7.9%)	43	Infs.
M&VRb	8	18(4.4%)	275	44
M&Va	Wrong	20(7.9%)	Wrong	Wrong
M&Vb	55(0.11%)	18(4.4%)	44(2.6%)	46(2%)
H&L	8	Infs.	Infs.	Infs.
Z&L&L	Wrong	32(11.2%)	Wrong	Wrong
Adaptive	6	46	48	73

Table 3.II – Number of PHA iterations with $\zeta = 0.01$

Method	KW3R	KW3Rb	app0110R	SGPF3Y3
Fixed	28	373	215	95
M&VRa	28	Limit	Limit	11(1.1%)
M&VRb	36	Limit	Limit	11(0.4%)
M&Va	28	299	102	11(1.08%)
M&Vb	25	234	86	11(0.4%)
H&L	83	Infs.	Infs.	49
Z&L&L	59	265	118	17(1.4%)
Adaptive	24	155	83	62

Method	Asset-Mgt	SGPF5Y4	wat10I16	wat10C32
Fixed	19	109	Limit	144
M&VRa	23	20(7.9%)	Limit	185
M&VRb	21	17(5%)	Limit	203
M&Va	Wrong	20(7.9%)	Wrong	Wrong
M&Vb	60(1.2%)	17(5%)	46(4.5%)	45(3.3%)
H&L	Wrong	Limit	286	95
Z&L&L	179	31(9.7%)	Wrong	Wrong
Adaptive	16	32	41	62

Table 3.III – Number of PHA iterations with $\zeta = 0.1$

Method	KW3R	KW3Rb	app0110R	SGPF3Y3
Fixed	44	324	109	467
M&VRa	29	234	126	11(1.2%)
M&VRb	60	Limit	Limit	11(0.8%)
M&Va	35	290	99	11(1.2%)
M&Vb	49	267	82	11(0.8%)
H&L	105	Limit	Infs.	45(0.4%)
Z&L&L	82	421	137	17(1.34%)
Adaptive	39	189	67	88

Method	Asset-Mgt	SGPF5Y4	wat10I16	wat10C32
Fixed	90	38	Limit	Limit
M&VRa	92	19(7.7%)	Limit	Limit
M&VRb	92	16(6.6%)	Limit	Limit
M&Va	Wrong	19(7.7%)	Wrong	Wrong
M&Vb	58(1.5%)	16(6.6%)	50(8.1%)	51(6.8%)
H&L	39(0.39%)	Limit	38(0.6%)	359(0.83%)
Z&L&L	Wrong	31(9.2%)	Wrong	Wrong
Adaptive	38	24	56	95

Table 3.IV – Number of PHA iterations with $\zeta = 0.5$

tive strategy is more efficient than keeping the penalty parameter fixed, and there is a slight advantage to start with a small initial value for the penalty parameter. The fixed approach is more sensitive to the choice of the initial penalty, a medium penalty being the best compromise in our experiments. The problems set being limited, we have to remain careful before we can derive strong conclusions, but the numerical results are nevertheless encouraging.

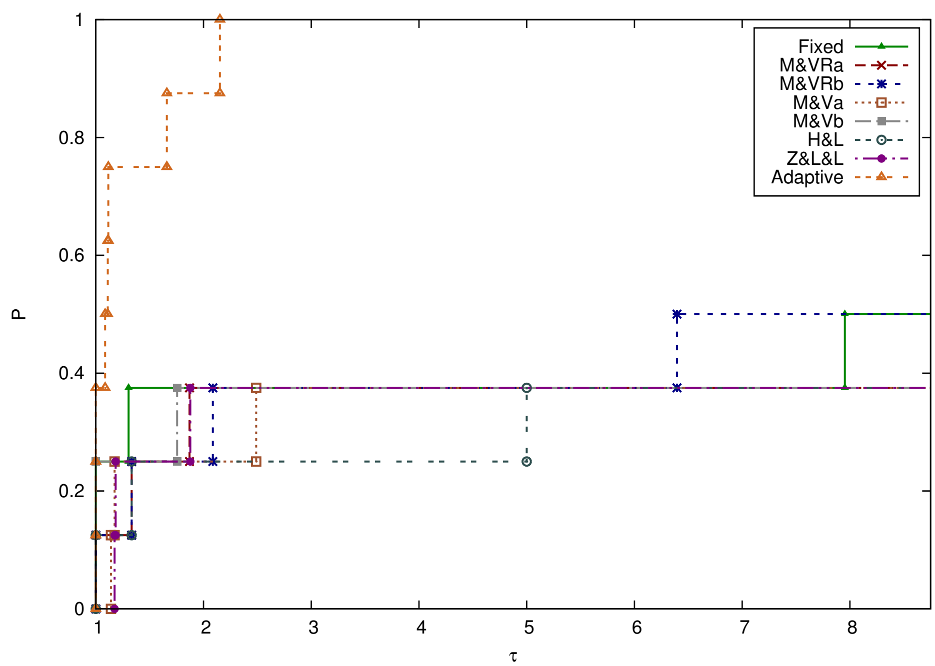

Figure 3.1 – Performance profile with $\zeta = 0.01$

3.3 Summary

PHA remains a popular scenarios decomposition method, but practical issues are still often observed. In particular, the choice of the penalty parameter value significantly influences the speed of convergence. A low value can produce a very slow convergence, while a large value will allow faster convergence, but the returned solution can be suboptimal. In order to circumvent these problems, many researchers have proposed heuristics to update the penalty parameters, but the study of their efficiency and robustness is of-

