Decision Rules for Optimization under Uncertainty: Algorithms, Advances, and Applications

by

Said Salim Rahal

A thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

in

Process Control

Department of Chemical and Materials Engineering

University of Alberta

Abstract

This thesis aims to investigate, develop and advance solution techniques for optimization under uncertainty in process control, scheduling and operations research applications. Decision rule methods offer a rich and flexible framework for solving these classes of problems. Recent literature has shown the promise of decision rules in which uncertainty-dependent decisions are represented as functions, whose parameters are decision variables to be optimized, of the underlying uncertain parameters.

First, we investigate hybrid strategies using linear and piecewise linear decision rule and we empirically illustrate that it is more favorable to have higher uncertainty refinement, equivalently better approximation quality of decisions, at the start of the decision-making process. We also demonstrate a case where, unexpectedly, a linear decision rule is superior to a more complex piecewise-linear decision rule within a simulator. This bolsters the need to assess the quality of decision rule in a simulator to obtain an impartial assessment of its solution quality.

Second, we develop a systematic approach to devise a linear decision rule for unit-specific event-based continuous-time formulation via steel-making and continuous casting problem. We illustrate the solution quality of reactive, proactive and hybrid scheduling strategies and we emphasize the added value of the latter strategy as an attractive trade-off between solution conservatism and excessive scheduling modifications.

Third, we utilize the concept of performing simple successive operations to extract complex features, borrowed from the deep learning community, in the context of optimization under uncertainty. It led to the development of deep lifted decision rules which are shown to offer attractive approximation quality. In this regard, we craft solution heuristics to optimize the aforementioned decision rule inspired by the stochastic gradient descent method.

Fourth, we propose an approach to construct polyhedral norm uncertainty sets, in particular, we characterize asymmetry in data-drive uncertain parameters using distribution information. We show the added value of capturing the asymmetry and the benefits of modelling the data-driven uncertain parameters as an intersection of two polyhedral norm sets. This work addresses the assumptions often made in optimization under uncertainty regarding a predefined polytopic uncertainty set.

Fifth, we draw a connection between linear parametric programming and decision rules. We suggest using decision rules to approximate parametric solutions for optimization problems with a large number of uncertain parameters and variables. Parametric programming is limited by the latter class of problems due to exacerbating complexity and memory storage requirements. We develop a rectilinear activation unit decision rule approximate algorithm which incorporates a branching scheme to refine the approximate solution. The concept of rectilinear activation unit decision rule is based on augmenting new flexible uncertain parameters (i.e., features) obtained from a 1-layer network to a linear decision rule. We demonstrate the benefits of the algorithm in terms of solution quality and computational cost.

In terms of its overall impact, this thesis makes several important contributions. From a methodological perspective, it introduces and advances the use of novel decision rules as promising solution techniques for optimization under uncertainty. From an application perspective, it develops and promotes the less-known decision rule methods in process scheduling and control.

Preface

This thesis is an original work of Said Rahal. It is conducted under the supervision of Dr. Zukui Li and in collaboration with Dr. Dimitri J. Papageorgiou. It is funded by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Chapter 2 of this thesis has been published as Rahal, S. Papageorgiou, D. Li, Z 2020. "Hybrid Strategies using Linear and Piecewise-Linear Decision Rules for Multistage Adaptive Linear Optimization". *European Journal of Operation Research*

Chapter 3 of this thesis has been published as Rahal, S. Li, Z. Papageorgiou, D, 2020. "Proactive and reactive scheduling of the steelmaking and continuous casting process through adaptive robust optimization". *Computers & Chemical Engineering, 133*, p. 106658.

Chapter 4 of this thesis has been submitted as Rahal, S. Li, Z. Papageorgiou, D, 2020. "Deep Lifted Decision Rules for Two-stage Adaptive Optimization". Submitted to *Computers & Operations Research.*

Chapter 5 of this thesis has been accepted as Rahal, S. Li, Z, 2020. "Norm Induced Polyhedral Uncertainty Set for Robust Linear Optimization". Submitted to *Optimization & Engineering* (minor revision in progress).

Chapter 6 of this thesis is a manuscript under preparation with the title "Decision Rule Methods for Multiparametric Linear Programming"

In chapters 2, 3, 4 and 6, I am responsible for theory development, algorithmic implementation and computational experiments set-up and analysis as well as the manuscript write-up. Dr. Zukui Li and Dr. Dimitri J. Papageorgiou are responsible for concept formation, manuscript compositions and edits.

In chapter 5, Dr. Zukui Li and I are both responsible for theory development, algorithmic implementation and computational experiments set-up and analysis as well as the manuscript write-up. Dr. Zukui Li is responsible for concept formation.

Dedication

No matter how well a society's basic institutions are devised, failures of some actors to live up to the behavior which is expected of them are bound to occur, if only for all kinds of accidental reasons. Each society learns to live with a certain amount of such dysfunctional or mis-behavior ; but lest the misbehavior feed on itself and lead to general decay, society must be able to marshal from within itself forces which will make as many of the faltering actors as possible revert to the behavior required for its proper functioning.

Exit,
Voice,
and
Loyalty
Responses to Decline
in Firms, Organizations,
and States
Albert O. Hirschman

Acknowledgements

- I would like to thank and dedicate this work to my parents.
- I would like to thank Dr. Zukui Li for his supervision and insights throughout.
- I would like to thank Dr. Dimitri J. Papageorgiou for his collaboration, industrial insights and the session chair invitation at the INFORMS 2019 annual meeting.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AOD	Argon oxygen decarburization
CC	Continuous caster
CVaR	Conditional value at risk with $1-\alpha$ tail distribution
DFO	Derivative free optimization
DNN	Deep neural network
DR	Decision rule
EAF	Electric arm furnace
HDR$< 3^l, 2^k, 1^p, 0^s >$	Hybrid DR where the uncertainty in the first l stages is lifted by 3 breakpoints; the next k stages is lifted by 2 breakpoints; the next p stages is lifted by 1 breakpoint, and the last s stages is not lifted
KEDM	Kernel density estimation method
LBFGS	limited-memory Broyden-Fletcher-Goldfarb-Shanno
LDR	Linear DR
LF	Ladle furnace
LSC	Logical sequencing constraint
mpLP	Multiparametric linear programming
mpNLP	Multiparametric non-linear programming
MPP	Multiparametric programming
NAC	Non-anticipativity constraints
NIS	No intermediate storage
NIS (FIS)	No (Finite) intermediate storage
PLDR-i ($\boldsymbol{z}$)	Piecewise LDR with i breakpoints at $\boldsymbol{z}$ in each stage
ReLU	Rectified linear unit
RMSE	Root mean squared error
RO	Robust optimization
SAA	Sample average approximation
SDDP	Stochastic dual dynamic programming
SP	Stochastic programming
USEB (GEB)	Unit-specific (Global) event-based

Chapter 1

Introduction

Optimization combines applied mathematics, numerical analysis and algorithmic implementation. It aims to determine the best set of decisions that increases a specific reward value and accommodates the surrounding circumstances. Optimization begins with building a description for the decision-making process (i.e., a system) via an empirical set of equations (i.e., a model). Then, solution techniques are developed and deployed to generate optimal decisions. The reward is reflected via an objective function value which is either maximized (e.g., profit) or minimized (e.g., cost). The required conditions related to the system or the surroundings are termed as the optimization constraints. In general, optimization is a decision-making problem that optimally allocates limited resources based on a specific objective.

Optimization is inevitable in the planning and scheduling community (Verderame et al. 2010) where an optimizer has to decide on the optimal schedule of jobs performed in a specific plant or operation to meet customers demand at the maximal profit or minimal time/cost. Planning and scheduling have seen applications in several process industries such as oil and gas (Li, Misener & Floudas 2012), chemical and petrochemical (Verderame & Floudas 2009). Optimization is a widely interdisciplinary knowledge. It is extensively present in the operations research community, for example, supply chain management (Peidro et al. 2009). It is present in the healthcare community in the form of inventory management of blood supply (Dillon et al. 2017) and surgeries scheduling in hospitals (Bruni et al. 2015).

Uncertainty is arguably present in most decision-making problems. Traditionally, uncertainty has been ignored in optimization, also known as deterministic optimization, mainly due to computational limitations which did not encourage algorithmic developments. The failure to incorporate uncertainty in the decisions leads to suboptimal decisions. In the worst-case, it may lead to decisions violating regulatory constraints (e.g., carbon emission limit) or contract-based constraints which consequently leads to high penalty fines. In the rest of this introduction, we will introduce an overview of uncertainty characterization in optimization, risk measures needed to quantify an uncertain objective function value and technical preliminaries about modelling and solution framework for linear adaptive optimization and decision rules.

1.1 Uncertainty characterization

Uncertainty takes one of four forms in an optimization (Pistikopoulos 1995): (1) Model-related uncertainty which is present in the estimated empirical parameters such as reaction kinetics constants, mass and heat transfer coefficients, etc. (2) Process-related uncertainty which is present in the unit operations processing time and variations in the measuring sensors, (3) External factors such as variations in market prices, demands, future environmental regulations, weather conditions and others, (4) Discrete uncertainty which is reflected in the availability of resources such as equipment, labor and raw material.

Uncertainty is described in optimization problems in three different representations: (i) Set-based, (ii) Probability distribution and (iii) Fuzzy set (Li & Ierapetritou 2008). The set-based representation is used in the case where no or scarce information about the uncertainty distribution is available. This is due to the lack of long historical data, or the nature of the uncertainty as in the case of estimated empirical parameters. Examples of commonly used set-based uncertainty are hyper-rectangles, polytopes and ellipsoids. An uncertainty probability distribution is constructed from historical data, and it is available in either a discrete or a continuous representation. It provides useful uncertainty correlation properties. Lastly, a fuzzy set is the ordered pairs of a fuzzy number and a membership function. It has applicability in cases where information about the probability distribution of the uncertainty is not available. The membership function is an indicator of constraint violation; as the value of the membership function is higher the degree of satisfaction is higher.

Uncertainty availability may depend on the implemented decisions by the optimizer. This feature gives rise to a new type of classification coined in Jonsbraten et al. (1998): endogenous uncertainty and exogenous uncertainty. On one hand, endogenous uncertainty availability depends on the decision-maker's action; for example, the uncertainty regarding the size of an oil well is revealed once the action of drilling a well is made. On the other hand, exogenous uncertainty availability is independent of any decisions made, for example, the uncertainty in consumers' demand for a certain product is not affected by the ordered amount of the same product by a seller.

1.2 Risk measures

Risk is closely related to uncertainty. For some, it is a measure of the degree of uncertainty, while for others, it is a measure of the objective function value deviation from its deterministic value. In a minimization problem, a positive deviation entails an increased level of risk. Rockafellar (2007) has emphasized the importance of quantifying the risk to carry an optimization under uncertainty problem. This is accomplished by a coherent risk measure which is a function mapping the uncertainty from its full dimension (i.e., all uncertain parameters) to a single quantity in $\mathbb{R}$.

Coherent risk measure is a term first presented in Artzner et al. (1999). A risk measure is deemed coherent if it satisfies the following axioms: (i) convexity, (ii) monotonicity, (iii) closedness, (iv) positive homogeneity, and (v) sub-additivity. In addition to the property that a mapping of a constant is the constant itself. The importance of coherency in optimization stems out from the features it exhibits: preservation of convexity, preservation of certainty and insensitivity to scaling. Rockafellar (2007) described three widely used risk measures in optimization under uncertainty. First, the expectation measure is implemented in stochastic programming (SP) (Birge & Louveaux 2011), where the optimal decisions minimize the expected value of the objective function. SP measures are used when the uncertainty probability distribution is known. Second, the worst-case scenario measure is implemented in robust optimization (RO) (Ben-Tal et al. 2009). As its name indicates, the optimal decisions minimize the worst possible outcome of the objective function. Though it is conservative, it ensures that the realized outcome after observing the uncertainty will not be worse.

In terms of risk, SP and RO are at opposite ends. The first is risk-neutral, whereas, the second is risk-averse and is completely immunized against risk. In general, RO is preferred in problems where infeasibility, such as violating safety and/or contractual requirements, has prohibitive consequences. SP is favorable for long-term planning, scheduling and strategic decisions (Grossmann et al. 2016). The long-time horizon allows the decision maker to benefit from the less conservative solution and to adjust decisions previously made in recourse to unexpected events.

The third risk measure is the conditional value at risk $CVaR_\alpha$. It is a trade-off between SP and RO, equivalently, between risk and cost. The optimal decisions are based on the level of confidence α, where $\alpha \in [0, 1]$. It was first proposed for portfolio optimization in the financial industry (Rockafellar & Uryasev 2000), where a quantified measure of risk is highly appreciated. $CVaR_\alpha$ is the expected value of the objective function tail distribution whose cumulative probability is equal to $1 - \alpha$ (Rockafellar 2007). For example, a $CVaR_{0.95}$ cost value of 1 thousand dollars means that (1) there is a 95% chance that the cost value will be less than 1 thousand dollars and (2) the expected cost of the worst 5% occurrences will be 1 thousand dollars. For the extreme α values at 0 and 1, $CVaR_\alpha$ is equivalent to expected value and worst-case measures, respectively.

1.3 Literature review

1.3.1 Adaptive optimization

Ben-Tal et al. (2004) introduced, in his seminal work, the notion of robust adaptive optimization to address optimization under uncertainty. The framework includes (1) non-adaptive decisions, also known as *"here-and-now"* decisions, which are independent of any future uncertainty and (2) adaptive decisions, also known as *"wait-and-see"* decisions, which are determined after the uncertainty is observed. Non-adaptive decisions are called first-stage decisions as they are made at the outset of the optimization. Similarly, adaptive decisions are called recourse decisions or policies. They manage the future uncertainty consequences to ensure the solution's feasibility and reliability.

Adaptive optimization problems are cast into two types based on the timing of the observed uncertainty: two-stage and multistage adaptive optimization. In the former, uncertainty is revealed once and for all, and the second-stage recourse decisions either profit from the realized uncertainty or mitigate its impact. In the latter, the uncertainty is revealed gradually at each time stage followed by a set of implemented recourse decisions dependent on all the previously revealed uncertain information. In the rest of this section, we will introduce the mathematical formulation of linear adaptive optimization using the stochastic, robust and conditional value at risk paradigms.

Two-stage linear adaptive optimization

The general two-stage continuous linear adaptive optimization is given in equation (1.1) where $\mathcal{X}_1 := \big\{\boldsymbol{x}_1 |\ \boldsymbol{A}_1\boldsymbol{x}_1 \geq \boldsymbol{b}_1\big\}$ is a polytopic set representing the feasible region for the first-stage deterministic decisions $\boldsymbol{x}_1$ and $\boldsymbol{c}_1^\top$ is a cost vector.

$$\min_{\boldsymbol{x}_1 \in \mathcal{X}_1} \quad \boldsymbol{c}^\top \boldsymbol{x}_1 + \rho\big(f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2)\big) \tag{1.1}$$

The primitive uncertainty vector $\boldsymbol{\xi}_2$ is observed in the second and only time stage. $\rho(\cdot)$ is a risk measure and $f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2)$ is the optimal value function of the second-stage optimization problem defined in equation (1.2) where $\mathbf{c_2}(\boldsymbol{\xi}_2),\ \boldsymbol{A}_{2i}(\boldsymbol{\xi}_2)$ for $i = \{1, 2\}$ and $\boldsymbol{b}_2(\boldsymbol{\xi}_2)$ represent the cost vector, recourse matrices and right-hand-side vector, respectively. Ξ is the uncertainty set which includes all possible outcomes of $\boldsymbol{\xi}_2$ and $\boldsymbol{x}_2(\boldsymbol{\xi}_2)$ is the adaptive policy implemented after the uncertainty is observed.

$$\begin{aligned} f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2) = \min_{\boldsymbol{x}_2} \quad & \boldsymbol{c}_2(\boldsymbol{\xi}_2)^\top \boldsymbol{x}_2 \\ \text{s.t.} \quad & \boldsymbol{A}_{21}(\boldsymbol{\xi}_2)\boldsymbol{x}_1 + \boldsymbol{A}_{22}(\boldsymbol{\xi}_2)\boldsymbol{x}_2(\boldsymbol{\xi}_2) \geq \boldsymbol{h}_2(\boldsymbol{\xi}_2); \quad \forall \boldsymbol{\xi}_2 \in \Xi \end{aligned} \tag{1.2}$$

Table 1.1 depicts the stochastic, robust and conditional value at risk two-stage adaptive continuous linear formulations. The only difference is in the risk measure ρ definition. Rockafellar & Uryasev (2000) applies a minimization rule to $\mathbb{CVaR}_\alpha\big[f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2)\big]$, so it can be implementable in an optimization problem where $[y]^+ = \max\{y, 0\}$ and θ is the starting point of the cost tail distribution. The min operator over θ can be dropped (Krokhmal et al. 2002).

Table 1.1: Two-stage continuous linear adaptive optimization with three risk measures.

Paradigm	Formulation
Stochastic programming	$\min\limits_{\boldsymbol{x}_1 \in \mathcal{X}_1} \left\{ \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \mathbb{E}\big[f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2)\big] \right\}$
Robust optimization	$\min\limits_{\boldsymbol{x}_1 \in \mathcal{X}_1} \left\{ \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \max\limits_{\boldsymbol{\xi}_2 \in \Xi} f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2) \right\}$
Conditional value at risk	$\min\limits_{\boldsymbol{x}_1 \in \mathcal{X}_1} \left\{ \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \min\limits_{\theta} \left\{ \theta + \frac{1}{1-\alpha} \mathbb{E}\big[(f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_2) - \theta)^+\big] \right\} \right\}$

Multistage linear adaptive optimization

Multistage adaptive optimization is a natural extension of the two-stage setting and it is cast as sequential decision-making under uncertainty. Instead of taking all decisions at once, without any previous knowledge of the progressively revealed uncertainty, the decision-maker first implements a set of deterministic decisions $\boldsymbol{x}_1$ which is known to be independent of any future uncertainty. Afterward, the decision-maker waits for the gradual unfolding of the uncertainty to implement optimal recourse decisions. In this sequence, the first set of uncertain parameters $\boldsymbol{\xi}_2$ is revealed in the second time stage followed by a set of recourse decisions which are functions of the realized uncertain parameters $\boldsymbol{x}_2 \equiv \boldsymbol{x}_2(\boldsymbol{\xi}_2)$. After which, the sequence of alternating observations and recourse decisions unfolds over T stages, wherein each stage $t = \{2, \ldots, T\}$, the decision-maker observes a set of uncertain parameters $\boldsymbol{\xi}_t$ and selects a set of recourse decisions $\boldsymbol{x}_t(\boldsymbol{\xi}_{[t]})$.

Implement $(\boldsymbol{x}_1)$ → observe $(\boldsymbol{\xi}_2)$ → Implement $(\boldsymbol{x}_2)$ → ⋯ → observe $(\boldsymbol{\xi}_T)$ → Implement $(\boldsymbol{x}_T)$

The implementable policies $\boldsymbol{x}_t(\boldsymbol{\xi}_{[t]})$ for all t must satisfy the non-anticipativity property. They area functions of only the history of uncertainty realizations, i.e., $\boldsymbol{\xi}_{[t]} = (\boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t)$, but not on any future realizations $(\boldsymbol{\xi}_{t+1}, \ldots, \boldsymbol{\xi}_T)$. For notation purposes, we denote $\boldsymbol{\xi}_{[T]}$ as $\boldsymbol{\xi}$. According to Shapiro et al. (2009), a general formulation of a multistage linear adaptive optimization problem is given in equation (1.3) where $f_t(\boldsymbol{x}_{t-1}, \boldsymbol{\xi}_{t-1})$ is the objective function value generated by the implemented policy $\boldsymbol{x}_t(\boldsymbol{\xi}_{[t]})$ at all $t = \{2, \cdots, T\}$.

$$\min_{\boldsymbol{x}_1 \in \mathcal{X}_1} \quad \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \rho\bigg(f_2(\boldsymbol{x}_1, \boldsymbol{\xi}_{[2]}) + \rho\Big(f_3(\boldsymbol{x}_2, \boldsymbol{\xi}_{[3]}) + \cdots + \rho\big(f_T(\mathrm{x}_{T-1}, \boldsymbol{\xi}_{[T]}) \big) \Big) \bigg) \tag{1.3}$$

An equivalent formulation for the robust optimization and stochastic paradigms is given in equation (1.4) where the risk measure is $\rho(.) = \max_{\boldsymbol{\xi} \in \Xi}(.)$ and $\rho(.) = \mathbb{E}(.)$, respectively. The equivalence does not hold for the conditional value at risk measure because of time inconsistency issue.

$$\min_{\boldsymbol{x}_t(\cdot)} \quad \boldsymbol{c}_1^\top \boldsymbol{x}_1 + \rho \left[\sum_{t=2}^{T} \boldsymbol{c}_t(\boldsymbol{\xi}_{[t]})^\top \boldsymbol{x}_t(\boldsymbol{\xi}_{[t]}) \right] \tag{1.4a}$$

$$\text{s.t.} \quad \boldsymbol{A}_1 \boldsymbol{x}_1 \geq \boldsymbol{b}_1 \tag{1.4b}$$

$$\sum_{s=2}^{t} \boldsymbol{A}_{ts}(\boldsymbol{\xi}_{[s]}) \boldsymbol{x}_s(\boldsymbol{\xi}_{[s]}) \geq \boldsymbol{b}_t(\boldsymbol{\xi}_{[t]}) \qquad \forall \boldsymbol{\xi} \in \Xi, \ t = \{2, \ldots, T\} \tag{1.4c}$$

Common assumptions in the literature include constant recourse cost coefficients (i.e., $\boldsymbol{c}_t(\boldsymbol{\xi}_{[t]}) \equiv \boldsymbol{c}_t$ for all t), constant recourse matrices (i.e., $\boldsymbol{A}_t(\boldsymbol{\xi}_{[t]}) \equiv \boldsymbol{A}_t$ for all t) and a set-based uncertainty representation of Ξ.

1.3.2 Solution approximation methods

The main computational obstacle in solving adaptive optimization is the so-called intractability. It is induced by the presence of semi-infinite uncertain constraints which are constraints that must be satisfied over a continuous uncertainty set (i.e., not discrete). By definition, a compact continuous uncertainty set consists of an infinite number of points. Enforcing the feasibility of a constraint over an infinite number of points is impractical, if not impossible. Adaptive optimization in their general form in (1.4) can not be solved. Approximation methods have been developed to circumvent computational intractability. Two commonly used methods are sample average approximation and decision rule-based approximation.

Sample average approximation

Sample average approximation (SAA) is used in stochastic programming. The concept is based on the approximation of the uncertainty $\boldsymbol{\xi}$ via a set of generated scenarios $\hat{\boldsymbol{\xi}}_s$ for all s. Each scenario is assigned a discrete probability value P_s that is used to approximate the expected objective function value $\mathbb{E}[\boldsymbol{\xi}] = \sum_{s \in \mathcal{S}} P_s \boldsymbol{\xi}_s$. The semi-infinite uncertain constraints are reformulated as deterministic constraints enforced at the generated set of scenarios only. For multistage adaptive optimization, the scenario generation process takes the form of a tree whose root node represents the first-stage. An element of the SAA method is the sampling technique. Niederreiter (1992) discusses the rate of convergence of the expected value estimates using Monte-Carlo, Quasi Monte-Carlo sampling techniques. Linderoth et al. (2006) showed, that for specific instances, an excellent solution can be obtained with a small sampled set when compared to the size of the uncertainty space. The computational complexity of the two-stage and multistage SP adaptive optimization using SAA is found (Shapiro & Nemirovski 2005).

The main SAA method limitation is the exponential increase of the number of scenarios, hence the dimension of the adaptive optimization problem with the number of stages and uncertainty dimension. This is known in the literature as the "curse of dimensionality" which leads to prohibitive computational burden. As an illustration, consider that a single stage-independent uncertain parameter may take a low, a medium or a high value in each week (i.e., 3 values). For a twelve weeks planning period, we will have $3^{12} = 531441$ possible scenarios. Equivalently, we have to replace each semi-infinite uncertain constraint with 531441 deterministic constraints. Scenario aggregation heuristic is used to mitigate the exponential increase in the number of scenarios without a significant loss in approximation quality. Scenario aggregation is based on the fact that some nodes in the late stages of a scenario tree have very low probability and that the time value of money is not influenced by the decisions done in the far future. These types of heuristics are generally based on the problem's specific structure and they can not be generalized.

Decision rule-based approximation

Decision rule-based approximation does not suffer from the same curse of dimensionality. First introduced by Garstka & Wets (1974), if not earlier, the significance of the approach was not fully realized until 2004 when Ben-Tal et al. (2004) demonstrated that the linear decision rule (LDR) approximation for robust linear adaptive optimization can be solved in polynomial time. In their framework, Ben-Tal et al. (2004) defined uncertainty-independent decision variables as static, while uncertainty-dependent adaptive decisions are defined as linear functions, or rules, of the uncertain parameters. The main advantage of linear decision rules is that, under certain convexity assumptions of the underlying uncertainty set, they give rise to tractable robust counterparts that are efficiently solvable by today's ever-improving optimization engines. The robust counterparts are derived using the strong duality theorem for convex optimization. Consider a semi-infinite constraint in equation (1.5) where we assume that the uncertainty is only present at the right-hand-side term.

$$\boldsymbol{x}_1 + \boldsymbol{x}_2(\boldsymbol{\xi}_2) \geq h(\boldsymbol{\xi}_2), \quad \boldsymbol{\xi}_2 \in \Xi \tag{1.5}$$

The LDR approximating $\boldsymbol{x}_2(\boldsymbol{\xi}_2)$ is nothing more than a function mapping realized uncertainty $\boldsymbol{\xi}_2$ into an implementable action as in equation (1.6) where $\boldsymbol{x}_2^0$ and $\boldsymbol{X}_2^1$ are the intercepts and slopes, respectively.

$$\boldsymbol{x}_2(\boldsymbol{\xi}_2) = \boldsymbol{x}_2^0 + \boldsymbol{X}_2^1\boldsymbol{\xi}_2, \quad \boldsymbol{\xi}_2 \in \Xi_2 \tag{1.6}$$

Another common format of an LDR in the literature is $\boldsymbol{x}_2(\boldsymbol{\xi}_2) = \boldsymbol{X}_2^1(1; \boldsymbol{\xi}_2)$ where the intercept and the slope values are incorporated in the same entity. We let $h(\boldsymbol{\xi}_2) = h_0 + \mathbf{h}_1^\top \boldsymbol{\xi}_2$ and we assume a polytopic uncertainty set $\Xi := \{\boldsymbol{\xi}_2 \in \mathbb{R}^n | \ \boldsymbol{A}^{\text{th}}\boldsymbol{\xi}_2 \geq \boldsymbol{b}^{\text{th}}\}$. Equation (1.5) is equivalently rewritten in equation (1.7).

$$\boldsymbol{x}_1 + \boldsymbol{x}_2^0 + \left\{ \begin{array}{ll} \min\limits_{\boldsymbol{\xi}_2} & (\boldsymbol{X}_2^1 - \mathbf{h}_1^\top)\boldsymbol{\xi}_2 \\ \text{s.t.} & \boldsymbol{A}^{\text{th}}\boldsymbol{\xi}_2 \geq \boldsymbol{b}^{\text{th}} \end{array} \right\} \geq 0 \tag{1.7}$$

Stated differently, if we plug the solution obtained from (1.7) (i.e., $\boldsymbol{\xi}_2$, $\boldsymbol{x}_2^0$ and $\boldsymbol{X}_2^1$) in equation

(1.5), it will be feasible for all $\boldsymbol{\xi}_2$. The equivalent dual of the inner minimization problem is given in equation (1.8) where $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the free dual variable.

$$\boldsymbol{x}_1 + \boldsymbol{x}_2^0 + \left\{ \begin{array}{ll} \max\limits_{\boldsymbol{\lambda}} & \boldsymbol{b}^{\mathrm{th}\top}\boldsymbol{\lambda} \\ \text{s.t.} & \boldsymbol{A}^{\mathrm{th}\top}\boldsymbol{\lambda} \geq \boldsymbol{X}_2^{1\top} - \mathbf{h}_1 \end{array} \right\} \geq 0 \tag{1.8}$$

The affine tractable robust counterpart is obtained after dropping the max operator which does not impact the feasibility of the constraint. The overall affine tractable counterpart is constructed by performing the same procedure for every semi-infinite uncertain constraint in the model.

1.3.3 Decision rules: Approximation quality and computational cost

Though LDRs are optimal for some multistage robust optimization instances (Bertsimas et al. 2010), its attractive modeling feature comes generally at the expense of an inferior approximation/solution quality (Kuhn et al. 2011). Variants of nonlinear decision rules which improve the solution quality of the adaptive decisions have been proposed in the literature. Better approximation quality comes at the cost of increased computational complexity. For example, quadratic decision rules have been introduced in the context of robust optimization (Ben-Tal & Den Hertog 2011). The obtained tractable counterpart, under an ellipsoidal uncertainty set, is a second-order cone optimization problem. As for polynomial decision rules in multistage robust dynamic problems (Bertsimas et al. 2011), the tractable counterpart, under an intersection of convex uncertainty sets, is a semidefinite optimization problem. Polynomial decision rules were later refined in the context of stochastic programming (Bampou & Kuhn 2011).

A specific class of nonlinear decision rules is the piecewise linear decision rule (PLDR). It improves the flexibility of an LDR by having multiple slopes (i.e., decision variables) while inheriting its modeling features due to its linear nature. Chen & Zhang (2009) proposed an extended linear decision rule using an extended uncertainty set defined via the positive and negative perturbations of the original uncertainty set. The decision rule is equivalent to a PLDR with two linear pieces. Later, Georghiou et al. (2015) introduced a lifting methodology which generalizes the construct of piecewise linear decision rules.

The increase in a decision rule complexity (i.e., approximation quality) does not only increase the incurred computational cost. It also limits the available uncertainty set types which satisfy the strong duality theorem required for the derivation of the robust counterparts. A more attractive strategy is to maintain the linear form of a decision rule and improve the approximation quality by increasing the number and complexity of its basis. This opportunity is exploited in Chapters 1, 3 and 5 of the thesis via non-decreasing hybrid decision rules, deep lifted decision rules and ReLU-based decision rules, respectively.

1.3.4 Decision rules in process scheduling

Proactive and reactive scheduling are two common scheduling under uncertainty methods. The former considers the worst-case uncertainty to avoid future changes to the initial schedule in the event of disruption while the latter reacts to each unexpected disruption by adjusting the initial schedule or generating a new schedule from scratch. Though not very common in the literature yet, a hybrid scheduling approach may be attractive (Iglesias Escudero et al. 2019). It immunizes the schedule against part of the uncertainty set (i.e., less conservative solution) and adjusts the current schedule whenever the realized uncertainty falls outside the set (i.e., fewer changes to the schedule) (Chaari et al. 2014).

Shaik et al. (2006) classified continuous time models used in process scheduling problems into three main classes: (i) unit-specific event-based (USEB), (ii) global event-based and (GEB) (iii) slot-based formulations. In the first two formulations, the model optimizes the start and finish timing decisions of a task, whereas in the last formulation the duration of a time slot is the decision to be optimized. A global event-based formulation adopts a single time grid for all units. This provides some information about the temporal correlation of tasks across different units. On the other hand, a unit-specific event-based formulation implements a unit-specific time grid. This feature has been shown to offer superior performance in the deterministic setting and is used by Li, Xiao, Tang & Floudas (2012) to model the SCC deterministic scheduling problem.

Endogenous uncertainty in process scheduling has been receiving increased research interests. Poss (2014) introduced a budget uncertainty set defined by the problem's decision variables. The new set is shown to reduce the price-of-robustness when compared to the classical budget uncertainty set provided by Bertsimas & Sim (2004) at the expense of a small increase in computational cost. Lappas & Gounaris (2016) introduced decision-dependent uncertainty sets for global event-based affine adaptive robust optimization problems. Lappas & Gounaris (2018) introduced a general formulation for the DDUS and illustrated the improvement in performance in comparison to the static uncertainty set.

We addressed in Chapter 3 two gaps in the process scheduling under uncertainty using decision rules literature. The first is related to the implementation of adaptive optimization framework to USEB continuous-time models and the second is the investigation of the less common hybrid scheduling method as a competitive candidate for the trade-off between solution conservativeness and frequent rescheduling events.

1.3.5 Decision rules in multiparametric programming

Multiparametric linear programming (mpLP) was first developed by Gal & Nedoma (1972). The solution of mpLPs is obtained by defining a set of critical regions or partitions that cover the uncertain parameters space. In each critical region, optimal decisions and cost value are defined as affine functions of the uncertain parameters which is similar to the concept of approximating adaptive decisions as linear functions of the uncertain parameters. The mpLP solution follows a bottom-up strategy in constructing the critical regions that cover the uncertainty space.

A wide variety of decision-making problems in process system engineering include nonlinearities in their model. Hence, research works in developing multiparametric nonlinear programming (mpNLP) solution methods soon followed. On one hand, the solution technique for convex mpNLP is based on constructing a linear (Acevedo & Salgueiro 2003, Dua & Pistikopoulos 1999) or quadratic (Domínguez & Pistikopoulos 2013) approximations coupled with mpLP solution techniques. In case the maximum error does not meet a prescribed tolerance, the uncertainty space is partitioned and the procedure reiterated. On the other hand, spatial branch and bound (Dua et al. 2004) and decomposition (Fotiou, Rostalski, Parrilo & Morari 2006,?, Fotiou, Rostalski, Sturmfels & Morari 2006) Fotiou et al. (2005) solution methods are developed for non-convex mpNLP.

In general, for the case where a Multiparametric programming problem (MPP) has a large number of uncertain parameters and small number of decision variables, an active set-based algorithm construct the critical regions regions by finding the combinations of active constraints. These types of algorithms do not need to observe the parameter space (e.g., Gupta et al. (2011)). For the case where an MPP has a small number of uncertain parameters and a large number of decision variables, a geometric algorithm finds an initial critical region and then explore the remaining uncertainty space by flipping the facet of each polyhedron. These types of algorithms only observe the parameter space (e.g., Bemporad et al. (2002)). However, for the case where an MPP has a large number of uncertain parameters and a large number of decision variables, the solution becomes difficult and limited by scalability. The solution of these types of problems is a present gap in the state-of-the-art multiprogramming literature which is addressed in Chapter 6 of the thesis using a novel decision rule.

1.4 Thesis contributions

The thesis thrust lies in three contributions: (1) devising novel decision rules with a competitive approximation quality and computational cost trade-off, (2) implementing decision rules for unit-specific event-based continuous scheduling models (e.g., in steelmaking scheduling) and (3) addressing multiparametric linear programming problems using decision rule-based solution techniques.

Chapters 2 and 4 contributes to the first contribution, where Chapter 3 fulfills the second contribution and Chapter 6 addresses the third contribution. Chapter 5 offers polyhedral-norm uncertain based construction technique for uncertain data that are normally present in a historical or distribution form (i.e., relax the compact polytope uncertainty set assumption). A detailed description of each chapter contribution is presented in a specific section in the thereof. Table 1.2 presents and overview of the main points, advances and features in the five coming chapters.

Table 1.2: An overview of the thesis five main chapters.

	Contributions	Frameworks	Decision rules	Paradigms	Models	Simulators	Applications
Chapter 2	Non-decreasing HDRs Highlighting simulator's value	Multistage	LDR/PLDR	SP	LP/MIP	Rolling-horizon/ Sample-based	Newsvendor/ Transportation
Chapter 3	LDRs for USEB time models Exploration of hybrid scheduling method.	Multistage	LDR	RO	MIP	Sample-based	SCC scheduling
Chapter 4	Deep lifted decision rules inspired from concepts in the machine learning community	Two-stage	Deep lifted DR	SP	LP/MIP	Not required	Transportation/ airlift operations scheduling
Chapter 5	Polyhedral-norm uncertainty set construction method. Asymmetric set construction using distribution information	Static	Not applicable	RO	LP	Not applicable	Numerical example and a reactor design problem
Chapter 6	mpLP approximation method using ReLU-based decision rule	Two-stage	ReLU-based DR	RO	LP	Not applicable	Numerical examples

Chapter 2

Hybrid Strategies using Linear and Piecewise-Linear Decision Rules

2.1 Introduction

In recent years, multistage adaptive optimization has received growing interest as a tool to address parameter uncertainty in decision making problems. In particular, multistage adaptive optimization can be cast as a sequential decision making problem under uncertainty. Instead of taking all decisions at once, without any previous knowledge of the progressively revealed uncertainty, the decision maker first implements a set of static decisions $\mathbf{x}_1$ which is known to be independent of future uncertainty. Afterwards, the decision maker waits for the gradual unfolding of uncertainty to implement optimal recourse decisions. In this sequence, the first set of uncertain parameters $\boldsymbol{\xi}_2$ is revealed[1], followed by a set of recourse decisions $\mathbf{x}_2$. Practically, the recourse decisions are functions of the realized uncertain parameters $\mathbf{x}_2 \equiv \mathbf{x}_2(\boldsymbol{\xi}_2)$. After which, the sequence of alternating observations and recourse decisions unfolds over T stages, where in each stage $t \in \mathcal{T}_{-1} = \{2, \ldots, T\}$, the decision maker observes a set of uncertain parameters $\boldsymbol{\xi}_t$ and selects a set of recourse decisions $\mathbf{x}_t(\boldsymbol{\xi}_{[t]})$, which depends on the whole history of past observations $\boldsymbol{\xi}_{[t]} = (\boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t)$, but not on any future observations $\boldsymbol{\xi}_{t+1}, \ldots, \boldsymbol{\xi}_T$. According to Shapiro et al. (2009), a general formulation of a multistage adaptive optimization problem is given as

$$\min_{\mathbf{x}_t(\cdot)} \quad \mathbf{c}_1^\top \mathbf{x}_1 + \rho \left[\sum_{t=2}^{T} \mathbf{c}_t(\boldsymbol{\xi}_{[t]})^\top \mathbf{x}_t(\boldsymbol{\xi}_{[t]}) \right] \tag{2.1a}$$

$$\text{s.t.} \quad \mathbf{A}_1 \mathbf{x}_1 \geq \mathbf{b}_1 \tag{2.1b}$$

$$\sum_{s=2}^{t} \mathbf{A}_s(\boldsymbol{\xi}_{[s]}) \mathbf{x}_s(\boldsymbol{\xi}_{[s]}) \geq \mathbf{b}_t(\boldsymbol{\xi}_{[t]}) \qquad \forall \boldsymbol{\xi} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.1c}$$

[1] The time subscript refers to the stage by which the information of a given variable is available. The first observed uncertainty $\boldsymbol{\xi}_2$ is first available in stage 2.

where ρ is a coherent risk measure (Rockafellar 2007), $\mathbf{A}_1$, $\mathbf{b}_1$, $\mathbf{c}_1$ are static parameters and $\mathbf{x}_1$ is the first-stage static (or *here-and-now*) decision. Similarly, $\mathbf{A}_t(\boldsymbol{\xi}_{[t]})$, $\mathbf{b}_t(\boldsymbol{\xi}_{[t]})$, $\mathbf{c}_t(\boldsymbol{\xi}_{[t]})$ are uncertain recourse matrices, right-hand-side vectors and cost coefficients, respectively and $\mathbf{x}_t(\boldsymbol{\xi}_{[t]})$ is the recourse/adaptive (or *wait-and-see*) decision. The possible realizations of $\boldsymbol{\xi}$ (i.e., $\boldsymbol{\xi}_{[T]}$) are defined by the underlying uncertainty set Ξ.

There are numerous paradigms to tackle sequential decision making problems. Arguably the three most popular within the operations research community are stochastic programming (SP) (Shapiro et al. 2009), adaptive robust optimization (Ben-Tal et al. 2009), and stochastic dynamic programming (Powell 2011). The first two grew out of a mathematical programming tradition, while the third has roots in control theory and reinforcement learning. While all paradigms aim to compute optimal decision policies, they do so based on assumptions about the underlying uncertainty required to define the risk measure. SP typically requires some underlying distributional assumptions about the uncertain parameters to determine the optimal expected objective function value, while robust optimization only assumes that the uncertain parameters belong to a known uncertainty set to optimize the objective function value under the worst-case scenario. Meanwhile, stochastic dynamic programming commonly relies on the solution of Bellman's equations, or approximations thereof, to generate policies. Today, one could argue that the lines between the three domains are becoming even more blurred as a cross-pollination of ideas continues to flourish.