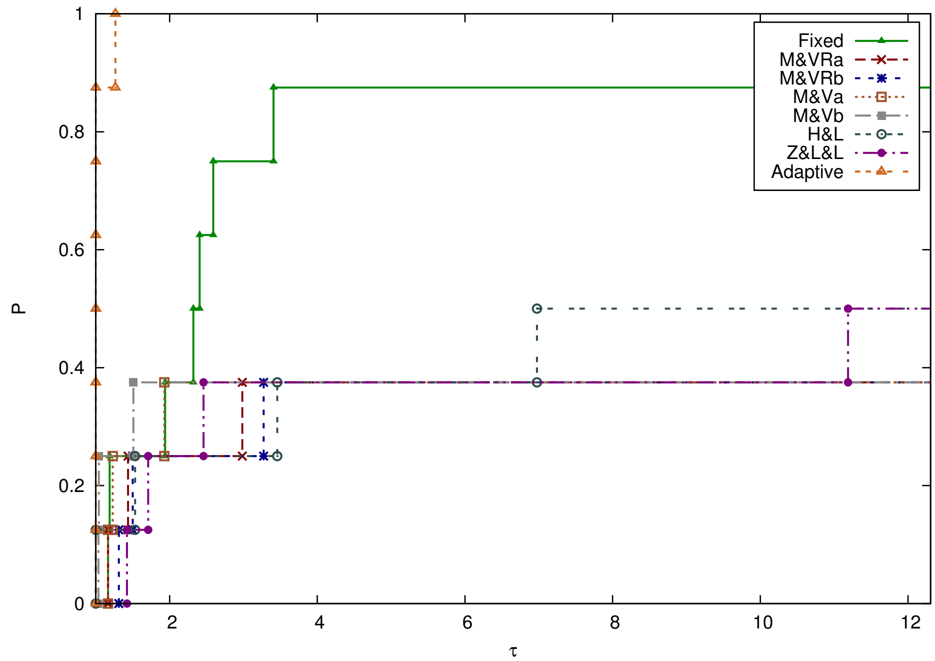

Figure 3.2 – Performance profile with $\zeta = 0.1$

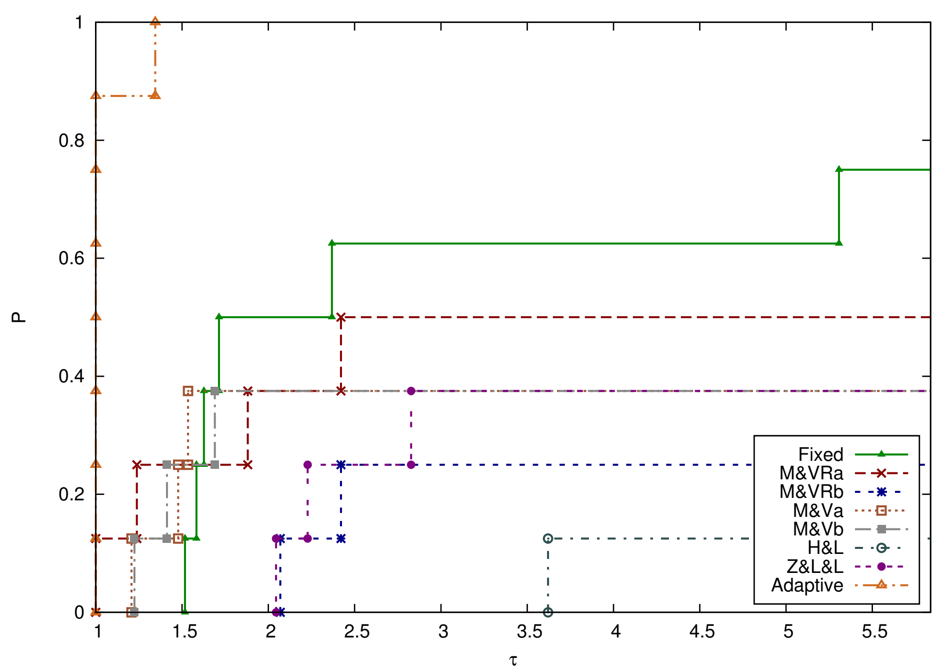

Figure 3.3 – Performance profile with $\zeta = 0.5$

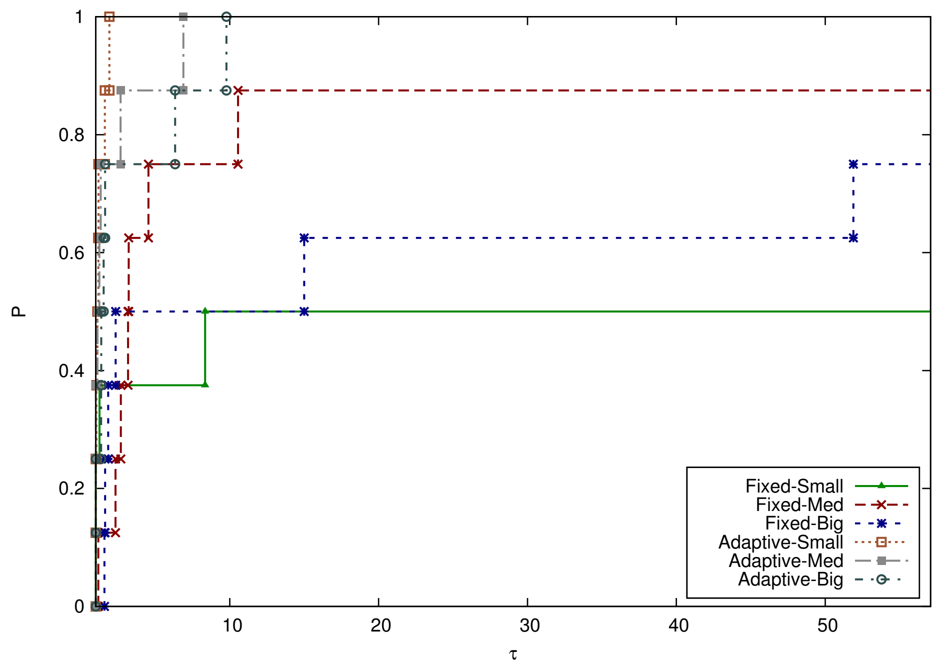

Figure 3.4 – Performance profile with fixed and adaptive strategies

ten limited, and valid for the application under consideration only. In this chapter, we have reviewed several approaches proposed in the literature, and observed that even for simple problems, we can face convergence issues. We then proposed a dynamic update that allows to increase or decrease the penalty parameter value, aiming to enforce the NA constraints only when they are correctly identified. While the proposed approach is still heuristic, we have observed a large improvement over the other strategies for the test problems, the method being fast and robust.

CHAPTER 4

CONCLUSION AND FUTURE WORKS

In this research, we focused on one of the decomposition techniques which uses the concept of scenario analysis to solve stochastic programming problems. PHA is such a method and it relaxes the NA constraints by using the principles of augmented Lagrangian method and proximal methods. Although PHA has been used in a vast range of stochastic programming problems, it still faces some practical issues. Many of these issues have been circumvented according to the nature the application in which PHA is involved.