Stochastic dual dynamic programming (SDDP) is yet another competing solution approach to tackle stochastic sequential decision problems, although Powell (2014) argues that SDDP falls in the class of approximate dynamic programming using a "look-ahead approximation" policy. Since Pereira & Pinto (1991) introduced it, SDDP has been used heavily for hydroelectric generation planning and other applications. It has since been extended to handle binary state variables (Zou et al. 2019). Perhaps the most well-known deficiencies of SDDP are its requirement for stagewise independence and its convergence properties. The stagewise independence assumption requires that the random vector $\boldsymbol{\xi}_{t+1}$ is independent of $\boldsymbol{\xi}_{[t]}$ for $t \in \{2, \ldots, T-1\}$, i.e., new information available in stage $t+1$ does not depend on the history of the process. This deficiency is often easy to overcome by augmenting the state space, but this comes at the cost of managing a cost-to-go function in a potentially much higher dimension. As for convergence properties, as described in Shapiro (2011), when applied to a T-stage stochastic linear problem with an infinite number of scenarios, an SDDP method using sample average approximation requires roughly $(T-1)$ linear programs to be solved in the forward pass and $\sum_{t=2}^{T} N_t$ in the backward pass, where N_t denotes the number of scenarios sampled in stage t. Assuming these forward and backward steps are run $N^{\max}$ times and that M repetitions are needed to achieve statistically significant results, Shapiro argues that a sample-based SDDP approach requires the solution of approximately $N^{\max}M\left(T + \sum_{t=2}^{T} N_t\right)$ linear programs. Practically speaking, this suggests that one can likely obtain meaningful results by solving a number of linear programs that is linear in the number of stages, similar to the justification for using decision rules. In the worst-case, however, Shapiro points out that "for two-stage programming the SDDP algorithm becomes the classical Kelley's cutting plane algorithm," whose worst-case performance can be exponential in the problem dimension. In short, while SDDP is a viable method for solving this class of problems, we do not compare our methods against it. Instead, we restrict our comparisons to the already widely used class of linear decision rules, which by most accounts serves as the state-of-the-art in practice for decision rule-based methods.

2.1.1 Literature review

Of the numerous algorithmic advances devised to construct optimal policies, two of the most prominent are scenario- and decision rule (DR)-based methods. The former is typically used in the context of SP where a set of discrete scenarios representing the uncertainty set is used to compute the expected objective function value. Scenarios are often represented via a scenario tree whose size increases exponentially with the size of the decision making sequence. This limits the use of scenario-based SP methods for high dimension problems due to the prohibitive increase in the computational overhead; a limitation that is also known as the curse of dimensionality.

On the other hand, DR-based methods do not suffer from the same curse of dimensionality. First introduced by Garstka & Wets (1974), if not earlier, the significance of the approach was not fully realized until 2004 when Ben-Tal et al. (2004) demonstrated that the linear decision rule (LDR) approximation for robust and stochastic optimization can be solved in polynomial time. In their framework, Ben-Tal et al. (2004) defined uncertainty-independent decision variables as static, while uncertainty-dependent adaptive decisions are defined as linear functions, or rules, of the uncertain parameters. The main advantage of LDRs is that, under certain convexity assumptions of the underlying uncertainty set, they give rise to tractable robust counterparts that are efficiently solvable by today's ever-improving optimization engines. Ben-Tal et al. (2005) applied the robust LDR method to the retailer supplier flexible commitment problem with uncertain demand. However, this modelling feature may come at the expense of deteriorated solution quality. Kuhn et al. (2011) proposed a method to estimate the loss in optimality by an LDR approximation using the solution gap between the primal and the dual version of the same problem. Bodur & Luedtke (2018) proposed a new approximate method for multistage linear SP problems which improves the solution quality of an LDR policy by using a sample average approximation approach and an approximate two-stage linear SP problem. For an example of robust LDR methods in process scheduling problems, see Lappas & Gounaris (2016).

Variants of nonlinear decision rules which improve the solution quality of the adaptive decisions have been proposed in the literature. Chen et al. (2008) introduced deflected and segregated decision rules used for stochastic programming problems with semi-complete and general recourse. See & Sim (2010) implemented a truncated LDR for their inventory problem which was proven to do better than an LDR policy. Both decision rules are generalized in Goh & Sim (2010) as bi-deflected LDRs. Ben-Tal & Den Hertog (2011) restricted the adaptive decisions into quadratic decision rules in the context of robust optimization. The obtained tractable counterpart, under an ellipsoidal uncertainty set, is a second-order cone programming problem. Further, Bertsimas et al. (2011) proposed polynomial decision rules in multistage robust dynamic problems. The increase in solution quality comes at the cost of increased computational complexity; the tractable counterpart, under an intersection of convex uncertainty sets, is a semidefinite programming problem. Polynomial decision rules were later refined by Bampou & Kuhn (2011) in the context of stochastic programming problems.

A specific class of nonlinear DRs is the piecewise linear decision rule (PLDR). It improves the flexibility of an LDR by having multiple slopes (i.e., decision variables) while inheriting its modelling features due to its linear nature. Chen & Zhang (2009) proposed an extended linear decision rule using an extended uncertainty set defined via the positive and negative perturbations of the original uncertainty set. The decision rule is equivalent to a PLDR with two linear pieces. Later, Georghiou et al. (2015) introduced the concept of generalized decision rules via liftings. The key property is the one-to-one correspondence between the LDRs in the lifted problem and a family of nonlinear decision rules in the original problem. Hence, the modelling features of LDRs in the lifted problem are exploited, while still exhibiting the flexibility of nonlinear decision rules in the original problem. A PLDR is an example of lifted decision rules; it is constructed for any number of breakpoints or linear pieces as in Georghiou et al. (2015). Ben-Tal et al. (2018) proposed a novel piecewise linear decision rule for linear dynamic robust problems. The framework is based on approximating the uncertainty set with a simplex where constructing the PLDR, with an exponential number of pieces, can be performed efficiently. The discussed PLDRs apply for real valued decisions. Recently, Guigues et al. (2020) introduced a constant depth decision rule for discrete and polytopic uncertainties applied to a hydro-thermal production planning problem. For binary adaptive decisions, piecewise-constant decision rules are used where the main concept is to assign a value of 0 or 1 to each partition in the uncertainty set. Several methodologies have been proposed in this regard in Bertsimas & Georghiou (2015), Hanasusanto et al. (2015) and Vayanos et al. (2011). Daryalal et al. (2020) derived integer-based primal policies from the proposed lagrangian dual decision rules for multistage mixed integer programming problems. For a recent survey on decision rules, see Yanıkoğlu et al. (2019).

Since our focus is on hybridizing LDR with the liftings in PLDRs, it is worth calling attention to several noteworthy applications where PLDRs have been employed to demonstrate their growing popularity. In Munoz-Alvarez et al. (2014), PLDRs were used to approximate recourse decisions when dispatching electric power given random power supply and consumption. Gauvin et al. (2017) evaluated various LDRs and PLDRs for managing reservoirs in Canada for electric power generation. Braathen & Eriksrud (2013) compared scenario-, LDR- and PLDR-based approaches in optimizing the hydropower bidding process for Nordic producers. Pan et al. (2015) extended the application of LDRs and segregated DRs to nonlinear objective functions for optimal reservoir operation. Beuchat et al. (2016) integrated power dispatch and reserve models and illustrated that the least flexible PLDR (specifically, two pieces for each uncertain parameter) provides substantial performance improvement with respect to an LDR. Further, in the control community, Jin & Xu (2018) implemented segregated LDRs to solve multistage stochastic control problem with linear dynamics and quadratic cost. Zhang et al. (2015) considered linear chance constrained predictive control problems subject to additive disturbance. For their problem, both the randomization approach (i.e., scenario-based approach) and the PLDR-based method were found to be computationally expensive. As an alternative, they implemented a combination of the two approaches which allowed them to exploit the flexibility of a PLDR.

2.1.2 Chapter contributions

The choice of which decision rule to implement is governed by two competing objectives: high solution quality and low computational cost. The latter objective favours LDRs while the former is typically achieved by nonlinear DRs. To the best of our knowledge, decision rule-based methods found in the literature implement homogeneous decision rules, i.e., the same form of linear or nonlinear decision rule is applied to every adaptive decision variable that is chosen to be represented with a decision rule. This observation motivates our work to investigate hybrid decision rules combining the salient features of both types. We limit our study of nonlinear decision rules to PLDRs only. For example, lifting the uncertainty for near-term stages (e.g., stages $2-4$) and keeping the original uncertainty for the remaining stages (e.g., stages $5-T$) gives increased flexibility in the types of permissible decisions/actions in the immediate future while giving limited recourse actions in subsequent long-term decisions. It is somewhat analogous to an approach used in scenario-based stochastic programming in which a scenario tree containing many branches per node (i.e., higher uncertainty resolution) in early stages and relatively few branches per node (i.e., lower uncertainty resolution) in later stages is shown to be attractive (Bakkehaug et al. (2014), Arslan & Papageorgiou (2017)).

The contributions of this chapter are:

1. Similar to what has been illustrated in scenario-based stochastic programming methods, we empirically show that "it is more important to model the uncertainty of the near future with more details than it is for the later stages" (Bakkehaug et al. 2014, p.73). We demonstrate this result for the first time using decision rules where having higher uncertainty resolution or more linear pieces in early stages improves the flexibility of a policy more than having it in late stages. This observation is shown via an empirical sensitivity analysis for two computational settings and two different number of stages. It motivates the design of PLDRs with axial segmentation using a hybrid combination of liftings (i.e., hybrid decision rules). It is also important to credit Georghiou et al. (2015) with conceiving the idea of exploiting the modularity of decision rules in stochastic programming. At the same time, these authors did not pursue this modular design with detailed computational experiments. Our work attempts to fill this gap.

2. We perform a computational study, using a moving horizon simulator, to evaluate the quality of hybrid decision rules (HDRs) with non-increasing (i.e., higher uncertainty res-

olution or more linear pieces in early stages) and non-decreasing (i.e., higher uncertainty resolution or more linear pieces in late stages) lifting strategies. We demonstrate empirically that (1) non-increasing HDRs are more flexible than non-decreasing HDRs in terms of solution quality, and (2) the computational benefits of non-increasing HDRs in terms of the trade-off between solution quality and computational cost is best manifested using a well-designed lifting strategy. We also show that a poorly designed non-increasing HDR loses these computational benefits.

3. We show a rather counter-intuitive result in which less complex decision rules outperform their more complex counterparts, even when the same look-ahead horizon lengths and uncertainty sets are assumed when the policies are constructed. In a first experiment, we demonstrate that this behavior can occur because a look-ahead model may generate policies in which *mutually exclusive* state variables are simultaneously positive for a set of scenarios (e.g., a policy in which a supplier simultaneously has positive inventory and a positive backlog for randomly realized demand). In a second experiment, we highlight that a less complex non-increasing HDR can be superior to a more complex PLDR, which contains the HDR as a special case, a behavior attributed to the overfit of the look-ahead policies at the end of the simulator's horizon. The two case studies reveal that it is crucial to evaluate policies within a simulation environment to obtain an impartial assessment of how various policies perform in practice (Powell 2014).

The rest of the chapter is structured as follows. In section 2, we illustrate, by way of example, the implementation of LDRs and PLDRs for a multistage stochastic newsvendor problem. The observations made about the look-ahead approximations pave the way for section 3 where we demonstrate that a less complex decision rule (i.e., LDR) is superior to a more complex decision rule (i.e., PLDR with a single breakpoint) within a simulator. In section 4, we empirically illustrate that having higher uncertainty resolution in early stages is more important than in late stages. We also demonstrate the computational benefits of non-increasing hybrid decision rules through a set of computational experiments. In section 5, we offer conclusions and future research directions.

2.2 Linear vs. piecewise-linear decision rules

Throughout our study, we assume that the cost coefficients and recourse matrices are fixed and the risk measure in equation (2.1) is the expected value function. The simplified multistage stochastic adaptive optimization problem is given in equation (2.2). In its current form, the model is computationally intractable due to the presence of semi-infinite constraints (2.2c). Decision rule based-methods circumvent this intractability by approximating the adaptive decision with a specific function of the uncertain parameters.

$$\min_{\mathbf{x}_1,\mathbf{x}_t(\cdot)} \quad \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E}\left[\sum_{t=2}^{T} \mathbf{c}_t^\top \mathbf{x}_t(\boldsymbol{\xi}_{[t]})\right] \tag{2.2a}$$

$$\text{s.t.} \quad \mathbf{A}_1\mathbf{x}_1 \geq \mathbf{b}_1 \tag{2.2b}$$

$$\sum_{s=2}^{t} \mathbf{A}_s\mathbf{x}_s(\boldsymbol{\xi}_{[s]}) \geq \mathbf{b}_t(\boldsymbol{\xi}_{[t]}) \qquad \forall \boldsymbol{\xi} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.2c}$$

For example, an LDR, as its name reflects, assumes a linear dependence between the adaptive ordering decision and uncertainty,

$$\mathbf{x}_t(\boldsymbol{\xi}_{[t]}) = \mathbf{x}_t^0 + \sum_{s=2}^{t} \mathbf{X}_s^1 \boldsymbol{\xi}_s \qquad \forall \boldsymbol{\xi} \in \Xi,\ t \in \mathcal{T}_{-1}$$

where $\mathbf{x}_t^0$ and $\mathbf{X}_t^1$ are the intercepts and slopes, respectively. As is typically done, we assume that $\mathbf{b}_t(\boldsymbol{\xi}_{[t]}) = \mathbf{b}_t^0 + \sum_{s=2}^{t} \mathbf{B}_s^1 \boldsymbol{\xi}_s$, where $\mathbf{B}_s^1$ defines the linear dependence. The approximated look-ahead model is given in equation (2.3). The tractable counterpart, for a polyhedral uncertainty set Ξ, is derived by exploiting the strong duality property of linear programming problems (Bertsimas & Tsitsiklis 1997) and is given by

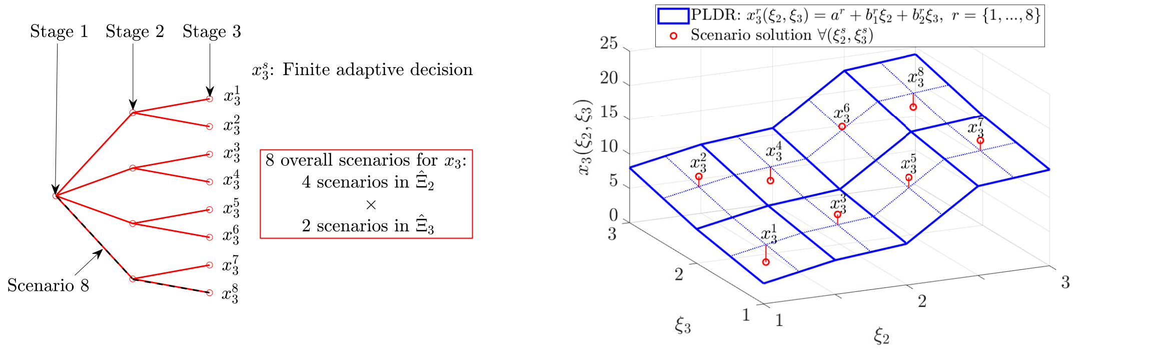
$$\min_{\mathbf{x}_1,\mathbf{x}_t^0,\mathbf{X}_s^1} \quad \mathbf{c}_1^\top \mathbf{x}_1 + \sum_{t=2}^{T} \mathbf{c}_t^\top \left(\mathbf{x}_t^0 + \sum_{s=2}^{t} \mathbf{X}_s^1 \mathbb{E}[\boldsymbol{\xi}_s]\right) \tag{2.3a}$$

$$\text{s.t.} \quad \mathbf{A}_1\mathbf{x}_1 \geq \mathbf{b}_1 \tag{2.3b}$$

$$\sum_{s=2}^{t} \mathbf{A}_s \left(\mathbf{x}_s^0 + \sum_{p=2}^{s} \mathbf{X}_p^1 \boldsymbol{\xi}_p\right) \geq \mathbf{b}_t^0 + \sum_{s=2}^{t} \mathbf{B}_s^1 \boldsymbol{\xi}_s \qquad \forall \boldsymbol{\xi} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.3c}$$

Extending the idea above, Georghiou et al. (2015) introduced PLDRs where a linear approximation of an adaptive decision in the lifted problem corresponds to a piecewise linear approximation in the original problem. In this regard, the policy is defined as $\mathbf{x}_t(\boldsymbol{\xi}'_{[t]}) = \mathbf{x}_t'^0 + \sum_{s=2}^{t} \mathbf{X}_s'^1 \boldsymbol{\xi}'_s$ for all $\boldsymbol{\xi}' \in \Xi'$, $t \in \mathcal{T}_{-1}$ where Ξ' is the underlying lifted uncertainty set.

Interestingly, some similarities can be drawn between PLDR-based methods and scenario-based stochastic programming methods. In the latter approach, the uncertainty set is approximated by a discrete set of scenarios for which a discrete set of optimal recourse decisions is computed. However, a common feature of both approaches is that the solution quality of the recourse decisions increases as the granularity/resolution of the uncertainty increases (be it more discrete scenarios or more lifted elements). To this end, Figure 2.1 attempts to graphically contrast the two approaches in a 3-stage example by illustrating the solution of the recourse decision in stage 3 where the uncertain parameters ξ_2 and ξ_3 are one-dimensional. While scenario-based stochastic programming methods consider discrete uncertainty sets $\hat{\Xi}_2$ and $\hat{\Xi}_3$ of scenarios and determine a recourse action for each scenario, decision rule-based methods provide an infinite number of recourse actions; one for every realization of the uncertain parameter in Ξ_2 and Ξ_3. Still, the impact of the number of scenarios and lifted elements on the recourse decision x_3 is comparable. For example, implementing four scenarios in stage 2 and two scenarios in stage 3 generate eight possible finite decisions x_3^s, where x_t^s denotes a recourse decision from a scenario-based method in stage t for scenario s. Likewise, introducing four lifted elements in stage 2 and two lifted elements in stage 3 generates eight linear decision functions $x_3^r(\xi_2, \xi_3)$, where x_t^r denotes a recourse function from a PLDR-based method in stage t and partition r of the lifted uncertainty set.


Stage 1
Stage 2
Stage 3
x_3^s: Finite adaptive decision
x_3^1
x_3^2
x_3^3
x_3^4
x_3^5
x_3^6
x_3^7
x_3^8
8 overall scenarios for x_3:
4 scenarios in $\hat{\Xi}_2$
×
2 scenarios in $\hat{\Xi}_3$
Scenario 8
PLDR: $x_3^r(\xi_2, \xi_3) = a^r + b_1^r\xi_2 + b_2^r\xi_3,\ r = \{1, ..., 8\}$
Scenario solution $\forall(\xi_2^s, \xi_3^s)$
$x_3(\xi_2, \xi_3)$
25
20
15
10
5
0
3
2
1
1
2
3
ξ_3
ξ_2

(a) Three-stage scenario tree

(b) Scenario- and DR-based recourse decisions in stage 3

Figure 2.1: Scenario- and decision rule-based recourse decisions analogy in stage 3 for a multi-stage stochastic adaptive optimization problem where $\xi_2,\ \xi_3 \in \mathbb{R}$.

Throughout this chapter, we make the following assumptions:

- **Assumption 1:** Piecewise linear decision rules and hybrid decision rules are constructed via lifting with *axial segmentation* as described in Georghiou et al. (2015). We do not address lifting with general segmentation.
- **Assumption 2:** The set of potential breakpoints to construct PLDRs is given. The search for an optimal set of breakpoints in each stage is still an open question but beyond the scope of this work.
- **Assumption 3:** The set of breakpoints implemented in a PLDR and for a specific uncertainty resolution in an HDR is *the same in all stages*. This is not optimal, but it allows us to perform systematic computational experiments within reasonable amount of time.

In the rest of this section, we introduce key concepts about LDR and PLDR methods by way of example. We first present a multistage stochastic newsvendor problem to illustrate how decision rules are applied. Then, we compare LDRs and PLDRs and illustrate the improvement obtained by the additional flexibility of PLDRs.

2.2.1 Illustrative example: multistage stochastic newsvendor problem

In a multistage newsvendor problem, a seller has to satisfy the demand of a perishable good at the minimal total cost. In each stage t, an order x_t is placed which becomes available in the next stage $t+1$. Figure 2.2 illustrates the chronological sequence of placing and receiving x_t with respect to observing the demand d_{t+1}; the latter is observed after the order x_t is placed, but before it is received.

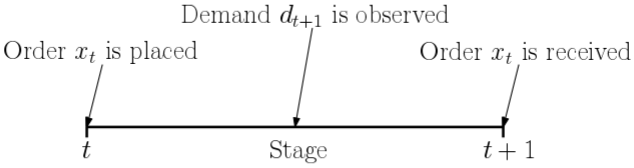


Figure 2.2: Chronological sequence of placing, receiving an order x_t and observing a demand d_{t+1} in a multistage newsvendor problem.

The cumulative difference between x_t and d_{t+1} defines the inventory and backlog: a positive value (i.e., $\max(0, I_t)$) indicates inventory amounts, while a negative value (i.e., $\max(0, -I_t)$) indicates backlog amounts. A multistage deterministic newsvendor problem is given in equation (2.4).

$$\min_{x_t, I_t} \quad \sum_{t=1}^{T-1} C_t x_t + \sum_{t=2}^{T} (H_t (I_t)^+ + B_t(-I_t)^+) \tag{2.4a}$$

$$\text{s.t.} \quad I_t = I_{t-1} + x_{t-1} - d_t \qquad \forall t \in \mathcal{T}_{-1} \tag{2.4b}$$

$$0 \leq x_t \leq U^{\mathrm{x}} \qquad \forall t \in \mathcal{T}_{-T} \tag{2.4c}$$

where I_1 is the initial inventory and $[\cdot]^+ = \max(0, \cdot)$. The per unit ordering, holding and backlog costs are given by C_t, H_t and B_t, respectively. The objective function minimizes the total ordering, holding and backlogging costs. Ordering too much results in high ordering and additional holding costs, whereas ordering too little leads to expensive backlogging costs. Equation (2.4b) dictates the balance of goods in stage t and equation (2.4c) defines the minimum and maximum ordering limits.

Equation (2.4) is reformulated as a linear programming problem by introducing the storage s_t^+ and backlog s_t^- auxiliary variables. Modelling the demand as uncertain, a multistage linear adaptive stochastic newsvendor problem is introduced in equation (2.5).

$$\min_{\substack{x_t(\cdot),I_t(\cdot)\\ s_t^+(\cdot),s_t^-(\cdot)}} \quad \mathbb{E}\left[\sum_{t=1}^{T-1} C_t x_t(\boldsymbol{d}_{[t]}) + \sum_{t=2}^{T}(H_t s_t^+(\boldsymbol{d}_{[t]}) + B_t s_t^-(\boldsymbol{d}_{[t]}))\right] \tag{2.5a}$$

$$\text{s.t.} \quad I_t(\boldsymbol{d}_{[t]}) = I_{t-1}(\boldsymbol{d}_{[t-1]}) + x_{t-1}(\boldsymbol{d}_{[t-1]}) - d_t \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.5b}$$

$$s_t^+(\boldsymbol{d}_{[t]}) \geq I_t(\boldsymbol{d}_{[t]}) \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.5c}$$

$$s_t^-(\boldsymbol{d}_{[t]}) \geq -I_t(\boldsymbol{d}_{[t]}) \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.5d}$$

$$0 \leq x_t(\boldsymbol{d}_{[t]}) \leq U^{\mathrm{x}} \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-T} \tag{2.5e}$$

$$s_t^+(\boldsymbol{d}_{[t]}),\ s_t^-(\boldsymbol{d}_{[t]}) \geq 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{2.5f}$$

where $\boldsymbol{d}_{[t]} = [d_2, \cdots, d_t]$, $x_1(\boldsymbol{d}_{[1]}) \equiv x_1$, $I_1(\boldsymbol{d}_{[1]}) \equiv I_1$, and Ξ is the underlying uncertainty set. The expectation is computed with respect to the distribution of $\boldsymbol{d}$ (i.e., $\boldsymbol{d}_{[T]}$).

2.2.2 Linear adaptive stochastic counterpart of the newsvendor problem

The adaptive decisions are defined in terms of the past demand $\boldsymbol{d}_{[t]} = (d_2, \ldots, d_t)$, which are observed and known in stage t; they cannot be a function of future unrealized demand parameters $(d_{t+1}, \ldots, d_T)$ (Ben-Tal et al. 2004). To make this temporal dependence concrete, an observation matrix $\boldsymbol{V}_t \in \mathbb{R}^{(T-1)\times(T-1)}$ for all $t \in \mathcal{T}_{-1}$ is introduced.

$$\boldsymbol{V}_t = \begin{bmatrix} \boldsymbol{I}_{t-1} & \boldsymbol{0}_{(t-1)\times(T-t)} \\ \boldsymbol{0}_{(T-t)\times(T-1)} & \boldsymbol{0}_{(T-t)\times(T-t)} \end{bmatrix} \quad \forall t \in \mathcal{T}_{-1} \tag{2.6}$$

where $\boldsymbol{I}_{t-1} \in \mathbb{R}^{(t-1)\times(t-1)}$ is the identity matrix. It relates $\boldsymbol{d}$ to $\boldsymbol{d}_{[t]}$ as follows

$$\boldsymbol{d}_{[t]} = \boldsymbol{V}_t \boldsymbol{d} = [d_2, d_3, \ldots, d_t, 0, \cdots, 0]^\top \quad \forall t \in \mathcal{T}_{-1} \tag{2.7}$$

The linear ordering decision rule is equivalent to $x_t(\boldsymbol{d}_{[t]}) = x_t^0 + \boldsymbol{X}_t^1 \boldsymbol{V}_t \boldsymbol{d}$ for all $t \in \mathcal{T}_{-T}$. Similar LDRs for $I_t(\boldsymbol{d}_{[t]})$, $s_t^+(\boldsymbol{d}_{[t]})$ and $s_t^-(\boldsymbol{d}_{[t]})$ are used. Assuming a polyhedral uncertainty set

$$\Xi := \left\{\boldsymbol{d} \in \mathbb{R}^{T-1} \middle|\ \boldsymbol{W}\boldsymbol{d} \geq \boldsymbol{h}\right\} \tag{2.8}$$

where $\boldsymbol{W} \in \mathbb{R}^{m\times(T-1)}$ and $\boldsymbol{h} \in \mathbb{R}^m$, the newsvendor linear affine stochastic counterpart (LASC) is derived in the supplementary material.

2.2.3 Piecewise linear adaptive stochastic counterpart of the newsvendor problem

Georghiou et al. (2015) developed a generic lifting operator $L : \mathbb{R}^k \to \mathbb{R}^{k'}$ that maps $\boldsymbol{d}$ to $\boldsymbol{d}'$ and an inverse retraction operator $R : \mathbb{R}^{k'} \to \mathbb{R}^k$ that projects $\boldsymbol{d}'$ to $\boldsymbol{d}$. The original and lifted uncertainty sets are $\Xi \subset \mathbb{R}^k$ and $\Xi' \subset \mathbb{R}^{k'}$ $(k' > k)$, respectively. The lifted uncertain demand is given as $\boldsymbol{d}' = (\boldsymbol{d}'_2, \ldots, \boldsymbol{d}'_T)^\top = (d'_{21}, \ldots, d'_{2r_2}, \ldots, d'_{T1}, \ldots, d'_{Tr_T})^\top$. Assuming a bounded demand,

$l_t \leq d_t \leq u_t$, the lifting operator $L_{tj}(d_t)$ defined in Georghiou et al. (2015), and restated in equation (2.9), maps d_t to d'_{tj} for all $j = \{1, \cdots, r_t\}$.

$$L_{tj}(d_t) = \begin{cases} d_t & \text{if } r_t = 1, \\ \min\{d_t, z^t_j\} & \text{if } r_t > 1,\ j = 1, \\ \max\{\min\{d_t, z^t_j\} - z^t_{j-1}, 0\} & \text{if } r_t > 1,\ j = 2, \ldots, r_t - 1 \\ \max\{d_t - z^t_{j-1}, 0\} & \text{if } r_t > 1,\ j = r_t, \end{cases} \quad \forall t = \{2, \cdots, T\} \tag{2.9}$$

where z^t_j is the j^{th} breakpoint, where a slope change may occur, and r_t is the number of linear pieces in d_t. The retraction operator defined in Georghiou et al. (2015) is the sum of the lifted elements

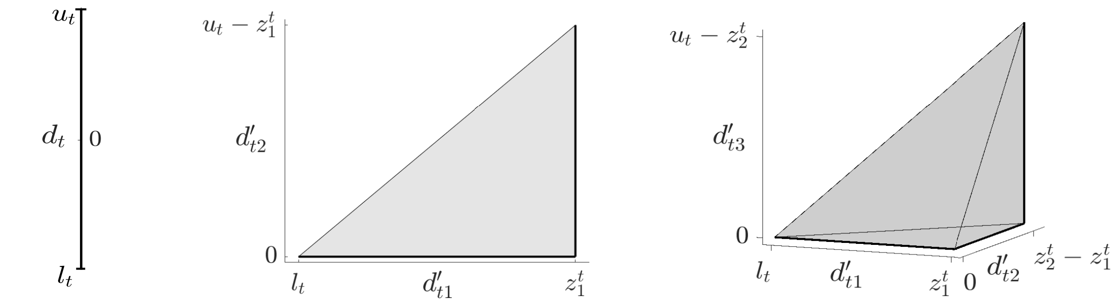
$$d_t = R\boldsymbol{d}'_t = \sum_{j=1}^{r_t} d'_{tj} \quad \forall t = \{2, \cdots, T\} \tag{2.10}$$

PLDRs are, in theory, linear functions in the lifted problem. To differentiate notations between a PLDR and an LDR, we use the superscript " $'$ " to indicate that the variable/parameter is used in a PLDR context. The observation matrix $\boldsymbol{V}'_t \in \mathbb{R}^{k' \times k'}$ for all $t \in \mathcal{T}_{-1}$ is reformulated in equation (2.11) where $k_t = \sum_{i=2}^{t} r_i$ and $k' \equiv k_T$.

$$\boldsymbol{d}'_{[t]} = \boldsymbol{V}'_t \boldsymbol{d}' = \begin{bmatrix} \boldsymbol{I}_{k_t} & \boldsymbol{0}_{k_t \times (k'-k_t)} \\ \boldsymbol{0}_{(k'-k_t) \times k_t} & \boldsymbol{0}_{(k'-k_t) \times (k'-k_t)} \end{bmatrix} \boldsymbol{d}' = [\boldsymbol{d}'_2, \cdots, \boldsymbol{d}'_t, 0, \ldots, 0]^\top \quad \forall t \in \mathcal{T}_{-1} \tag{2.11}$$

The non-convexity of the lifted uncertainty set Ξ'_t, consequently the non-convexity of Ξ', violates the strong duality property required to derive the stochastic counterpart. Alternatively, the convex hull Ξ'_t is used without affecting the optimal solution (Georghiou et al. 2015). Figure 2.3 illustrates a bounded uncertain demand $d_t \in \Xi_t$, the non-convex lifted uncertainty sets Ξ'_t and the convex hulls using one and two breakpoints (i.e., $r_t = 2$ and 3), respectively. When Ξ is a generic polytope, not a hyper-rectangle, the lifted uncertainty set Ξ' has no tractable convex hull representation. To overcome this issue, Georghiou et al. (2015) presented a tractable polyhedral outer approximation of the convex hull.

The dimension of the lifted uncertain parameter in stage t is analogous to the number of scenarios in stage t in a scenario tree. Figure 2.1 emphasizes the analogy which supports the motivation behind relating an empirical efficient construct of a scenario tree to the lifting strategy in a hybrid decision rule.



(a) No lifting (b) Single breakpoint (c) Two breakpoints

Figure 2.3: Lifted uncertainty set Ξ'_t (i.e., bold line) of uncertain demand d_t and its convex hull (i.e., shaded region)

The details of the newsvendor piecewise linear adaptive stochastic counterpart (PWLASC) are included in the supplementary material.

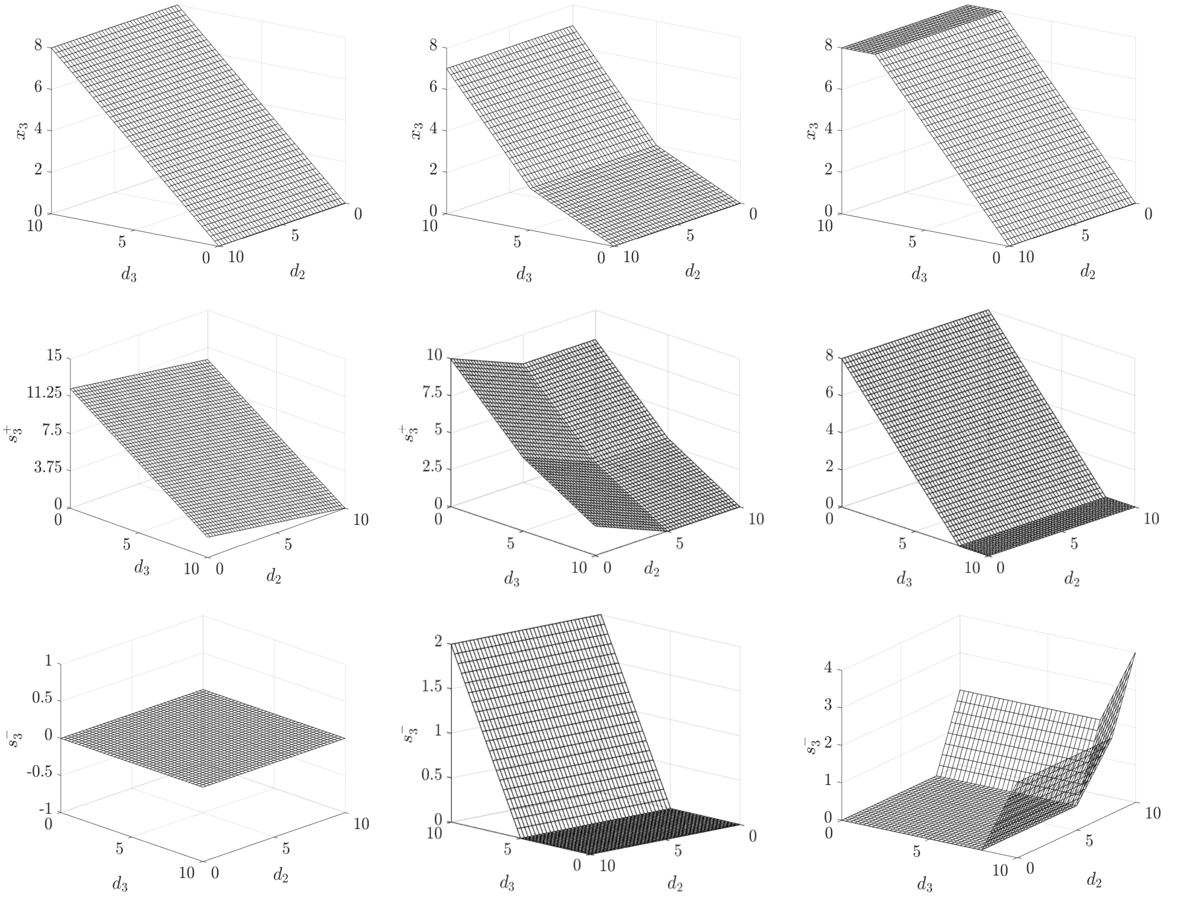
2.2.4 Numerical results for the multistage stochastic newsvendor problem

This section illustrates the solution quality improvements generated by PLDRs relative to LDRs, corroborating the empirical results found in Georghiou et al. (2015). Since the thrust of our chapter is to showcase the benefits of HDRs that rely on PLDR structures, and since PLDRs are themselves rather new in the stochastic/robust optimization community, it is valuable to highlight the potential solution quality improvements possible with PLDRs. It is also a predecessor to the next section where we address the following question: Does the additional flexibility of PLDRs (i.e., a more complex decision rule) always lead to better practical policies? We demonstrate that improvement is not guaranteed; in fact, the solution can deteriorate. Unless otherwise stated, the uncertain demand is stagewise independent and follows a uniform distribution between 0 and 10 in each stage. The cost coefficients are equal to $C_t = 3,\ H_t = 1.5,\ B_t = 7$ in all stages. The ordering amount limit U^{x} and the initial inventory I_1 are equal to 8 and 4, respectively.

Three decision rules are studied: (i) LDR, which serves as a lower bound, (ii) PLDR-1 ($\mathbb{E}[d_t]$) and (iii) PLDR-1 (U^{x}). The two PLDRs apply a single breakpoint in each d_t at $\mathbb{E}[d_t]$ and U^{x}, respectively. While there is not a systematic method to identify the optimal set of breakpoints, we think that relating it to a characteristic of the uncertainty distribution (e.g., $\mathbb{E}[d_t]$) or a physical state/resource of the model (e.g., U^{x}) is an intuitive and a practical choice.

For $T = 4$, the LDR optimal cost is equal to 83.5, and the first stage solution x_1 is equal to 8. As expected, a PLDR-1 ($\mathbb{E}[d_t]$) reduces the optimal cost to 66.25 which reflects an improvement of 20.66% with respect to an LDR. Likewise, x_1 has decreased by 25% to a value of 6. Meanwhile, a PLDR-1 (U^{x}) solution quality exceeds that of a PLDR-1 ($\mathbb{E}[d_t]$) where the optimal cost is equal to 63.60, and x_1 is reduced to 4. The solution of the optimal policies is reported in the supplementary material.

The ordering, inventory and backlog policies in stage 3 are demonstrated in Figure 2.4. On the one hand, we observe that PLDRs provide the flexibility to implement different recourse strategies based on the previous realization of the uncertain demand. For example, the policy for s_3^+ generated by a PLDR-1 ($\mathbb{E}[d_t]$) exhibits four different recourse strategies due to a single breakpoint in d_2 and d_3, while an LDR generates only a single recourse strategy. On the other hand, we observe that the look-ahead approximations of the mutually exclusive state variables output simultaneous positive values. As an illustration, for (d_2, d_3)=$(7, 9)$, the values of $(s_3^+(d_2, d_3)$,$s_3^-(d_2, d_3))$ are $(0.6, 1.6)$ and $(1, 1)$ using a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^{x}), respectively. It is fair to argue that the look-ahead model-based solution quality has some degree of overestimation and is not accurate.



(a)

(b)

(c)

Figure 2.4: Optimal $x_3(d_2, d_3)$, $s_3^+(d_2, d_3)$ and $s_3^-(d_2, d_3)$ policies obtained using (a) an LDR, (b) a PLDR-1 ($\mathbb{E}[d_t]$) and (c) a PLDR-1 (U^{x}) in a multistage stochastic newsvendor problem.

2.3 Is more complexity necessary? Motivating hybrid decision rules

To motivate our investigation of hybrid decision rules, we begin this section by posing a succession of questions under increasingly restrictive conditions to better understand when higher complexity decision rules should be sacrificed for lower complexity ones. We start with the general, yet provocative question: Is a more complex decision rule guaranteed to outperform a less complex one? Fortunately, Bertsimas et al. (2010) provided a negative answer to this question by proving the optimality of LDR policies in the context of one-dimensional, constrained, multistage robust optimization.

Moving beyond this special case when LDRs are known to be optimal, we ask the same question under more restrictive conditions: Can a less complex decision rule outperform a more complex one when the following assumptions, labeled (A1)-(A3), hold: (A1) The latter includes the former as a special case; (A2) Both decision rule-based look-ahead models assume the same "forecasted" uncertainty set; (A3) This "forecasted" uncertainty set coincides with the true underlying uncertainty set? de Ruiter et al. (2016) provided a partial answer to this question when a worst-case objective is being optimized. More concretely, building on the insightful work of Iancu & Trichakis (2013), which introduced the notion of Pareto robust optimality and the adverse consequences of using dominated robust optima in practice, de Ruiter et al. (2016) demonstrated empirically that an optimal solution to a static robust problem can outperform the solution of an affinely adjustable robust counterpart within a folding horizon simulation. In other words, they answered this question in the affirmative by showing that a static robust policy (i.e., a constant decision rule as all decisions are treated as here-and-now decisions) can exhibit superior *average* performance relative to an LDR, which contains the static policy as a special case. They attributed this unexpected behavior to the presence of multiple robust optima. It is important to stress that de Ruiter et al. (2016) "trained" these policies using a robust look-ahead model and then "tested" (i.e., evaluated) them based on their simulated expected value performance.[2] To overcome this issue, they applied a two-step approach suggested in Iancu & Trichakis (2013) to generate non-dominated robust LDRs, which then outperformed the static robust counterparts with respect to average performance. In the context of our complexity question, they found that, after finding non-dominated policies, lower complexity static robust policies never outperform higher complexity LDRs. More details about training and testing policies are provided in the supplementary material.

Finally, moving beyond the robust (e.g., min-max) optimization setting to our risk-neutral setting, we pose an even more restrictive version of the above question: Can a less complex decision rule outperform a more complex one when assumptions (A1)-(A3) as well as the following conditions hold: (A4) The objective function of the look-ahead model used to "train" the policies is risk neutral, i.e., based on expected value performance; (A5) The policies are

[2] de Ruiter et al. (2016) also "trained" LDRs using a risk neutral look-ahead model, and surprisingly always obtained a unique optimum, to demonstrate a different point, but this is not a concern for our purposes.

non-dominated or Pareto optimal as defined by Iancu & Trichakis (2013)? In this section, we answer this question in the affirmative and show that the presence of multiple optima is *not* the reason for this behavior.

In our study, the look-ahead models are the derived adaptive stochastic counterparts. For brevity, we will refer to the "look-ahead model" as the "model" for the rest of the chapter. We used two simulators to test the model-based policies. The design of a simulator is user-specific. For clarity, we describe them in the following two points:

- **First simulator**: The first simulator is a folding horizon simulator with no re-optimization in stages $t \in \{2, \ldots, T-1\}$. It uses the model-based ordering policy computed at the outset to evaluate the ordering decisions in all stages. The state variables are evaluated by their definitions: $s_t^+ = \max(0, I_t)$ and $s_t^- = \max(0, -I_t)$ for all $t \in \mathcal{T}_{-1}$, rather than by their model-based approximations.
- **Second simulator**: The second simulator is a moving horizon simulator. It re-optimizes the model-based ordering policy to evaluate the ordering decision in each stage (Powell 2014). We assume that the uncertain demand forecasting horizon in each re-optimization is the same. The state variables are always computed by their definitions: $s_t^+ = \max(0, I_t)$ and $s_t^- = \max(0, -I_t)$ for all $t \in \mathcal{T}_{-1}$. The last re-optimization is done in stage $t = T$ where the simulator is truncated.

We test the policies using 10^5 and 10^2 scenarios in the first and second simulators, respectively, where a scenario consists of uncorrelated demand realizations sampled from the uniform distribution $\mathcal{U}(0, 10)$ in stages $t \in \mathcal{T}_{-1}$.

A comparison between the model- and simulator-based costs of an LDR, a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^{x}) is shown in Table 2.1. It is immediately apparent that the model tends to overestimate decision rule quality, i.e., projected expected cost relative to simulated expected cost. For example, a PLDR-1 (U^{x}) is superior to a PLDR-1 ($\mathbb{E}[d_t]$) within the model, but not when evaluated within the first simulator. We found that the quality of a PLDR-1 ($\mathbb{E}[d_t]$) within the first simulator (i.e., 59.07) is better than its quality within the second simulator (i.e., 84.26). This observation may look counter-intuitive as one would expect that by re-optimizing in each stage, the added value of knowing the uncertainty information in the previous stages would be reflected in a lower optimal cost. We emphasize that the second simulator is designed in a moving horizon framework, which in contrast to the folding horizon framework (i.e., first simulator), is truncated in $t = T$ where the ordering costs are implemented as well. The uncertain demand forecasting horizon in the re-optimizations in $t \geq 2$ exceeds the simulator end of horizon which may lead to an overfit of the optimal ordering policy, i.e., excessively ordering units to hedge against uncertainty beyond T. We also reported the standard deviation (i.e., σ) and minimum/maximum values (i.e., min/max) of the optimal cost distribution in each simulator.

Table 2.1: Model- and simulator-based costs comparison for a multistage stochastic newsvendor problem using an LDR, a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^{x}).

	Model	First simulator				Second simulator			
Decision rule	$\mathbb{E}$[cost]	$\mathbb{E}$[cost]	σ	min	max	$\mathbb{E}$[cost]	σ	min	max
LDR	83.50	75.14	4.72	63.00	88.20	99.22	6.29	83.24	110.59
PLDR-1($\mathbb{E}[d_t]$)	66.25	59.07	9.97	45.30	104.40	84.26	6.33	68.25	95.59
PLDR-1(U^{x})	63.60	59.88	11.23	36.52	139.44	74.35	11.95	53.25	110.84

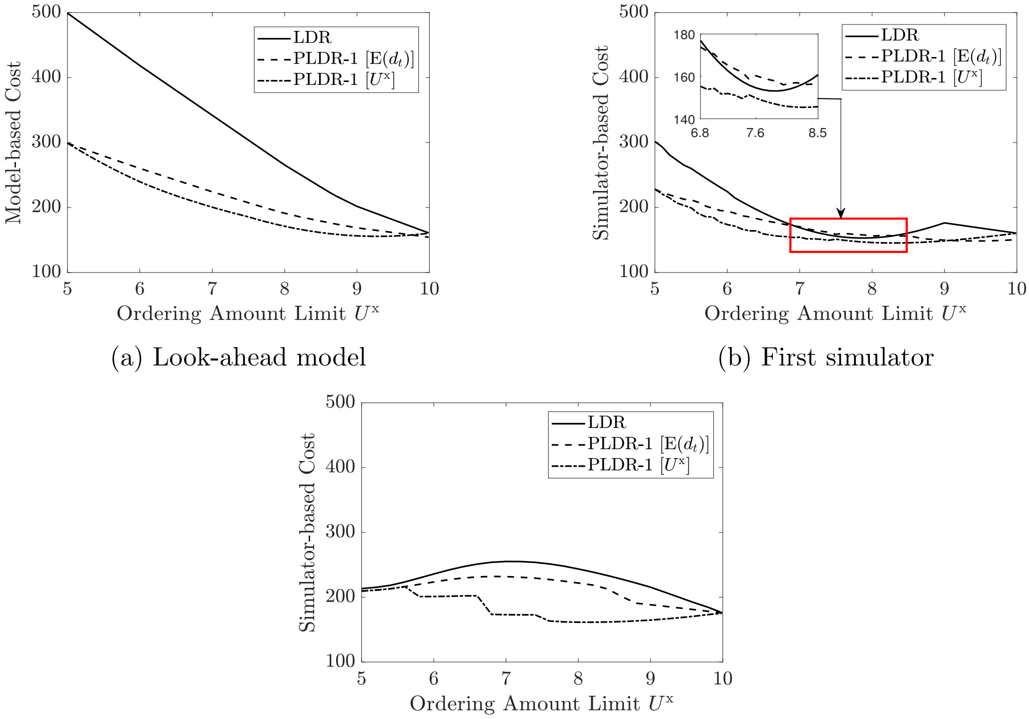
The overestimation of the model-based cost, with respect to simulator 1, stems from the approximations of the mutually exclusive state variables $s_t^+(\boldsymbol{d}_{[t]})$ and $s_t^-(\boldsymbol{d}_{[t]})$ via the model-based decision rules. The model-based inventory and backlog decision rules *may* output, for a given scenario, simultaneous positive decisions for what are supposed to be in practice mutually exclusive (i.e., the seller either has a deficit or a surplus, but not both). Given the scenario $(d_2, d_3, d_4) = (5, 9, 3)$, LDR and PLDR-1($U^{\mathrm{x}}$) output positive inventory and backlog decisions in $t = 4$, whereas PLDR-1($\mathbb{E}[d_t]$) exhibits similar behavior in $t = 3$.

Mathematically, the complementary condition $s_t^+(\boldsymbol{d}_{[t]})s_t^-(\boldsymbol{d}_{[t]}) = 0$ for all $t \in \mathcal{T}_{-1}$, which is always satisfied in deterministic and scenario-based stochastic settings, is not guaranteed by the model-based decision rules for all $\boldsymbol{d}_{[t]} \in \Xi$. This is the cost of deriving tractable stochastic counterparts for the intractable semi-infinite model in equation (2.5). On the one hand, the stochastic counterparts immunize the model against the worst-case scenario and, as a consequence of assuming a convex uncertainty set, ensure feasibility for all scenarios in the uncertainty set. On the other hand, enforcing the complementary condition for the worst-case scenario does not guarantee the complementary condition for the other scenarios within the uncertainty set.

It is fair to claim that the inventory and backlog model-based policies, and as a consequence the model-based solution quality, cannot be "fully trusted." Based on that, is it still true that less complex decision rules are never superior to their more complex counterparts? To address this question, we extend the planning horizon T from 4 to 8 and we compute the model- and simulator-based costs for an LDR, a PLDR-1($\mathbb{E}[d_t]$) and a PLDR-1 (U^{x}) for $U^{\mathrm{x}} \in [5, 9.99]$[3] at increments of 0.2. Figure 2.5 illustrates the obtained results. The model shows that the more complex decision rule (i.e., PLDR-1) outperforms the less complex decision rule (i.e., LDR) for all U^{x}. More precisely, the added value of a PLDR-1 in the model is most significant at low values of U^{x} which, as suggested by the model, means that the added computational cost by the increased complexity of PLDR-1 is most justifiable. Figure 2.5b demonstrates the results within the first simulator, in which an LDR does indeed outperform a PLDR-1 ($\mathbb{E}[d_t]$) for $U^{\mathrm{x}} \in [6.9, 8.3]$. This observation is statistically verified using 100 replications. For example, at $U^{\mathrm{x}} = 7.5$, the simulated cost confidence intervals of an LDR, PLDR-1 ($\mathbb{E}[d_t]$) and PLDR-1 (U^{x}) are equal to 155.9689 ± 0.2324, 158.8269 ± 0.2658 and 151.2677 ± 0.2406 in the first simulator,

[3] Note that we chose the upper bound to be 9.99 because PLDR-1 (U^{x}) is not defined at $U^{\mathrm{x}} = 10$.

respectively. We verified the outcome for 10^4, 10^3 and 10^2 scenarios as well.



(a) Look-ahead model

(b) First simulator

(c) Second simulator

Figure 2.5: Model- and simulator-based costs using an LDR, a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^x) as a function of U^x for a multistage stochastic newsvendor problem.

Apparently, an LDR is never found to be superior to a PLDR-1 within the second simulator as shown in Figure 2.5c. Nevertheless, we observe that the LDR quality is not as inferior as depicted by the model in Figure 2.5a for low values of U^x. A similar observation is made within the first simulator in Figure 2.5b. The model seems to overestimate the added value by the more expensive complex decision rule. It is more intuitive to think that a lack of resources (e.g., limited ordering limit U^x) restricts the ability of a decision maker to address uncertainty irrespective of the complexity of a policy. Likewise, the surplus of resources, which usually comes at a price not reflected in our work, provides a buffer by which the decision maker does not necessarily need a complex policy to pro-actively hedge against future uncertainty. Figure 2.5c is a good illustration of the relationship between the level resources or physical states in a model and the need for complex model-based policies. The added value of a more complex decision rule is mostly appreciated in the instances where there is neither a lack nor a surplus of resources. In our problem, this is observed within the second simulator for $U^x \in [7, 8]$.

To justify our results in Figure 2.5b and appreciate the overestimation in the model-based total cost, Figure 2.6 compares the total holding and backlog costs for the model and the first simulator for $T = 8$ and $U^x = 8$. The overestimation is most significant in the backlog cost and is more prominent in the LDR policy. The LDR model-based total backlog cost is the highest, while the LDR simulator-based counterpart is the lowest. The latter is due to a more conservative LDR ordering policy, compared to a PLDR-1, that reduces the magnitude of the simulated mean backlog cost. It also empirically explains why an LDR outperforms a PLDR-1 ($\mathbb{E}[d_t]$) for $U^{\mathrm{x}} \in [6.9, 8.3]$.

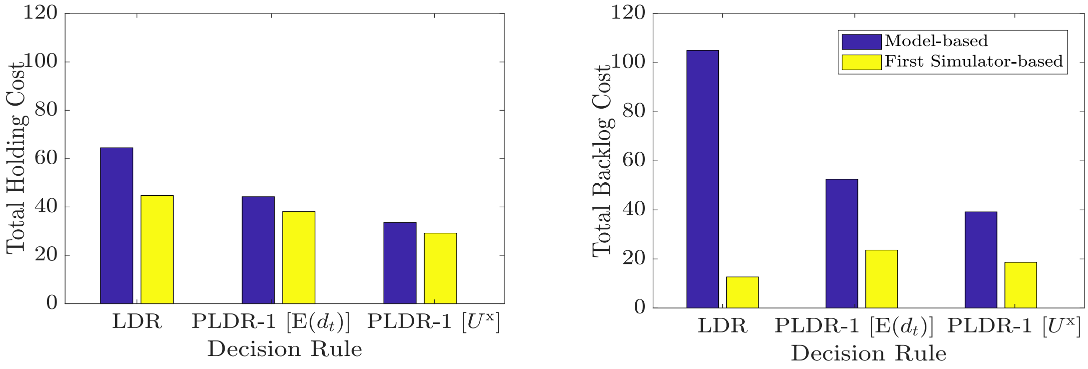


Figure 2.6: Total holding and backlog costs of a multistage stochastic newsvendor problem using an LDR, a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^{x}) within the model and the first simulator.

We conclude this section by highlighting a few points. First, the uncertainty set used in this example is a hyper-rectangle. The convex hull of the lifted uncertainty set is constructed exactly with no overestimation. Thus, the deterioration in the flexibility of a PLDR-1 ($\mathbb{E}[d_t]$) in a simulator, and its possible inferiority to an LDR, cannot be attributed to an overestimation in the lifted uncertainty set.

Second, we implemented a two-step approach similar to what was introduced in de Ruiter et al. (2016), but with an expected objective function in the first step and a worst-case objective function in the second step. The optimal expected value in the second step is guaranteed to be no worse than the optimal expected value obtained in the first step. The purpose of this procedure is to find a different optimal solution with the same expected cost and lower worst-case cost. We found that multiple optima do exist and we verified that, an LDR can outperform a PLDR with a single breakpoint using the second optimal solution. More details are included in the supplementary material.

Third, we highlight that even though we did not introduce last-stage penalties to eliminate the end of horizon effect (Fisher et al. 2001), we empirically verified that the outcome in Figure 2.5b is still valid when these penalties are present.

2.4 Hybrid lifting strategies

We showed in the previous section that the superiority of a complex decision rule could be overestimated by a look-ahead model, when, in fact, the actual improvement in the solution quality may not justify the added computational overhead. In this section, we exploit the modularity of PLDRs with axial segmentation to propose less complex decision rules that present a better trade-off between solution quality and computational cost. Using a multistage stochastic transportation problem, we empirically illustrate that having higher uncertainty resolution (i.e., more linear pieces) in early stages is more significant than having it in late stages using model-based results. Then, we use this empirical evidence to devise decisions rules with non-increasing hybrid lifting strategies that are competitive to, in some cases better than, the more complex PLDR with a similar number of breakpoints. At the same time, we note that a poorly designed non-increasing hybrid lifting strategy loses these computational benefits. All hybrid decision rules are evaluated in a simulator.

2.4.1 Multistage stochastic transportation model

We present a multistage stochastic transportation problem in equation (2.12). The framework involves a set of suppliers $\mathcal{I}$ and a set of customers $\mathcal{J}$. Suppliers are able to store some of their products for future time as inventory. In each stage t, a decision maker must determine (1) the number of units x_{it} produced by supplier i which will first become available for distribution in the subsequent stage $t+1$; (2) the number of transported units y_{ijt} from supplier i to customer j; (3) the number of units I_{it} carried as inventory by supplier i.

$$\begin{aligned}
\max_{\substack{x_{it}(\cdot),I_{it}(\cdot)\\ y_{ijt}(\cdot)}} \quad & \mathbb{E}\Bigg[\sum_{t\in\mathcal{T}_{-1}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}(R_{jt}-T_{ijt})y_{ijt}(\boldsymbol{\xi}_{[t]}) - \sum_{t\in\mathcal{T}_{-T}}\sum_{i\in\mathcal{I}} C_{it}x_{it}(\boldsymbol{\xi}_{[t]}) \\
& - \sum_{t\in\mathcal{T}_{-1}}\sum_{i\in\mathcal{I}} H_{it}I_{it}(\boldsymbol{\xi}_{[t]}) + \sum_{i\in\mathcal{I}} S_i I_{iT}(\boldsymbol{\xi}_{[T]})\Bigg] && \text{(2.12a)}\\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} y_{ijt}(\boldsymbol{\xi}_{[t]}) \le I_{it}(\boldsymbol{\xi}_{[t]}) \qquad \forall i\in\mathcal{I},\ \boldsymbol{\xi}\in\Xi,\ t\in\mathcal{T}_{-1} && \text{(2.12b)}\\
& \sum_{i\in\mathcal{I}} y_{ijt}(\boldsymbol{\xi}_{[t]}) \le D_{jt}(\boldsymbol{\xi}_t) \qquad \forall j\in\mathcal{J},\ \boldsymbol{\xi}\in\Xi,\ t\in\mathcal{T}_{-1} && \text{(2.12c)}\\
& I_{it}(\boldsymbol{\xi}_{[t]}) = I_{it-1}(\boldsymbol{\xi}_{[t-1]}) + x_{it-1}(\boldsymbol{\xi}_{[t-1]}) - \sum_{j\in\mathcal{J}} y_{ijt-1}(\boldsymbol{\xi}_{[t-1]}) \\
& \qquad \forall i\in\mathcal{I},\ \boldsymbol{\xi}\in\Xi,\ t\in\mathcal{T}_{-1} && \text{(2.12d)}\\
& 0 \le x_{it}(\boldsymbol{\xi}_{[t]}) \le U_i^{\max} \qquad \forall i\in\mathcal{I},\ \boldsymbol{\xi}\in\Xi,\ t\in\mathcal{T}_{-T} && \text{(2.12e)}\\
& I_{it}(\boldsymbol{\xi}_{[t]}),\ y_{ijt}(\boldsymbol{\xi}_{[t]}) \ge 0 \qquad \forall i\in\mathcal{I},\ j\in\mathcal{J},\ \boldsymbol{\xi}\in\Xi,\ t\in\mathcal{T}_{-1} && \text{(2.12f)}
\end{aligned}$$

where Ξ is the underlying polyhedral uncertainty set. We let $y_{ij1}(\boldsymbol{\xi}_{[1]}) = 0$ for all $i,\ j$, $x_{i1}(\boldsymbol{\xi}_{[1]}) = x_{i1}$ and $I_{i1}(\boldsymbol{\xi}_{[1]}) = I_{i1}$ for all i.

The objective function (2.12a) seeks to maximize profit. It includes revenue, transportation, production, holding costs and salvage value to mitigate the end of horizon effect (Fisher et al. 2001). Constraint (2.12b) ensures that the transported units out from each supplier do not exceed the available inventory. Meanwhile, constraint (2.12c) dictates that the transported units to each customer do not surpass the uncertain demand. Inventory balance is governed by constraint (2.12d), while constraint (2.12e) limits the produced units by supplier i within 0 and

$U_i^{\max}$. We assume that the uncertain customer demand is an affine function of the uncertainty

$$D_{jt}(\boldsymbol{\xi}_t) = D_{jt}^0 + (\mathbf{D}_{jt}^1)^\top \boldsymbol{\xi}_t \quad \forall j \in \mathcal{J},\ \boldsymbol{\xi}_t \in \Xi_t, \quad t \in \mathcal{T}_{-1}. \tag{2.13}$$

The multistage transportation LASC and PWLASC are included in the supplementary material.

2.4.2 Numerical results for the multistage stochastic transportation problem

In this section, we demonstrate that having a higher number of linear pieces in early stages is more attractive than having it in late stages. This complements empirical evidence found in scenario-based stochastic programming methods where scenario trees with higher granularity in early stages are more attractive (Bakkehaug et al. (2014), Arslan & Papageorgiou (2017)). We verify the latter empirical evidence in the context of decision rule-based methods via a sensitivity analysis of the solution quality with respect to the number of linear pieces, equivalently the number of breakpoints, in each stage. Then, we introduce examples of non-increasing hybrid decision rules to demonstrate the computational benefits.

Our initial computational setting includes three suppliers and two customers. The uncertainty in each stage is uni-dimensional, independent and follows a uniform distribution between 0 and 3. Table 2.2 lists the remaining computational parameters.

Table 2.2: Computational setting for a multistage stochastic transportation problem. Parameters are constant in all stages (e.g., $C_{it} \equiv C_i$ for all t).

	C_i	H_i	S_i	$U_i^{\max}$		T_{ij}			D_j^0	D_j^1	R_j
$i \downarrow$					$i \downarrow j \rightarrow$	1	2	$j \downarrow$			
1	5	2	0	10	1	3	4	1	5	2	18
2	7	3	0	8	2	1	5	2	3	1	16
3	1	0.5	0	5	3	6	2	—	—	—	—

Limitation of PLDRs with a large number of breakpoints

An introductory experiment is conducted to appreciate the exponential increase in the computational cost exhibited from lifting the uncertainty using a large number of breakpoints in each stage. Figure 2.7 illustrates the increase in the model size (number of constraint and variables) and the computational time required to solve the multistage stochastic transportation problem for all $T \in \{2, \ldots, 30\}$ using an LDR, a PLDR-1 (0.5) and a PLDR-5 ([0.5,1,$\mathbb{E}[\xi_t]$,2,2.5]). The exponential increase in the case of PLDR-5 is evident.

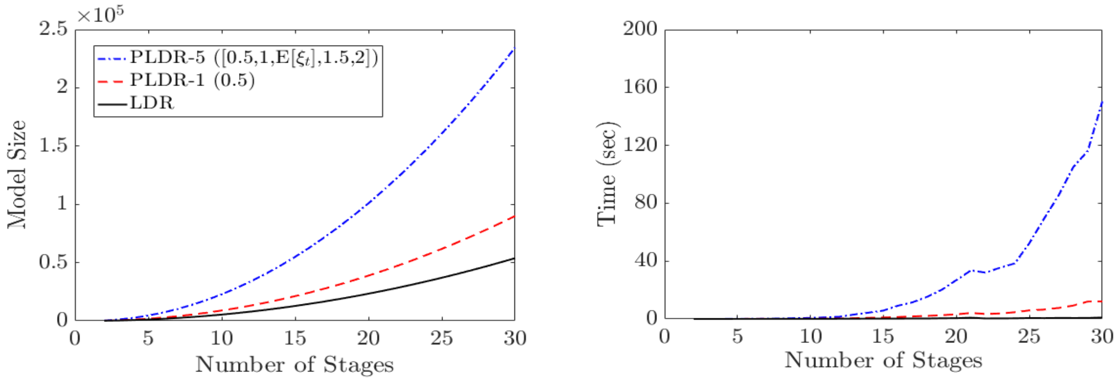
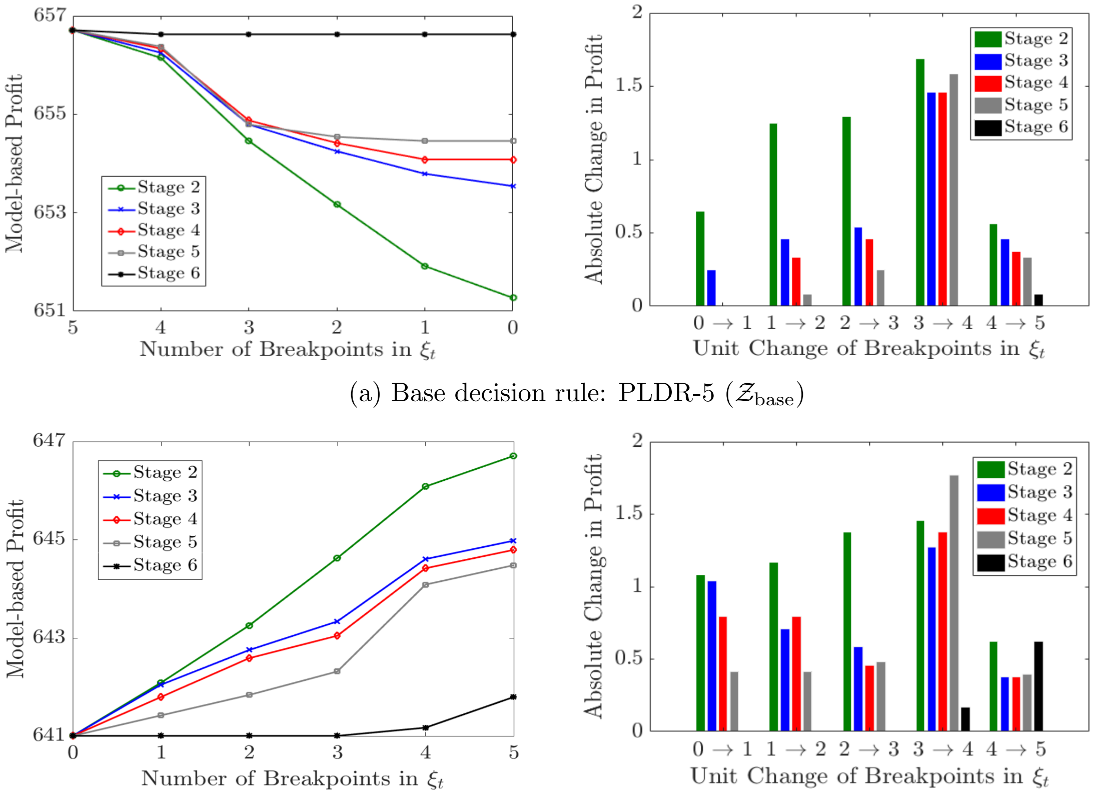


Figure 2.7: Model size (number of constraints and variables) and computational time required to solve a multistage stochastic transportation problem using an LDR, a PLDR-1 (0.5) and a PLDR-5 ([0.5,1,$\mathbb{E}[\xi_t]$,2,2.5]) for $T \in \{2, \dots, 30\}$.

Impact of uncertainty resolution in early and late stages on solution quality

In our sensitivity analysis, we introduced two base decision rules as two extreme references: (1) an LDR where there is only one linear piece in ξ_t for all $t \in \mathcal{T}_{-1}$ and (2) a PLDR-5 ([0.5,1,$\mathbb{E}[\xi_t]$,2,2.5]) which includes six linear pieces in ξ_t for all $t \in \mathcal{T}_{-1}$. The LDR and PLDR-5 ([0.5,1,$\mathbb{E}[\xi_t]$,2,2.5]) reflect the lowest and highest uncertainty resolutions in all stages. The sensitivity of the solution quality to the resolution of ξ_t for all $t \in \mathcal{T}_{-1}$ is measured by unit changes to the number of breakpoints used to lift ξ_t while keeping the resolution of $\xi_{t'}$ for all $t' \in \mathcal{T}_{-\{1,t\}}$ equal to that of the base decision rule.

In the first experiment, we set $T = 6$ and $S_i = 6$ for all i. Figure 2.8 illustrates the model-based profit as a function of the number of breakpoints used to lift ξ_t for all $t \in \mathcal{T}_{-1}$ using both base decision rules. The bar plots compare the absolute change in profit for a unit-change in the number of breakpoints in all stages. The overall trend shows that the solution quality is more sensitive to the change in resolution in early stages compared to late stages. Few exceptions are observed, but we emphasize that (1) the salvage values are determined empirically and they do not entirely eliminate the end of horizon effect and (2) the impact of lifting ξ_t for all $t \in \mathcal{T}_{-1}$ using an additional breakpoint depends highly on the breakpoint value and the nature of the true decision rule solution in stage t.

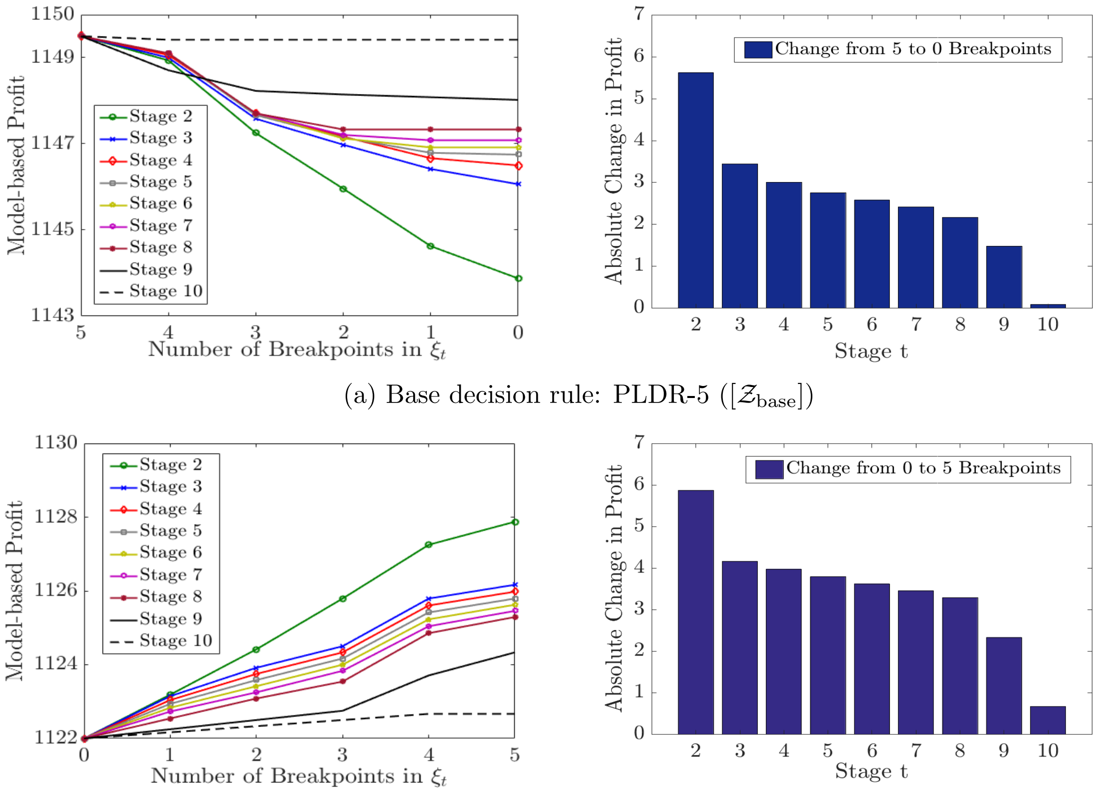


(a) Base decision rule: PLDR-5 ($\mathcal{Z}_{\text{base}}$)

(b) Base decision rule: LDR

Figure 2.8: Model-based profit sensitivity curves with respect to uncertainty resolution in all stages for a multistage stochastic transportation problem using a PLDR-5 ($p0.5, 1, \mathbb{E}[\xi_t], 2, 2.5]$) and an LDR as base decision rules.

In the second experiment, we set $T = 10$ and $S_i = 7.5$ for all i. Likewise, Figure 2.9 illustrates similar sensitivity curves for the model-based profit which reiterates the conclusion made in the first experiment. Figure 2.9b outputs a clear trend showing that higher uncertainty resolution in early stage of a decision rule is more favourable. There is one discrepancy in Figure 2.9a that is attributed to the same causes previously described for the first experiment. The bar plots depict the absolute change in profit obtained by changing the resolution of ξ_t in each stage between the two extrema, a result which aligns with the previously made observations.



(a) Base decision rule: PLDR-5 ($[\mathcal{Z}_{\text{base}}]$)

(b) Base decision rule: LDR

Figure 2.9: Model-based profit sensitivity curves with respect to the uncertainty resolution in a multistage stochastic transportation problem using a PLDR-5 ($[0.5, 1, \mathbb{E}[\xi_t], 2, 2.5]$) and an LDR as base decision rules.

Evaluating HDRs within a moving horizon simulator

We empirically demonstrated that higher uncertainty resolution in early stages offers more flexibility in a policy when compared to the same resolution in late stages. Still, we need to investigate various lifting strategies to answer the following question: Under what conditions is it more likely that non-increasing HDRs will provide an attractive trade-off between solution quality and computational cost?

An LDR is equivalent to an HDR<0. . . 0> where none of the uncertainty elements are lifted, while a PLDR-p is equivalent to an HDR<$pp \ldots p$> where the uncertainty elements are lifted using the same number of breakpoints p in all stages. Following up on the initial question, we investigate the following relation

$$\text{LDR} \preceq \text{non-increasing HDR } {<}p_2 p_3 \ldots p_T{>} \preceq \text{PLDR}{-}p$$

where $p_t \geq p_{t+1}$ and $p_t \leq p$ for all $t \in \mathcal{T}_{-1}$. The relation $\text{DR}_1 \preceq \text{DR}_2$ indicates that (1) DR_2 is more complex than DR_1; (2) DR_1 is a special case of DR_2; (3) DR_1 solution quality may be similar to that of DR_2 at a lower computational cost.

To investigate this relation, we increase the number of suppliers and customers to 22 and 11, respectively. We increase the dimension of the uncertainty in each stage from 1 to 8: $\boldsymbol{\xi}_t \in \mathbb{R}^8$ for all $t \in \mathcal{T}_{-1}$. Dependence of the uncertain customer demands remain unchanged as in equation (2.13). All uncertain elements within the same stage are independent and each follows a uniform distribution between 0 and 1. There is no stagewise dependence between the uncertain elements. The rest of the computational setting is included in the supplementary material.

In our experiment, the model-based policies are evaluated within a moving horizon simulator (see Figure 2.10). Given an initial inventory, a multistage transportation problem with n_p stages[4] is solved in $t = 1$. There are no distribution decisions at the outset, hence only production decisions are implemented. Afterwards, the inventory of each supplier is updated, the uncertain demand of each customer is realized, and the model-based time grid is shifted to the subsequent stage. For all $t \geq 2$, the procedure is similar except that both production and distribution decisions are now implemented after each re-optimization. The simulator is truncated after the last re-optimized decisions are implemented in $t = T$. For our study, both n_p and T are equal to six.

As shown in Figure 10, we allow all look-ahead models to “see” beyond the simulator evaluation horizon T after the first stage. Note that this is different from a folding horizon procedure in which one always uses a forecast up to and including stage T, but never beyond T. We chose this option because the simulator evaluation horizon T is somewhat arbitrary in practice (e.g., it could be one week, one month, one year, etc.) when, in fact, the actual business problem continues well beyond the evaluation horizon. Indeed, folding horizon approaches tend to yield solutions in which there is no ending inventory because the look-ahead models assume that “the world ends” in stage T. It is mainly for this reason that we chose a moving horizon procedure over a folding horizon procedure. Note that we could have used a shorter planning horizon (i.e., $n_p < T$) for each look-ahead model solve, but the question of whether or not to “see” beyond T would still arise in the later stages of the simulation. In summary, the most important point is that all policies were constructed (or “trained”) in a perfectly consistent manner using the same look-ahead forecast of n_p stages.

[4] The notation n_p is used to indicate that the model and simulator horizons may be different. In our experiment, they are equal.

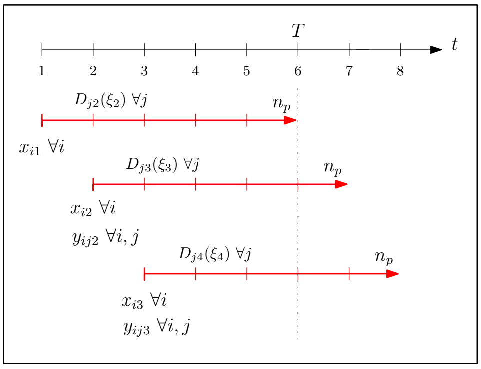


Figure 2.10: The first three iterations of a moving horizon simulator used to evaluate the model-based policies of a multistage stochastic transportation problem.

We use 200 scenarios to evaluate the model-based policies in the simulator where a scenario consists of the realized $\boldsymbol{\xi}_t$ for all $t \in \mathcal{T}_{-1}$. For each scenario, we solve six optimization problems and we compute the simulated profit and the simulation time. We define the simulation time as the sum of the six-optimization solution time.

The breakpoints used to lift the uncertainty set belong to $\mathcal{Z}_{\text{base}} := \{0.25, 0.375, 0.5, 0.625, 0.75\}$. Given $\mathcal{Z}_{\text{base}}$ and the assumption of using the same breakpoints to lift all elements in $\boldsymbol{\xi}_t$ for all stages, the total number of designs, defined by the breakpoints sampled from $\mathcal{Z}_{\text{base}}$, of PLDR-1, PLDR-2 and PLDR-3 are equal to 5, 10 and 10, respectively. We consider further restrictions on the selection of breakpoints. In particular, the difference between the minimum and maximum breakpoint values while lifting with two breakpoints is set to be at least equal to 0.325, while lifting with three breakpoints is only done using the points 0.25, 0.5 and 0.75. As a result, the total PLDR-2 and PLDR-3 designs are reduced to 6 and 1, respectively.

Figure 2.11 illustrates the percentage change in the mean simulated profit with respect to LDR obtained by PLDR-1, PLDR-2 and PLDR-3 clusters as a function of the mean simulation time. A cluster includes all possible designs of a policy based on our aforementioned breakpoint selection heuristics. The difference in the solution quality between LDR and PLDR-3 is close to 14%. Nonetheless, the computational overhead of the latter is significantly higher. PLDR-1 exhibits improved solution quality and a potential for further enhancement remains as shown by the PLDR-2 cluster. To quantify the obtained clusters, box plots are added. The middle mark in a box corresponds to the median, the lower and upper edge marks correspond to the 25^{th} and 75^{th} percentiles, respectively. The red "+" marks correspond to possible outliers in a cluster.

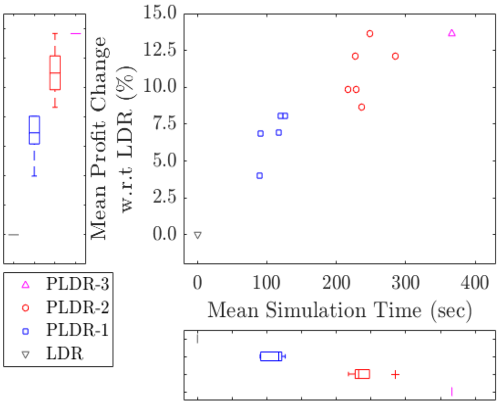


Figure 2.11: Mean profit change with respect to an LDR (in %) generated by PLDRs 1, 2 and 3, within a moving horizon simulator, for a multistage stochastic transportation problem (22 suppliers and 11 customers).

Figure 2.12 illustrates the clusters of six non-increasing HDRs that are grouped into three rows. The clusters of the six non-decreasing HDRs, obtained by inverting the lifting strategy of the non-increasing HDRs, are also shown. For notation purposes, HDR<33100> indicates that three breakpoints are used to lift each element in $\boldsymbol{\xi}_t$ in stages 2 and 3, one breakpoint is used to lift each element in $\boldsymbol{\xi}_t$ in stage 4 and there is no lifting in stages 5 and 6. We follow the same heuristics to select the breakpoint values in each stage. We also assume that if the number of breakpoints used to lift the uncertainty set in different stages is the same, then the set of breakpoints is the same too. This makes the number of HDR<33100> designs equal to five.

Essentially, the results in Figure 2.12 indicate that for any PLDR-p, there exists an attractive non-increasing HDR with a p number of breakpoints in the first stage and an equal or fewer number of breakpoints in the subsequent stages. For example, HDR<10000> and HDR<11000> are two examples that offer computationally less expensive policies in comparison to a PLDR-1 but with better solution quality than an LDR. The HDR<11000> (i.e., less complex) cluster recovers the same solution quality of the PLDR-1 (i.e., more complex) cluster with a small reduction in computational cost. Likewise, HDR<21100> and HDR<21111> are both less complex than PLDR-2, but with a similar solution quality. In particular HDR<21100> offers a clear advantage in terms of computational cost with even some specific designs outperforming the more complex decision rules PLDR-2 and PLDR-3.

Similarly, HDR<31000> and HDR<33100> reduce the average (over the five designs) computational cost of PLDR-3 by 50.89% and 28.66%, respectively, while offering similar solution quality. We also note that three HDR<31000> designs outperform the more complex PLDR-3. These unexpected observations are due to the fact that the simulator is truncated in stage T while the forecasting horizon of the model exceeds T for $t \geq 2$ as shown in Figure 2.10. The model may overfit certain policies by hedging against the future uncertainty beyond T, which is never realized due to the truncation. The additional production units by the suppliers, to react for future uncertainty, ends up as excessive inventory in stage T.

The attractiveness of the non-increasing HDRs in comparison with the corresponding non-decreasing HDRs for the six cases also stands out. The impact of low resolution in early stages is detrimental. Figure 2.12 reveals that, despite having a lower computational cost, non-decreasing HDRs are always inferior to the next complex PLDR (e.g., HDR<11222> and PLDR-2). For the HDRs where there is no lifting in early stages, the solution quality is no better than an LDR.

Corroborating the outcome in Figure 2.6, the look-ahead model in this experiment was not able to foretell the computational benefits of non-increasing HDRs. Instead, the model suggests that non-increasing HDRs are not attractive. A comparison between the model- and simulator-based results is shown in the supplementary material.

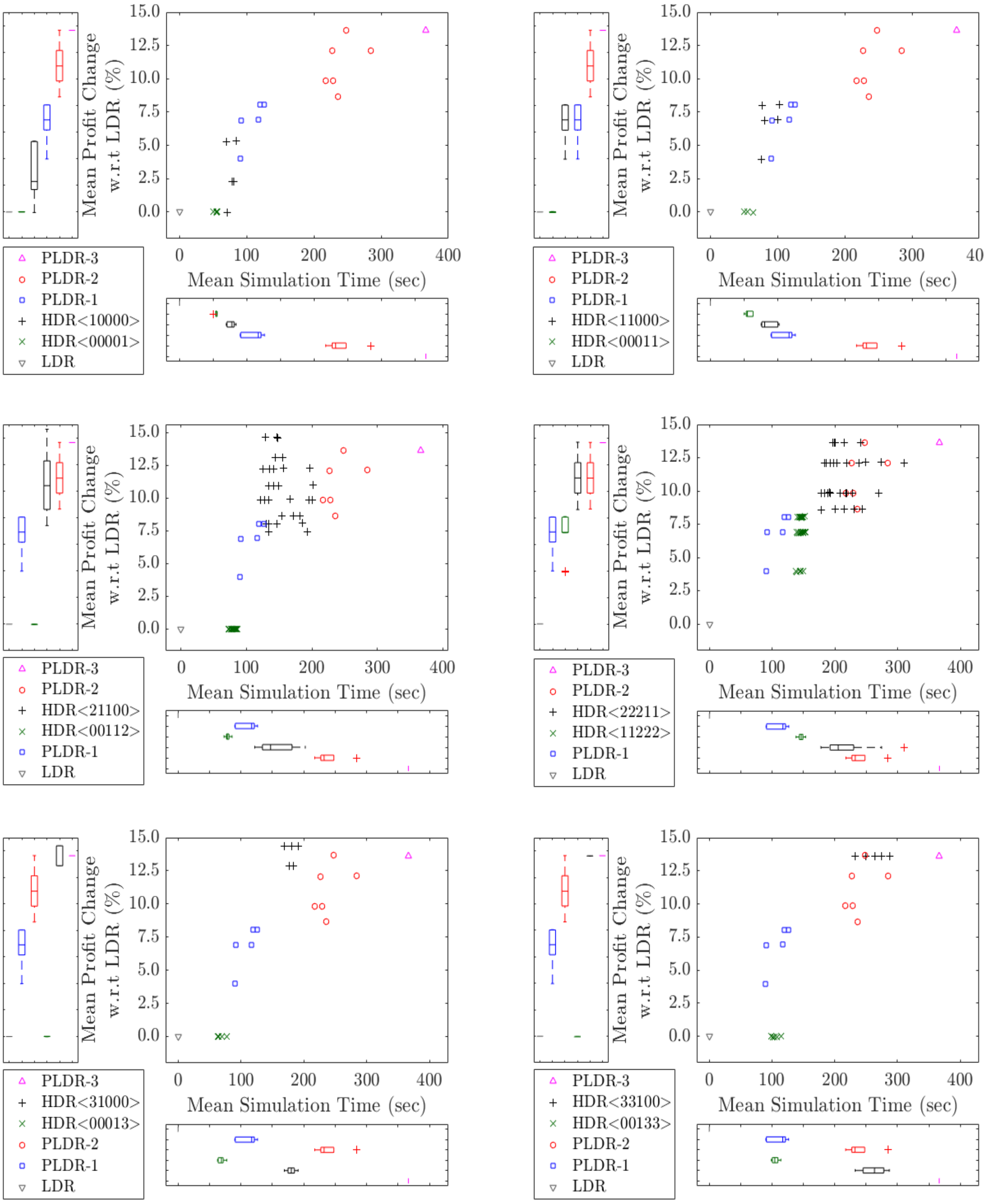


Figure 2.12: Mean profit change with respect to an LDR (in %) generated by six non-increasing HDRs, within a moving horizon simulator, for a multistage stochastic transportation problem (22 suppliers and 11 customers).