One of the challenges is the choice of the penalty parameter which affects the performance of PHA significantly. As part of this research, we reviewed some major suggestions in the literature related to the choice of this parameter. We also proposed a dynamic update which provides an increment or decrement to the penalty parameter, depending if the NA constrains are correctly enforced or not. We increase the parameter if the NA constraints are correctly enforced or none of the variables' change is dominant. In reverse, we decrease the penalty parameter to prevent enforcing NA, while we are still far from the optimal solution. We keep the current value of the parameter, if PHA is already enforcing enough NA. Our new update rule performed robustly and fast over the other techniques. However, the proof of convergence is yet to be studied as a future work.

Another obstacle for the researchers using PHA is how to handle the quadratic term in the objective function. Although the augmented penalty term is powerful in forcing the relaxed constraints, it is still numerically difficult to solve such objective function. Specially in some applications (like MIP problems) having some nonconvexity, handling the quadratic term can be quite troublesome. Since again, there are some case dependent suggestions in the literature, we decided to work on this matter as future research. One idea is to make the NA constraints elastic and add a linear penalty on the violation of constraints to the original objective function. This idea has yet to be numerically tested. However, we believe that the proof of convergence is possible. The empirical test and the proof convergence are to be done as future works.

As we worked on linear multi-stage stochastic test problems, another possible next step would be to test the new dynamic update and elastic PHA on a set of nonlinear stochastic test problems.

BIBLIOGRAPHY

[1] Diana Barro and Elio Canestreli. Dynamic portfolio optimization: Time decomposition using the maximum principle with scenario approach. *European Journal of Operational Research*, 163(1):217–229, 2005.

[2] Fabian Bastin. *Trust-Region Algorithms for Nonlinear Stochastic Programming and Mixed Logit Models*. PhD thesis, Department of Mathematics, Universty of Namur, Namur, Belgium, 2004.

[3] Nils Jacob Berland and Kjetil K. Haugen. Mixing stochastic dynamic programming and scenario aggregation. *Annals of Operations Research*, 64(1):1–19, 1996.

[4] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, NY, USA, 1997.

[5] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, NY, USA, 2nd edition, 2011.

[6] Erik G. Boman. *Infeasibility and Negative Curvature in Optimization*. PhD thesis, Institute for Computational and Mathematical Engineering, Stanford University, California, US, 1999.

[7] Gerald G. Brown and Glenn W. Graves. Elastic programming: A new approach to large-scale mixed integer optimization. In *ORSA/TIMS Conference*, Las Vegas, NV, US, 1975.

[8] Pierre-Luc Carpentier, Michel Gendreau, and Fabian Bastin. Long-term hydro generation scheduling under uncertainty using the progressive hedging algorithm. *Water Ressources Research*, 49(5):2812–2827, 2013.

[9] Pierre-Luc Carpentier, Michel Gendreau, and Fabian Bastin. Optimal scenario set partitioning for multistage stochastic programming with the progressive hedging algorithm. Technical Report 2013–55, CIRRELT, Montreal, QC, Canada, 2013.

[10] Pedro M. S. Carvalho, Luis A. F. M. Ferreira, and L.M.F. Barruncho. Hedging large-scale distribution system investments against uncertainty. In *IEEE Power Engineering Society 1999 Winter Meeting*, volume 2, pages 901–906, 1999.

[11] Alice Chiche. *Théorie et algorithmes pour la résolution de problèmes numériques de grande taille: application à la gestion de production d'électricité*. PhD thesis, Université Pierre et Marie Curie, Paris, France, 2012.

[12] Bock J. Chun and Stephen M. Robinson. Scenario analysis via bundle decomposition. *Annals of Operations Research*, 56(1):39–63, 1995.

[13] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, PA, USA, 2000.

[14] Teodor G. Crainic, Xiaorui Fu, Michel Gendreau, Walter Rei, and Stein W. Wallace. Progressive hedging-based metaheuristics for stochastic network design. *Networks*, 58(2):114–124, 2011.

[15] Teodor Gabriel Crainic, Mike Hewitt, and Walter Rei. Scenario grouping in a progressive hedging-based meta-heuristic for stochastic network design. *Computers & Operations Research*, 43:90–99, 2014.

[16] George B. Dantzig. Linear programming under uncertainty. *Management Science*, 1(3-4):197–206, 1955.

[17] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[18] Jitka Dupačová. Applications of stochastic programming: Achievements and questions. *European Journal of Operational Research*, 140(2):281–290, 2002.

[19] Yueyue Fan and Changzheng Liu. Solving stochastic transportation network protection problems using the progressive hedging-based method. *Networks and Spatial Economics*, 10(2):193–208, 2010.

[20] Cristinca Fulga. Dynamic portfolio optimization for utility-based models. In *2009 International Conference on Information and Financial Engineering*, pages 117–121, Singapore, 2009.

[21] Horand I. Gassmann and Bjarni Kristjánsson. The SMPS format explained. *IMA Journal of Managament Mathematics*, 19(4):47–377, 2008.

[22] Philip E. Gill, Walter Murray, and Michael A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005.

[23] Raphael E. C. Gonçalves, Erlon Cristian Finardi, and Edson Luiz da Silva. Applying different decomposition schemes using the progressive hedging algorithm to the operation planning problem of hydrothermal system. *Electric Power Systems Research*, 83(1):19–27, 2012.

[24] Raphael E.C. Gonçalves, Erlon Cristian Finardi, and Edson Luiz da Silva. Exploring the progressive hedging characteristics in the solution of the medium-term operation planing problem. In 17^{th} *Power Systems Computation Conference*, Stockholm, Sweden, 2011.

[25] Serhat Gul. *Optimization of Surgery Delivery Systems*. PhD thesis, Industrial Engineering Department, Arizona State University, Arizona, US, 2010.

[26] Kjetil Haugen, Arne Løkketangen, and David L. Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132(1):116–122, 2001.

[27] Thorkell Helgason and Stein W. Wallace. Approximation scenario solutions in the progressive hedging algorithm. *Annals of Operations Research*, 31(1):425–444, 1991.

[28] Arlid Helseth. Stochastic network constrained hydro-thermal scheduling using a linearized progressive hedgning algorithm. *Energy Systems*, Forthcoming.

[29] Lars Hvattum and Arns Løkketangen. Using scenario trees and progressive hedging for stochastic inventory routing problems. *Journal of Heuristics*, 15(6):527–557, 2009.

[30] Tore W. Jonsbråten. Oil field optimization under price uncertainty. *The Journal of the Operational Research Society*, 49(8):811–818, 1998.

[31] Henrik Jönsson, Kurt Jörnsten, and Edward A. Silver. Application of the scenario aggregation approach to a two-stage, stochastic, common component, inventory problem with a budget constraint. *European Journal of Operational Research*, 68 (2):196–211, 1993.

[32] Kurt Jörnsten and Mette Bjorndal. Dynamic location under uncertainty. *Studies in Regional and Urban Planning*, 3:163–184, 1994.

[33] Kurt Jörnsten and Rainer Leisten. Scenario aggregation in single-resource production planning models with unceratain demand. *Production Planning & Control: The Management of Operations*, 5(3):271–281, 1994.

[34] Kurt O. Jørnsten. Sequencing offshore oil and gas fields under uncertainty. *European Journal of Operational Research*, 58(2):191–201, 1992.

[35] Peter Kall and Stein W. Wallace. *Stochastic Programming*. John Wiley & Sons, 1994.

[36] Michal Kaut. COIN-OR tools for stochastic programming. In Miloš Kopa, editor, *On Selected Software for Stochastic Programming*, chapter 6, pages 88–116. MatfyzPress, Prague, Czech Republic, 2008.

[37] Lubomír Klimeš, Tomáš Mauder, and Josef Štětina. Stochastic approach and optimal control of continuous steel casting process by using progressive hedging algorithm. In 20^{th} *International Conference on Metallurgy and Metals METAL*, pages 146–151, Czech Republic, 2011.

[38] Amina Lamghari and Roussos Dimitrakopoulos. Progressive hedging applied as a metaheuristic to schedule production in open-pit mines accounting for reserve uncertainty. *European Journal of Operational Research*, 2016.

[39] Jeff Linderoth and Stephen Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2–3):207–250, 2003.

[40] Ovidiu Listes and Rommert Dekker. A scenario aggregation based approach for determining a robust airline fleet composition. *Transportation Science*, 39(3):367–382, 2005.

[41] Xinwei Liu, Kim-Chuan Toh, and Gongyun Zhao. On the implementation of a log-barrier progressive hedging method for multistage stochastic programs. *Journal of Computational and Applied Mathematics*, 234(2):579–592, 2010.

[42] Arne Løkketangen and David L. Woodruff. Progressive hedging and tabu search applied to mixed integer (0,1) multistage stochastic programming. *Journal of Heuristics*, 2(2):111–128, 1996.

[43] John M. Mulvey and Hercules Vladimirou. Evaluation of a parallel hedging algorithm for stochastic network programming. In Ramesh Sharda, Bruce L. Golden, Edward Wasil, Osman Balci, and William Stewart, editors, *Impacts of Recent Computer Advances on Operations Research*, Publications in Operations Research Series, pages 106–119. Elsevier Science Publishing, 1989.

[44] John M. Mulvey and Hercules Vladimirou. Solving multistage stochastic networks: An application of scenario aggregation. *Networks*, 21(6):619–643, 1991.

[45] John M. Mulvey and Hercules Vladimirou. Applying the progressive hedging algorithm to stochastic generalized networks. *Annals of Operation Research*, 31(1):399–424, 1991.

[46] John M. Mulvey and Hercules Vladimirou. Stochastic network programming for financial planning problems. *Management Science*, 38(11):1642–1664, 1992.

[47] Bruce A. Murtagh and Michael A. Saunders. Minos 5.5 user's guide. Technical Report SOL 83-20R, Department of Operations Research, Standford University, 1998.

[48] Tiziano Parriani, Guojing Cong, Carol Meyers, and Deepak Rajan. An efficient approach for solving large stochastic unit commitment problems arising in a california iso planning model. In *2014 IEEE Power and Energy Society General Meeting Conference & Exposition*, pages 1–5, National Harbor, MD, USA, 2014.

[49] Guido Perboli, Luca Gobbato, and Francesca Maggioni. A progressive hedging method for the multi-path traveling salesman problem with stochastic travel times. *IMA Journal of Management Mathematics*, pages 1–22, 2015.

[50] Francisco S. Reis, Pedro M. S. Carvalho, and Luis A. F. M. Ferreira. Reinforcement scheduling convergence in power systems transmission planning. *IEEE Transactions on Power Systems*, 20(2):1151–1157, 2005.

[51] R. Tyrrell Rockafellar and Roger J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[52] Sarah M. Ryan, Roger J.-B. Wets, David L. Woodruff, César Silva-Monroy, and Jean-Paul Watson. Toward scalable, parallel progressive hedging for stochastic unit commitment. In *2013 IEEE Power and Energy Society General Meeting*, pages 1–5, 2013.

[53] Marcelo L. L. dos Santos, Edson Luiz da Silva, Erlon Christian Finardi, and Raphael E. C. Gonçalves. Using the progressive hedging method to solve the short term operating planning problem for hydrothermal systems. In *EngOpt 2008: International Conference on Engineering Optimization*, Rio de Janeiro, Brazil, 2008.

[54] Alexander Shapiro. Monte Carlo sampling methods. In Alexander Shapiro and Andrzej Ruszczyński, editors, *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*, pages 353–425. Elsevier, 2003.

[55] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on Stochastic Programming*. SIAM, Philadelphia, PA, USA, 2009.

[56] Michael Somervell. *Progressive Hedging in Parallel*. PhD thesis, University of Auckland, Auckland, New Zealand, 1998.

[57] Samer Takriti and John R. Birge. Lagrangian solution techniques and bounds for loosely coupled mixed-integer stochastic programs. *Operations Research*, 48(1): 91–98, 2000.

[58] Samer Takriti, John R. Birge, and Erik Long. A stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems*, 11(3):1497–1508, 1996.