The nature of the non-increasing lifting strategy of an HDR is key in dictating its flexibility. A poorly designed non-increasing HDR loses its computational benefits. Two poor design choices are demonstrated in Figure 2.13. The HDR<33322> cluster is an example where lifting in late stages is unnecessary, while the HDR<55320>[5] cluster is an example where the solution quality

[5] Lifting with five breakpoints is done using all elements in $\mathcal{Z}_{\text{base}}$.

obtained from over-lifting the uncertainty in early stages does not justify the computational overhead.

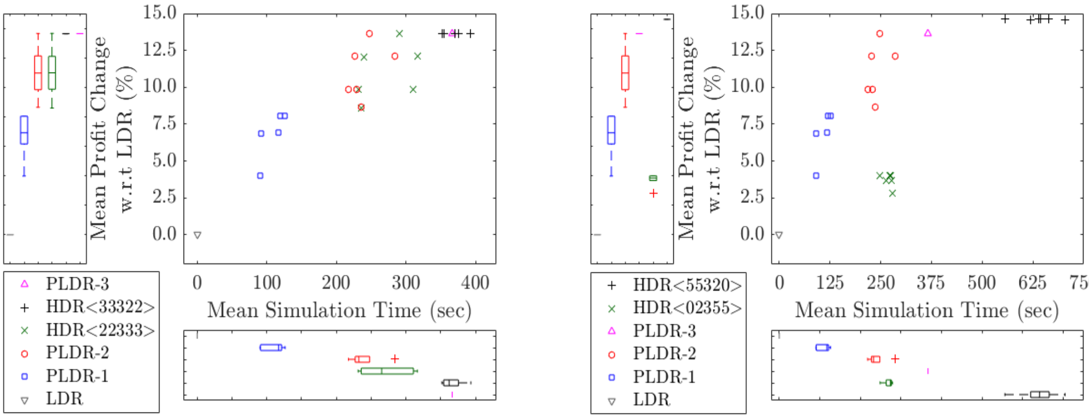


Figure 2.13: Poorly designed non-increasing HDRs for a multistage stochastic transportation problem (22 suppliers and 11 customers).

2.5 Conclusion

Since their recent inception, piecewise linear decision rules have received increasing attention in the stochastic and robust optimization communities. This is mainly due to the flexibility and solution quality improvements that they provide while maintaining a tractable linear structure closely resembling that used for LDRs. However, the increase in solution quality comes at the expense of a significant computational burden for large-scale problems. This study first provides an unexpected result in which a less complex decision rule (e.g., LDR) is *superior* to a more complex decision rule (e.g., PLDR with a single breakpoint) when assessed within a given simulator, a result that is not due to the presence of multiple robust optima. The comparison is done using a hyper-rectangle uncertainty set where there is an exact tractable representation of the convex hull of the lifted uncertainty set. We also showed that a model may not be reliable in predicting the actual improvement in solution quality by a complex decision rule. These findings highlight the need for assessing the quality of the look-ahead policies within a given simulator, instead of just relying on the look-ahead model's objective function value. Then, the study emphasizes the concept of implementing hybrid decision rules as a promising direction to mitigate the increased computational burden in high dimensional multistage adaptive optimization problems. The main aspect of HDRs explored is the lifting strategies or the axial combinations of the LDR and PLDRs where it is empirically illustrated that having higher uncertainty resolution (i.e., more linear pieces) in early stages is more important than having it in late stages. Though the empirical observation is verified via a single numerical example (i.e., mutlitstage transportation problem), its validity for other problems is supported by the logic that decisions in the short term have higher time-value than the decision made in the long term. Further, the impact of inferior decisions in the short term may limit the potential of generating high quality decisions on the long term as the latter depends on the former.

Several open questions are ripe for additional research. First, the flexibility of a non-increasing HDR highly depends on the design of the lifting strategy. Currently, we do not provide a systematic way to design an optimal lifting strategy. Second, removing the assumptions on the general model will increase the complexity of the problem. Uncertain cost coefficients lead to quadratic uncertainty terms in the objective function. An uncertain recourse matrix generates nonlinear uncertain constraints whose tractable stochastic counterparts are more challenging to derive. Third, the hybrid decision rule approach can be extended to conditional value at risk and robust multistage adaptive optimization problems. Fourth, it would be interesting to explore HDRs in conjunction with the Fourier-Motzkin Elimination (FME) procedure tailored to adaptive optimization problems introduced by Zhen et al. (2018). Using this approach, one can systematically remove adaptive decision variables via FME before attempting to restrict any remaining adaptive decision variables to adhere to some tractable decision rules. Of course, the FME approach comes at a cost of having to append possibly a quadratic number of constraints for each adaptive decision variable removed. The authors combat this challenge by employing a heuristic procedure to identify and remove redundant constraints. Our findings suggest that there may be greater benefit to removing adaptive decision variables in earlier stages and approximating the remaining with HDRs. Lastly, a comparative study between scenario-based stochastic programming via sample average approximation, stochastic dual dynamic programming, approximate dynamic programming/reinforcement learning, and decision rule-based solution methods might lead to interesting insights.

Bibliography

Ang, M., Lim, Y. F. & Sim, M. (2012), 'Robust storage assignment in unit-load warehouses', *Management Science* **58**(11), 2114–2130.

Arslan, A. N. & Papageorgiou, D. J. (2017), 'Bulk ship fleet renewal and deployment under uncertainty: A multi-stage stochastic programming approach', *Transportation Research Part E: Logistics and Transportation Review* **97**, 69–96.

Bakkehaug, R., Eidem, E. S., Fagerholt, K. & Hvattum, L. M. (2014), 'A stochastic programming formulation for strategic fleet renewal in shipping', *Transportation Research Part E: Logistics and Transportation Review* **72**, 60–76.

Bampou, D. & Kuhn, D. (2011), Scenario-free stochastic programming with polynomial decision rules, *in* 'Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on', IEEE, pp. 7806–7812.

Baptista, S., Gomes, M. I. & Barbosa-Povoa, A. P. (2012), A two-stage stochastic model for the design and planning of a multi-product closed loop supply chain, *in* 'Computer Aided Chemical Engineering', Vol. 30, Elsevier, pp. 412–416.

Ben-Tal, A. & Den Hertog, D. (2011), Immunizing conic quadratic optimization problems against implementation errors, Technical report, Tilburg Univeristy.

Ben-Tal, A., El Ghaoui, L. & Nemirovski, A. (2009), *Robust optimization*, Princeton University Press.

Ben-Tal, A., El Housni, O. & Goyal, V. (2018), 'A tractable approach for designing piecewise affine policies in two-stage adjustable robust optimization', *Mathematical Programming* pp. 1–46.

Ben-Tal, A., Golany, B., Nemirovski, A. & Vial, J.-P. (2005), 'Retailer-supplier flexible commitments contracts: A robust optimization approach', *Manufacturing & Service Operations Management* **7**(3), 248–271.

Ben-Tal, A., Goryashko, A., Guslitzer, E. & Nemirovski, A. (2004), 'Adjustable robust solutions of uncertain linear programs', *Mathematical Programming* **99**(2), 351–376.

Bertsimas, D. & Georghiou, A. (2015), 'Design of near optimal decision rules in multistage adaptive mixed-integer optimization', *Operations Research* **63**(3), 610–627.

Bertsimas, D., Iancu, D. A. & Parrilo, P. A. (2010), 'Optimality of affine policies in multistage robust optimization', *Mathematics of Operations Research* **35**(2), 363–394.

Bertsimas, D., Iancu, D. A. & Parrilo, P. A. (2011), 'A hierarchy of near-optimal policies for multistage adaptive optimization', *IEEE Transactions on Automatic Control* **56**(12), 2809–2824.

Bertsimas, D. & Thiele, A. (2006), 'Robust and data-driven optimization: Modern decision-making under uncertainty', *INFORMS tutorials in operations research: models, methods, and applications for innovative decision making* **3**.

Bertsimas, D. & Tsitsiklis, J. N. (1997), *Introduction to linear optimization*, Vol. 6, Athena Scientific Belmont, MA.

Beuchat, P. N., Warrington, J., Summers, T. H. & Morari, M. (2016), 'Performance bounds for look-ahead power system dispatch using generalized multistage policies', *IEEE Transactions on Power Systems* **31**(1), 474–484.

Birge, J. R. & Louveaux, F. (2011), *Introduction to stochastic programming*, Springer Science & Business Media.

Bodur, M. & Luedtke, J. R. (2018), 'Two-stage linear decision rules for multi-stage stochastic programming', *Mathematical Programming* pp. 1–34.

Braathen, J. & Eriksrud, A. L. (2013), Hydropower bidding using linear decision rules, Master's thesis, Institutt for industriell økonomi og teknologiledelse.

Bruni, M. E., Beraldi, P. & Conforti, D. (2015), 'A stochastic programming approach for operating theatre scheduling under uncertainty', *IMA Journal of Management Mathematics* **26**(1), 99–119.

Chen, X., Sim, M. & Sun, P. (2007), 'A robust optimization perspective on stochastic programming', *Operations Research* **55**(6), 1058–1071.

Chen, X., Sim, M., Sun, P. & Zhang, J. (2008), 'A linear decision-based approximation approach to stochastic programming', *Operations Research* **56**(2), 344–357.

Chen, X. & Zhang, Y. (2009), 'Uncertain linear programs: Extended affinely adjustable robust counterparts', *Operations Research* **57**(6), 1469–1482.

Daryalal, M., Bodur, M. & Luedtke, J. R. (2020), 'Lagrangian dual decision rules for multistage stochastic mixed integer programming', *arXiv preprint arXiv:2001.00761* .

de Ruiter, F. J., Brekelmans, R. C. & den Hertog, D. (2016), 'The impact of the existence of multiple adjustable robust solutions', *Mathematical Programming* **160**(1-2), 531–545.

Dillon, M., Oliveira, F. & Abbasi, B. (2017), 'A two-stage stochastic programming model for inventory management in the blood supply chain', *International Journal of Production Economics* **187**(Supplement C), 27–41. ID: 271692.

Fisher, M., Ramdas, K. & Zheng, Y.-S. (2001), 'Ending inventory valuation in multiperiod production scheduling', *Management Science* **47**(5), 679–692.

Garstka, S. J. & Wets, R. J.-B. (1974), 'On decision rules in stochastic programming', *Mathematical Programming* **7**(1), 117–143.

Gauvin, C., Delage, E. & Gendreau, M. (2017), 'Decision rule approximations for the risk averse reservoir management problem', *European Journal of Operational Research* **261**(1), 317 – 336.

Georghiou, A., Wiesemann, W. & Kuhn, D. (2015), 'Generalized decision rule approximations for stochastic programming via liftings', *Mathematical Programming* **152**(1-2), 301–338.

Goh, J. & Sim, M. (2010), 'Distributionally robust optimization and its tractable approximations', *Operations research* **58**, 902–917.

Grass, E. & Fischer, K. (2016), 'Two-stage stochastic programming in disaster management: A literature survey', *Surveys in Operations Research and Management Science* **21**(2), 85–100. ID: 280252.

Guigues, V., Juditski, A. & Nemirovski, A. (2020), 'Constant depth decision rules for multistage optimization under uncertainty', *arXiv preprint arXiv:2005.13387* .

Hanasusanto, G. A., Kuhn, D. & Wiesemann, W. (2015), 'K-adaptability in two-stage robust binary programming', *Operations Research* **63**(4), 877–891.

Iancu, D. A. & Trichakis, N. (2013), 'Pareto efficiency in robust optimization', *Management Science* **60**(1), 130–147.

Jin, J. & Xu, Y. (2018), Segregated linear decision rules for distributionally robust control with linear dynamics and quadratic cost, *in* '2018 Annual American Control Conference (ACC)', IEEE, pp. 2687–2694.

Krokhmal, P., Palmquist, J. & Uryasev, S. (2002), 'Portfolio optimization with conditional value-at-risk objective and constraints', *Journal of risk* **4**, 43–68.

Kuhn, D., Wiesemann, W. & Georghiou, A. (2011), 'Primal and dual linear decision rules in stochastic and robust optimization', *Mathematical Programming* **130**(1), 177–209.

Lappas, N. H. & Gounaris, C. E. (2016), 'Multi-stage adjustable robust optimization for process scheduling under uncertainty', *AIChE Journal* **62**(5), 1646–1667.

Li, J., Misener, R. & Floudas, C. A. (2012), 'Scheduling of crude oil operations under demand uncertainty: A robust optimization framework coupled with global optimization', *AIChE Journal* **58**(8), 2373–2396.

Li, Z. & Ierapetritou, M. (2008), 'Process scheduling under uncertainty: Review and challenges', *Computers and Chemical Engineering* **32**(4-5), 715–727.

Lorca, A., Sun, X. A., Litvinov, E. & Zheng, T. (2016), 'Multistage adaptive robust optimization for the unit commitment problem', *Operations Research* **64**(1), 32–51.

Munoz-Alvarez, D., Bitar, E., Tong, L. & Wang, J. (2014), Piecewise affine dispatch policies for economic dispatch under uncertainty, *in* 'PES General Meeting— Conference & Exposition, 2014 IEEE', IEEE, pp. 1–5.

Pan, L., Housh, M., Liu, P., Cai, X. & Chen, X. (2015), 'Robust stochastic optimization for reservoir operation', *Water Resources Research* **51**(1), 409–429.

Peidro, D., Mula, J., Poler, R. & Lario, F.-C. (2009), 'Quantitative models for supply chain planning under uncertainty: a review', *The International Journal of Advanced Manufacturing Technology* **43**(3-4), 400–420.

Pereira, M. V. & Pinto, L. M. (1991), 'Multi-stage stochastic optimization applied to energy planning', *Mathematical programming* **52**(1-3), 359–375.

Powell, W. B. (2009), 'What you should know about approximate dynamic programming', *Naval Research Logistics (NRL)* **56**(3), 239–249.

Powell, W. B. (2011), *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, Vol. 842, John Wiley & Sons.

Powell, W. B. (2014), *Clearing the jungle of stochastic optimization*, Informs, chapter Chapter 4, pp. 109–137.

Revelle, C., Joeres, E. & Kirby, W. (1969), 'The linear decision rule in reservoir management and design: 1, development of the stochastic model', *Water Resources Research* **5**(4), 767–777.

Rocha, P. & Kuhn, D. (2012), 'Multistage stochastic portfolio optimisation in deregulated electricity markets using linear decision rules', *European Journal of Operational Research* **216**(2), 397–408.

Rockafellar, R. T. (2007), 'Coherent approaches to risk in optimization under uncertainty', *Tutorials in Operations Research* pp. 38–61.

Rockafellar, R. T. & Uryasev, S. (2000), 'Optimization of conditional value-at-risk', *Journal of risk* **2**, 21–42.

Sahinidis, N. V. (2004), 'Optimization under uncertainty: state-of-the-art and opportunities', *Computers & Chemical Engineering* **28**(6-7), 971–983.

See, C.-T. & Sim, M. (2010), 'Robust approximation to multiperiod inventory management', *Operations research* **58**(3), 583–594.

Shapiro, A. (2011), 'Analysis of stochastic dual dynamic programming method', *European Journal of Operational Research* **209**(1), 63–72.

Shapiro, A., Dentcheva, D. & Ruszczynski, A. (2009), 'Lectures on stochastic programming', *MPS-SIAM series on optimization* **9**, 3.

Vayanos, P., Kuhn, D. & Rustem, B. (2011), Decision rules for information discovery in multi-stage stochastic programming, *in* '2011 50th IEEE Conference on Decision and Control and European Control Conference', pp. 7368–7373.

Verderame, P. M., Elia, J. A., Li, J. & Floudas, C. A. (2010), 'Planning and scheduling under uncertainty: a review across multiple sectors', *Industrial & engineering chemistry research* **49**(9), 3993–4017.

Verderame, P. M. & Floudas, C. A. (2009), 'Operational planning of large-scale industrial batch plants under demand due date and amount uncertainty. i. robust optimization framework', *Industrial and Engineering Chemistry Research* **48**(15), 7214–7231.

Yanıkoğlu, İ., Gorissen, B. L. & den Hertog, D. (2019), 'A survey of adjustable robust optimization', *European Journal of Operational Research* **277**(3), 799–813.

Zhang, Q., Morari, M. F., Grossmann, I. E., Sundaramoorthy, A. & Pinto, J. M. (2016), 'An adjustable robust optimization approach to scheduling of continuous industrial processes providing interruptible load', *Computers and Chemical Engineering* **86**, 106–119.

Zhang, X., Georghiou, A. & Lygeros, J. (2015), Convex approximation of chance-constrained MPC through piecewise affine policies using randomized and robust optimization, *in* 'Decision and Control (CDC), 2015 IEEE 54th Annual Conference on', IEEE, pp. 3038–3043.

Zhen, J., den Hertog, D. & Sim, M. (2018), 'Adjustable robust optimization via fourier–motzkin elimination', *Operations Research* **66**(4), 1086–1100.

Zou, J., Ahmed, S. & Sun, X. A. (2019), 'Stochastic dual dynamic integer programming', *Mathematical Programming* **175**(1-2), 461–502.

Chapter 3

Steelmaking and Continuous Casting Scheduling under Uncertainty

3.1 Introduction

Steel is a major component in most industries including but not limited to transportation, construction, manufacturing and defense. It is produced via three consecutive processes: (i) iron ore mining and processing, (ii) steel-making and continuous-casting (SCC) and (iii) rolling and finishing mills. In terms of scheduling operations, the SCC process is the bottleneck and has attracted most of the research initiatives (Iglesias Escudero et al. 2019). It is a complex hybrid flow shop problem with strict production requirements (Ruiz & Vázquez-Rodríguez 2010). Despite the complexity of the process, manual generation of the schedule by experienced schedulers used to be the norm in the industry with the aid of computerized-based simulator to compare different feasible schedules (Pacciarelli & Pranzo 2004). This has started to change in the last decade as solution techniques offering higher quality schedules have been progressively developed (Harjunkoski et al. 2014, Iglesias Escudero et al. 2019).

The generated schedule must satisfy the production requirements in the face of unexpected disruptions. Uncertainties in the SCC are attributed to four main factors: (i) processing/operation time, (ii) machine availability, (iii) product specifications, and (iv) orders/demand.

Solution techniques for scheduling under uncertainty can be classified into two groups: proactive scheduling methods and reactive scheduling method. Proactive scheduling considers the worst case uncertainty to avoid future changes to the initial schedule in the event of a disruption. Reactive scheduling reacts to each unexpected disruption by adjusting the initial schedule or generating a new schedule from scratch. Gupta & Maravelias (2016) and Gupta et al. (2016) suggested that rescheduling should not necessarily be triggered by only a disruption, rather it should be done periodically to improve the quality of the implemented schedule. Though not very common in the literature yet, a hybrid scheduling approach may be attractive (Iglesias Escudero et al. 2019). It immunizes the schedule against part of the uncertainty set (i.e., less conservative solution) and adjusts the current schedule whenever the realized uncertainty falls outside the set (i.e., fewer changes to the schedule) (Chaari et al. 2014).

For a general process scheduling under uncertainty review, please refer to the work of Li & Ierapetritou (2008). As for the SCC scheduling problem, both proactive and reactive methods have been explored in the literature, Yu (2013) developed a forecasting method to predict infeasibility in the initial schedule due to uncertainty in operation time, then adjusted only the affected ladles accordingly to minimize the number of changes to the initial schedule. Jiang et al. (2016) used a non-parametric machine learning technique (i.e., Gaussian process regression) to predict a characteristic index that immune casts' due date in the caster unit against processing time uncertainty. Based on the predicted value, dynamic optimization is used to generate a soft schedule and dispatching rules or heuristics are then used to react for other disruptions like machine breakdown. Similar work is presented in Jiang et al. (2017) where the soft-form schedule is generated via two phases using continuous/discrete estimation distribution algorithms. Mori & Mahalec (2017) used a Bayesian network model to predict the distribution of the unit processing time given the customers' demand and operational parameters. Given the expected value of the processing time, the two-level scheduling problem is solved using a simulated annealing algorithm. Kammammettu & Li (2018) implemented affine adaptive stochastic and robust optimization to solve the SCC scheduling problem under processing time uncertainty.

Rong & Lahdelma (2008) and Yu et al. (2009) used fuzzy programming to address the uncertainty in the raw materials' composition and operation time in the SCC process, respectively. Gerardi et al. (2013) addressed the uncertainty in the composition of the raw material via a two-stage scenario based stochastic programming method. The first stage decisions include the raw material purchasing amounts, whereas the second stage decisions include operational parameters. Hong & Wang (2014) addressed product specification uncertainties in the cold rolling stage via robust optimization and solved the non-linear problem using a particle swarm algorithm. Sun et al. (2015) used stochastic dynamic programming to address the uncertainty in the number of machine runs (i.e., uncertainty in final product specification). Noshadravan et al. (2017) used fuzzy programming to address uncertainty in the raw materials' composition. They considered right-skewed uncertainty distribution which is more common in real practice.

Ye et al. (2014) addressed demand uncertainty, in the form of the number of ladles within a cast, using robust optimization and two-stage scenario-based stochastic optimization with scenario reduction technique. Schedules generated from both solution methods are shown to have comparable performance.

Worapradya & Thanakijkasem (2010) proposed a worst case performance approach to address both machine failure and processing time uncertainties. Historical data were fitted into beta and log-normal distributions and a genetic algorithm is used. Wang et al. (2017) introduced a proactive multi-objective schedule for the rolling production phase with stochastic machine breakdown and controllable processing time. It is one of the few works in the literature that consider controlling the processing time.

In this work, we focus on dealing with the processing time uncertainty in the SCC scheduling problem. Uncertainty in the scheduling problem is classified based on its dependence upon the problem's decisions. Exogenous uncertainty is a term used for those whose time of availability or revelation does not depend on any decision. For example, the observation of an uncertain demand or a rush order is not related to the assignment of a specific operation to a specific unit. However, that same decision does dictate the observability of the uncertain processing time of the cast in the unit. Uncertainties whose observability is governed by decisions made in the problem are called endogenous (Jonsbraten et al. 1998).

Goel & Grossmann (2004) and Goel & Grossmann (2006) proposed a hybrid mixed-integer disjunctive programming model to address endogenous uncertainty in a multi-stage scenario-based stochastic optimization problem. The superstructure of the scenario tree is constructed taking all the possible outcomes of the endogenous uncertainty and the conditional non anticipativity constraints (NACs) are modeled in disjunctions. Using model-specific properties, lagrangian duality branch and bound algorithm are used to solve the problem. Gupta & Grossmann (2011) extended the previous work by adding a new property that characterized this class of problems and proposed new solution strategies using the fact that (i) most NACs are not active at optimality (i.e., NACs relaxation strategies) and (ii) the scenario tree can be decomposed into independent sub-problems (i.e., Lagrangian decomposition strategy). Apap & Grossmann (2017) further developed the solution method by considering both exogenous and endogenous uncertainties in the scenario tree. They proposed a sequential reduction technique to eliminate redundant NACs and reduce the prohibitive model size.

In the robust optimization community, the specific consideration of endogenous uncertainty is very recent. Poss (2014) introduced a budget uncertainty set defined by the problem's decision variables. The new set is shown to reduce the price-of-robustness when compared to the classical budget uncertainty set provided by Bertsimas & Sim (2004) at the expense of a small increase in computational cost. Lappas & Gounaris (2016) introduced decision-dependent uncertainty sets (DDUS) for global event-based affine adaptive robust optimization problems. The DDUS is defined by affine inequalities where the coefficient of an uncertain parameter is either a constant or a binary decision that dictates the observability of the parameter. Lappas & Gounaris (2018) introduced a general formulation for the DDUS and illustrated the improvement in performance in comparison to the static uncertainty set. Nohadani & Sharma (2018) introduced DDUSs in the context of static robust optimization.

The two competing objectives for scheduling SCC under uncertainty are solution quality and number/extent of changes made to the initial schedule. Furthermore, the use of proactive scheduling methods is more recent and less common than reactive scheduling methods. In particular, when addressing the uncertain processing time, the proactive solution methods do not model the gradual realization of the uncertainty across processing units. In most cases, prediction methods are used to estimate the uncertain parameters (i.e., deterministic) and heuristics are used to react for the error in the prediction when the real value of the uncertain parameter is observed. Based on these existing gaps in the literature, the contributions of this chapter are:

1. We present a robust multistage adaptive optimization model for the steel-making and continuous-casting process under endogenous processing time uncertainty. The static (i.e., "here and now") decisions are the allocation of casts to processing units and their sequence in each unit. The recourse (i.e.,"wait and see") decisions are the ladle start and finish processing time in each processing stage.

2. We devise a systematic method to construct decision rules that satisfy the non anticipativity property using "logical sequencing constraints" (LSCs) for a unit-specific event-based (USEB) continuous time formulation. LSCs are used to infer temporal correlations within the unit-specific time grids. To the best of our knowledge, no previous attempt to extend the use of multistage adaptive robust optimization in USEB scheduling formulation has appeared in the literature.

3. We emphasize the significance of computing high quality cast allocation and sequencing (i.e., "here and now") decisions. We do so by comparing the worst case performance of deterministic reactive and adaptive robust proactive schedules. Despite the former method's opportunity to re-optimize the "wait and see" decisions, it does not incorporate any uncertainty information in the solution of the "here and now" decisions. This hinders the ability of the re-optimized recourse decisions to improve the overall schedule performance.

4. To compare deterministic reactive, adaptive robust proactive, and adaptive robust hybrid scheduling, we assess each approach using two competing objectives: solution quality and the number of rescheduling events. Using three different distributions, we show that adaptive robust hybrid schedules that are robust over a small uncertainty set are superior to the deterministic reactive schedule. Further, we demonstrate how knowledge of the distribution provides insightful guidelines in determining a preferable scheduling method. This contribution addresses an existing gap in the literature regarding the added value from hybrid scheduling methods (Iglesias Escudero et al. 2019).

The remainder of the chapter is organized as follows. Section 2 presents a general description of the SCC process, specifies the propositions used in representing the process network and states the problem statement. Section 3 briefly describes the unit-specific event-based formulation and presents the deterministic scheduling model. Section 4 introduces endogenous processing time uncertainty, static/adaptive decisions, the robust adaptive multi-stage SCC problem and describes the mathematical tools used to construct the linear decision rules that satisfy the non-anticipativity property. Section 5 includes the three computational experiments conducted in this work. The chapter is concluded in section 6.

3.2 Steel-making and continuous casting problem

In this section, a general description of the steel-making and continuous casting process is introduced followed by the problem statement.

3.2.1 Process description

The SCC process transforms iron coke/scraps into steel casts of specific composition and dimension. It operates in batch mode where a batch of molten steel is termed a *ladle*. Ladles with similar production requirements are grouped in a *cast*.

Figure 3.1 demonstrates a general schematic overview of the SCC process. At the outset, a mix of iron scrap and coke is melted in an electric arc furnace (EAF) unit. After which the ladle is processed in an argon oxygen decarburization (AOD) unit where alloy metals are added to attain a specific composition. The processing time of a ladle in EAF and AOD units varies from 70 to 80 minutes. Next, the ladle is further refined in a ladle furnace (LF) unit for around 30 minutes before being processed in a continuous caster (CC) unit. The casting time of a ladle ranges from 50 to 70 minutes. Ladles grouped in the same cast *must* be continuously processed in the CC unit without any interruptions.

Finite intermediate storage (FIS) is available before the AOD and CC processing units. While there is no restriction on the storage duration before the AOD unit, the storage duration of a ladle before the CC unit must not exceed 10 minutes. Otherwise, the ladle will cool down and be deemed inappropriate for casting. This production requirement is also known as the perishability constraint (Pacciarelli & Pranzo 2004).

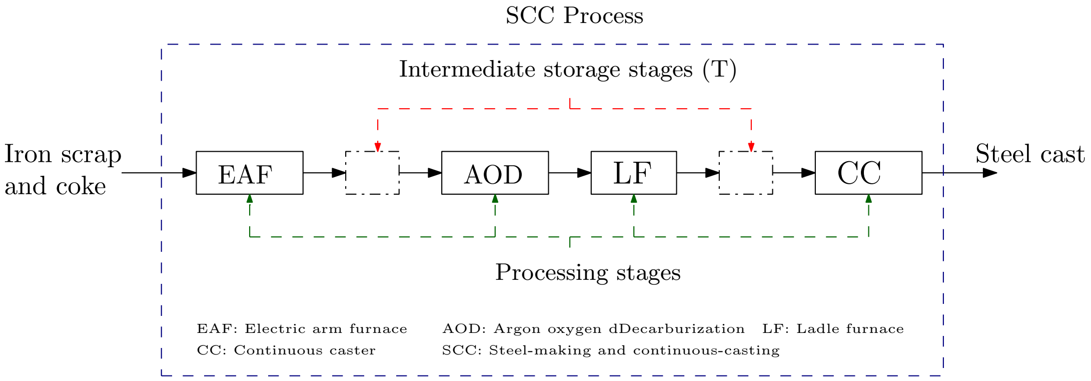

SCC Process
Intermediate storage stages (T)
Iron scrap and coke
EAF
AOD
LF
CC
Steel cast
Processing stages
EAF: Electric arm furnace
AOD: Argon oxygen dDecarburization
LF: Ladle furnace
CC: Continuous caster
SCC: Steel-making and continuous-casting

Figure 3.1: Steel-making and continuous casting process.

3.2.2 Process representation used for modelling

Li, Xiao, Tang & Floudas (2012) implements a sequential representation for the SCC scheduling problem. Batch sizing and material utilization constraints are not included in the scheduling problem. The propositions made in constructing the process standard network/structure are given below:

1. Each of the parallel EAF, AOD and CC units corresponds to a single processing stage. Each unit has a capacity of one ladle.

2. The FIS policy is converted to a "no intermediate storage" (NIS) policy by modelling a storage unit as a processing unit with *zero* processing time (Romero et al. 2004) .

3. A storage unit with a capacity of n^{T}-ladles before the AOD processing stage is represented by n^{T}-serial storage units. Each unit has a capacity of one ladle and corresponds to a distinct processing stage.

4. The LF processing stage with n^{LF}-parallel units, each with a capacity of one ladle and equal processing time (i.e., PT), are represented as n^{LF}-serial LF units. Each unit has a capacity of one ladle and a processing time equivalent to $\frac{1}{n^{\mathrm{LF}}}PT$.

5. There is always a single storage unit before the CC processing stage.

Figure 3.2 illustrates a general representation of the process network used for the SCC scheduling. Li, Xiao, Tang & Floudas (2012) claim that modelling LF units in series does not affect the correctness of the model. As explained in Pacciarelli & Pranzo (2004), this is justified by the assumption of smaller ladle processing time in an LF unit in comparison to that in AOD and CC units. They also considered LF units as high temperature buffers before the CC processing stage. We agree that this assumption does not affect the correctness of the model, but we emphasize that it is implemented to increase the flexibility of the model to satisfy the production requirements.

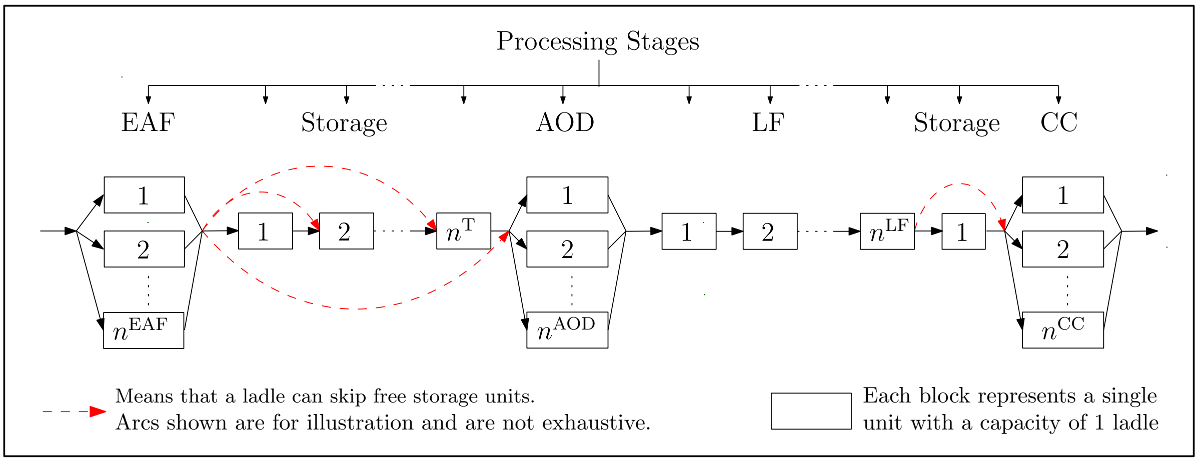

Processing Stages
EAF
Storage
AOD
LF
Storage
CC
1
2
n^{EAF}
1
2
n^{T}
1
2
n^{AOD}
1
2
n^{LF}
1
1
2
n^{CC}
Means that a ladle can skip free storage units.
Arcs shown are for illustration and are not exhaustive.
Each block represents a single unit with a capacity of 1 ladle

Figure 3.2: Steel-making and continuous casting scheduling process network.

The objective of the scheduling problem is to minimize the total makespan. It is assumed that the ladles-cast assignments are pre-defined and the nominal processing times for each ladle is known. The decisions to be optimized are the allocation of casts to the units, the sequencing of casts in each unit and the start and finish processing time for all ladles in each processing stage.

3.3 Deterministic scheduling problem

Scheduling problems are either modelled using a discrete time or a continuous time representation. In the former, time grid is fixed before the solution and time stages where a task may start and finish are known. In the latter, time stages where a task may start are only known after the optimal schedule is obtained. The continuous time representation used for the SCC scheduling problem in Li, Xiao, Tang & Floudas (2012) is next introduced followed by the deterministic mathematical model.

3.3.1 Unit-specific event-based continuous time formulation

Shaik et al. (2006) classified continuous time models into three main classes: (i) unit-specific event-based (USEB), (ii) global event-based and (GEB) (iii) slot-based formulations. In the first two formulations, the model optimizes the start and finish timing decisions of a task, whereas in the last formulation the duration of a time slot is the decision to be optimized. A global event-based formulation adopt a single time grid for all units. This provides some information about the temporal correlation of tasks across different units. On the other hand, a unit-specific event-based formulation implements a unit-specific time grid. This feature has been shown to offer superior performance in the deterministic setting and is used by Li, Xiao, Tang & Floudas (2012) to model the SCC deterministic scheduling problem.

Figure 3.3 illustrates a schedule for 4 ladles across 3 units using USEB and GEB formulations. The number of event points required for the USEB formulation is half the number of events needed for the GEB formulation (i.e., 4 instead of 8). In an extensive comparison study, Shaik et al. (2006) demonstrated the significant reduction in computational cost and superior solution quality offered by the USEB formulation in comparison to GEB and other continuous time formulations.

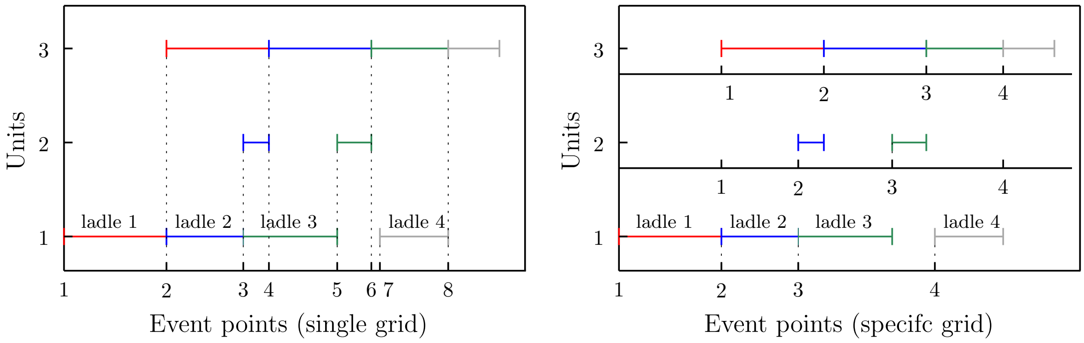


Figure 3.3: A schematic schedule of 4 ladles and 3 units using global (left) and unit-specific (right) event-based formulations.

3.3.2 Mathematical model

The SCC deterministic scheduling optimization model in equation (3.1) is adapted from Li, Xiao, Tang & Floudas (2012) with few modifications. The release and due dates are not considered in this work and the tightening constraints are not included. The nomenclature is presented in Appendix B.1.

Equation (3.1b) depicts the total makespan being minimized. Equations (3.1c) and (3.1d) dictate that only one cast can start processing at an event point. Equation (3.1e) allocates one cast to one processing unit in each processing stage. Equation (3.1f) states that at most one cast is processed on a unit in one event point. Equations (3.1g) and (3.1h) describe the logic relation between the binary variables.

The events' sequencing constraints in each processing unit are given by equations (3.1i) and (3.1j). The equality sign in (3.1j) guarantees the continuous casting requirement of all ladles within the same cast. Similarly, the ladles' sequencing constraints across processing stages are described by equations (3.1k) and (3.1l). The equality sign in (3.1l) reflects the "zero wait" (ZW) policy implemented for the EAF, AOD and CC processing units. The policy states that a ladle has to be immediately transferred out from the processing unit as soon as it finishes processing.

The processing order of ladles within the same cast is governed by equation (3.1m). The perishability constraint is given by equation (3.1n) where P_j^{perish} is equal to 10 min.