[59] Marcelo L. L. dos Santos, Edson Luiz da Silva, Erlon Christian Finardi, and Raphael E. C. Gonçalves. Practical aspects in solving the medium-term operation planning problem of hydrothermal power systems by using the progressive hedging method. *International Journal of Electrical Power & Energy Systems*, 31 (9):546–552, 2009.

[60] Fernando B. Veliz, Jean-Paul Watson, Andres Weintraub, Roger J.-B. Wets, and David L. Woodruff. Stochastic optimization models in forest planning: A progressive hedging solution approach. *Annals of Operations Research*, 232(1):259–274, 2014.

[61] Jean-Paul Watson and David L. Woodruff. Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, 8(4):355–370, 2011.

[62] Jean-Paul Watson, David L. Woodruff, and David R. Strip. Progressive hedging innovations for a stochastic spare parts support enterprise. *Naval Research Logistics*, 2007.

[63] Jean-Paul Watson, David L. Woodruff, and William E. Hart. PySP: Modeling and solving stochastic programs in Python. *Mathematical Programming computation*, 4(2):109–149, 2012.

[64] Roger J-B Wets. The aggregation principle in scenario analysis and stochastic optimization. *Algorithms and Modeling in Mathematical Programming*, 51:91–113, 1989.

[65] Hongyu Wu, Erik Ela, Ibrahim Krad, Anthony Florita, Jie Zhang, Bri-Mathias Hodge, Eduardo Ibanez, and Wenzhong Gao. An assessment of the impact of stochastic day-ahead scuc on economic and reliability metrics at multiple timescales. In *2015 IEEE Power and Energy Society General Meeting*, pages 1–5, 2015.

[66] Masoumeh K. Zanjani, Mustapha Nourelfath, and Daoud Ait-Kadi. A scenario decomposition approach for stochastic production planning in sawmills. *Journal of Operational Research Society*, 64(1):48–59, 2013.

[67] Shohre Zehtabian and Fabian Bastin. Penalty parameter strategies in progressive hedging algorithm. Technical Report 2016–12, CIRRELT, Montreal, Qc, Canada, 2016.

[68] Luckny Zéphyr, Pascal Lang, and Bernard F. Lamond. Adaptive monitoring of the progressive hedging penalty for reservoir systems management. *Energy Systems*, 5 (2):307–322, 2014.

[69] Gongyun Zhao. A Lagrangian dual method with self-concordant barriers for multi-stage stochastic convex programming. *Mathematical Programming*, 102(1):1–24, 2004.

Appendix I

Data sets preparations

In this appendix, we explain our major data sets as well as how we managed to have the information for scenario subproblems by modifying an open source interface. First we introduce this interface briefly and then discuss our modifications.

I.1 Stochastic Modeling Interface

The Computational Infrastructure for Operation Research (COIN-OR) (`http://www.coin-or.org`) is a collection of open-source projects developing software for the operations research (OR) community. There are two software projects for stochastic optimization programs available from COIN-OR): COmmon Optimization Python Repository (Cooper) and SMI. Cooper software project integrates a variety of Python optimization-related packages [36], while SMI is an interface for stochastic programming models developed in C++. As we aim to study the performance of our suggested strategy update for PHA on linear problems with C++ programming language, we use SMI (`http://www.coin-or.org/projects/Smi.xml`) to provide our required data sets. Here are the main features of SMI:

— it constructs a scenario tree structure for multi period stochastic data,

— it is a stochastic MPS [1] (SMPS) reader,

— it is an interface for generating scenario trees from paths and from discrete random variables,

— it generates an open source interface (OSI) object with the deterministic equivalent problem,

— it is a parser of the solutions by stage and scenario.

1. Mathematical Programming System (MPS) is a file format for presenting and archiving linear programming (LP) and MIP problems.

Since almost all stochastic test problems from the literature exist in SMPS format, we should introduce this data format.

The SMPS format is available to describe stochastic linear programs, as proposed by Gassmann and Kristjánsson [21]. SMPS makes use of three text files with an MPS record layout. The role of each file is briefly explained in the following:

— the core file gives a list of all the deterministic information of the problem. The information include the name and type of each constraint, the names of rows and columns, matrix of coefficients in a column-order, right hand-side of constraints and bounds on the variables. It also gives the locations of the stochastic elements,

— the time file describes the dynamic structure of the problem and breaks the data into stages. If the core file is given in time-ordered fashion, then this is a simple matter of recording the first row and column of each stage, otherwise a full list of rows and columns must be given along with the stage to which each of them belongs,

— the stoch file gives the stochastic data. There are several ways to present the information. The purpose of stoch file is to generate an event tree. However there are other features considered for stoch file, such as linear and quadratic penalties for violating a stochastic constraint, probabilistic constraints and objectives and integrated chance constraints.

For more details about SMPS, we refer to [21].

I.2 Modified Stochastic Modeling Interface: A stochastic benchmark tool for scenario decomposition techniques

According to our requirements in PHA, none of the original features of SMI could exactly be used to construct each scenario subproblem and solve it separately. In order to achieve our goal we decided to make some modifications in SMI. The original version of SMI either can get the data from SMPS files by a solver or directly from a user and then give the access only to the solution. In both methods, SMI creates a deterministic equivalent of the stochastic problem which is not accessible to us. So we made changes such that it is now possible to have access to the information per scenario and save them as well as to solve each scenario subproblem separately. Through the modification process, we found out that there was a modeling bug while SMI was trying to construct the whole scenario tree. At each stage, instead of copying the information from a parent node to its child nodes, it copied the information from the root node in the scenario tree. We managed to have this issue solved through personal contacts with SMI project manager, Dr. Alan King, whom we are grateful for his collaboration. Now we can say that with the new changes we created a stochastic benchmark tool for scenario decomposition techniques. The modified codes of SMI are available upon request.

Appendix II

Supplement to Chapter 3

II.1 Proof of equality (2.52)

In this section, we develop the proof of the equality (2.52)

$$\sum_{s\in\mathcal{S}} p_s\|x^{s,k+1}-\hat{x}^{s,k}\|^2 = \sum_{s\in\mathcal{S}} p_s\|x^{s,k+1}-\hat{x}^{s,k+1}\|^2 + \sum_{s\in\mathcal{S}} p_s\|\hat{x}^{s,k+1}-\hat{x}^{s,k}\|^2.$$

To prove the equality (2.52), we work on each term separately. For that we use the notations and definitions introduced in Chapter 2.