Equation (3.1o) is one of the two sequencing constraints that characterize a unit-specific event-based formulation. It prevents any overlapping of tasks processed in two consecutive event points in the same unit. For this specific problem, there is a setup time requirement for the CC units (i.e., $PT_j^{\text{setup}} = 60$ min). For the remaining units, setup time is not required (i.e., $PT_j^{\text{setup}} = 0$ min). Equation (3.1p) defines the second sequencing constraint where the start time of a ladle i in processing stage $s + 1$ is equal to the finishing time of the same ladle i in processing stage s. At last, equations (3.1q) and (3.1r) are the big-M formulation constraints which couple the timing decisions ($t_{i,s}^{\text{s}}$, $t_{j,n}^{\text{s}}$) where $i \in \mathcal{I}^{\text{first}}$ and ($t_{i,s}^{\text{f}}$, $t_{j,n}^{\text{f}}$) where $i \in \mathcal{I}^{\text{last}}$, respectively.

$$\min \quad MS \tag{3.1a}$$

$$\text{s.t.} \quad MS \geq t^{\mathrm{f}}_{j,n} \qquad \forall j \in \mathcal{J}_S,\ n = N \tag{3.1b}$$

$$\sum_{c} x_{c,n} = 1 \qquad \forall n \tag{3.1c}$$

$$\sum_{n} x_{c,n} = 1 \qquad \forall c \tag{3.1d}$$

$$\sum_{j \in \mathcal{J}_s \cap \mathcal{J}_c} y_{c,j} = 1 \qquad \forall c, s \tag{3.1e}$$

$$\sum_{c \in \mathcal{C}_j} z_{c,j,n} \leq 1 \qquad \forall j, n \tag{3.1f}$$

$$\sum_{n} z_{c,j,n} = y_{c,j} \qquad \forall j, c \in \mathcal{C}_j \tag{3.1g}$$

$$\sum_{j \in \mathcal{J}_c} z_{c,j,n} \leq S x_{c,n} \qquad \forall c, n \tag{3.1h}$$

$$t^{\mathrm{f}}_{j,n} \geq t^{\mathrm{s}}_{j,n} + \sum_{c \in \mathcal{C}_j} \sum_{i \in \mathcal{I}_c} \overline{PT}_{i,j} z_{c,j,n} \qquad \forall s < S, j \in \mathcal{J}_s, n \tag{3.1i}$$

$$t^{\mathrm{f}}_{j,n} = t^{\mathrm{s}}_{j,n} + \sum_{c \in \mathcal{C}_j} \sum_{i \in \mathcal{I}_c} \overline{PT}_{i,j} z_{c,j,n} \qquad \forall s = S, j \in \mathcal{J}_s, n \tag{3.1j}$$

$$t^{\mathrm{f}}_{i,s} \geq t^{\mathrm{s}}_{i,s} + \sum_{j \in \mathcal{J}_s \cap \mathcal{J}_c} \overline{PT}_{i,j} y_{c,j} \qquad \forall i, s \notin \mathcal{S}^{\mathrm{ZW}} \tag{3.1k}$$

$$t^{\mathrm{f}}_{i,s} = t^{\mathrm{s}}_{i,s} + \sum_{j \in \mathcal{J}_s \cap \mathcal{J}_c} \overline{PT}_{i,j} y_{c,j} \qquad \forall i, s \in \mathcal{S}^{\mathrm{ZW}} \tag{3.1l}$$

$$t^{\mathrm{s}}_{i+1,s} \geq t^{\mathrm{f}}_{i,s} \qquad \forall s, c, i \notin \mathcal{I}^{\mathrm{last}}_{\mathrm{c}} \tag{3.1m}$$

$$t^{\mathrm{f}}_{i,s} \leq t^{\mathrm{s}}_{i,s} + P^{\mathrm{perish}}_{j} y_{c,j} \qquad \forall i,\ s \in \mathcal{S}^{\mathrm{perish}},\ j \in \mathcal{J}_s \cap \mathcal{J}_c \tag{3.1n}$$

$$t^{\mathrm{s}}_{j,n+1} \geq t^{\mathrm{f}}_{j,n} + \sum_{c \in \mathcal{C}_j} P^{\mathrm{setup}}_{j} z_{c,j,n} \qquad \forall j, n \tag{3.1o}$$

$$t^{\mathrm{s}}_{i,s+1} = t^{\mathrm{f}}_{i,s} \qquad \forall i, s < S \tag{3.1p}$$

$$t^{\mathrm{s}}_{i,s} \geq t^{\mathrm{s}}_{j,n} - M\left[1 - z_{c,j,n}\right] \qquad \forall s,\ c,\ n,\ i \in \mathcal{I}^{\mathrm{first}}_{\mathrm{c}},\ j \in \mathcal{J}_s \tag{3.1q}$$

$$t^{\mathrm{f}}_{i,s} \leq t^{\mathrm{f}}_{j,n} + M\left[1 - z_{c,j,n}\right] \qquad \forall s,\ c,\ n,\ i \in \mathcal{I}^{\mathrm{last}}_{\mathrm{c}},\ j \in \mathcal{J}_s \tag{3.1r}$$

$$t^{\mathrm{s}}_{i,s}, t^{\mathrm{f}}_{i,s} \geq 0 \qquad \forall i, s \tag{3.1s}$$

$$t^{\mathrm{s}}_{j,n}, t^{\mathrm{f}}_{j,n} \geq 0 \qquad \forall j, n \tag{3.1t}$$

$$x_{c,n}, y_{c,j}, z_{c,j,n} \in \{0, 1\} \qquad \forall c,\ n,\ j \tag{3.1u}$$

3.4 Adaptive robust scheduling problem

Uncertainty set characterization

Uncertain processing time of a ladle i in a unit j is defined as a function of the primitive uncertainty $\xi_{i,j}$ in equation (3.2) where $\overline{PT}_{i,j}$ is the nominal processing time and $\mathcal{J}^{\mathrm{p}}$:={EAF, AOD, LF, CC} is the set of processing units.

$$PT_{i,j}(\xi_{i,j}) = \overline{PT}_{i,j}(1 + \xi_{i,j}) \qquad \forall j \in \mathcal{J}^{\text{p}},\ i \in \mathcal{I}_j \tag{3.2}$$

Uncertainty is not defined for a storage unit as there is no processing taking place. In this work, we are considering only delay in the processing time. The uncertainty set is defined in equation (3.3) and Δ_j is a parameter describing the maximum delay in a unit j. The primitive uncertainty is endogenous and its observability is dictated by the allocation decision $z_{c,j,n}$.

$$\Xi := \{\boldsymbol{\xi} \mid 0 \leq \xi_{i,j} \leq \Delta_j \quad \forall j \in \mathcal{J}^{\text{p}},\ i \in \mathcal{I}_j\} \tag{3.3}$$

Static and adaptive variables

In this work, we set the allocation and sequencing decisions to be first stage static variables (i.e., $x_{c,n}$, $y_{c,j}$ and $z_{c,j,n}$). The timing decisions $t^{\text{s}}_{j,n}(\boldsymbol{\xi})$, $t^{\text{f}}_{j,n}(\boldsymbol{\xi})$ $t^{\text{s}}_{i,s}(\boldsymbol{\xi})$, and $t^{\text{f}}_{i,s}(\boldsymbol{\xi})$ are considered recourse adaptive decisions. Few exceptions include the start time of the first event in all processing units (i.e., $t^{\text{s}}_{j,1}$ $\forall j$) and the start time of the first ladle i in a cast c in the first processing stage (i.e., $t^{\text{s}}_{i,1}$ $\forall i \in \mathcal{I}^{\text{first}}_{c(i)}$).

3.4.1 Mathematical model

The SCC adaptive robust optimization model (3.4) minimizes the worst case makespan. It is a natural extension of the deterministic setting.

$$\begin{aligned}
\min \quad & MS && \text{(3.4a)}\\
\text{s.t.} \quad & MS \geq \max_{\xi \in \Xi}[t^{\text{f}}_{j,n}(\boldsymbol{\xi})] \qquad \forall \boldsymbol{\xi} \in \Xi,\ n = N,\ j \in \mathcal{J}_S && \text{(3.4b)}\\
& \text{eq.(3.1c)} - \text{eq.(3.1u)} \qquad \forall \boldsymbol{\xi} \in \Xi\\
\text{where} \quad & t^{\text{f}}_{j,n}, t^{\text{s}}_{j,n}, t^{\text{s}}_{j,n+1} \equiv t^{\text{f}}_{j,n}(\boldsymbol{\xi}), t^{\text{s}}_{j,n}(\boldsymbol{\xi}), t^{\text{s}}_{j,n+1}(\boldsymbol{\xi})\\
& t^{\text{s}}_{i,s}, t^{\text{f}}_{i,s}, t^{\text{s}}_{i+1,s}, t^{\text{s}}_{i,s+1} \equiv t^{\text{s}}_{i,s}(\boldsymbol{\xi}), t^{\text{f}}_{i,s}(\boldsymbol{\xi}), t^{\text{s}}_{i+1,s}(\boldsymbol{\xi}), t^{\text{s}}_{i,s+1}(\boldsymbol{\xi})
\end{aligned}$$

The size of the model is reduced by removing the equality constraints $t^{\text{s}}_{i,s+1}(\boldsymbol{\xi}) = t^{\text{f}}_{i,s}(\boldsymbol{\xi})$ and replacing $t^{\text{f}}_{i,s}(\boldsymbol{\xi})$ with $t^{\text{s}}_{i,s+1}(\boldsymbol{\xi})$. Model (3.4) is computationally intractable due to the presence of the semi-infinite constraints.

Decision rule based-solution method circumvents the intractability by approximating the adaptive decision functionals via functions of the observed uncertainty. Linear decision rule (LDR), as its name indicates, restricts the adaptive decisions to a linear dependence of the primitive uncertainty $\boldsymbol{\xi}$. LDR is the simplest form of decision rules and offers attractive modelling features. Practically, an adaptive decision at any time stage can not depend on any uncertainty information/realization to be known in the future. Hence, the corresponding LDR must contain future realized uncertainties (through the slope terms) in its definition. This property is known as non-anticipativity and must be satisfied by any decision rule.

In a USEB formulation, the same event point in each unit is different which allows the same ladle to start processing in the same event point at different times. To maintain consistency among the timing decisions of ladles, the formulation depends on special sequencing constraints (Janak & Floudas 2008). These constraints will be used to systematically design an LDR that satisfy the non-anticipativity property.

3.4.2 Logical sequencing constraints

Three temporal correlations within the unit-specific time grid are investigated. The first two are special sequencing constraints inherently present in USEB formulations and the third is exhibited by the NIS policy. The context of each temporal correlation is described below:

1. Logical Sequencing Constraint 1 (LSC 1) represents the first temporal correlation used. It dictates the timing decisions of processed casts in a unit through sequential event points. The logic is inferred from the constraints $t^{\mathrm{s}}_{j,n}(\boldsymbol{\xi}) \geq t^{\mathrm{f}}_{j,n-1}(\boldsymbol{\xi})$ for all j and $n > 1$. It implies that if a cast initiates processing in a unit j in an event point n, then the casts that has initiated processing in the previous event points $n' < n$ in the same unit have already finished. Consequently, the uncertainty realizations related to the ladles processed in $n' < n$ and unit j satisfy the non-anticipativity property for the decision rule approximating $t^{\mathrm{s}}_{j,n}(\boldsymbol{\xi})$. This is extended for the decision rule defining $t^{\mathrm{f}}_{j,n}(\boldsymbol{\xi})$ with $n' \leq n$.

2. Logical Sequencing Constraint 2 (LSC 2) represents the second temporal correlation. It dictates the finish and start timing decisions of a ladle across processing stages. The logic is inferred from the constraints $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi}) \geq t^{\mathrm{f}}_{i,s-1}(\boldsymbol{\xi})$ for all i and $s > 1$. In our problem, NIS policy is implemented in the model which imposes the strict equality sign $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi}) = t^{\mathrm{f}}_{i,s-1}(\boldsymbol{\xi})$. Regardless of the storage policy, it implies that a ladle i initiates processing in a processing stage s only after it has finished processing in the previous processing stages $s' < s$. Consequently, the uncertainty related information for a ladle i in $s' < s$ satisfies the non-anticipativity property in the decision rule approximating $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$.

3. The third temporal correlation is exhibited by a predetermined processing sequence of ladles within a cast ($t^{\mathrm{s}}_{i+1,s}(\boldsymbol{\xi}) \geq t^{\mathrm{f}}_{i,s}(\boldsymbol{\xi}),\ i \in \mathcal{I}_c, i \neq \mathcal{I}^{\mathrm{last}}_c$) and the NIS policy ($t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi}) = t^{\mathrm{f}}_{i,s-1}(\boldsymbol{\xi})\ \forall i,\ s > 1$). They are assumptions made for this scheduling problem. The logic is only applied to the ladles within the same cast. Let $p(i)$ be the processing order of a ladle i in a cast c. We claim that (i) none of the ladles' uncertainty information in future processing stages are guaranteed to satisfy the non-anticipativity property in the decision rule approximating $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$ when $p(i) < 3$ (first and second processed ladles in a cast) and (ii) the last processing stage at which a ladle's uncertainty information in $c(i)$ is guaranteed to satisfy the non-anticipativity property is $s + p(i) - 2$. The number of uncertainty realizations is the highest in $s+1$ and includes the first $p(i) - 2$ ladles within cast $c(i)$. After that, it decreases by one ladle in each consequent processing stage (i.e., in processing stage $s+2$, the first $p(i) - 3$ uncertainty elements satisfy the non-anticipativity property).

For example, assume a set of ladles $\mathcal{I}_2 = \{3, 4, 5, 6\}$ belongs to the second cast $c = 2$. In the first processing stage $s = 1$, we need to determine which uncertainty information in future processing stages are guaranteed to satisfy the non-anticipativity property in the LDR approximating $t^{\mathrm{s}}_{6,1}(\boldsymbol{\xi})$ (i.e., $p(6) = 4$). Applying the logic, we conclude that in processing stage $s = 2$, we have the uncertainty information of ladles 3 and 4, and in processing stage $s = 3$, we have the uncertainty information of only ladle 3.

3.4.3 Linear decision rules

Timing decisions of a ladle i in a processing stage s

The main factor in constructing the LDR is the number of logical inference steps implemented. For example, LSC 1 or LSC 2 is considered a single logical inference step. For multi-alternating logical inference steps, only two forms exist. For example, LSC 1 given LSC 2 (i.e., LSC 1/LSC 2) and LSC 2 given LSC 1 (i.e., LSC 2/LSC 1) are the only two logical inference steps.

The LDR approximating $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$ is given in eq (3.5) using a maximum of two logical inference steps and the NIS policy in the current processing stage s is given in equation (3.5).

$$
\begin{aligned}
t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi}) = \; & t^{0,\mathrm{s}}_{i,s} + \underbrace{\sum_{i'=l^{\mathrm{first}}_{c(i)}}^{i} \sum_{s'=1}^{s-1} \sum_{j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} t^{1,\mathrm{s}}_{i,s,i',j',n} \xi_{i',j'} + \sum_{i'=l^{\mathrm{first}}_{c(i)}}^{i-1} \sum_{j \in \mathcal{J}_{s} \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} t^{1,\mathrm{s}}_{i,s,i',j,n} \xi_{i',j}}_{\text{LSC 2: ladles within cast } c(i) \text{ in past processing stages}} \\
& + \underbrace{\sum_{c' \in \mathcal{C},\ c' \neq c(i)} \sum_{i' \in \mathcal{I}_{c'}} \sum_{j \in \mathcal{J}_{s} \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{i,s,i',j,n'} \xi_{i',j}}_{\text{LSC 1: ladles in past events in current processing stage}} \\
& + \underbrace{\sum_{c' \in \mathcal{C},\ c' \neq c(i)} \sum_{i' \in \mathcal{I}_{c'}} \sum_{s'=1}^{s-1} \sum_{j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{i,s,i',j',n'} \xi_{i',j'}}_{\text{LSC 1/LSC 2: ladles in past events given past processing stages}} \\
& + \underbrace{\sum_{c' \in \mathcal{C},\ c' \neq c(i)} \sum_{i' \in \mathcal{I}_{c'}} \sum_{s'=1}^{s-1} \sum_{j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{i,s,i',j',n'} \xi_{i',j'}}_{\text{LSC 2/LSC 1: ladles in past processing stages given past events}} \\
& + \underbrace{\sum_{k=1}^{p(i)-2} \sum_{i'=l^{\mathrm{first}}_{c(i)}}^{l^{\mathrm{first}}_{c(i)}+p(i)-k-2} \sum_{j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} t^{1,\mathrm{s}}_{i,s,i',j',n} \xi_{i',j'}}_{\text{NIS: applied to ladles within cast } c(i) \text{ in current processing stage}} \\
& + \underbrace{\sum_{c' \in \mathcal{C},\ c' \neq c(i)} \sum_{k=1}^{L_c-1} \sum_{i'=l^{\mathrm{first}}_{c'}}^{l^{\mathrm{first}}_{c'}+L_c-k-1} \sum_{j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n=1}^{N} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{i,s,i',j',n'} \xi_{i',j'}}_{\text{NIS: applied to ladles within casts } c' \text{ in past events and in current processing stage}} \quad \forall i \in \mathcal{I}, s \in \mathcal{S}
\end{aligned}
\tag{3.5}
$$

where $l_{c(i)}^{\text{first}}$ is the absolute index of the first ladle i in a cast c and L_c is the total number of ladles in a cast c ($L_c = p(i)$ where $i \in \mathcal{I}_{c(i)}^{\text{last}}$).

Materialization constraints that dictate the observability of the endogenous uncertainty in an event point n in a processing unit j are required to complement the LDRs. Basically, for the non-materialized uncertainties, the corresponding coefficients in the LDR are set to zero. These constraints are modelled using the allocation decision variables $z_{c,j,n}$.

For the uncertain parameters inferred by LSC 2, the materialization constraints are defined in eq. (3.6). The linearity is due to the fact that a cast initiates processing in the same event point in all processing stages. The M_1 value is equal to 80 which is an upper bound for ladle processing time in all units.

$$\left.\begin{array}{l} t_{i,s,i',j',n}^{1,\text{s}} \leq M_1 z_{c(i'),j',n} \\ t_{i,s,i',j',n}^{1,\text{s}} \geq -M_1 z_{c(i'),j',n} \end{array}\right\} \ \forall s \in \mathcal{S},\ i' \in \mathcal{I}_{c(i)},\ i' \leq i,\ s' < s,\ j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\text{p}},\ n \in \mathcal{N}$$

$$\left.\begin{array}{l} t_{i,s,i',j,n}^{1,\text{s}} \leq M_1 z_{c(i'),j,n} \\ t_{i,s,i',j,n}^{1,\text{s}} \geq -M_1 z_{c(i'),j,n} \end{array}\right\} \ \forall s \in \mathcal{S},\ i' \in \mathcal{I}_{c(i)},\ i' \leq i-1,\ j \in \mathcal{J}_s \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\text{p}},\ n \in \mathcal{N} \tag{3.6}$$

For the uncertain parameters inferred using LSC 1, the materialization constraints in equation (3.7) first identify the event point and processing unit at which the cast $c(i)$ has initiated processing (i.e., $z_{c(i),j,n} = 1$). Then, it identifies the casts c' that have finished processing in past event points in the same unit (i.e., $z_{c'(i'),j,n'} = 1$).

$$\left.\begin{array}{l} t_{i,s,i',j,n'}^{1,\text{s}} \leq M_1 z_{c(i),j,n} z_{c'(i'),j,n'} \\ t_{i,s,i',j,n'}^{1,\text{s}} \geq -M_1 z_{c(i),j,n} z_{c'(i'),j,n'} \end{array}\right\} \begin{array}{l} \forall s \in \mathcal{S},\ c,c' \in \mathcal{C}, c' \neq c,\ i \in \mathcal{I}_{\text{c}},\ i' \in \mathcal{I}_{\text{c}'}, \\ j \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\text{p}},\ n \in \mathcal{N},\ n' < n \end{array} \tag{3.7}$$

The bilinear term is linearized by introducing the auxiliary variable $v_{c(i),c'(i'),j,j',n,n'}$ and the following set of constraints

$$\left.\begin{array}{l} v_{c(i),c'(i'),j,j',n,n'} \leq z_{c'(i'),j,n'} \\ v_{c(i),c'(i'),j,j',n,n'} \leq z_{c(i),j,n} \\ v_{c(i),c'(i'),j,j',n,n'} \geq z_{c'(i'),j,n'} + z_{c(i),j,n} - 1 \\ v_{c(i),c'(i'),j,j',n,n'} \in \{0,1\} \end{array}\right\} \begin{array}{l} \forall c,c' \in \mathcal{C},\ c' \neq c, i \in \mathcal{I}_{\text{c}},\ i' \in \mathcal{I}_{\text{c}'}, n \in \mathcal{N}, \\ n' < n,\ j \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\text{p}}, j' = j \end{array} \tag{3.8}$$

For the uncertain parameters inferred by LSC 1/LSC 2, LSC 2/LSC 1 and the NIS policy, the materialization constraints are included in Appendix B.2. In general, the complexity of the materialization constraints increases as the number of logical inference steps used to obtain the uncertain parameters increases.

Timing decisions of an event n in a processing unit j

The LDR approximating $t^{\mathrm{s}}_{j,n}(\boldsymbol{\xi})$ is given in equation (3.9) using a maximum of two logical inference steps and the NIS policy for ladles processed in unit j.

$$\begin{aligned}
t^{\mathrm{s}}_{j,n}(\boldsymbol{\xi}) = t^{0,\mathrm{s}}_{j,n} &+ \underbrace{\sum_{n'=1}^{n-1} \sum_{c' \in \mathcal{C}} \sum_{i' \in \mathcal{I}_{c'}} t^{1,\mathrm{s}}_{j,n,i',j,n'} \xi_{i',j}}_{\text{LSC 1: ladles in past events in unit } j} + \underbrace{\sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{I}^{\mathrm{first}}_{c}} \sum_{s=1}^{s_j - 1 + \eta} \sum_{j' \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}}} t^{1,\mathrm{s}}_{j,n,i,j',n} \xi_{i,j'}}_{\text{LSC 2: ladles in event n in past processing stages}} \\
&+ \underbrace{\sum_{c' \in \mathcal{C}, c' \neq c} \sum_{i' \in \mathcal{I}_{c'}} \sum_{s=1}^{s_j - 1} \sum_{j' \in \mathcal{J}_s \cap \mathcal{J}_c \cap \mathcal{J}^{\mathrm{p}}} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{j,n,i',j',n'} \xi_{i',j'}}_{\text{LSC 1/LSC 2: ladles in past events given past processing stages}} \\
&+ \underbrace{\sum_{c' \in \mathcal{C}, c' \neq c} \sum_{i' \in \mathcal{I}_{c'}} \sum_{s=1}^{s_j - 1} \sum_{j' \in \mathcal{J}_s \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n'=1}^{n-1} t^{1,\mathrm{s}}_{j,n,i',j',n'} \xi_{i',j'}}_{\text{LSC 2/LSC 1: ladles in past processing stages given past events}} \\
&+ \underbrace{\sum_{c' \in \mathcal{C}} \sum_{k=1}^{L_{c'} - 1} \sum_{i = l^{\mathrm{first}}_{c'}}^{l^{\mathrm{first}}_{c'} + L'_c - k - 1} \sum_{j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}}} \sum_{n'=1}^{n-1+\eta} t^{1,\mathrm{s}}_{j,n,i',j',n'} \xi_{i',j'}}_{\text{NIS for ladles in unit } j \text{ in past events}} \qquad \forall j \in \mathcal{J}, n \in \mathcal{N}
\end{aligned} \tag{3.9}$$

where $\eta = 0$.

Recall that uncertainty is not defined in the storage units (see equation (3.3)), hence the terms related to LSC 1 do not exist in the decision rules for all $j \in \mathcal{J}_s$.

The materialization constraints for the uncertain parameters in LSC 1 and LSC 2 blocks are defined in equations (3.10) and (3.11), respectively.

$$\left.\begin{aligned} t^{1,\mathrm{s}}_{j,n,i',j,n'} &\leq M_1 z_{c'(i'),j,n'} \\ t^{1,\mathrm{s}}_{j,n,i',j,n'} &\geq -M_1 z_{c'(i'),j,n'} \end{aligned}\right\} j \in \mathcal{J}^{\mathrm{p}},\ n \in \mathcal{N},\ i' \in \mathcal{I}_j,\ n' < n \tag{3.10}$$

$$\left.\begin{aligned} t^{1,\mathrm{s}}_{j,n,i,j',n'} &\leq M_1 z_{c(i),j,n} z_{c(i),j',n'} \\ t^{1,\mathrm{s}}_{j,n,i,j',n'} &\geq -M_1 z_{c(i),j,n} z_{c(i),j',n'} \end{aligned}\right\} \begin{aligned} & j \in \mathcal{J},\ n, n' \in \mathcal{N},\ c \in \mathcal{C},\ i \in \mathcal{I}^{\mathrm{first}}_{c},\ s \leq s_j - 1 + \eta, \\ & n' < n,\ j' \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}} \end{aligned} \tag{3.11}$$

Equation (3.11) first identifies the cast $c(i)$ processed in unit j and event point n (i.e., $z_{c(i),j,n} = 1$), then it identifies the event points n' in units j' in past processing stages where the same cast $c(i)$ is processed (i.e., $z_{c(i),j',n'} = 1$). The model assumes that $c(i)$ initiates processing in the same event point n in all processing stages ($n' = n$). The bilinear term $z_{c(i),j,n} z_{c(i),j',n'}$ reduces to $z_{c(i),j',n}$ as in equation (3.12)

$$\left.\begin{aligned} t^{1,\mathrm{s}}_{j,n,i,j',n} &\leq M_1 z_{c(i),j',n} \\ t^{1,\mathrm{s}}_{j,n,i,j',n} &\geq -M_1 z_{c(i),j',n} \end{aligned}\right\} \begin{aligned} & j \in \mathcal{J},\ n \in \mathcal{N},\ c \in \mathcal{C},\ i \in \mathcal{I}^{\mathrm{first}}_{c},\ s \leq s_j - 1 + \eta, \\ & j' \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}} \end{aligned} \tag{3.12}$$

The materialization constraints for the uncertain parameters obtained by LSC 1/LSC 2, LSC 2/LSC 1 and the NIS policy are included in Appendix B.2.

The LDR of the state variable $t^{\mathrm{f}}_{j,n}(\boldsymbol{\xi})$ is similar to equation (3.9) with $\eta = 1$ and $\mathcal{I}^{\mathrm{first}}_c \equiv \mathcal{I}_c$ in the block of terms corresponding to LSC 2. The intercept $t^{0,\mathrm{s}}_{j,n}$ is replaced by $t^{0,\mathrm{f}}_{j,n}$. The slopes $t^{1,\mathrm{s}}_{j,n,i',j,n'}$, $t^{1,\mathrm{s}}_{j,n,i,j',n}$ and $t^{1,\mathrm{s}}_{j,n,i',j',n'}$ are replaced by $t^{1,\mathrm{f}}_{j,n,i',j,n'}$, $t^{1,\mathrm{f}}_{j,n,i,j',n}$ and $t^{1,\mathrm{f}}_{j,n,i',j',n'}$, respectively. These changes also apply to the materialization constraints.

Figure 3.4 illustrates the observed uncertain parameters under various forms of the decision rules using a schematic schedule. Redundant ladle uncertain parameters may appear in an LDR when using more than one logical inference step. The redundancy cannot be avoided due to the endogenous nature of the uncertainty.

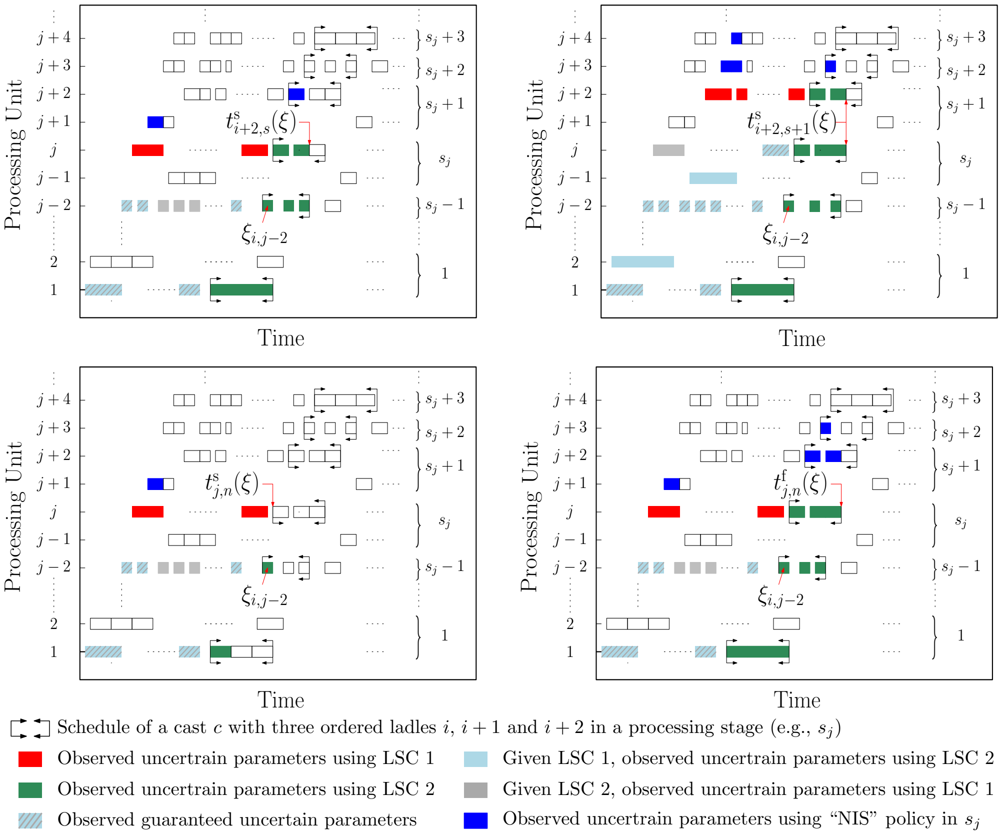

Processing Unit
$j+4$
$j+3$
$j+2$
$j+1$
j
$j-1$
$j-2$
2
1
s_j+3
s_j+2
s_j+1
s_j
s_j-1
1
$t^{\mathrm{s}}_{i+2,s}(\xi)$
$t^{\mathrm{s}}_{i+2,s+1}(\xi)$
$t^{\mathrm{s}}_{j,n}(\xi)$
$t^{\mathrm{f}}_{j,n}(\xi)$
$\xi_{i,j-2}$
Time
Schedule of a cast c with three ordered ladles i, $i+1$ and $i+2$ in a processing stage (e.g., s_j)
Observed uncertrain parameters using LSC 1
Observed uncertrain parameters using LSC 2
Observed guaranteed uncertain parameters
Given LSC 1, observed uncertrain parameters using LSC 2
Given LSC 2, observed uncertrain parameters using LSC 1
Observed uncertrain parameters using “NIS” policy in s_j

Figure 3.4: Realized uncertainties, in a schematic schedule, using a maximum of two logical inference steps and the no intermediate storage policy are illustrated for the $t^{\mathrm{s}}_{i+2,s}(\boldsymbol{\xi})$, $t^{\mathrm{s}}_{i+2,s+1}(\boldsymbol{\xi})$ and $t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$ linear decision rules.

3.5 Computational results

Model (3.4) is computationally intractable in its given form due to the presence of the semi-infinite constraints. The deterministic robust counterpart is derived constraint wise. For the semi-infinite inequality constraints, strong duality for LP is used to derive the equivalent counterpart, whereas for the semi-infinite equality constraints the counterpart is obtained by forcing each of the slope terms (i.e., coefficients of the uncertainty) and the intercept terms to be equal to zero. This procedure is done by a modified version of JuMPeR v0.5.0 package which is an extension to the mathematical programming package JuMP v0.15 in Julia v0.5.2. The modification to JuMPeR v0.5.0 includes the derivation of the stochastic counterpart for semi-infinite equality constraints. The MIP problems are solved using ILOG CLPEX 12.7

Three computational studies are executed in this work. The first compares four different proposed LDRs. The LDR which offers the best trade-off between solution quality and computational cost will be adopted. The second compares the worst case performance of adaptive robust proactive and deterministic reactive schedules. The third simulates the performance of adaptive robust proactive, deterministic reactive and adaptive robust hybrid schedules over various uncertainty distributions.

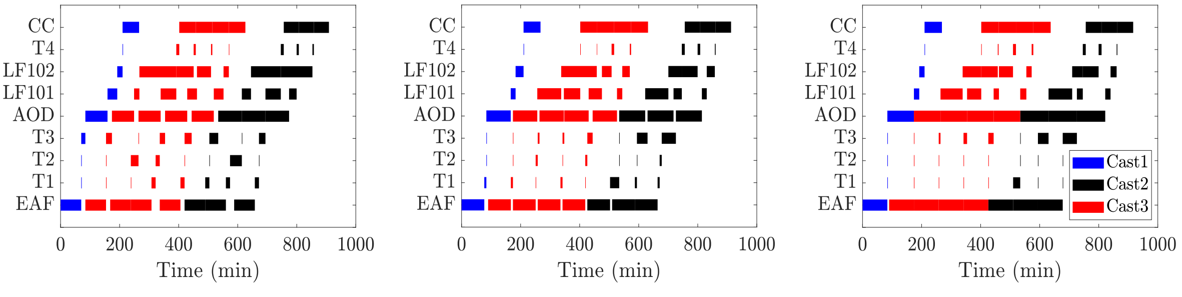
For all computational experiments, the production orders describing the ladle to cast assignment and the ladles' processing time in each unit is given in Appendix B.3 (Li, Xiao, Tang & Floudas 2012).

3.5.1 Illustrative example

Table 3.1 presents the adaptive timing decision rule solution for a 3-cast/8-ladle schedule using the decision rule definition in equations (3.9) and (3.5). The process structure does not include any parallel units. The uncertain processing time delay in the EAF, AOD, LF101 and LF102 units varies from 0 to 20% (i.e., $\xi_{i,j} \in [0, 0.2]$), whereas the uncertain delay in the CC units varies from 0 to 5% (i.e., $\xi_{i,j} \in [0, 0.05]$). The optimal makespan is equal to 916.5 min and the optimal sequence of casts in the CC is cast 1 → cast 3 → cast 2. Most of the independent timing decisions which include the starting time of all ladles in the EAF and AOD units are statically robust. Figure 3.5 illustrates the simulated schedule for a realized minimum, average and maximum delay scenarios. We observe processing gaps between ladles in the EAF and AOD schedule in figure 3.5a and 3.5b because the start timing decisions of consequent ladles do not adapt to the fact that the realized delay is less than the worst cast value (i.e., suboptimal). Meanwhile, we observe no processing gaps in the EAF and AOD units for the worst case realized delay as in Figure 3.5c.

Table 3.1: The ladle related adaptive decision rule solution for a 3-cast/8-ladle schedule using a process structure with no parallel units.

Units	$t^{\mathrm{s}}_{1,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{1,s}(\boldsymbol{\xi})$	$t^{\mathrm{s}}_{5,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{5,s}(\boldsymbol{\xi})$
EAF	$\mathbf{0}$	$70+70\xi_{1,EAF}$	$\mathbf{90}$	$160+70\xi_{5,EAF}$
T1	$70+70\xi_{1,EAF}$	84	$160+70\xi_{5,EAF}$	174
T2	84	84	174	174
T3	84	84	174	174
AOD	$\mathbf{84}$	$159 + 75\xi_{1,AOD}$	$\mathbf{174}$	$249+75\xi_{5,AOD}$
LF1	$159 + 75\xi_{1,AOD}$	$174 + 75\xi_{1,AOD} + 15\xi_{1,LF1}$	$249+75\xi_{5,AOD}$	$336+15\xi_{5,LF1}$
LF2	$\mathbf{174 + 75\xi_{1,AOD} + 15\xi_{1,LF1}}$	210	$\mathbf{336 + 15\xi_{5,LF1}}$	402
T4	210	210	402	402
CC	$\mathbf{210}$	$266 + 56\xi_{1,CC}$	$\mathbf{402}$	$458+56\xi_{5,CC}$
Units	$t^{\mathrm{s}}_{6,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{6,s}(\boldsymbol{\xi})$	$t^{\mathrm{s}}_{7,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{7,s}(\boldsymbol{\xi})$
EAF	$\mathbf{174}$	$244+70\xi_{6,EAF}$	$\mathbf{258}$	$328+70\xi_{7,EAF}$
T1	$244+70\xi_{6,EAF}$	$244+70\xi_{6,EAF}$	$328+70\xi_{7,EAF}$	342
T2	$244+70\xi_{6,EAF}$	258	342	342
T3	258	264	342	$342+60\xi_{6,AOD}$
AOD	$\mathbf{264}$	$339+75\xi_{6,AOD}$	$\mathbf{342 + 60\xi_{6,AOD}}$	$417+60\xi_{6,AOD}+75\xi_{7,AOD}$
LF101	$339+75\xi_{6,AOD}$	402	$417+60\xi_{6,AOD}+75\xi_{7,AOD}$	$488.8\text{-}80\xi_{5,EAF}\text{-}54\xi_{7,LF101}$
LF102	$\mathbf{402}$	458	$\mathbf{488.8 - 80\xi_{5,EAF} - 54\xi_{7,LF101}}$	$506.8+56\xi_{5,CC}$
T4	458	$458+56\xi_{5,CC}$	$506.8+56\xi_{5,CC}$	$514+56\xi_{5,CC}+56\xi_{6,CC}$
CC	$\mathbf{458 + 56\xi_{5,CC}}$	$514+56\xi_{5,CC}+56\xi_{6,CC}$	$\mathbf{514 + 56\xi_{5,CC} + 56\xi_{6,CC}}$	$570+56\xi_{5,CC}+56\xi_{6,CC}+56\xi_{7,CC}$
Units	$t^{\mathrm{s}}_{8,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{8,s}(\boldsymbol{\xi})$	$t^{\mathrm{s}}_{2,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{2,s}(\boldsymbol{\xi})$
EAF	$\mathbf{342}$	$412+70\xi_{8,EAF}$	$\mathbf{426}$	$496 +70\xi_{2,EAF}$
T1	$412+70\xi_{8,EAF}$	$412+70\xi_{8,EAF}$	$496+70\xi_{2,EAF}$	534
T2	$412+70\xi_{8,EAF}$	426	534	534
T3	426	444	534	534
AOD	$\mathbf{444}$	$519+75\xi_{8,AOD}$	$\mathbf{534}$	$614+80\xi_{2,AOD}$
LF1	$519+75\xi_{8,AOD}$	$534+56\xi_{5,CC}+75\xi_{8,AOD}+15\xi_{8,LF1}$	$614+80\xi_{2,AOD}$	$691+15\xi_{2,LF1}+80\xi_{5,LF2}$
LF2	$\mathbf{534 + 56\xi_{5,CC} + 75\xi_{8,AOD} + 15\xi_{8,LF1}}$	$567+56\xi_{5,CC}+15\xi_{8,LF2}$	$\mathbf{691 + 15\xi_{2,LF1} + 80\xi_{5,LF2}}$	746
T4	$567+56\xi_{5,CC}+15\xi_{8,LF2}$	$570+56\xi_{5,CC}+56\xi_{6,CC}+56\xi_{7,CC}$	746	756
CC	$\mathbf{570 + 56\xi_{5,CC} + 56\xi_{6,CC} + 56\xi_{7,CC}}$	$626+56\xi_{5,CC}+56\xi_{6,CC}+56\xi_{7,CC}+56\xi_{8,CC}$	$\mathbf{756}$	$807+51\xi_{2,CC}$
Units	$t^{\mathrm{s}}_{3,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{3,s}(\boldsymbol{\xi})$	$t^{\mathrm{s}}_{4,s}(\boldsymbol{\xi})$	$t^{\mathrm{f}}_{4,s}(\boldsymbol{\xi})$
EAF	$\mathbf{510}$	$580+70{*}\xi_{3,EAF}$	$\mathbf{580 + 70\xi_{3,EAF}}$	$650+70\xi_{3,EAF}+70\xi_{4,EAF}$
T1	$580+70\xi_{3,EAF}$	594	$650+70\xi_{3,EAF}+70\xi_{4,EAF}$	$664+70\xi_{4,EAF}$
T2	594	594	$664+70\xi_{4,EAF}$	678
T3	594	630	678	726
AOD	$\mathbf{630}$	$710+80\xi_{3,AOD}$	$\mathbf{726}$	$806+80\xi_{4,AOD}$
LF1	$710+80\xi_{3,AOD}$	746	$806+80\xi_{4,AOD}$	$821+80\xi_{4,AOD}+15\xi_{4,LF1}$
LF2	$\mathbf{746}$	799.55	$\mathbf{821 + 80\xi_{4,AOD} + 15\xi_{4,LF1}}$	$855+51\xi_{2,CC}+15\xi_{4,LF1}$
T4	799.55	$807+51\xi_{2,CC}$	$855+51\xi_{2,CC}+15\xi_{4,LF1}$	$858+51\xi_{2,CC}+51\xi_{3,CC}$
CC	$\mathbf{807 + 51\xi_{2,CC}}$	$858+51\xi_{2,CC}+51\xi_{3,CC}$	$\mathbf{858 + 51\xi_{2,CC} + 51\xi_{3,CC}}$	$909+51\xi_{2,CC}+51\xi_{3,CC}+51\xi_{4,CC}$



(a) Minimum $PT_{i,j}(\xi_{i,j})$ (b) Average $PT_{i,j}(\xi_{i,j})$ (c) Maximum $PT_{i,j}(\xi_{i,j})$

Figure 3.5: Simulated schedules for a 3-cast/8-ladle instance at three different scenarios.

3.5.2 Comparison of linear decision rules

Four different LDRs for $t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$ and $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$ are shown in Table 3.2. LDR version 1 is equivalent to the decision rule in equations (3.9) and (3.5), while the remaining LDR versions are derived from the latter two equations by only including the uncertain parameters dictated by the corresponding block of terms in the design strategy. The intuition behind LDR version 3 is that LSC 1 and LSC 2 are directly related to $t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$ and $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$, respectively. As for LDR version 4, the state decisions $t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$ are assumed to be deterministic as it may be sufficient to provide a tight robust static bounds for the adaptive ladle timing decisions within an event n. The NIS policy is present in all LDRs. The information regarding the processing progress of the ladles within the same cast in future processing stages is necessary to satisfy the continuous-casting requirement.

Table 3.2: Four LDR versions for $t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$ and $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$ used in the comparison study in decreasing complexity.

	Decision rule design strategy	
	$t^{\mathrm{s/f}}_{j,n}(\boldsymbol{\xi})$	$t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$
LDR version 1	1,2 logical inference steps and NIS	1,2 logical inference steps and NIS
LDR version 2	1 logical inference step and NIS	1 logical inference step and NIS
LDR version 3	LSC 1 and NIS	LSC 2 and NIS for ladles in $c(i)$
LDR version 4	Static/Deterministic	LSC 2 and NIS for ladles in $c(i)$

The maximum delay for a ladle in the EAF, AOD and LF units is equal to 20% of the ladles' processing time. For the CC units, the maximum delay is equal to 5%. Larger delays in the CC unit will lead to infeasible schedules (i.e., violate the perishability constraint).

The optimal makespan and computational cost for 3-cast and 5-cast instances are shown in Table 3.3 using the processing network with no parallel units. No correlation between the uncertain parameters is considered. The solution quality is the same for all LDRs. The increase in the approximation quality for LDRs 1, 2 and 3 do not add any value to the solution quality.

Table 3.3: Makespan (min), solution time (sec) and size of a 3-cast and a 5-cast SCC scheduling problem using the four LDRs and uncorrelated uncertainty. Unit setup: $n^{\mathrm{EAF}} = n^{\mathrm{AOD}} = n^{\mathrm{CC}} = 1$, $n^{\mathrm{T}} = 3$, $n^{\mathrm{LF}} = 2$.

	3-cast instance				5-cast instance			
LDR version	1	2	3	4	1	2	3	4
Makespan	916.65	916.65	916.65	916.65	1816.50	1816.50	1816.50	1816.65
Solution time	23.79	4.67	2.61	0.32	4197.02	210.98	125.15	3.53
Binary variables	4167	3762	117	117	31545	28420	295	295
Continuous variables	51576	51576	49656	37416	309692	309692	303392	237512
Constraints	40665	25867	21265	17261	434071	180287	114905	92819

In the next set of experiments, the correlation between the uncertain delay is introduced as in equation (3.13). It reflects that the realizations of the delay of all ladles in the same unit are close.

$$|\xi_{1,j} - \xi_{i,j}| \leq 0.05\Delta_j \quad \forall j \in \mathcal{J}^{\mathrm{P}},\ i > 1 \tag{3.13}$$

Tables 3.4 and 3.5 illustrate the results for the process networks without and with parallel units, respectively. As expected, LDR version 1 can capture part of the correlation to generate a slightly better schedule (i.e., lower makespan), however this comes at a prohibitive computational cost. The solution of LDR versions 1 and 2 did not converge to optimality within the 10 hr time limit for the 5-cast instance in Table 3.5. LDR version 4 presents the best trade-off between solution quality and computational cost for the scheduling problem in hand. It will be used in the subsequent computational experiments.

Table 3.4: Makespan (min), solution time (sec) and size of a 3-cast and a 5-cast SCC scheduling problem using the four LDRs and correlated uncertainty. Unit setup: $n^{\mathrm{EAF}} = n^{\mathrm{AOD}} = n^{\mathrm{CC}} = 1$, $n^{\mathrm{T}} = 3$, $n^{\mathrm{LF}} = 2$.

	3-cast instance				5-cast instance			
LDR version	1	2	3	4	1	2	3	4
Makespan	912.57	916.65	916.65	916.65	1812.57	1816.50	1816.50	1816.65
Solution time	138.72	16.92	7.03	6.07	30849.52	5445.75	365.13	57.52
Binary variables	4167	3762	117	117	31545	28420	295	295
Continuous variables	82516	82516	78916	61636	487172	487172	474922	385072
Constraints	40665	25867	21265	17216	434071	180287	114905	92819

Table 3.5: Makespan (min), solution time (sec) and size of a 3-cast and a 5-cast SCC scheduling problem using the four LDRs and correlated uncertainty. Unit setup: $n^{\mathrm{EAF}} = n^{\mathrm{AOD}} = n^{\mathrm{CC}} = 2$, $n^{\mathrm{T}} = 3$, $n^{\mathrm{LF}} = 2$.

	3-cast instance				5-cast instance			
LDR version	1	2	3	4	1	2	3	4
Makespan	786.48	790.65	790.65	790.65	1810.05	1583.65	1551.65	1551.65
Solution time	1471.68	252.48	50.92	43.90	36000	36000	9231.4	4776.04
Binary variables	8577	7929	153	153	65385	60385	385	385
Continuous variables	156298	156298	148858	111514	916862	921342	892782	694382
Constraints	99484	48054	39170	31020	1339120	359655	213664	168910

3.5.3 Worst-case performance comparison

In this section, adaptive robust proactive and deterministic reactive scheduling methods are compared with respect to the worst case scenario. By deterministic reactive scheduling, we mean that the ladles' processing time are assumed to take the nominal values with no uncertainty and a new schedule is generated from scratch once a ladle finished processing in a given unit. The realized processing time of any ladle will always exceed the nominal value which invalidates the current schedule.

Further, we have two assumptions. First, the casting speed of ladles are adjustable during a rescheduling via a slack variable τ_i for all i. Equation (3.1l) for the CC processing stage is modified as equation (3.14).

$$t^{\mathrm{s}}_{i,s+1}(\boldsymbol{\xi}) = t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi}) + \sum_{j \in \mathcal{J}_s \cap \mathcal{J}_c} y_{c,j}(\overline{PT}_{i,j}(1 + \xi_{i,j}) + \tau_i) \qquad \forall i,\ s = S \tag{3.14}$$

Equation (3.1j) are similarly modified for the CC units.

The second assumption is related to the LF units. It is assumed that an LF can store a ladle after it finishes processing. This means that a delay in the processing time of a ladle is not considered a disruption unless the delay exceeds the allocated storage duration. Consequently, there is no need to react unless the latter condition is satisfied. For ease of implementation, we implicitly include the maximum delay into the deterministic processing time of all ladles in LF units (i.e., $\overline{PT}_{i,j}(1 + \Delta_j)$) and assume no disruption can take place at an LF unit. As for proactive scheduling, the uncertainty in the LF units are always set to the maximum delay value, so we have the same basis for comparison. The latter point is also applied in the simulation based study in the next section.

A 3-cast, a 5-cast and an 8-cast instances are used for the worst case performance comparison. The instances contain 8, 18 and 24 ladles, respectively (see Appendix B.3). For each instance, three maximum delay magnitudes in EAF, AOD and LF units are considered: 5%, 10% and 20% of the ladle's nominal processing time. The maximum delay in the CC is always fixed at

5% due to guarantee continuous casting (higher delays lead to infeasibility). LDR version 4 is implemented in the adaptive robust proactive scheduling method.

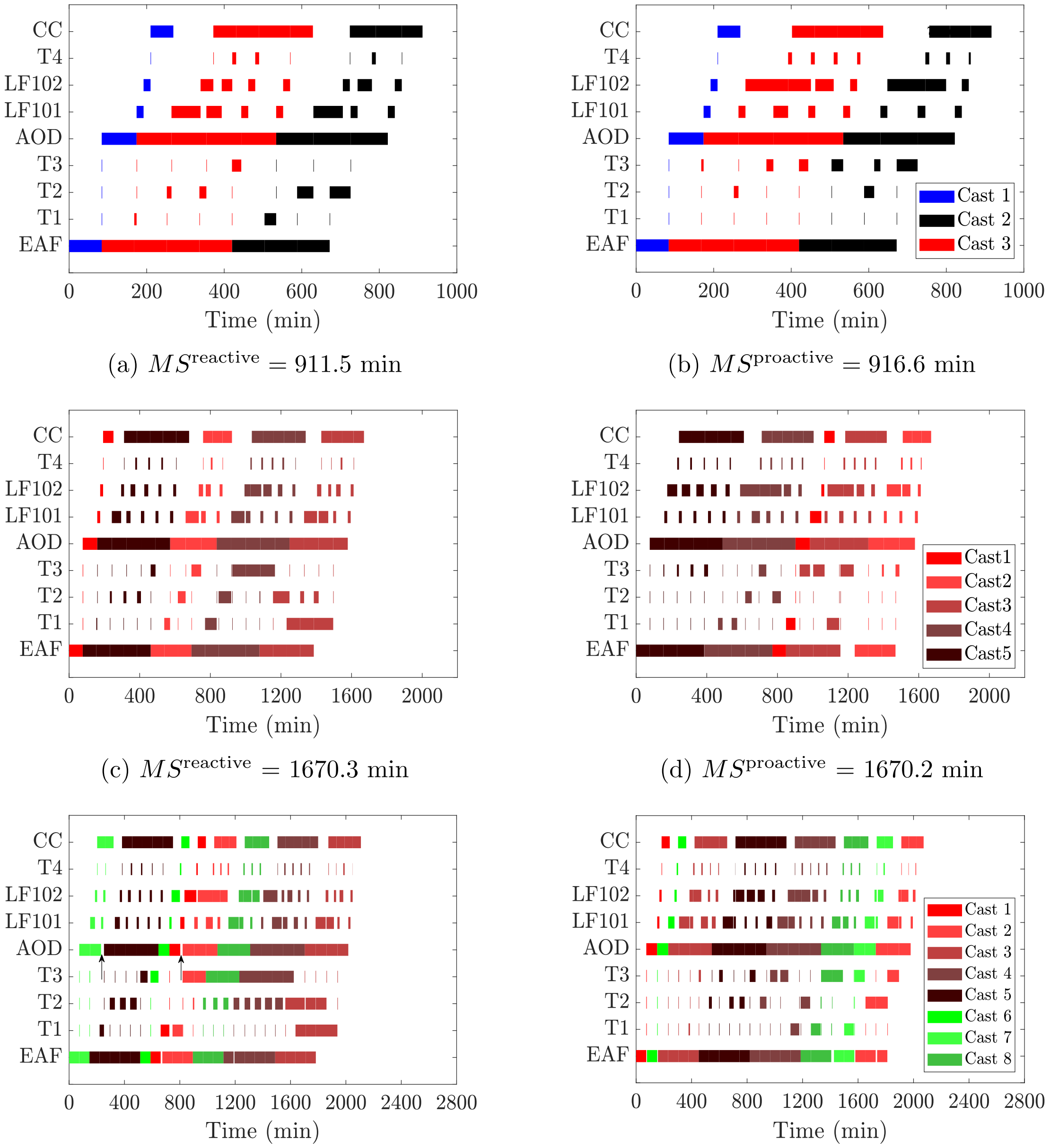
The worst case scenario corresponds to the maximum delay realization for each ladle in EAF, AOD, LF101, LF102 and CC units. For the robust adaptive proactive approach, the schedule is obtained by simply evaluating the optimal policies for $\xi_{i,j} = \Delta_j$ for all i, j (i.e., optimization is done only once; no rescheduling). While for the deterministic reactive approach, a rescheduling from scratch is triggered after each finished ladle in EAF, AOD and CC. This means that for a 3-cast, a 5-cast and an 8-cast instance, we have 24, 54 and 72 rescheduling events, respectively.

The simulated makespan obtained by the two scheduling approaches is tabulated in Table 3.6. The change in the makespan is computed for $MS^{\text{proactive}}$ with respect to MS^{reactive}: positive change indicates inferior performance, while negative change indicates superior performance.

Table 3.6: Simulated makespan (min) generated using the worst-case scenario via the deterministic reactive and robust proactive scheduling approaches.

	3-cast instance			5-cast instance			8-cast instance		
	5%	10%	20%	5%	10%	20%	5%	10%	20%
MS^{reactive}	805.4	840.0	911.5	1592.9	1670.3	1811.6	2108.5	2190.3	2382.8
$MS^{\text{proactive}}$	809.4	845.1	916.6	1596.9	1670.2	1816.6	2072.7	2165.1	2356.6
Change	4	5.1	5.1	4	-0.1	5	-35.8	-25.1	-26.2
Change (%)	0.50	0.61	0.56	0.25	-0.01	0.28	-1.70	-1.15	-1.01

The quality of the proactive schedule is superior to the reactive schedule for the 8-cast instance, whereas the opposite is true for the 3- and 5-cast instances. In the instances where the sequencing of casts are the same for both reactive and proactive schedule as for a 3-cast and 20% maximum delay in Figure 3.6, the quality of the reactive schedule is superior due to the re-optimized timing decisions via rescheduling and the implementation of controlled casting speed. The latter will provide the ability to prolong the processing time of earlier ladles in a given cast, consequently the first ladle in a cast may start processing earlier in a CC unit (without violating continuous casting requirement) compared to the same instance with same sequencing decisions in a robust proactive schedule. This results in a lower total makespan. On the other hand, when the cast sequencing decisions are inferior in the reactive schedule as in the 8-cast and 5% maximum delay in Figure 3.6, the overall quality of the reactive schedule is deteriorated. In particular, the sequencing decisions restricted possible improvements by re-optimized timing decisions. This can be verified by the processing gaps in the AOD unit. Such gaps do not exist in the proactive schedule. This outcome highlights the importance of incorporating uncertainty information for identifying resilient sequencing decisions. The 5-cast instance in Figure 3.6 illustrates a case where suboptimal sequencing decisions can be corrected by frequent rescheduling.



(a) $MS^{\text{reactive}} = 911.5$ min

(b) $MS^{\text{proactive}} = 916.6$ min

(c) $MS^{\text{reactive}} = 1670.3$ min

(d) $MS^{\text{proactive}} = 1670.2$ min

(e) $MS^{\text{reactive}} = 2108.5$ min

(f) $MS^{\text{proactive}} = 2072.7$ min

Figure 3.6: Adaptive robust proactive and deterministic reactive worst-case simulated schedules for a 3-cast instance with 20% maximum delay, a 5-cast instance with 10% maximum delay and an 8-cast instance with a 5% maximum delay.

3.5.4 Simulation over differently skewed uncertainty distributions

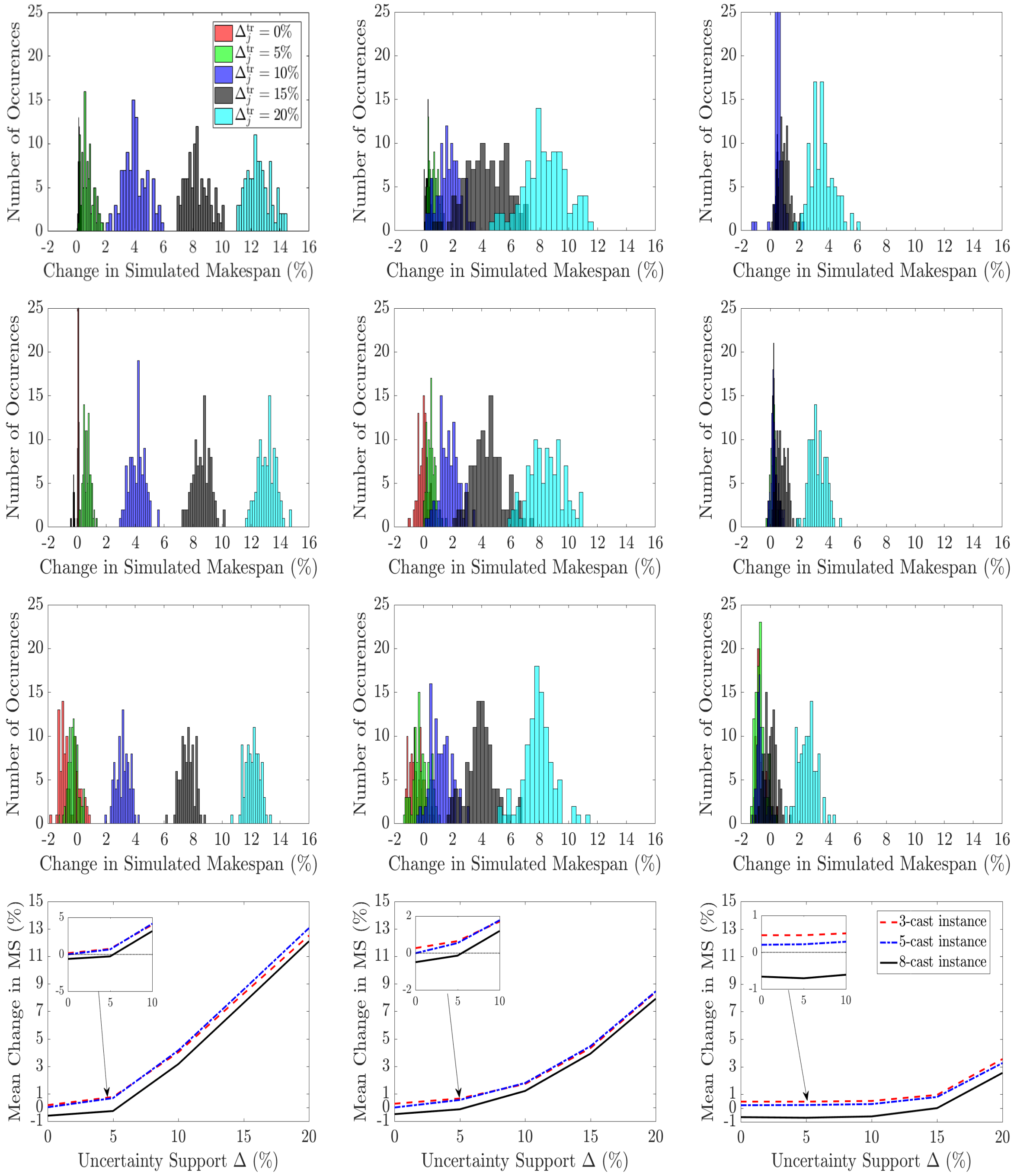
In practice, the processing time delay is not always equal to the worst case value. A simulation over various uncertainty distributions is meaningful to appreciate the quality of adaptive robust proactive and deterministic reactive schedules. Also, a hybrid scheduling method is introduced that combines the features of the latter two methods.

A hybrid schedule is modeled as an adaptive robust schedule with a truncated uncertainty set. Whenever the realized uncertain processing time delay falls outside the truncated support, a rescheduling event is triggered and a new schedule is generated from scratch. The aim is to incorporate enough uncertainty information in the cast-sequencing decisions so that they are neither conservative nor limiting for the potential future rescheduling.

In this experiment, the process structure does not include parallel units and the uncertain parameters are uncorrelated. The maximum possible delay is 20% for all ladles in all processing units except the CC (i.e., 5%). The design parameter for the hybrid schedule is the magnitude of the truncated delay Δ_j^{tr} in the EAF and the AOD units. We consider the following cases: 15%, 10%, 5% and 0%. Note that a hybrid schedule with $\Delta_j^{\text{tr}} = 20\%$ is equivalent to an adaptive robust proactive schedule. However, a hybrid schedule with 0% is not equivalent to a deterministic reactive approach since the LF and CC are immune to maximum delays of 20% and 5%, respectively. Recall that the assumption regarding the LF units made in the previous section still applies.

Three uncertainty distributions depicted in Figure 3.8a are studied. The main charactaristic of the distribution is skewness. The beta distributions $\beta(3, 8)$ with mean $\mu = 5.45\%$, $\beta(1, 1)$ with mean $\mu = 10\%$ and $\beta(16, 4)$ with mean $\mu = 16\%$ are left, uniform and right skewed distributions, respectively.

Figure 3.7 illustrates the relative change in simulated makespan exhibited by the adaptive robust proactive and hybrid schedules with respect to the deterministic reactive schedule using 100 scenarios from the different distributions. The conservatism of a proactive schedule decreases as the skewness shifts from left to right for the 3-cast, 5-cast and 8-cast instances. For each distribution, the mean change in simulated makespan for a given Δ_j^{tr} value decreases as the number of casts increases. The increase in the number of casts elevates the significance of determining good sequencing decisions. Regardless of the distribution, we observe that the quality of the hybrid schedule with $\Delta_j^{\text{tr}} = 0\%$ and $\Delta_j^{\text{tr}} = 5\%$ are superior to the deterministic reactive schedule for an 8-cast instance. This is despite the fewer available opportunities to re-optimize the ladle timing decisions in the hybrid schedule (i.e., better quality with less rescheduling).



(a) Left Skewed: $\beta(3,8)$

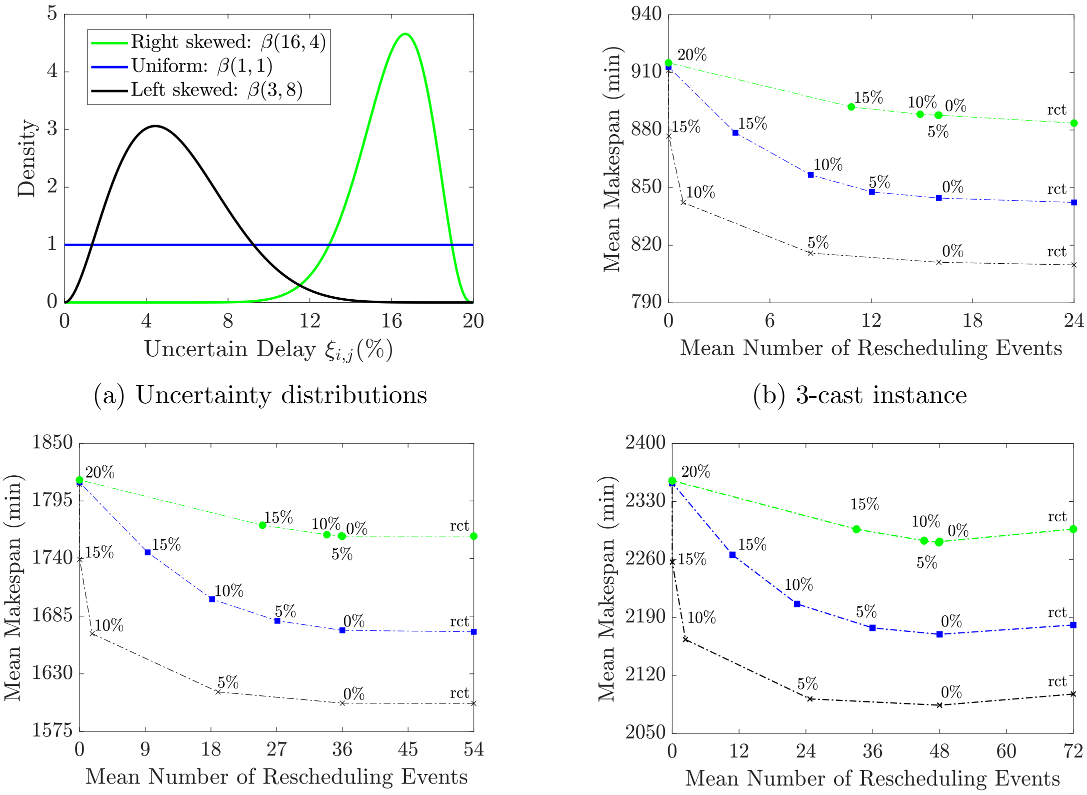
(b) Uniform: $\beta(1,1)$

(c) Right Skewed: $\beta(16,4)$

Figure 3.7: Change in hybrid schedules simulated makespan with respect to the deterministic reactive schedules for the 3-cast, 5-cast and 8-cast instances in the 1^{st}, 2^{nd} and 3^{rd} rows, respectively.

Using the mean simulated makespan and the mean rescheduling events, the pareto fronts for the 3-cast, 5-cast and 8-cast instances are illustrated in Figure 3.8 for three different skewed uncertainty distributions. Adaptive robust proactive scheduling is not desirable due to its conservative solution quality, while the same applies to the deterministic reactive schedule due to the high number of rescheduling events. The lower left region presents the best trade-off between solution quality and frequent rescheduling. The mean simulated makespans exhibited by the adaptive-robust proactive schedule $\Delta_j^{\mathrm{tr}} = 20\%$ for the different distributions at each instance are approximately the same. This outcome is specific for the scheduling problem studied as most independent adaptive decisions are static, not adaptive. Consequently, the schedule does not adapt to the fact that the past realized uncertainty, regardless of the distribution, is less than the anticipated worst case. The pareto fronts also verify that the conservatism of a proactive schedule is dictated by the skewness of the distribution: less conservative with the right-skewed distribution and more conservative with the left-skewed distribution.

The quality of hybrid schedules with $\Delta_j^{\mathrm{tr}} = 0\%$ and $\Delta_j^{\mathrm{tr}} = 5\%$ for the 8-cast instance in the three distributions is comparable if not better than the deterministic reactive schedule marked as "rct" in the Figure. This re-emphasizes the added value from incorporating some of the uncertainty information in determining the sequencing decisions. The number of rescheduling events required for a hybrid schedule can be inferred with the knowledge of the distribution. For example, assuming the delay follows a uniform distribution in EAF and AOD units, the mean rescheduling events for a hybrid schedule with $\Delta_j^{\mathrm{tr}} = \mu = 10\%$ is approximately equal to half of the maximum rescheduling events in the two mentioned units. For the 3-,5- and 8-cast instances, it is approximately equal to 8, 18 and 24, respectively.



(a) Uncertainty distributions

(b) 3-cast instance

(c) 5-cast instance

(d) 8-cast instance

Figure 3.8: Pareto fronts with the mean simulated makespan and mean rescheduling events as the two competing objectives for a 3-, a 5- and an 8-cast instances for a left-, a uniform- and a right-skewed uncertainty distributions.

3.6 Conclusion

This work investigates reactive and proactive scheduling methods for the steel-making and continuous-casting process under processing time uncertainty via adaptive robust optimization. The problem is represented using the unit-specific event-based continuous time formulation. The unit-specific time grid feature in the formulation does not provide any temporal correlation between the realized endogenous uncertainty across processing units. We used mathematical tools named “logical sequencing constraints” to infer temporal precedence of the uncertain parameters in different units and to derive linear decision rules that satisfy the non-anticipativity property.

We then compared the worst case performance of the adaptive robust proactive and the deterministic reactive schedules. For large instances, where the quality of the casts’ sequencing decisions is significant, the proactive schedule exhibits superior performance. This observation showcase how “bad” implemented sequencing decisions limit the possible improvements induced by future rescheduling. Knowing that in real practice processing time delay follows a distribution, we drew the Pareto fronts using mean simulated makespan and mean rescheduling events as the two competing objectives for a right, a uniform and a left skewed distributions. Each front is described by an adaptive robust proactive, a deterministic reactive and an adaptive robust hybrid schedules. The latter schedule immune the process against truncated uncertainty support and triggers a rescheduling only if the realized uncertain delay falls outside the support. Irrespective of the type of the distribution or the number of casts being scheduled, the Pareto fronts highlight the attractiveness of adaptive robust hybrid schedules when compared to either adaptive robust proactive or deterministic reactive schedules. Knowing the characteristics and skewness of the uncertainty distribution offers insightful guidelines to tune the magnitude of the truncated uncertainty support in an adaptive robust hybrid schedule.

Bibliography

Apap, R. M. & Grossmann, I. E. (2017), 'Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties', *Computers & Chemical Engineering* **103**, 233–274.

Bertsimas, D. & Sim, M. (2004), 'The price of robustness', *Operations Research* **52**(1), 35–53.

Chaari, T., Chaabane, S., Aissani, N. & Trentesaux, D. (2014), Scheduling under uncertainty: Survey and research directions, *in* '2014 International conference on advanced logistics and transport (ICALT)', IEEE, pp. 229–234.

Gerardi, D., Marlin, T. E. & Swartz, C. L. E. (2013), 'Optimization of primary steelmaking purchasing and operation under raw material uncertainty', *Industrial & Engineering Chemistry Research* **52**(35), 12383–12398.

Goel, V. & Grossmann, I. E. (2004), 'A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves', *Computers and Chemical Engineering* **28**(8), 1409–1429.

Goel, V. & Grossmann, I. E. (2006), 'A class of stochastic programs with decision dependent uncertainty', *Mathematical Programming* **108**(2), 355–394.

Gupta, D. & Maravelias, C. T. (2016), 'On deterministic online scheduling: Major considerations, paradoxes and remedies', *Computers & Chemical Engineering* **94**, 312–330.

Gupta, D., Maravelias, C. T. & Wassick, J. M. (2016), 'From rescheduling to online scheduling', *Chemical Engineering Research and Design* **116**, 83–97.

Gupta, V. & Grossmann, I. E. (2011), 'Solution strategies for multistage stochastic programming with endogenous uncertainties', *Computers and Chemical Engineering* **35**(11), 2235–2247.

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G. & Wassick, J. (2014), 'Scope for industrial applications of production scheduling models and solution methods', *Computers and Chemical Engineering* **62**, 161–193.

Hong, Y. & Wang, X. (2014), Robust operation optimization in cold rolling production process, *in* 'The 26th Chinese Control and Decision Conference (2014 CCDC)', IEEE, pp. 1365–1370.

Iglesias Escudero, M., Villanueva Balsera, J., Ortega Fernandez, F. & Rodriguez Montequín, V. (2019), 'Planning and scheduling with uncertainty in the steel sector: A review', *Applied Sciences* **9**(13).

Jiang, S. L., Liu, M., Lin, J. H. & Zhong, H. X. (2016), 'A prediction-based online soft scheduling algorithm for the real-world steelmaking-continuous casting production', *Knowledge-Based Systems* **111**, 159–172.

Jiang, S., Liu, M. & Hao, J. (2017), 'A two-phase soft optimization method for the uncertain scheduling problem in the steelmaking industry', *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **47**(3), 416–431.

Jonsbraten, T. W., Wets, R. J. & Woodruff, D. L. (1998), 'A class of stochastic programs with decision dependent random elements', *Annals of Operations Research* **82**, 83–106.

Kammammettu, S. & Li, Z. (2018), 'Multistage adaptive optimization for steelmaking and continuous casting scheduling under processing time uncertainty', *IFAC-PapersOnLine* **51**(21), 262–267.

Lappas, N. H. & Gounaris, C. E. (2016), 'Multi-stage adjustable robust optimization for process scheduling under uncertainty', *AIChE Journal* **62**(5), 1646–1667.

Lappas, N. H. & Gounaris, C. E. (2018), 'Robust optimization for decision-making under endogenous uncertainty', *Computers and Chemical Engineering* **111**, 252–266.

Li, J., Xiao, X., Tang, Q. & Floudas, C. A. (2012), 'Production scheduling of a large-scale steelmaking continuous casting process via unit-specific event-based continuous-time models: Short-term and medium-term scheduling', *Industrial & Engineering Chemistry Research* **51**(21), 7300–7319.

Li, Z. & Ierapetritou, M. (2008), 'Process scheduling under uncertainty: Review and challenges', *Computers & Chemical Engineering* **32**(4-5), 715–727.

Mori, J. & Mahalec, V. (2017), 'Planning and scheduling of steel plates production. Part II: Scheduling of continuous casting', *Computers & Chemical Engineering* **101**, 312–325.

Nohadani, O. & Sharma, K. (2018), 'Optimization under decision-dependent uncertainty', *SIAM Journal on Optimization* **28**(2), 1773–1795.

Noshadravan, A., Gaustad, G., Kirchain, R. & Olivetti, E. (2017), 'Operational strategies for increasing secondary materials in metals production under uncertainty', *Journal of Sustainable Metallurgy* **3**(2), 350–361.

Ouelhadj, D. & Petrovic, S. (2009), 'A survey of dynamic scheduling in manufacturing systems', *Journal of Scheduling* **12**(4), 417–431.

Pacciarelli, D. & Pranzo, M. (2004), 'Production scheduling in a steelmaking-continuous casting plant', *Computers and Chemical Engineering* **28**(12), 2823–2835.

Poss, M. (2014), 'Robust combinatorial optimization with variable cost uncertainty', *European Journal of Operational Research* **237**(3), 836–845.

Romero, J., Puigjaner, L., Holczinger, T. & Friedler, F. (2004), 'Scheduling intermediate storage multipurpose batch plants using the s-graph', *American Institute of Chemical Engineers. AIChE Journal* **50**(2), 403–417.

Rong, A. & Lahdelma, R. (2008), 'Fuzzy chance constrained linear programming model for optimizing the scrap charge in steel production', *European Journal of Operational Research* **186**(3), 953–964.

Ruiz, R. & Vázquez-Rodríguez, J. A. (2010), 'The hybrid flow shop scheduling problem', *European Journal of Operational Research* **205**(1), 1–18.

Shaik, M. A., Janak, S. L. & Floudas, C. A. (2006), 'Continuous-time models for short-term scheduling of multipurpose batch plants: A comparative study', *Industrial & Engineering Chemistry Research* **45**(18), 6190–6209.

Sun, L., Luan, F. & Pian, J. (2015), 'An effective approach for the scheduling of refining process with uncertain iterations in steel-making and continuous casting process', *IFAC PapersOnLine* **48**(3), 1966–1972.

Wang, D. J., Liu, F. & Jin, Y. (2017), 'A proactive scheduling approach to steel rolling process with stochastic machine breakdown', *Natural Computing* .

Worapradya, K. & Thanakijkasem, P. (2010), Worst case performance scheduling facing uncertain disruption in a continuous casting process, *in* '2010 IEEE International Conference on Industrial Engineering and Engineering Management', IEEE, pp. 291–295.

Ye, Y., Li, Z., Li, J., Tang, Q., Xiao, X. & Floudas, C. A. (2014), 'Robust optimization and stochastic programming approaches for medium-term production scheduling of a large-scale steelmaking continuous casting process under demand uncertainty', *Computers and Chemical Engineering* **66**, 165–185.

Yu, S. (2013), 'A prediction method for abnormal condition of scheduling plan with operation time delay in steelmaking and continuous casting production process', *ISIJ International* **53**(6), 1028–1041.

Yu, S. P., Pang, X. F., Chai, T. Y. & Zheng, B. L. (2009), 'Research on production scheduling for steelmaking and continuous casting with processing time uncertainty [J]', *Control and Decision* **24**(10), 1467–1472.

Chapter 4

Deep Lifted Decision Rules for Two-Stage Adaptive Optimization

4.1 Introduction

Multi-stage adaptive optimization is a framework to handle sequential decision-making problems that arise in numerous business and engineering disciplines. It has elicited growing attention due to its ability to manage uncertainty in the decision making process. Arguably the three most popular modeling paradigms for multi-stage adaptive optimization are stochastic programming (SP) (Shapiro et al. 2009), robust optimization (RO) (Ben-Tal et al. 2009), and stochastic dynamic programming (SDP) (Ross 2014). SP is widely used in operations research applications, SDP has roots in control theory and RO is popular in engineering and operations research problems where the failure to manage all uncertain events can lead to severe consequences.

As described by Georghiou et al. (2019), the decision rule approach to decision making under uncertainty is quite expressive and flexible, encompassing multi-stage stochastic problems, chance-constrained problems, and static and adaptive robust optimization problems. When coupled with closed polyhedral or conic uncertainty sets, the approach inherits the tractability of robust optimization, without sacrificing the essential dynamics of a multi-stage optimization problem under uncertainty. It can therefore be seen as a hybrid approach, although RO is its nearest neighbour methodologically.

A decision rule is nothing more than a function mapping realized uncertain parameters to implementable decisions. In the reinforcement learning community, a decision rule is known more generally as a *policy*, "*any* function that returns an action given a state" (Powell 2011, p.233). Multi-stage adaptive optimization categorizes decisions into non-adaptive and adaptive groups. The former, also known as "here-and-now" or "first-stage" decisions, are independent of any future uncertain information and are made at the outset of the problem. In contrast, the latter, also known as "wait-and-see" decisions, are implemented gradually in response to the revealed uncertain information. Since, at the outset of a multi-stage adaptive optimization problem, we do not know the realizations of future uncertain parameters, a decision rule prescribes a recourse decision to take once the history of the uncertain parameters has been realized up to and including that decision stage.

One of the main limitations of decision rule-based methods, and ultimately a key motivation for this work, is that they require the user to specify the structural form of the approximate policy before optimizing. To date, practitioners must propose a structural form - with linear, quadratic, and piecewise linear decision rules being by far the most popular - and then empirically test its suitability. Unfortunately, it is not possible, in general, to deduce an optimal policy's structure a priori. This conundrum brings us to the driving question for our research: Can we relax the requirement of having the user specify a fixed policy structure and instead allow this structure to be as flexible as possible via a sequence of simple operations?

To answer this question, we turn to basic concepts used to train deep neural networks (DNNs), which have ultimately helped popularize deep learning. At its core, a DNN is a functional approximator. Given a set of inputs, the user would like to determine outputs that best match the existing sample output. To determine what could be a complex approximation, a DNN performs a succession of simple operations, namely by passing input through a feedforward network with simple activation functions and optimizing activation function weights in the process. Most importantly, the user does not need to assume the approximate function's structure a priori; it is learned during the training process. In this work, we attempt to demonstrate that cross-pollinating deep learning concepts (which are used for learning) with decision rule concepts (which are used for stochastic optimization) can lead to improved solution methods for stochastic adaptive optimization problems.

4.1.1 Literature Review

Initially proposed by Garstka & Wets (1974), if not early, decision rule-based methods came to prominence as a viable approach for solving stochastic optimization problems only after the seminal chapter of Ben-Tal et al. (2004). They assumed adaptive policies are linearly dependent on the uncertain parameters to derive a linear deterministic counterpart, under a polytopic uncertainty set, of a multi-stage linear adaptive optimization problem. Due to their favorable modeling characteristics, linear decision rules have been implemented in several fields such as supply chain (Ben-Tal et al. 2005), power systems management (Zugno et al. 2016) and scheduling (Rahal et al. 2020, Zhang et al. 2016), to name few.

Ben-Tal & Den Hertog (2011) introduced a more complex approximation which defines the adaptive policies as quadratic functions of the uncertain parameters. The derived deterministic counterpart, under an ellipsoidal uncertainty set, is a second-order cone programming problem. Bertsimas et al. (2011) proposed polynomial decision rules in multi-stage robust dynamic problems. The increase in the complexity/flexibility of the adaptive policies comes at the expense of a more complex deterministic counterpart which, under an intersection of convex uncertainty sets, is a semidefinite programming problem. Bampou & Kuhn (2011) similarly proposed polynomial decision rules in the context of stochastic programming.

The well-known trade-off between solution quality and computational cost is evident here. Increasing an adaptive policy flexibility by simply increasing its complexity is not an attractive direction. An alternative paradigm is to refine or transform the uncertainty set to improve the quality of less complex policies. In this regard, Chen & Zhang (2009) proposed an extended linear decision rule using an extended uncertainty set defined via the positive and negative perturbations of the original uncertain parameters. The linear approximation over the extended uncertainty space projects into a piecewise linear approximation over the original uncertainty set with two linear segments. Georghiou et al. (2015) introduced lifted decision rules which are based on a one-to-one correspondence between a set of linear functions in the lifted uncertainty space and a family of piecewise linear functions in the original uncertainty space. Rahal et al. (2021) devised hybrid lifted decision rules which emphasize higher refinement of the lifted uncertainty space in the short- rather than the long-term time horizon.

Hanasusanto et al. (2015) partitioned the uncertainty space to compute k-adaptive linear policies. The optimal implemented policy depends on the value of the observed uncertainty. Ben-Tal et al. (2018) proposed an approximate piecewise-linear policy for two-stage robust linear problems by approximating the uncertainty set via a dominated simplex. The proposed approximation method scales well and is computationally efficient, however the constructed policy is not guaranteed to outperform a linear decision rule approximation for budget uncertainty sets or an intersection of thereof. Yanıkoğlu et al. (2019) surveyed decision rule theory, applications and methodologies in the context of robust optimization. Computational tools such as ROME (Goh & Sim 2011), RSOME (Chen et al. 2020) and JuMPeR (Dunning 2016) have been developed in response to the wide implementation of decision rule-based methods for optimization under uncertainty.

Reinforcement learning, also known as approximate dynamic programming, is a common solution method that utilizes, in one of its forms, the prowess of an artificial neural network in approximating multi-stage stochastic decision problems (Lee et al. 2018). The problem is typically modeled as a Markov decision process after which a designed network approximates complex functions to output predictions, after being trained, based on a given input (i.e., observed uncertainty) (Lee & Lee 2006). Similarly, Han (2016) introduced deep learning approximations for multi-stage stochastic control problems. The network architecture is composed of interconnected sub-networks; each sub-network has multiple hidden layers. The authors advocate that their approach overcomes the "curse of dimensionality" in solving high-dimensional problems. Huré et al. (2018) developed convergence analysis for a proposed deep neural network algorithm for stochastic control problems followed by a set of numerical applications in Bachouch et al. (2018).

Deep learning via a deep neural network (DNN) has been increasingly incorporated in speech recognition, recommender systems, objection detection and other applications (LeCun et al.

2015). Though the focus of this chapter is on solving two-stage adaptive stochastic optimization problems, we emphasize the capability of DNNs, through successive applications of simple operations, to extract complex features from big data sets. Each layer consists of an affine transformation followed by a non-linear operator, known as "activation function," at each node. Among others, the rectified linear activation unit (ReLU) is a piecewise linear function with two linear segments and a breakpoint at 0. The slope of the positive and negative ranges are 1 and 0, respectively. The idea of using a sequence of simple operations to construct complex features/structures is a catalyst for this work. Instead of using a ReLU as the non-linear operator, we use the lifting operator introduced by Georghiou et al. (2015). Further, we introduce different solution methodologies to optimize/train our proposed network to devise the optimal decision rule structure.

4.1.2 Contributions

The contributions of this chapter are:

1. We propose a "deep lifting" network to generate complex functions to approximate the adaptive policies in two-stage stochastic adaptive optimization problems. Taking the original uncertain parameters as inputs, the network consists of multiple processing layers that enable the construction of complex lifted parameters. Each layer consists of two operations: an affine transformation followed by a lifting operator at each node. The coupling was introduced by Georghiou et al. (2015) and coined as generalized lifting, which is equivalent to a deep lifting network with a single layer. The functions used to define the adaptive policies are linear functions of the uncertain primitive vector, the lifted uncertain parameters in all layers and an intercept. We call the resulting policies "deep lifted decision rules" to reflect the two critical components upon which the approach is founded: (a) the lifting operations applied to the uncertainty set in each layer and (b) the connections to deep neural networks through the succession of simple operations.

2. We present two solution methods for two-stage stochastic adaptive optimization problems with fixed cost coefficients, recourse coefficients and right-hand-side uncertainty. In the first method, we optimize the deep lifting network within a black-box optimizer using a differential evolutionary algorithm. In the second method, the solution is obtained using a derivative-based approach via first- and second-order approximate derivatives. We further show the benefits of piecewise linear decision rules (i.e., PLDRs) using a deep lifting network in comparison to linear and axial piecewise linear decision rules.