To start, we first have

$$\sum_{s\in\mathcal{S}} p_s\|x^{(s),k+1}-\hat{x}^{(s),k}\|^2 = \|x^{k+1}-\hat{x}^k\|^2_{\mathcal{D}} = \|x^{k+1}\|^2_{\mathcal{D}} - 2\langle x^{k+1},\hat{x}^k\rangle_{\mathcal{D}} + \|\hat{x}^k\|^2_{\mathcal{D}}. \quad \text{(II.1)}$$

Similarly,

$$\sum_{s\in\mathcal{S}} p_s\|\hat{x}^{(s),k+1}-\hat{x}^{(s),k}\|^2 = \|\hat{x}^{k+1}-\hat{x}^k\|^2_{\mathcal{D}} = \|\hat{x}^{k+1}\|^2_{\mathcal{D}} - 2\langle \hat{x}^{k+1},\hat{x}^k\rangle_{\mathcal{D}} + \|\hat{x}^k\|^2_{\mathcal{D}}, \quad \text{(II.2)}$$

Moreover, we have

$$\begin{aligned}
\sum_{s\in\mathcal{S}} p_s\|x^{(s),k+1}-\hat{x}^{(s),k+1}\|^2 &= \sum_t \sum_{\mathcal{S}_t^{(s)}} \sum_{s'\in\mathcal{S}_t^{(s)}} p_{s'}\|x_t^{(s'),k+1}-\hat{x}_t^{(s'),k+1}\|^2 \\
&= \sum_t \sum_{\mathcal{S}_t^{(s)}} \sum_{s'\in\mathcal{S}_t^{(s)}} p_{s'}\left(\|x_t^{(s'),k+1}\|^2-\|\hat{x}_t^{(s'),k+1}\|^2\right) \quad \text{(II.3)} \\
&= \|x^{k+1}\|^2_{\mathcal{D}} - \|\hat{x}^{k+1}\|^2_{\mathcal{D}},
\end{aligned}$$

Combining (II.2) and (II.3), we have

$$\sum_{s\in\mathcal{S}} p_s\|x^{(s),k+1}-\hat{x}^{(s),k+1}\|^2 + \sum_{s\in\mathcal{S}} p_s\|\hat{x}^{(s),k+1}-\hat{x}^{(s),k}\|^2 = \|x^{k+1}\|^2_{\mathcal{D}} - 2\langle \hat{x}^{k+1},\hat{x}^k\rangle_{\mathcal{D}} + \|\hat{x}^k\|^2_{\mathcal{D}},$$

that corresponds to (II.1) as

$$\langle x^{k+1} - \hat{x}^{k+1}, \hat{x}^k \rangle_{\mathcal{D}} = \langle K x^{k+1}, J x^k \rangle_{\mathcal{D}} = \langle x^{k+1}, K J x^k \rangle_{\mathcal{D}} = 0,$$

since K is an orthogonal projection operator. This concludes the proof.

II.2 Detailed numerical results

We provide in Tables II.I–II.VIII the detailed results of our numerical experimentations over the eight test problems. For each problem, we compare thirty penalty parameter update strategies, and provide the final objective value along with the gap to optimal value in brackets. We also report the number of iterations and the computation time. In case we reach the iteration or time limit before declaring convergence, we mention "Limit" in the corresponding cell.

Update	Objective value	Iterations	Time (s)
Fixed-Sml	2613 (0)	30	3.53
Fixed-Med	2613 (0)	28	3.14
Fixed-Big	2613 (0)	44	4.77
M&VRa	2613 (0)	Limit	122.22
M&VRa-Small	2613 (0)	43	5.11
M&VRa-Medium	2613 (0)	28	3.18
M&VRa-Large	2613 (0)	29	3.32
M&VRb	2613 (0)	64	8.1
M&VRb-Small	2613 (0)	48	5.7
M&VRb-Medium	2613 (0)	36	4.07
M&VRb-Large	2613 (0)	60	6.94
M&Va	2613 (0)	29	3.56
M&Va-Small	2613 (0)	27	3.1
M&Va-Medium	2613 (0)	28	3.24
M&Va-Large	2613 (0)	35	3.97
M&Vb	2613 (0)	25	2.88
M&Vb-Small	2613 (0)	23	2.6
M&Vb-Medium	2613 (0)	25	2.97
M&Vb-Large	2613 (0)	49	5.77
H&L	2613 (0)	118	17.6
H&L-Small	2613 (0)	115	16.53
H&L-Medium	2613 (0)	83	11.85
H&L-Large	2613 (0)	105	15.3
Z&L&L	2613 (0)	28	2.53
Z&L&L-Small	2613 (0)	27	3.15
Z&L&L-Medium	2613 (0)	59	7.14
Z&L&L-Large	2613 (0)	82	10.99
Adaptive-Small	2613 (0)	25	2.87
Adaptive-Medium	2613 (0)	24	2.65
Adaptive-Large	2613 (0)	39	4.36

Table II.I – KW3R problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	3197.34 (-0.21)	Limit	123.92
Fixed-Med	3204 (0)	373	74.8
Fixed-Big	3204 (0)	324	60.67
M&VRa	3204.25 (0.008)	Limit	124.33
M&VRa-Small	3204.93 (0.03)	Limit	124.31
M&VRa-Medium	3204.05 (0.001)	Limit	123.91
M&VRa-Large	3204 (0)	234	39.58
M&VRb	3204 (0)	Limit	122.23
M&VRb-Small	3204.16 (0.005)	Limit	121.3
M&VRb-Medium	3204 (0)	Limit	120.62
M&VRb-Large	3204.26 (0.008)	Limit	118.19
M&Va	3204 (0)	348	69.29
M&Va-Small	3204 (0)	346	69.81
M&Va-Medium	3204 (0)	299	56.12
M&Va-Large	3204 (0)	290	52.48
M&Vb	3204 (0)	244	41.04
M&Vb-Small	3204 (0)	244	40.6
M&Vb-Medium	3204 (0)	234	39
M&Vb-Large	3204 (0)	267	46.39
H&L	3193 (-0.3)	Limit	123.75
H&L-Small	3151.08 (-2)	Limit	125.7
H&L-Medium	3154.16 (-1)	Limit	124.58
H&L-Large	3204 (0)	Limit	122.29
Z&L&L	3204 (0)	266	41.92
Z&L&L-Small	3204 (0)	261	44.38
Z&L&L-Medium	3204 (0)	265	44.85
Z&L&L-Large	3204 (0)	421	91.72
Adaptive-Small	3204 (0)	139	20.98
Adaptive-Medium	3204 (0)	155	23.55
Adaptive-Large	3204 (0)	189	29.91