3. We introduce local-search heuristics to optimize deep lifting networks. We empirically verify that the proposed heuristics generate flexible deep lifted decision rules while reducing the computational cost by several folds. We also illustrate that the proposed heuristics scale well with large dimensional instances.

4.2 Problem statement

In this work, we address the solution of a two-stage stochastic adaptive optimization problem with fixed cost coefficients and a recourse matrix given in Model (4.1).

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{y}(\boldsymbol{\xi})} \quad & \mathbf{c}^\top \mathbf{x} + \mathbb{E}\left[\mathbf{q}^\top \mathbf{y}(\boldsymbol{\xi})\right] && \text{(4.1a)}\\
\text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} && \text{(4.1b)}\\
& \mathbf{T}\mathbf{x} + \mathbf{D}\mathbf{y}(\boldsymbol{\xi}) \geq \mathbf{h}(\boldsymbol{\xi}) \qquad \forall \boldsymbol{\xi} \in \Xi && \text{(4.1c)}
\end{aligned}$$

We assume that the uncertainty set Ξ is a well-defined polytope. We also assume a linear uncertainty dependence for the right-hand-side uncertain vector $\mathbf{h}(\boldsymbol{\xi}) = \mathbf{H}\boldsymbol{\xi}$. Model (4.1) is intractable in its general form due to the presence of the semi-infinite constraints required to be satisfied over Ξ. This is circumvented by approximating the adaptive policies via either a discrete set of deterministic scenario-based decisions or decision rules to derive a tractable deterministic counterpart.

Motivating deep lifted decision rules

In this section, we motivate the need for more complex decision rules by illustrating the limitations of an axial lifting-based PLDR using a two-stage adaptive transportation problem given in equation (4.2). It is fair to assume that a piecewise linear function, with a large number of linear segments, can accurately approximate a general function. Likewise, one may expect that axial PLDRs, with large number of linear segments, can accurately approximate complex second-stage policies and thus drive the approximate optimal solution towards the true optimal solution. While Georghiou et al. (2015) also showed the potential inferiority of axial lifting-based PLDRs, we present a new example to clearly motivate our approach.

The transportation problem objective is to maximize the expected profit by a set of suppliers $\mathcal{I}$ given uncertain consumer demand $D_j(\boldsymbol{\xi})$ for all $j \in \mathcal{J}$. The production amount x_i by supplier i is determined at the outset of the problem, while the distribution policy $y_{i,j}(\boldsymbol{\xi})$ from supplier i to consumer j is determined after the uncertain demands are observed. Equation (4.2b) dictates that the maximum distributed amount out of supplier i does not exceed the produced amount and equation (4.2c) limits the distributed amount to consumer j by its uncertain demand.

$$\begin{aligned}
\min_{x_i, y_{i,j}(\boldsymbol{\xi})} \quad & \sum_{i\in\mathcal{I}} C_i x_i - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \Big(R_{i,j} - T_{i,j}\Big)\mathbb{E}\big[y_{i,j}(\boldsymbol{\xi})\big] && && \text{(4.2a)}\\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} y_{i,j}(\boldsymbol{\xi}) \leq x_i && \forall i \in \mathcal{I}, \boldsymbol{\xi} \in \Xi && \text{(4.2b)}\\
& \sum_{i\in\mathcal{I}} y_{i,j}(\boldsymbol{\xi}) \leq D_j(\boldsymbol{\xi}) && \forall j \in \mathcal{J}, \boldsymbol{\xi} \in \Xi && \text{(4.2c)}\\
& y_{i,j}(\boldsymbol{\xi}) \geq 0 && \forall i \in \mathcal{I}, j \in \mathcal{J}, \boldsymbol{\xi} \in \Xi && \text{(4.2d)}\\
& x_i \geq 0 && \forall i \in \mathcal{I} && \text{(4.2e)}
\end{aligned}$$

We have three suppliers, two consumers and uniformly distributed uncertain parameters $\xi_j \sim \mathcal{U}(1,3)$ for all j. We assume a polyhedral uncertainty set $\Xi := \{\boldsymbol{\xi} = (\xi_1, \xi_2) | \, \boldsymbol{A}\boldsymbol{\xi} \geq \boldsymbol{b}\}$ where $\boldsymbol{A} = (-1, 0; 0, 1; 1, 1)$ and $\boldsymbol{b} = (-3; -3; 4)$. The uncertain consumer demand is given as $D_j(\boldsymbol{\xi}) = \xi_j$ for all j and the cost parameters are shown in Table 4.1 and explained in the nomenclature section.

Table 4.1: Two-stage transportation problem cost parameters.

	C_i	T_{ij}		R_{ij}	
$i \downarrow, j \rightarrow$		1	2	1	2
1	10	1	5	13	19
2	12	3	4	16.5	20
3	10	6	2	21	14

To give an idea of what the true optimal second-stage adaptive solution looks like, we solve a scenario-based approximation of equation (4.2). For 1,326 scenarios obtained by discretizing Ξ into a two-dimensional grid with equidistant scenarios taken at step size of 0.04, the optimal scenario-based profit is 18.20. The scenario-based distribution decisions are illustrated in Figure 4.1. Non-axial facets on the surface of $y_{1,1}(\xi_1, \xi_2)$, $y_{1,2}(\xi_1, \xi_2)$ and $y_{3,2}(\xi_1, \xi_2)$ are highlighted in red. We also observe at most 3 linear pieces on the surface of the scenario-based distribution decisions. The main motivation behind our work is the fact that a decision maker does not know and can rarely predict the complex structure of adaptive policies. Instead, we propose a systematic approach to recover more complex decision rules that better approximate an optimal policy. In the next section, we empirically illustrate that, even for this simple two-stage stochastic program with a two-dimensional uncertainty set, the state-of-the-art PLDR approach is indeed limited and offers no guarantee on improving the solution consistently with the increase in the PLDR complexity.

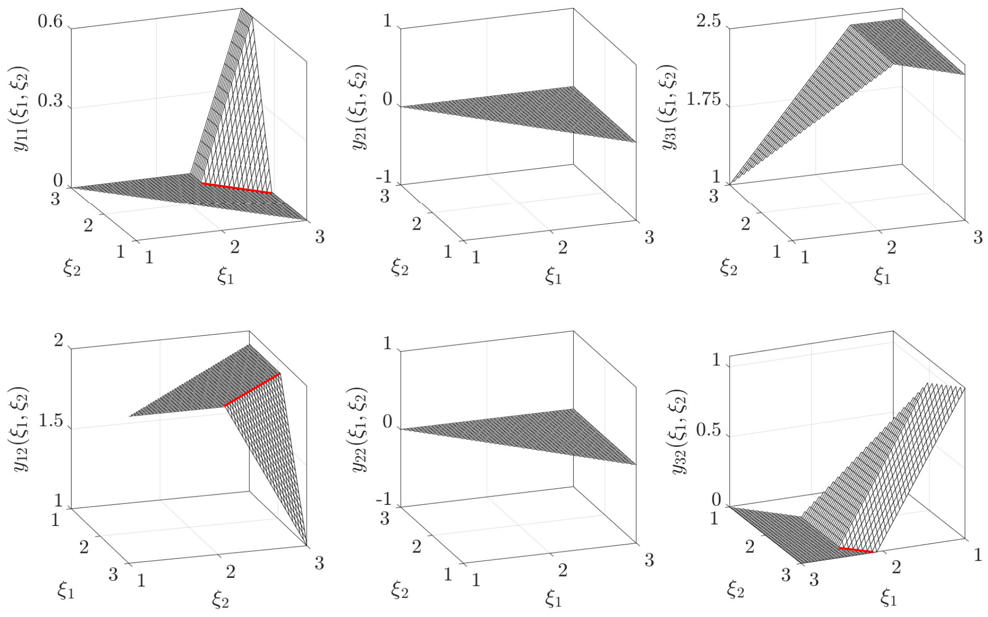


Figure 4.1: Optimal scenario-based recourse decisions for a two-stage transportation problem instance using 1,326 equidistant scenarios.

4.3 Limitation of LDRs and axial lifting based PLDRs

In this section, we introduce linear decision rules and illustrate the optimal distribution policies of the transportation problem given by formulation (4.2). We then restate concepts of lifted decision rules originally introduced by Georghiou et al. (2015) and highlight a possible limitation of this type of decision rule.

4.3.1 Linear decision rules

Popular thanks to their simplicity and interpretability, linear decision rules define a linear dependence between the adaptive policies and uncertain parameters. The general form for second-stage decisions is given in equation (4.3) where $y(\boldsymbol{\xi})$ is a vector in $\mathbb{R}^m$, $\mathbf{Y}$ is a matrix in $\mathbb{R}^{m\times(n+1)}$ and $\xi_0 = 1$.

$$y(\boldsymbol{\xi}) = \mathbf{Y}(\xi_0; \xi_1; \ldots; \xi_n) = \mathbf{Y}\boldsymbol{\xi} \tag{4.3}$$

Also known as affine decision rules because of the presence of an intercept, the LDR-based approximation model possesses a tractable stochastic counterpart using robust optimization techniques. As shown in C.2, when the uncertainty set Ξ is a polytope, the stochastic counterpart of (4.1) is a linear program. Revisiting our two-stage transportation problem in (4.2), its

LDR-based approximation is given in equation (4.4).

$$
\begin{aligned}
\min_{x_i, \mathbf{Y}_{i,j}} \quad & \sum_{i \in \mathcal{I}} C_i x_i - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \Big(R_{i,j} - T_{i,j} \Big) \mathbf{Y}_{i,j} \mathbb{E}[\boldsymbol{\xi}] && && \text{(4.4a)} \\
\text{s.t.} \quad & \sum_{j \in \mathcal{J}} \mathbf{Y}_{i,j} \boldsymbol{\xi} \leq x_i && \forall i \in \mathcal{I}, \boldsymbol{\xi} \in \Xi && \text{(4.4b)} \\
& \sum_{i \in \mathcal{I}} \mathbf{Y}_{i,j} \boldsymbol{\xi} \leq \xi_j && \forall j \in \mathcal{J}, \boldsymbol{\xi} \in \Xi && \text{(4.4c)} \\
& \mathbf{Y}_{i,j} \boldsymbol{\xi} \geq 0 && \forall i \in \mathcal{I}, j \in \mathcal{J}, \boldsymbol{\xi} \in \Xi && \text{(4.4d)} \\
& x_i \geq 0 && \forall i \in \mathcal{I} && \text{(4.4e)}
\end{aligned}
$$

where $\mathbb{E}[\boldsymbol{\xi}]$ is the expected uncertainty vector in $\mathbb{R}^{n+1}$ and the polyhedral uncertainty set Ξ is reformulated to accommodate ξ_0.

The optimal linear decision rule-based profit is 17 which corresponds to a deterioration of 6.60% in comparison with the scenario-based solution. Figure 4.2 illustrates the optimal linear distribution policies. Similar to the scenario-based solution, the LDR-based solution does not include any produced amount via supplier 2. On the other hand, each of the distribution policies out of suppliers 1 and 3 exhibits a single partition and poorly approximates the structure observed in the scenario-based distribution decisions in Figure 4.1.

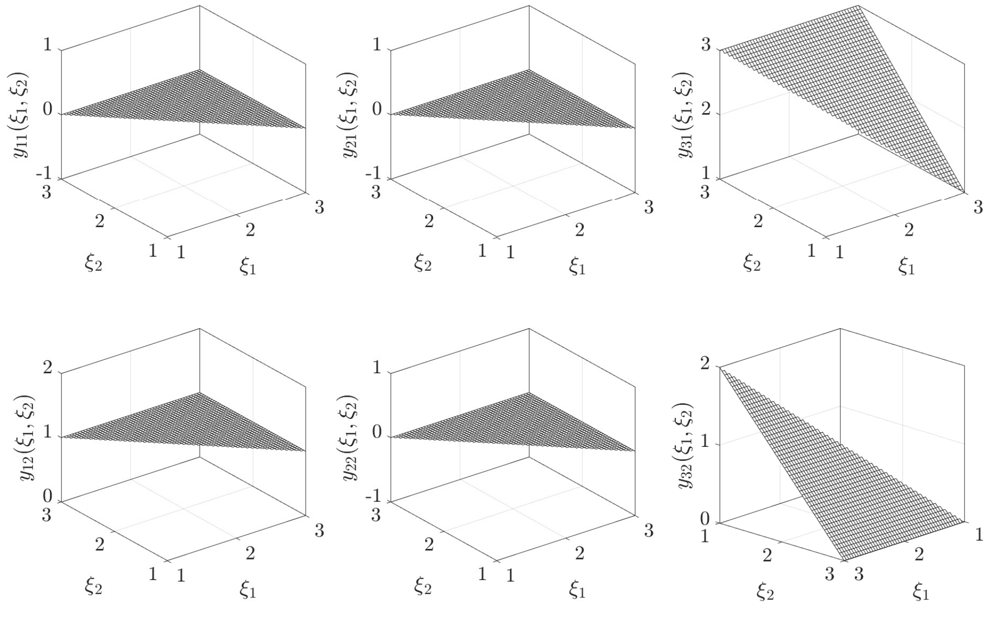


Figure 4.2: Optimal linear adaptive policies for a two-stage transportation problem with a profit 6.6% less than the stochastic scenario-based optimal profit.

4.3.2 Piecewise linear decision rules with axial lifting

Before illustrating the limited ability of PLDRs with axial lifting to recover non-axial facets present in the second-stage policies, we restate the lifted decision rules methodology which was introduced by Georghiou et al. (2015). In their framework, a lifting operator $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}^{n'}$ maps an uncertain vector to a lifted, higher dimension ($n' > n$), space. Inversely, they introduce a retraction operator $\mathcal{R} : \mathbb{R}^{n'} \to \mathbb{R}^n$ that retracts the lifted uncertain vector back to the original space. The adaptive linear approximation in the lifted space represents a piecewise linear approximation in the original space. Hence, the modeling features of a linear decision rule is maintained while offering the solution quality of a non-linear policy. The lifted uncertain vector obtained by lifting an original parameter ξ_i is $\boldsymbol{\xi}'_i = (\xi'_{i,1}, \dots, \xi'_{i,r_i})^\top$ where r_i is the number of linear pieces in ξ_i. Given uncertain parameter bounds $l_j \leq \xi_j \leq u_j$, the lifting operator $\mathcal{L}_{i,j}(\xi_i)$, defined in Georghiou et al. (2015) and restated in equation (4.5), maps ξ_i to $\xi'_{i,j}$.

$$\xi'_{i,j} = \begin{cases} \xi_i & \text{if } r_i = 1, \\ \min\{\xi_i, z_{i,j}\} & \text{if } r_i > 1,\ j = 1, \\ \max\big\{\min\{\xi_i, z_{i,j}\} - z_{i,j-1}, 0\big\} & \text{if } r_i > 1,\ j = 2, \dots, r_i - 1 \\ \max\{\xi_i - z_{i,j-1}, 0\} & \text{if } r_i > 1,\ j = r_i, \end{cases} \quad \forall i = 1, \dots, n, j = 1, \dots, r_i \tag{4.5}$$

where $z_{i,j}$ is the j^{th} breakpoint in ξ_i where a slope change may occur. The retraction defined in Georghiou et al. (2015) is given in equation (4.6) where $\mathbf{R}_i$ is a row vector in $\mathbb{R}^{r_i}$.

$$\xi_i = \mathbf{R}_i \boldsymbol{\xi}'_i = \sum_{j=1}^{r_i} \xi'_{i,j} \quad \forall i = 1, \dots, n \tag{4.6}$$

The lifted polyhedral uncertainty set Ξ' is a subspace of $\mathbb{R}^{n'}$ where the dimension parameter n' is equal to the summation of the linear pieces in all original uncertain parameters $\sum_{i=1}^{n} r_i$. It was introduced by Georghiou et al. (2015) and it is included in C.3 for a quick reference. The overall lifted stochastic counterpart derivation is similar to that of the linear stochastic counterpart since both decision rules are linear, though in different spaces, and both uncertainty sets are polyhedral.

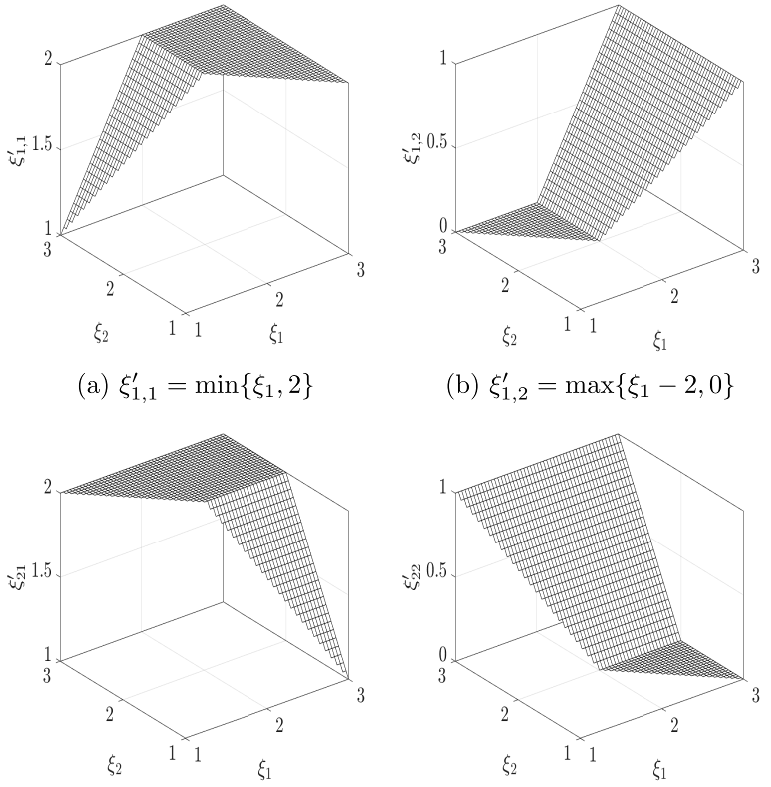
PLDRs define a linear dependence between the adaptive policies and the lifted uncertain parameters. The general form of second-stage PLDRs is given in equation (4.7) where $y(\boldsymbol{\xi}')$ is a vector in $\mathbb{R}^m$, $\mathbf{Y}'$ is a matrix in $\mathbb{R}^{m \times (n'+1)}$ and $\xi_0 = 1$.

$$y(\boldsymbol{\xi}') = \mathbf{Y}'(\xi_0; \boldsymbol{\xi}'_1; \dots; \boldsymbol{\xi}'_n) = \mathbf{Y}'\boldsymbol{\xi}' \tag{4.7}$$

The lifted uncertain vector is modified to accommodate an intercept term in the decision rule, via the parameter ξ_0, in a more compact representation. The polyhedral lifted uncertainty set is also adjusted accordingly. Revisiting the two-stage transportation problem in (4.2), the PLDR-based approximation is given in equation (4.8).

$$
\begin{aligned}
\min_{x_i, \mathbf{Y}'_{i,j}} \quad & \sum_{i \in \mathcal{I}} C_i x_i - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \Big(R_{i,j} - T_{i,j} \Big) \mathbf{Y}'_{i,j} \mathbb{E}[\boldsymbol{\xi}'] && && \text{(4.8a)} \\
\text{s.t.} \quad & \sum_{j \in \mathcal{J}} \mathbf{Y}'_{i,j} \boldsymbol{\xi}' \leq x_i && \forall i \in \mathcal{I}, \boldsymbol{\xi}' \in \Xi' && \text{(4.8b)} \\
& \sum_{i \in \mathcal{I}} \mathbf{Y}'_{i,j} \boldsymbol{\xi}' \leq \sum_{p=1}^{r_j} \xi'_{j,p} && \forall j \in \mathcal{J}, \boldsymbol{\xi}' \in \Xi' && \text{(4.8c)} \\
& \mathbf{Y}'_{i,j} \boldsymbol{\xi}' \geq 0 && \forall i \in \mathcal{I}, j \in \mathcal{J}, \boldsymbol{\xi}' \in \Xi' && \text{(4.8d)} \\
& x_i \geq 0 && \forall i \in \mathcal{I} && \text{(4.8e)}
\end{aligned}
$$

where $\mathbb{E}[\boldsymbol{\xi}']$ is the expected lifted uncertain vector in $\mathbb{R}^{n'+1}$. A decision maker has to determine the number and location of all breakpoints for PLDRs before the solution. Figure 4.3 illustrates the lifted parameters generated via lifting the two uncertain parameters ξ_1 and ξ_2 using a single breakpoint at 2 in each parameter. For the adaptive distribution decision, we define a PLDR as the linear combination of the four lifted parameters and an intercept. This limits the possible structure exhibited by the optimal solution as the optimizer can only change the slope of the approximation across each parameter only at $\xi_j = 2$. As a result, a PLDR will have at most four linear pieces.



(a) $\xi'_{1,1} = \min\{\xi_1, 2\}$

(b) $\xi'_{1,2} = \max\{\xi_1 - 2, 0\}$

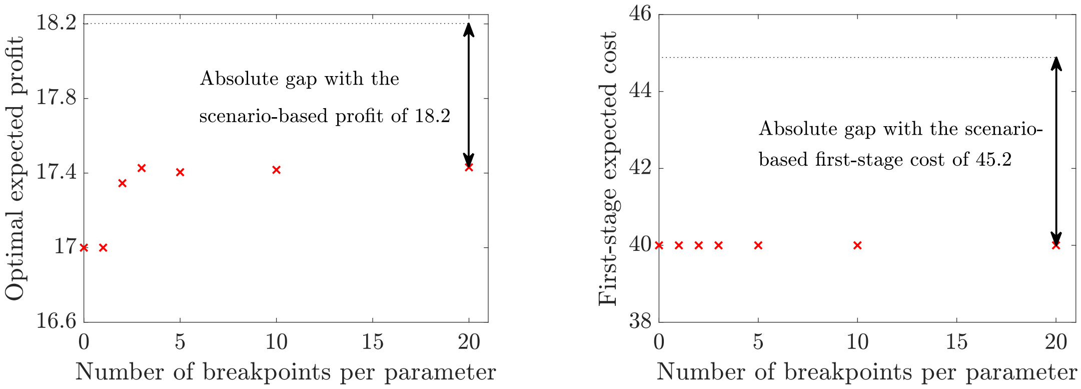
(c) $\xi'_{2,1} = \min\{\xi_2, 2\}$

(d) $\xi'_{2,2} = \max\{\xi_2 - 2, 0\}$

Figure 4.3: Four lifted uncertain parameters in the two-stage transportation instance given by (4.2) using a breakpoint at $\xi_1 = \xi_2 = 2$.

One may argue that complex axial lifting-based PLDRs with a large number of breakpoints in each parameter will recover the true optimal distribution policies despite the fixed a priori structures. To address this argument, a modified version of JuMPeR which is an extension of JuMP (Dunning et al. 2017), a modeling language for mathematical optimization embedded in Julia, is used to solve (4.8) for different breakpoint settings. The modifications include an automated construction of PLDRs with axial lifting, for any given number of breakpoints and breakpoint values, and the modeling of an expectation-based objective function.

Figure 4.4a shows the PLDR-based optimal profit with an increasing number of breakpoints in both parameters ξ_1 and ξ_2. Knowing that the scenario-based solution is equal to 18.2, we found that the solution of axial PLDRs plateaus at an optimal profit of 17.41 (i.e., a 4.3% gap with the scenario-based profit) using 5 evenly distributed breakpoints in each parameter. Figure 4.5 illustrates the optimal distribution policies using 20 evenly distributed breakpoints. Going a step further, we verified that this plateau occurs for an axial PLDR with 100 evenly distributed breakpoints in each parameter. This behavior even with 100 breakpoints in each parameter is not shown in Figure 4.5. Likewise, the increase in the axial PLDRs complexity does not improve the first-stage optimal costs/decisions as illustrated in Figure 4.4b. The complex axial PLDRs failed to realize an opportunity to pay more in the first-stage in return for a better return in the second-stage. The complex axial PLDRs limitation is caused by the poorly approximate non-axial facets shown in Figure 4.1.



(a)

(b)

Figure 4.4: (a) Axial PLDR solution plateaus at an optimal profit of 17.41 for 5 evenly distributed breakpoints in both ξ_1 and ξ_2 for the two-stage transportation problem in (4.8). (b) Axial PLDRs do not realize an opportunity to pay a higher first-stage cost in return for a higher overall profit .

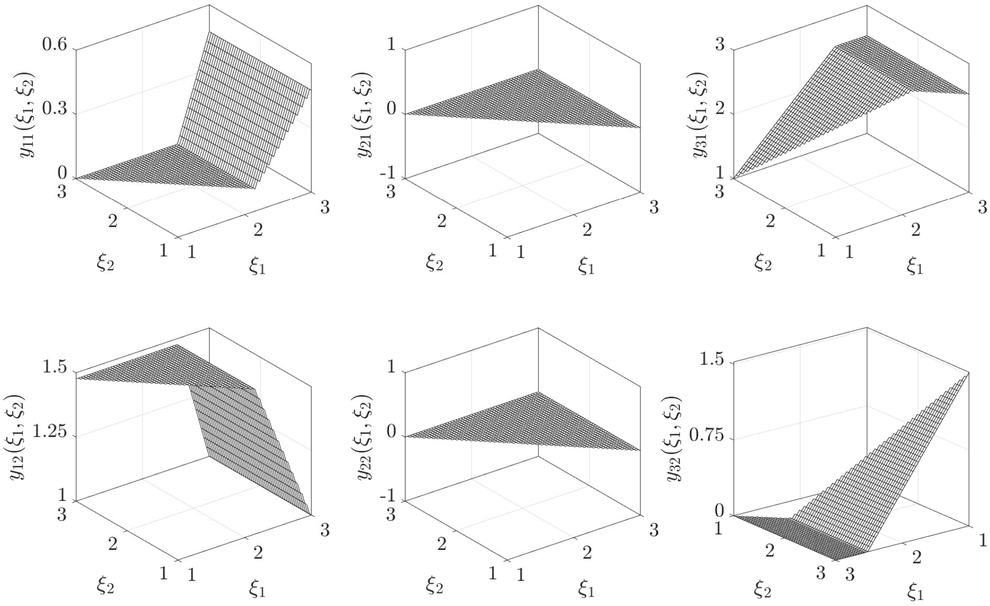


Figure 4.5: Optimal axial lifting-based piecewise linear adaptive policies for the two-stage transportation problem with 20 evenly distributed breakpoints in ξ_1 and ξ_2.

Georghiou et al. (2015) also introduced lifted decision rules based on lifting the affine images of the original parameters (e.g., $\xi_3 = 0.8\xi_1 - 0.4\xi_2$). The constructed decision rule is also known as a PLDR with general lifting and it represents the foundation at which we develop the deep lifting strategy in the next section.

4.4 Constructing piecewise linear decision rules using deep lifting

4.4.1 Deep lifting

A deep lifting network comprises a sequence of two simple operations: an affine transformation followed by a lifting operator at each node. Combining ideas from deep neural networks and lifted decision rules, we believe that a deep lifting network is capable of generating complex and flexible policies at an attractive computational cost. To be clear, we highlight that the term "deep lifting" is not meant for machine learning. Rather, it describes a lifting strategy used in constructing decision rules for two-stage stochastic adaptive optimization.

Figure 4.6 illustrates an example of a deep lifting network. The input is a two-dimensional uncertain vector and the number of layers is three. The image of the transformation in each layer is reflected in two nodes/parameters (i.e., unfilled circles) and each node is lifted using a single breakpoint. An intercept value is included in each affine transformation. The weight matrices in all layers share the same size except in the first layer. It depends on the input uncertainty dimension. A deep lifted decision rule is defined as an affine function of the uncertain input parameters (i.e., squares) and the lifted parameters (i.e., filled circles) in all layers. For notation purposes, the input uncertain vector is given as $\boldsymbol{\xi}^0 \equiv \boldsymbol{\xi} \in \mathbb{R}^n$. The original and lifted parameters in all layers $k \in \{1, ..., L\}$ are denoted by $\boldsymbol{\xi}^k \in \mathbb{R}^{n^{\text{node}}}$ and $\boldsymbol{\alpha}^k \in \mathbb{R}^{n^{\text{node}}(n^{\text{brkp}}+1)}$, respectively. While the number of nodes in each layer n^{node} and number of breakpoints for each node n^{brkp} is flexible, we assume (i) the number of nodes is the same in all layers, (ii) the number of breakpoints is the same at all nodes and (ii) the breakpoints are evenly distributed for each uncertain parameter.

We emphasize two features that differentiate a deep lifting network from a DNN. First, the usage of the deep lifting network is not restricted to the last layer. In a deep lifting network, the optimal adaptive decision rules are constructed as a linear combination of the original uncertain parameters (i.e., $\boldsymbol{\xi}^0$), the image of the affine mapping in all layers (i.e., $\boldsymbol{\xi}^k$), lifted parameters of all $\boldsymbol{\xi}^k$ (i.e., $\boldsymbol{\alpha}^k$) and an intercept. Second, the output of lifting at a given node in a deep lifting network leads to the "creation" of multiple new nodes. The behaviour, which does not exist in a DNN, is reflected in each layer by two types/columns of variables $\boldsymbol{\xi}^k$ and $\boldsymbol{\alpha}^k$ present in the definition of an adaptive policy. It is illustrated in Figure 4.6 using the unfilled and filled circles, respectively.

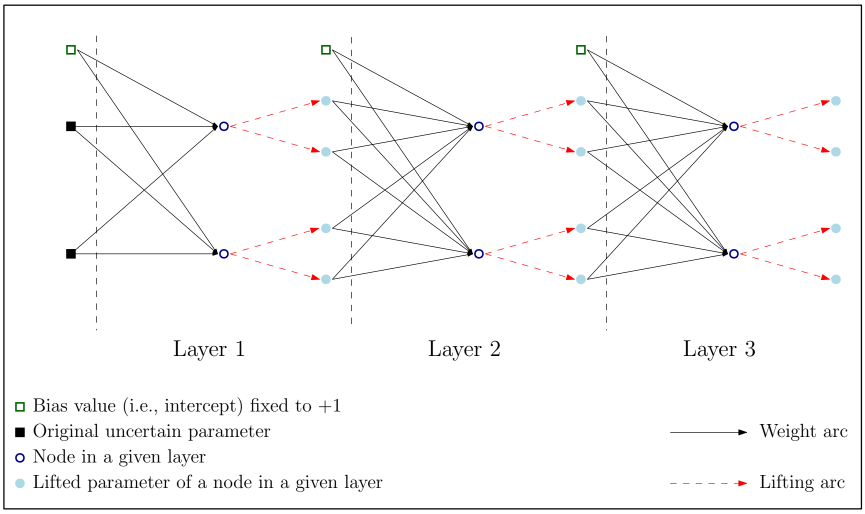

Layer 1
Layer 2
Layer 3
□ Bias value (i.e., intercept) fixed to +1
■ Original uncertain parameter
○ Node in a given layer
● Lifted parameter of a node in a given layer
Weight arc
Lifting arc

Figure 4.6: A deep lifting network with three layers, two nodes in each layer and one breakpoint at each node.

The affine transformation in each layer of a deep lifting network is given in equation (4.9). A special deep lifting network corresponds to a piecewise linear decision rule with general lifting introduced by Georghiou et al. (2015). It arises when the affine mapping image dimension is the same as the source dimension. For the specific network setting shown in Figure 4.6, we get $\boldsymbol{\xi}^1 \in \mathbb{R}^2$, $\boldsymbol{\alpha}^1 \in \mathbb{R}^4$, $\boldsymbol{\xi}^2 \in \mathbb{R}^4$, $\boldsymbol{\alpha}^2 \in \mathbb{R}^8$, $\boldsymbol{\xi}^3 \in \mathbb{R}^8$ and $\boldsymbol{\alpha}^3 \in \mathbb{R}^{16}$. A PLDR with axial lifting is a specific case of the special deep lifting network where there is only a single non-zero weight arc out of a source parameter and a single non-zero weight arc out into an image parameter. The weight arcs related to the intercept values are insignificant in this case.

$$\begin{aligned} \boldsymbol{\xi}^1 &= \boldsymbol{W}^1[1; \boldsymbol{\xi}^0] \\ \boldsymbol{\xi}^k &= \boldsymbol{W}^k[1; \boldsymbol{\alpha}^{k-1}] \qquad \forall k = 2, \ldots, L \end{aligned} \tag{4.9}$$

The lifting operation is borrowed from Georghiou et al. (2015). We demonstrate its implementation in the context of a deep lifting network. Consider the bounded uncertain parameter $l_i^k \le \xi_i^k \le u_i^k$ at layer k and node i. The lifting operator which maps ξ_i^k to $\alpha_{i,j}^k$ is given in equation (4.10).

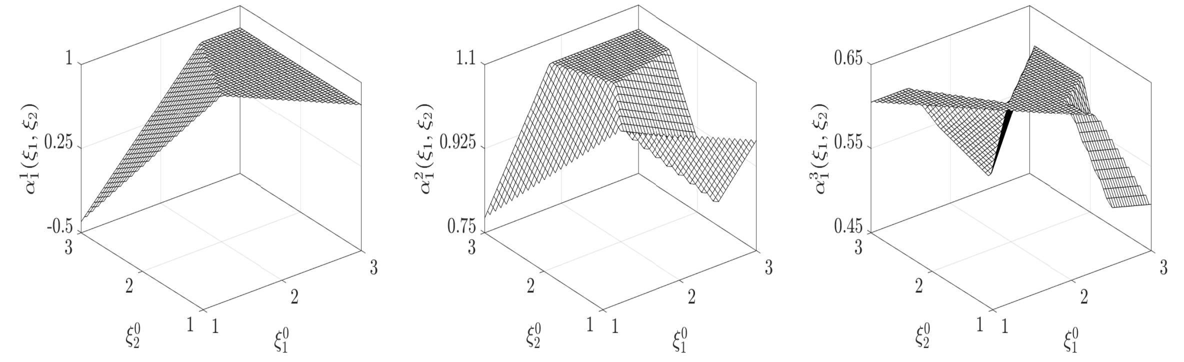
$$\alpha_{i,j}^k = \begin{cases} \xi_i^k & \text{if } r_i^k = 1, \\ \min\{\xi_i^k, z_{i,j}^k\} & \text{if } r_i^k > 1,\ j = 1, \\ \max\left\{\min\{\xi_i^k, z_{i,j}^k\} - z_{i,j-1}^k, 0\right\} & \text{if } r_i^k > 1,\ j = 2, \ldots, r_i^k - 1, \\ \max\{\xi_i^k - z_{i,j-1}^k, 0\} & \text{if } r_i^k > 1,\ j = r_i^k, \end{cases} \quad \begin{array}{l} \forall k = 1, \ldots, L,\ i = 1, \ldots, n^{\text{node}}, \\ j = 1, \ldots, r_i^k \end{array} \tag{4.10}$$

where $r_i^k - 1$ is the number of breakpoints introduced in ξ_i^k and $z_{i,j}^k$ is the j^{th} breakpoint in ξ_i^k. The retraction, originally defined in Georghiou et al. (2015), is given in equation (4.11) where

$\mathbf{R}_i^k \in \mathbb{R}^{1\times r_i^k}$.

$$\xi_i^k = \mathbf{R}_i^k \boldsymbol{\alpha}_i^k = \sum_{j=1}^{r_i^k} \alpha_{i,j}^k \quad \forall k = 1,\ldots,L,\ i = 1,\ldots,n^{\text{node}} \tag{4.11}$$

where $\boldsymbol{\alpha}_i^k = (\alpha_{i,1}^k;\ldots;\alpha_{i,r_i^k}^k)$. The lifted uncertain vector for all layers k is $\boldsymbol{\alpha}^k = (\boldsymbol{\alpha}_i^k;\ldots;\boldsymbol{\alpha}_{n^{\text{node}}}^k)$. Figure 4.7 illustrates a lifted uncertain parameter in each layer of the network in Figure 4.6. The input uncertainty set is the same polytope as introduced for the transportation problem. The weight matrices in the first, second and third layers are $\mathbf{W}^1 = [0, 0.8, -0.4; 1, -0.2, -0.6]$, $\mathbf{W}^2 = [0.5, 0.3, -0.2, -0.4, 0.6; 0.7, -0.2, 0.5, 0.4, -0.4]$ and $\mathbf{W}^3 = [0.3, 0.1, 0.2, 0.8, -0.6; 0.2, 0.1, -0.3, 0.2, 0.7]$, respectively. The single breakpoint at each node is at the midpoint of the ξ_i^k range. The increase in the lifted uncertain parameter complexity, equivalently the deep lifted decision rule approximation quality, with the network's depth is obvious. Note that a network with a single layer is equivalent to a PLDR with generalized lifting introduced by Georghiou et al. (2015).



(a) Lifted parameter in layer 1: $\alpha_{1,1}^1(\xi_1^0,\xi_2^0) = \min\{\xi_1^0, 0.2\}$

(b) Lifted parameter in layer 2: $\alpha_{1,1}^2(\xi_1^0,\xi_2^0) = \min\{\xi_1^1, 0.5250\}$

(c) Lifted parameter in layer 3: $\alpha_{1,1}^3(\xi_1^0,\xi_2^0) = \min\{\xi_1^2, 0.9485\}$

Figure 4.7: A lifted uncertain parameter in each layer of the deep lifting network in Figure 4.6 using the original uncertainty set of the two-stage transportation problem instance given by (4.2).

4.4.2 Optimization model using deep lifted decision rules

The sequence of simple affine and lifting operations introduces new complex features to the lifted uncertain parameters, which refine the approximation quality of the second-stage adaptive policies. The general form of deep lifted decision rules is given in equation (4.12) where $y(\boldsymbol{\alpha})$ is a vector in $\mathbb{R}^m$, $\mathbf{Y}^{\text{deep}}$ is a matrix in $\mathbb{R}^{m\times n^{\text{deep}}}$ and $\xi_0 = 1$. The dimension parameter n^{deep} is equal to $\sum_{i=1}^{n^{\text{node}}} \sum_{k=1}^{L} r_i^k + n + 1$.

$$\boldsymbol{y}(\boldsymbol{\alpha}) = \mathbf{Y}^{\text{deep}}(1;\boldsymbol{\xi}^0;\boldsymbol{\alpha}^1;\cdots;\boldsymbol{\alpha}^L) = \mathbf{Y}^{\text{deep}}\boldsymbol{\alpha} \tag{4.12}$$

To leverage the potential of a deep lifting network, the weights in all layers are optimized simultaneously with the the first-stage decisions and the decision rule slopes. Model (4.13) is obtained by substituting the deep lifted approximation in Model (4.1). Recall that the breakpoints are evenly distributed at each node.

$$\min_{\substack{\mathbf{x},\mathbf{Y}^{\text{deep}};\\ \mathbf{W}^1,\cdots,\mathbf{W}^L;\\ \mathbf{z}^1,\cdots,\mathbf{z}^L}} \quad \mathbf{c}_1^\top \mathbf{x} + \mathbf{q}^\top \mathbf{Y}^{\text{deep}} \mathbb{E}\left[\boldsymbol{\alpha}\right] \tag{4.13a}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} \geq \mathbf{b} \tag{4.13b}$$

$$\mathbf{T}\mathbf{x} + \mathbf{W}\mathbf{Y}^{\text{deep}}\boldsymbol{\alpha} \geq \mathbf{H}\boldsymbol{\xi}^0 \qquad \forall \boldsymbol{\xi}^0 \in \Xi,\ \boldsymbol{\alpha} \in \Xi^{\text{deep}}(\mathbf{z}^1, \cdots, \mathbf{z}^L) \tag{4.13c}$$

where $\mathbf{z}^k$ includes all the breakpoints used in lifting $\boldsymbol{\xi}^k$ and $\Xi^{\text{deep}}(\mathbf{z}^1, \cdots, \mathbf{z}^L) = \bigtimes_{k=0}^{L} \Xi_k^{\text{deep}}(\mathbf{z}^k)$ is the overall deep lifted uncertainty set. The layer-wise lifted uncertainty sets $\Xi_k^{\text{deep}}(\mathbf{z}^k)$ are defined in equation (4.14). For $k = 0$ ($\boldsymbol{z}^0$ is not defined), we have the original uncertainty set $\Xi_0^{\text{deep}} \equiv \Xi$.

$$\begin{aligned}
\Xi_1^{\text{deep}}(\mathbf{z}^1) &= \left\{ \boldsymbol{\xi}^0,\ \boldsymbol{\xi}^1,\ \boldsymbol{\alpha}^1 \;\middle|\; \begin{array}{ll} \boldsymbol{\xi}^1 & = \boldsymbol{W}^1[1;\boldsymbol{\xi}^0] \\ \mathbf{Q}^1(\mathbf{z}^1)\boldsymbol{\alpha}^1 & \geq \mathbf{p}^1(\mathbf{z}^1) \\ \boldsymbol{\xi}^1 & = \mathbf{R}^1\boldsymbol{\alpha}^1 \end{array} \right\} \\
\Xi_k^{\text{deep}}(\mathbf{z}^k) &= \left\{ \boldsymbol{\alpha}^{\text{k}-1}, \boldsymbol{\xi}^k, \boldsymbol{\alpha}^k \;\middle|\; \begin{array}{ll} \boldsymbol{\xi}^k & = \boldsymbol{W}^k\boldsymbol{\alpha}^{\text{k}-1} \\ \mathbf{Q}^k(\mathbf{z}^k)[1;\boldsymbol{\alpha}^k] & \geq \mathbf{p}^k(\mathbf{z}^k) \\ \boldsymbol{\xi}^k & = \mathbf{R}^k\boldsymbol{\alpha}^k \end{array} \right\} \quad \forall k = 2, \ldots, L
\end{aligned} \tag{4.14}$$

where $\mathbf{R}^k$ is a block diagonal matrix where the blocks are $\mathbf{R}_i^k$ for all nodes i. $\mathbf{Q}^k(\mathbf{z}^k)$ and $\mathbf{p}^k(\mathbf{z}^k)$ are the matrix and right-hand-side vector of the lifted uncertainty set at layer k. The representation of the two parameters are included in C.3 for a quick reference.