Table II.II – Modified KW3R problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	39.4 (-6)	Limit	424.77
Fixed-Med	41.96 (0)	215	91.35
Fixed-Big	41.96 (0)	109	30.75
M&VRa	41.96 (0)	Limit	421.14
M&VRa-Small	42.01 (0.12)	Limit	423.34
M&VRa-Medium	41.96 (0)	Limit	422.51
M&VRa-Large	41.96 (0)	126	38.72
M&VRb	41.963 (0.007)	Limit	407.18
M&VRb-Small	41.96 (0)	Limit	430.5
M&VRb-Medium	42 (0.09)	Limit	422.19
M&VRb-Large	41.96 (0)	Limit	422.97
M&Va	41.96 (0)	105	29.34
M&Va-Small	41.96 (0)	105	29.13
M&Va-Medium	41.96 (0)	102	28.42
M&Va-Large	41.96 (0)	99	25.79
M&Vb	41.96 (0)	88	21.89
M&Vb-Small	41.96 (0)	92	23.03
M&Vb-Medium	41.96 (0)	86	21.41
M&Vb-Large	41.96 (0)	82	20.1
H&L	34.83 (-17)	Limit	423.63
H&L-Small	19.25 (-54)	Limit	424.79
H&L-Medium	31.47 (-25)	Limit	421.84
H&L-Large	37.81 (-9.9)	Limit	422.05
Z&L&L	41.96 (0)	125	35.75
Z&L&L-Small	41.96 (0)	109	29.15
Z&L&L-Medium	41.96 (0)	118	34.58
Z&L&L-Large	41.96 (0)	137	43.91
Adaptive-Small	41.96 (0)	108	29.04
Adaptive-Medium	41.96 (0)	83	20.53
Adaptive-Large	41.96 (0)	67	14.48

Table II.III – app0110R problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	-2967.90 (0.0003)	9	6.93
Fixed-Med	-2967.90 (0.0003)	95	385.7
Fixed-Big	-2967.90 (0.0003)	467	8475.53
M&VRa	-2927.3 (1.4)	14	13.64
M&VRa-Small	-2935.81 (1.1)	12	11.09
M&VRa-Medium	-2935.86 (1.1)	11	9.66
M&VRa-Large	-2931.62 (1.22)	11	9.5
M&VRb	-2879.93 (3)	15	15.21
M&VRb-Small	-2958.85 (0.3)	13	12.3
M&VRb-Medium	-2954.63 (0.4)	11	9.93
M&VRb-Large	-2943.71 (0.8)	11	9.61
M&Va	-2927.30 (1.37)	14	13.52
M&Va-Small	-2935.81 (1.08)	12	10.82
M&Va-Medium	-2935.86 (1.08)	11	9.83
M&Va-Large	-2931.62 (1.22)	11	9.82
M&Vb	-2879.93 (3)	15	15.34
M&Vb-Small	-2958.85 (0.3)	13	12.38
M&Vb-Medium	-2954.63 (0.4)	11	9.64
M&Vb-Large	-2943.71 (0.8)	11	9.55
H&L	-2895.27 (2.4)	19	21.98
H&L-Small	-2967.90 (0.0003)	9	6.81
H&L-Medium	-2967.83 (0.003)	49	113.37
H&L-Large	-2957.07 (0.4)	45	98.7
Z&L&L	-2917.65 (1.7)	17	17.66
Z&L&L-Small	-2912.12 (1.9)	18	20.34
Z&L&L-Medium	-2926.65 (1.4)	17	18.79
Z&L&L-Large	-2928.06 (1.34)	17	18.95
Adaptive-Small	-2967.90 (0.0003)	10	8.07
Adaptive-Medium	-2967.84 (0.002)	62	176.83
Adaptive-Large	-2967.83 (0.003)	88	324.23

Table II.IV – SGPF3Y3 problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	-164.74 (0)	6	0.52
Fixed-Med	-164.74 (0)	19	1.63
Fixed-Big	-164.74 (0)	90	9.93
M&VRa	-164.74 (0)	173	25.05
M&VRa-Small	-164.74 (0)	8	0.7
M&VRa-Medium	-164.74 (0)	23	2.76
M&VRa-Large	-164.74 (0)	92	11.35
M&VRb	-164.74 (0)	175	20.86
M&VRb-Small	-164.74 (0)	8	0.68
M&VRb-Medium	-164.74 (0)	21	1.86
M&VRb-Large	-164.74 (0)	92	10.12
M&Va	-162.57 (1.3)	Limit	92.28
M&Va-Small	-163.18 (1.0)	Limit	94.27
M&Va-Medium	-162.84 (1.2)	Limit	94.61
M&Va-Large	-162.55 (1.3)	Limit	93.6
M&Vb	-162.12 (1.6)	57	6.21
M&Vb-Small	-164.55 (0.1)	55	5.42
M&Vb-Medium	-162.72 (1.2)	60	5.67
M&Vb-Large	-162.19 (1.5)	58	5.39
H&L	-162.09 (1.6)	29	2.25
H&L-Small	-164.74 (0)	8	0.68
H&L-Medium	-164.16 (0.4)	Limit	90.85
H&L-Large	-164.17 (0.3)	39	4.4
Z&L&L	-164.73 (0.006)	Limit	95.71
Z&L&L-Small	-164.44 (0.2)	Limit	89.82
Z&L&L-Medium	-164.74 (0)	179	24.1
Z&L&L-Large	-163.09 (1)	Limit	84.82
Adaptive-Small	-164.74 (0)	6	0.51
Adaptive-Medium	-164.74 (0)	16	1.36
Adaptive-Large	-164.74 (0)	38	3.43

Table II.V – Asset problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	-4031.30 (0)	134	Limit
Fixed-Med	-4031.30 (0)	109	24542.4
Fixed-Big	-4031.30 (0)	38	3132.49
M&VRa	-3573.08 (11.4)	26	1553.29
M&VRa-Small	-3711.83 (7.9)	20	897
M&VRa-Medium	-3711.38 (7.9)	20	898.47
M&VRa-Large	-3722.37 (7.7)	19	818.35
M&VRb	-3372.32 (16.3)	25	1447.19
M&VRb-Small	-3853.78 (4.4)	18	744.29
M&VRb-Medium	-3829.95 (5)	17	661.26
M&VRb-Large	-3764.81 (6.6)	16	605.09
M&Va	-3573.08 (11.4)	26	1588.28
M&Va-Small	-3711.83 (7.9)	20	897.03
M&Va-Medium	-3711.38 (7.9)	20	895.65
M&Va-Large	-3722.37 (7.7)	19	821.27
M&Vb	-3372.32 (16.3)	25	1440.74
M&Vb-Small	-3853.78 (4.4)	18	735.41
M&Vb-Medium	-3829.95 (5)	17	668.51
M&Vb-Large	-3764.81 (6.6)	16	603.77
H&L	-3575.56 (11.3)	111	25407.78
H&L-Small	-4138.81 (-2.7)	134	Limit
H&L-Medium	-4031.30 (0)	134	Limit
H&L-Large	-4029.73 (0.04)	134	Limit
Z&L&L	-3638.27 (9.8)	32	2318.17
Z&L&L-Small	-3578 (11.2)	32	2310.63
Z&L&L-Medium	-3640.91 (9.7)	31	2191.27
Z&L&L-Large	-3660.43 (9.2)	31	2169.85
Adaptive-Small	-4031.30 (0)	46	4551.63
Adaptive-Medium	-4031.30 (0)	32	2317.18
Adaptive-Large	-4031.30 (0)	24	1318.47

Table II.VI – SGPF5Y4 problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	-2158.75 (0)	342	3800.11
Fixed-Med	-2158.74 (0.0005)	Limit	8003.73
Fixed-Big	-2158.74 (0.0005)	Limit	8061.06
M&VRa	-2134.91 (1.1)	Limit	8108.3
M&VRa-Small	-2158.74 (0.0005)	43	78.45
M&VRa-Medium	-2158.74 (0.0005)	Limit	7921.8
M&VRa-Large	-2158.74 (0.0005)	Limit	7940.46
M&VRb	-2148.12 (0.5)	Limit	7789.87
M&VRb-Small	-2158.75 (0)	275	2476.09
M&VRb-Medium	-2158.74 (0.0005)	Limit	8026.43
M&VRb-Large	-2158.74 (0.0005)	Limit	8070.88
M&Va	-1931.3 (10.5)	Limit	8067.18
M&Va-Small	-2013.88 (6.7)	Limit	8078.32
M&Va-Medium	-2001.70 (7.3)	Limit	8094.26
M&Va-Large	-1975.13 (8.5)	Limit	8086.62
M&Vb	-1790.44 (17.1)	Limit	7962.29
M&Vb-Small	-2102.07 (2.6)	44	80.36
M&Vb-Medium	-2061.52 (4.5)	46	86.18
M&Vb-Large	-1983.14 (8.1)	50	100.94
H&L	-1728.57 (20)	67	172.87
H&L-Small	-2251.52 (-4.3)	Limit	7983.85
H&L-Medium	-2158.75 (0)	286	2687.91
H&L-Large	-2145.81 (0.6)	38	62.14
Z&L&L	-2100.79 (2.68)	Limit	8052.37
Z&L&L-Small	-2076.36 (3.8)	Limit	8092.45
Z&L&L-Medium	-2088 (3.3)	Limit	7864.39
Z&L&L-Large	-2110.99 (2.2)	Limit	8037.98
Adaptive-Small	-2158.74 (0.0005)	48	94.54
Adaptive-Medium	-2158.71 (0.002)	41	65.97
Adaptive-Large	-2158.71 (0.002)	56	121.74

Table II.VII – wat10I16 problem

Update	Objective value	Iterations	Time (s)
Fixed-Sml	-2611.92 (0)	Limit	33954.18
Fixed-Med	-2611.92 (0)	144	2930.41
Fixed-Big	-2611.87 (0.002)	Limit	33785.49
M&VRa	-2572.14 (1.5)	Limit	34108.57
M&VRa-Small	-2612.96 (-0.04)	Limit	34040.47
M&VRa-Medium	-2611.01 (0.03)	185	4960.15
M&VRa-Large	-2610.72 (0.04)	Limit	34069.22
M&VRb	-2585.06 (1)	Limit	34029.37
M&VRb-Small	-2611.21 (0.03)	44	289.91
M&VRb-Medium	-2611.84 (0.003)	203	5944.1
M&VRb-Large	-2611.77 (0.006)	Limit	34000.12
M&Va	-2377.84 (9)	Limit	34112.42
M&Va-Small	-2464.55 (5.6)	Limit	33858.15
M&Va-Medium	-2457.85 (5.9)	Limit	33813.32
M&Va-Large	-2428.80 (7)	Limit	33953.17
M&Vb	2260.01 (13.5)	72	777.52
M&Vb-Small	-2558.96 b	46	338.36
M&Vb-Medium	-2525.67 (3.3)	45	309.53
M&Vb-Large	-2434.35 (6.8)	51	406.14
H&L	-2216.59 (15.1)	35	203.9
H&L-Small	-2809.08 (-7.5)	Limit	34057
H&L-Medium	-2611.92 (0)	95	1353.35
H&L-Large	-2590.12 (0.83)	359	18151.14
Z&L&L	-2564.12 (1.83)	Limit	33928.35
Z&L&L-Small	-2530.93 (3.1)	Limit	33985.33
Z&L&L-Medium	-2551.90 (2.3)	Limit	34031.9
Z&L&L-Large	-2571.58 (1.5)	Limit	34090.2
Adaptive-Small	-2611.92 (0)	73	795.2
Adaptive-Medium	-2611.92 (0)	62	587.24
Adaptive-Large	-2611.87 (0.002)	95	1380.99

Table II.VIII – wat10C32 problem