Examining Model (4.13), we identify three challenges to obtain global optimal solution. First, the expected value vector $\mathbb{E}[\boldsymbol{\alpha}]$ in equation (4.13a) can only be computed via a sample-based approach due to the varying weights. This necessitates the presence of auxiliary variables and additional constraints which nullify the advantage of using a decision rule-based method. Second, fractional non-linearities in the definition of $\mathbf{Q}^k(\mathbf{z}^k)$ and $\mathbf{p}^k(\mathbf{z}^k)$ via the breakpoint values $\boldsymbol{z}^k$ for all k are present. Third, bilinear terms emerge between the weight and dual variables in the derived stochastic counterpart.

We circumvent these issues by re-postulating the problem as a bi-level optimization problem in equation (4.15). The weights are optimized in the upper level model, and they are treated as known parameters in the lower level model.

$$\min_{\substack{\mathbf{W}^1,\dots,\mathbf{W}^L;\\ \mathbf{z}^1,\dots,\mathbf{z}^L}} \quad \zeta(\mathbf{W}^1,\dots,\mathbf{W}^L,\mathbf{z}^1,\dots,\mathbf{z}^L) \tag{4.15a}$$

$$\text{s.t.} \quad \zeta(\mathbf{W}^1,\dots,\mathbf{W}^L,\mathbf{z}^1,\dots,\mathbf{z}^L) = \left\{ \begin{array}{ll} \displaystyle\min_{\mathbf{x},\mathbf{Y}^{\text{deep}}} & \mathbf{c}^\top\mathbf{x} + \mathbf{q}^\top\mathbf{Y}^{\text{deep}}\mathbb{E}\big[\boldsymbol{\alpha}\big] \\ \text{s.t.} & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{T}\mathbf{x} + \mathbf{D}\mathbf{Y}^{\text{deep}}\boldsymbol{\alpha} \geq \mathbf{H}\boldsymbol{\xi}^0 \\ & \qquad \forall \boldsymbol{\xi}^0 \in \Xi,\ \boldsymbol{\alpha} \in \Xi^{\text{deep}}(\mathbf{z}^1,\dots,\mathbf{z}^L) \end{array} \right\}$$

$$w^{\text{lb}} \leq \mathbf{W}^k \leq w^{\text{ub}} \qquad \forall\, k = 1,\dots,L \tag{4.15b}$$

where the deep lifted uncertainty set $\Xi^{\text{deep}}(\mathbf{z}^1,\dots,\mathbf{z}^L)$ is defined in equation (4.14). The lower and upper bounds on all weight coefficients are given by w^{lb} and w^{ub}, respectively.

4.5 Solution methods

We propose two solution methods to address the two-layer optimization problem: a derivative-free and a derivative-based method.

4.5.1 Derivative-free optimization

Derivative-free optimization (DFO) does not require the objective function to be differentiable. Instead, DFO mimics natural concepts and phenomena to generate near-optimal or best attainable solutions. Genetic algorithms, a class of DFO methods, were first introduced by Holland et al. (1992). They are also known as evolutionary algorithms (i.e., EAs) and are based on the survival of the fittest law in nature which is achieved through natural operations such as selection, recombination and mutation. For an extensive review of DFO algorithms and software implementation, see Rios & Sahinidis (2013).

In this chapter, we use a differential evolutionary algorithm (i.e., an improved version of EA) provided by the *BlackBoxOptim.jl* Julia package. Maier et al. (2019) provided an introductory overview of EAs. The algorithm starts with a group of initial points, known as the initial population, with the objective function value as a fitness measure. Then, the group members evolve with the number of iterations based on probabilistic rules which are aimed at improving the members' fitness. The initial population size plays a role in its evolution. Liepins & Hilliard (1989) suggested a size between 50 and 100 to overcome possible bias by the members with the higher solution quality. We use a population size of 50 in all of the derivative-free (i.e., black-box) computational experiments. EA manages constraints, such as equation (4.15b), by simply penalizing its violation in the objective function. Moreover, it enables the user to prioritize between two competing processes: exploration and exploitation. The former increases the chance of finding multiple diverse high fitness members, while the latter increases the local convergence rate of a given population. We emphasize that the search for a good trade-off between the two competing processes is beyond the scope of this work.

Motivating example revisited

In this section, we revisit the two-stage adaptive transportation problem in equation (4.2). We illustrate the solution of deep lifted decision rules constructed via 9 different network settings in Figure 4.8. All settings implement 1 breakpoint at each node. The expected profit does not plateau with the increase in the number of layers and nodes, as in the case of axial PLDRs in Figure 4.4a. For example, using a 3-layer network with four nodes per layer, the best expected profit, found via a layer-wise black-box optimization, is 17.92 which reflect an increase of 2.9% with respect to the best obtained axial PLDR-based optimal profit and an increase of 3.71% with respect to a PLDR with generalized lifting optimal profit (i.e., optimal profit using a 1-layer deep lifting network). The overall computational time for the three layers is 6 hours. The lower and upper weight bounds are -2 and 2, respectively. The 0-layer setting corresponds to a PLDR with axial lifting and is valid only at 2 nodes per layer ($\boldsymbol{\xi}^0 \in \mathbb{R}^2$).

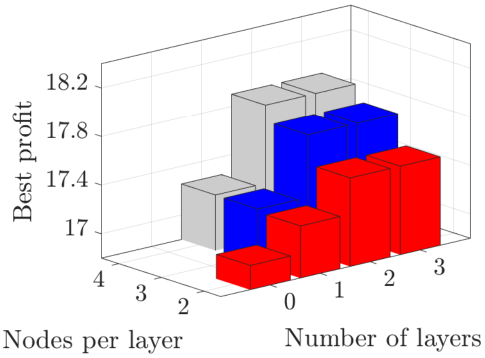


Figure 4.8: Deep lifted decision rules solution with various number of layers, nodes and 1 breakpoint at each node for a two-stage transportation problem using black-box optimization.

Figure 4.9 illustrates the distribution policies using a 3-layer deep lifted decision rule with 4 nodes per layer. The policies $y_{2,1}(\xi_1, \xi_2)$ and $y_{2,2}(\xi_1, \xi_2)$ are indifferent to those shown in Figure 4.1. We observe the non-axial facets in the remaining policies and we see an extra linear piece (i.e., 3 pieces) within the $y_{3,1}(\xi_1, \xi_2)$ and $y_{3,2}(\xi_1, \xi_2)$ policies in comparison to the PLDR-based solution in Figure 4.5.

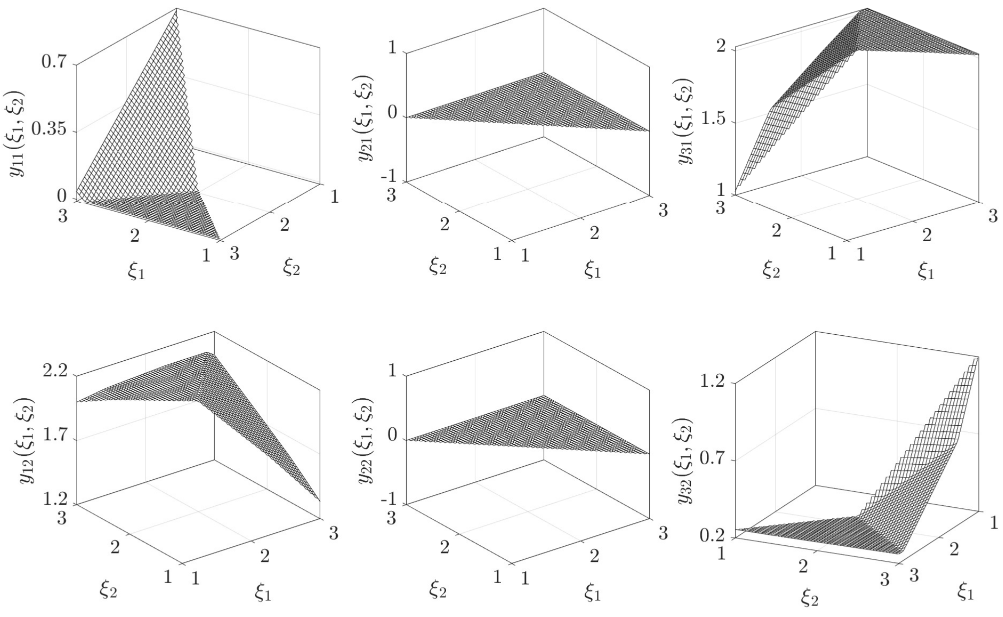


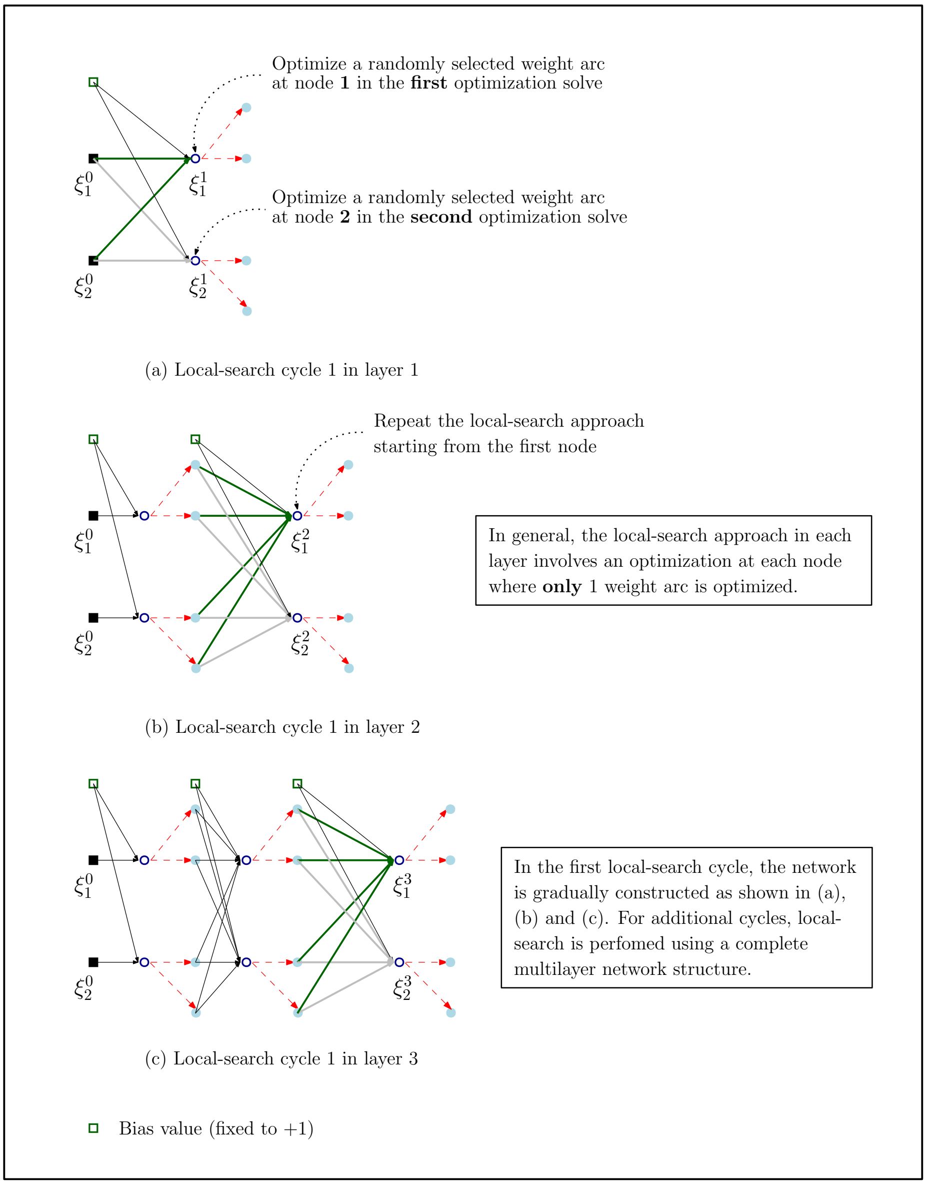
Figure 4.9: Optimal adaptive policies using a 3-layer deep lifting network with 4 nodes per layer and 1 breakpoint at each node for a two-stage transportation problem.

4.5.2 Derivative-based optimization

Analytical first- and second-order gradient information can not be derived for the objective function in equation (4.15a). Hence, we use the limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm which is a class of quasi-newton solution methods (Liu & Nocedal 1989). The first derivative is approximated using a finite difference method. This involves multiple solutions of the lower level model. The approximate inverse Hessian matrix, required to compute the improving quasi-newton direction, is estimated from the first derivative and is stored in a form of several vectors rather than a dense square matrix. This makes the LBFGS algorithm an attractive choice for large-scale optimization problems due to memory-wise management. In this work, we use the LBFGS algorithm via the *Optim.jl* Julia package developed by Mogensen & Riseth (2018). The package offers unconstrained and box-constrained optimization features.

A possible limitation of the derivative-based solution of equation (4.15) is the computational cost of approximating the first-order derivative information as it depends on multiple solutions of the lower level model. The computational cost ramps up as the number of weight variables increases. For a traditional gradient descent approach in which all weights are optimized simultaneously, we adopt a layer-wise optimization framework to reduce the impact of the increased computational load. For example, a 3-layer network is optimized by first optimizing the first layer, then optimizing the weights in the second layer while the weights in the first layer are fixed. Lastly, the weights in the third layer are similarly optimized.

Alternatively, we propose three different local-search heuristics to recover an attractive solution quality at a reduced computational cost. The first heuristic optimizes only one randomly selected weight arc per node per layer. The optimization is done for each node in a given layer and is executed layer-wise for multiple cycles. A single cycle includes a single pass of the local-search procedure in all layers. We obtain the second heuristic by relaxing the node-wise local-search structure so that we optimize only one random weight arc per layer, layer-wise, for multiple cycles. Lastly, we further relax the structure in our local-search heuristic so that we optimize only one random weight arc from the entire network. The concept of a cycle does not exist in this heuristic as there is no layer-wise optimization. Figure 4.10 visualizes the first local-search cycle using the first heuristic for a 3-layer deep lifting network with 2 nodes per layers and 1 breakpoint at each node. Note that for the first two heuristics, the network is gradually built within the first local-search cycle. For later cycles, local-search optimization is always performed using a 3-layer (i.e., complete) network structure. Also, we assume the weight arcs of bias values, which are fixed to 1, have minimal values. So, they are excluded from all local-search heuristics.



Optimize a randomly selected weight arc at node **1** in the **first** optimization solve

Optimize a randomly selected weight arc at node **2** in the **second** optimization solve

(a) Local-search cycle 1 in layer 1

Repeat the local-search approach starting from the first node

In general, the local-search approach in each layer involves an optimization at each node where **only** 1 weight arc is optimized.

(b) Local-search cycle 1 in layer 2

In the first local-search cycle, the network is gradually constructed as shown in (a), (b) and (c). For additional cycles, local-search is perfomed using a complete multilayer network structure.

(c) Local-search cycle 1 in layer 3

□ Bias value (fixed to +1)

Figure 4.10: First local-search cycle using 1 weight arc per node per layer heuristic for a 3-layer network with 2 nodes per layer and 1 breakpoint at each node.

Potential Enhancements: Initial weights selection

A good initial guess improves the local optima solution and the local convergence rate. We propose a two-step scenario-aided optimization framework to generate initial weights, for the upper level model, which may be more attractive than random initial weights. In the first step, a set of optimal scenario-based decisions $\boldsymbol{y}_s$ are obtained by solving a scenario-based stochastic approximation of Model (4.1),

$$\min_{\mathbf{x},\mathbf{y}_s} \quad \mathbf{c}^\top \mathbf{x} + \sum_{s\in\mathcal{S}} P_s\Big(\mathbf{q}^\top \mathbf{y}_s\Big) \tag{4.16a}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} \geq \mathbf{b} \tag{4.16b}$$

$$\mathbf{T}\mathbf{x} + \mathbf{W}\mathbf{y}_s \geq \mathbf{H}\boldsymbol{\xi}_s \qquad \forall s \in \mathcal{S} \tag{4.16c}$$

where $\boldsymbol{\xi}_s \in \hat{\Xi}$ is the set of discrete scenarios and P_s is the probability of the discrete scenario $\boldsymbol{\xi}_s$. The objective is not to generate an accurate set of optimal decisions. Rather, we aim to get "guiding" decisions to be used in the second step. The first step is, of course, only possible if an adequate scenario-based stochastic program can be constructed and easily solved. The second step is a bi-level optimization problem, shown in formulation (4.17), where the square error between the deep lifted decision rule $\boldsymbol{Y}^{\text{deep}}\boldsymbol{\alpha}_s$ and the scenario-based solution $\boldsymbol{y}_s$, for all discrete scenarios in $\hat{\Xi}$, is minimized in the lower level model. Similar to equation (4.15), the weights are optimized in the upper level model and are treated as known parameters in the lower level model. An evolutionary algorithm, via *BlackBoxOptim.jl*, is used to solve the upper level model, while a nonlinear optimization package, NLopt 2.6.2 developed by Johnson (2020), is used to solve the lower level model.

$$\min_{\substack{\mathbf{W}^1,\ldots,\mathbf{W}^L;\\ \mathbf{z}^1,\ldots,\mathbf{z}^L}} \quad \eta(\mathbf{W}^1,\ldots,\mathbf{W}^L,\mathbf{z}^1,\ldots,\mathbf{z}^L) \tag{4.17a}$$

$$\text{s.t.} \quad \eta(\mathbf{W}^1,\ldots,\mathbf{W}^L,\mathbf{z}^1,\ldots,\mathbf{z}^L) = \left\{ \begin{array}{lll} \min\limits_{\substack{\boldsymbol{Y}^{\text{deep}};\\ \boldsymbol{\xi}_s^1,\ldots,\boldsymbol{\xi}_s^L;\\ \boldsymbol{\alpha}_s^1,\ldots,\boldsymbol{\alpha}_s^L}} & \sum\limits_{s\in\mathcal{S}}\Big(\boldsymbol{Y}^{\text{deep}}\boldsymbol{\alpha}_s - \boldsymbol{y}_s\Big)^2 & \\ \text{s.t.} & \boldsymbol{\xi}_s^1 = \boldsymbol{W}^1\boldsymbol{\xi}_s^0 & \forall s\in\mathcal{S},\ \boldsymbol{\xi}_s^0 \in \hat{\Xi} \\ & \boldsymbol{\alpha}_{i,j,s}^k = \mathcal{L}_{i,j}(\xi_{i,s}^k) & \forall s\in\mathcal{S},\ k=1,\ldots,L; \\ & & \ i=1,\ldots,n^{\text{node}}; \\ & & \ j=1,\ldots,r_i^k \\ & \boldsymbol{\xi}_s^k = \boldsymbol{W}^k\boldsymbol{\alpha}_s^{k-1} & \forall s\in\mathcal{S},\ k=2,\ldots,L \\ & \boldsymbol{\xi}_s^0 \in \mathbb{R}^n & \forall s\in\mathcal{S} \\ & \boldsymbol{\xi}_s^k \in \mathbb{R}^{n^{\text{node}}} & \forall s\in\mathcal{S} \\ & \boldsymbol{\alpha}_s^k \in \mathbb{R}^{n^{\text{node}}(n^{\text{brkp}}+1)} & \forall s\in\mathcal{S},\ k=1,\ldots,L \end{array} \right\} \tag{4.17b}$$

$$w^{1\text{b}} \leq \mathbf{W}^k \leq w^{\text{ub}} \qquad \forall\ k=1,\ldots,L \tag{4.17c}$$

4.6 Computational Experiments

In this section, we illustrate the training of a deep lifting network using a black-box, a traditional gradient descent method and local-search heuristics for a two-stage airlift operations scheduling problem from Ariyawansa & Felt (2004) and also found in the GAMS library (GAMS Development Corporation 2018). We investigate the impact of increasing decision variables and uncertainty dimensions on the performance of the solution methods. The lower and upper weight bounds in all experiments are -2 and 2, respectively. In all computational experiments, we assume no lifting at nodes where the range of the uncertain parameters are less than 0.01 to avoid numerical issues which arise from the presence of a large difference in the coefficients magnitude in an optimization problem.

The objective of this airlift operations scheduling problem is to meet the uncertain capacities for all routes $j \in \mathcal{J}$ by a set of aircraft types $\mathcal{I}$ at the minimal expected total cost. The number of flights $x_{i,j}^{\text{orig}}$ originally planned for all aircraft types i and all routes j are determined at the outset of the problem. Meanwhile, the recourse decisions, taken after realizing the actual routes capacity, are: (1) the number of flights $x_{i,j,k}(\boldsymbol{\xi})$ by an aircraft type i originally planned for route j but switched to route k, (2) the commercially contracted capacity $y_j^{\text{com}}(\boldsymbol{\xi})$ for route j and (3) the unused capacity $y_j^{\text{emp}}(\boldsymbol{\xi})$ for route j.

A two-stage stochastic adaptive airlift operations scheduling model is given in equation (4.18). The expected cost, to be minimized, includes the costs of initially planned flights, change in costs due to change in planned flights, cost of commercially contracted and unused capacity assigned for route j. Equation (4.18b) limits the number of flight hours by an aircraft type to the maximum number of flight hours, F_i, allowed. The parameter $a_{i,j}$ is the number of flight hours per flight by an aircraft type i for route j, while $a_{i,j,k}$ is the increase in flight hours by an aircraft type i when switching from route j to route k. Equation (4.18c) translates the fact that the number of flight hours transferred from route j to all other routes must not exceed the originally planned flight hours. Equation (4.18d) dictates the capacity balance for each route. The parameter $b_{i,j}$ is the service capacity per flight by an aircraft type i for route j.

$$
\begin{aligned}
\min_{\substack{x_{i,j}^{\text{orig}},x_{i,j,k}(\boldsymbol{\xi});\\ y_j^{\text{com}}(\boldsymbol{\xi}),y_j^{\text{emp}}(\boldsymbol{\xi})}} \quad & \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\left(c_{i,j}x_{i,j}^{\text{orig}} + \sum_{\substack{k\in\mathcal{J};\\ k\neq j}}\Big(c_{i,j,k} - c_{i,j}\frac{a_{i,j,k}}{a_{i,j}}\Big)\mathbb{E}\big[x_{i,j,k}(\boldsymbol{\xi})\big]\right) \\
& + \sum_{j\in\mathcal{J}}\Big(c_j^{\text{com}}\mathbb{E}\big[y_j^{\text{com}}(\boldsymbol{\xi})\big] + p_j\mathbb{E}\big[y_j^{\text{emp}}(\boldsymbol{\xi})\big]\Big) && \text{(4.18a)} \\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} a_{i,j}x_{i,j}^{\text{orig}} \leq F_i \qquad \forall\, i\in\mathcal{I} && \text{(4.18b)} \\
& \sum_{k\in\mathcal{J},k\neq j} a_{i,j,k}x_{i,j,k}(\boldsymbol{\xi}) \leq a_{i,j}x_{i,j}^{\text{orig}} \qquad \forall\, i\in\mathcal{I}, j\in\mathcal{J}, \boldsymbol{\xi}\in\Xi && \text{(4.18c)} \\
& \sum_{i\in\mathcal{I}} b_{i,j}x_{i,j}^{\text{orig}} - \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{J},k\neq j} b_{i,j}\frac{a_{i,j,k}}{a_{i,j}}x_{i,j,k}(\boldsymbol{\xi}) + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{J},k\neq j} b_{i,j}x_{i,k,j}(\boldsymbol{\xi}) \\
& + y_j^{\text{com}}(\boldsymbol{\xi}) - y_j^{\text{emp}}(\boldsymbol{\xi}) = s_j\xi_j \qquad \forall\, j\in\mathcal{J}, \boldsymbol{\xi}\in\Xi && \text{(4.18d)} \\
& x_{i,j}^{\text{orig}} \geq 0 \qquad \forall\, i\in\mathcal{I}, j\in\mathcal{J} && \text{(4.18e)} \\
& x_{i,j,k}(\boldsymbol{\xi}) \geq 0 \qquad \forall\, i\in\mathcal{I}, j,k\in\mathcal{J}, k\neq j, \boldsymbol{\xi}\in\Xi && \text{(4.18f)} \\
& y_j^{\text{com}}(\boldsymbol{\xi}), y_j^{\text{emp}}(\boldsymbol{\xi}) \geq 0 \qquad \forall\, j\in\mathcal{J}, \boldsymbol{\xi}\in\Xi && \text{(4.18g)}
\end{aligned}
$$

We organize the computational experiments into three groups based on the uncertainty dimension.

- **Small dimensional instance** with 2 aircraft types, 2 routes and $\boldsymbol{\xi} \in \mathbb{R}^2$. All solution approaches are expected to perform efficiently.
- **Medium dimensional instance** with 4 aircraft types, 4 routes and $\boldsymbol{\xi} \in \mathbb{R}^4$. We show that dimensionality plagues a scenario-based solution approach and the computational cost increases for the traditional gradient descent method.
- **Large dimensional instance** with 10 aircraft types, 10 routes and $\boldsymbol{\xi} \in \mathbb{R}^{10}$. We highlight the effectiveness of local-search heuristics with respect to scaling.

Small dimensional instance

The computational parameters for the small dimensional instance are shown in Table 4.2. The first uncertain capacity follows a beta distribution $\beta(8,2)$ within [0,1250], whereas the second uncertain capacity follows a beta distribution $\beta(6,3)$ within [0,2500]. The scaling factor s_j is equal to 1 for both routes. The maximum flight hours F_i for each aircraft type is 7200 hr.

Table 4.2: Computational parameters for the small dimensional two-stage airlift operations scheduling problem.

Flying hours per flight			
$a_{1,1}$	$a_{2,1}$	$a_{1,2}$	$a_{2,2}$
24	49	14	29

Serviced capacity per flight			
$b_{1,1}$	$b_{2,1}$	$b_{1,2}$	$b_{2,2}$
50	20	75	20

Cost per flight			
$c_{1,1}$	$c_{2,1}$	$c_{1,2}$	$c_{2,2}$
7200	7200	6000	4000

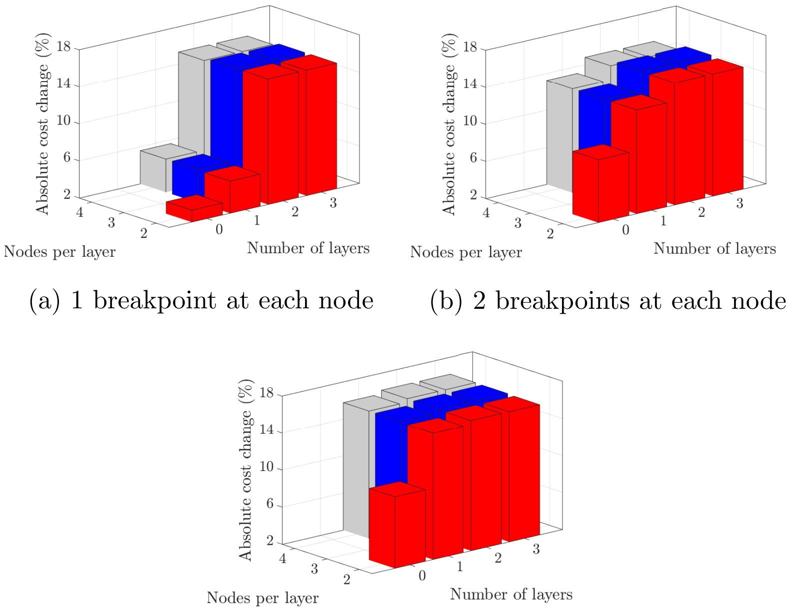
Unit auxiliary cost			
c_1^{com}	c_2^{com}	p_1	p_2
500	250	0	0

Increase in flying hours			
$a_{1,1,2}$	$a_{1,2,1}$	$a_{2,1,2}$	$a_{2,2,1}$
19	29	36	56

Cost per switched flight			
$c_{1,1,2}$	$c_{1,2,1}$	$c_{2,1,2}$	$c_{2,2,1}$
7000	8200	5500	8700

Figure 4.11 illustrates the best attainable solution quality improvement with respect to an LDR via layer-wise black-box optimization for various deep lifted decision rules. We investigated the flexibility of the piecewise linear decision rules with respect to the number of layers (i.e., depth), the number of nodes (i.e., breadth) and the number of breakpoints at each node (i.e., granularity). For this specific example, we observe that the depth of a network plays a greater role in improving the flexibility of decision rules at low granularity. The breadth of a network, on the other hand, has a minimal impact in defining the approximation quality of a deep

lifted decision rule whether it is for low/high granularity or low/high depth. Nonetheless, the attractiveness of a a deep lifted decision rule using a 1-layer network, equivalently a PLDR with generalized lifting, increases with the increase in breadth. The optimization time limit in each layer is 2 hours. The 0-layer index corresponds to an axial PLDR and it is only valid for the 2 nodes setting since $\boldsymbol{\xi}^0 \in \mathbb{R}^2$.

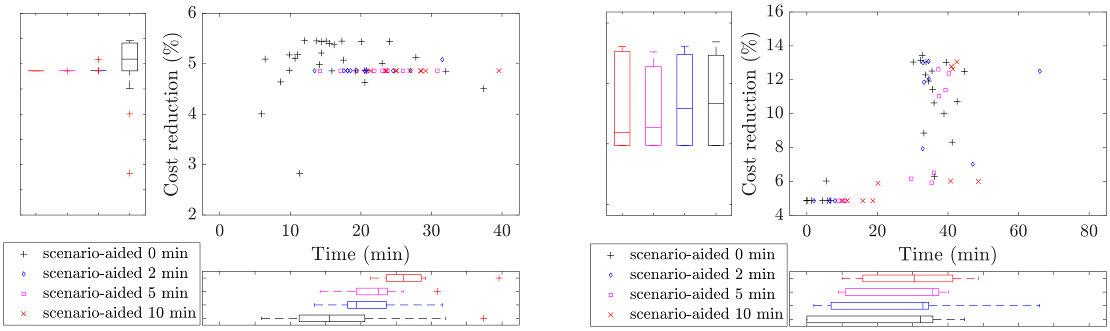


(a) 1 breakpoint at each node (b) 2 breakpoints at each node

(c) 3 breakpoints at each node

Figure 4.11: The best attainable solution improvement to an LDR using a black-box optimizer for the small dimensional two-stage airlift operations scheduling problem as a function of layers, nodes per layers and breakpoints at each node.

For the derivative-based approach, we first address the following question: Should we use a random or a scenario-aided initial guess for the upper level model in equation (4.15)? A scenario-aided initial guess is given by the solution of equation (4.17) using a black-box optimizer in the *BlackBoxOptim.jl* Julia package. Figure 4.12 illustrates the comparison of scenario-aided and random initial guesses derivative-based solution in 1- and 2-layer networks with 2 nodes per layer and 1 breakpoint at each node. Optimization is performed layer-wise. For notation purposes, the label “scenario-aided 2 min” corresponds to the solution using a scenario-aided initial guess obtained after 2 min in a black-box optimizer. In total, each of the random and scenario-aided initial guess clusters includes 30 different solutions. We empirically infer that the benefit of a scenario-aided initial guess is minimal and does not justify its use in the remaining derivative-based computational experiments. The only factor to be aware of when choosing a random initial guess is to avoid a trivial weight matrix which does not improve the solution of the new extended network (i.e., avoid a stationary point).

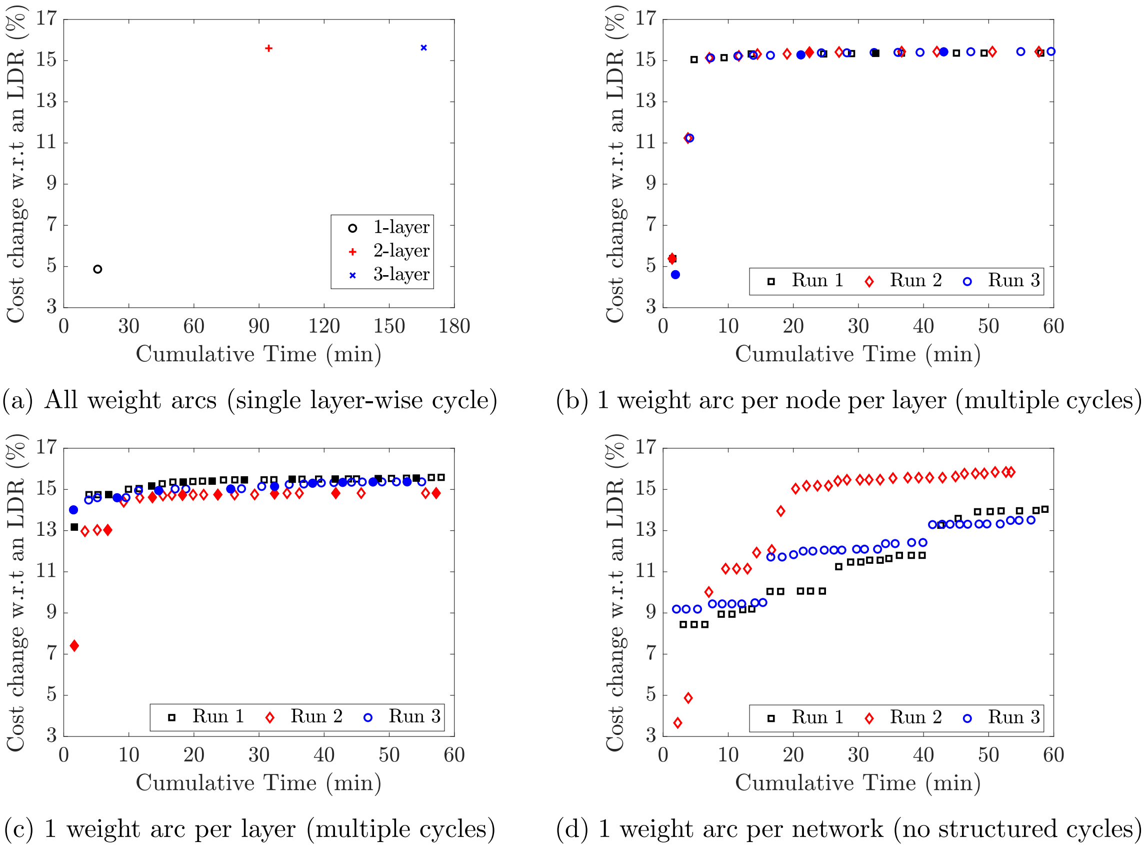


(a) 1-layer setting

(b) 2-layer setting

Figure 4.12: Solution of 1- and 2-layer networks using (i) a gradient-based solution method with scenario-aided and (ii) random (labeled "scenario-aided 0 min") initial guesses for the small dimension two-stage airlift operations scheduling problem.

The training of 1-, 2- and 3-layer networks with 2 nodes per layer and 1 breakpoint at each node is illustrated using a derivative-based approach. Figure 4.13a depicts the output of a traditional layer-wise optimization of all weights. The improvement with respect to an LDR is similar to that obtained via a black-box optimizer. Figures 4.13b-4.13d illustrate the local-search heuristics outputs where the start of a new cycle is denoted by a filled marker. For the first two heuristics, it is evident the significant reduction in computational cost, approximately 1 order of magnitude, when comparing to the traditional gradient descent. The attractiveness of the third heuristic, which lacks any optimization structure, is reduced. Nonetheless, it is still competitive with respect to the traditional gradient descent.



(a) All weight arcs (single layer-wise cycle)

(b) 1 weight arc per node per layer (multiple cycles)

(c) 1 weight arc per layer (multiple cycles)

(d) 1 weight arc per network (no structured cycles)

Figure 4.13: Derivative-based solution for the small dimensional two-stage airlift operations scheduling problem using a 3-layer deep lifting network via (a) a traditional gradient-descent, (b) a 1 weight arc per layer per node, (c) a 1 weight arc per layer and (d) a 1 weight arc per network local-search heuristics.

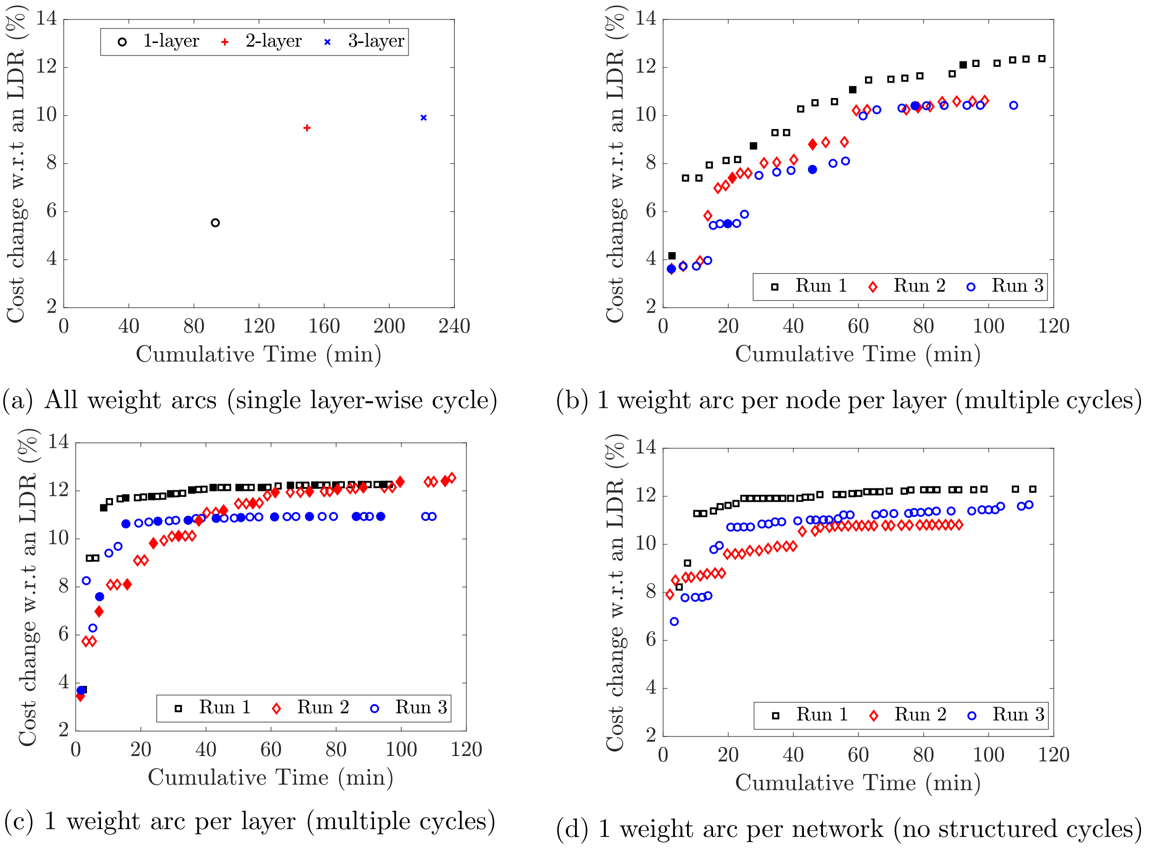
Medium dimensional instance

For the medium dimensional instance, we have 4 aircraft types, 4 routes and $\boldsymbol{\xi} \in \mathbb{R}^4$. All uncertain parameters follow a beta distribution and the detailed computational setting is included in C.4. Despite the two-stage nature of the problem, the scenario-based solution is plagued by memory limitation. This is shown in Table 4.3 as we increase the number of evenly distributed scenarios per dimension from 18 (i.e., 4^{18} ≈1.05e+05 total scenarios) to 21 (i.e., ≈2.4e+05 total scenarios). The memory requirement for the latter setting exceeds the available 16GB memory. Another prominent observation is the steep increase in computational time from 37.61 to 1915.09 sec as we increased the total number of scenarios by 1 order of magnitude. Assuming we have enough memory, this increase in computational cost is expected to ramp up as we further increase the number of scenarios.

Table 4.3: Scenario-based solution for the medium dimensional two-stage airlift operations scheduling problem.

# of scenarios [per dimension]	Cost	Time (sec)	# of constraints	# of variables
1.60e+1 [2]	619355.53	0.03	1.63e+3	5.85e+3
2.56e+2 [4]	604084.81	0.25	5.13e+3	1.84e+4
1.30e+3 [6]	588821.77	2.59	2.59e+4	9.33e+4
1.00e+4 [10]	577047.62	37.61	2.00e+5	7.20e+5
1.05e+5 [18]	569099.86	1915.09	2.10e+6	7.56e+6
2.34e+5 [21]	N/A	N/A	—	—

The increase in the uncertainty dimension leads to an increase in the number of optimized weight variables, which translates to a higher computational load for the traditional gradient descent. This is shown in Figure 4.14a. As for the local-search heuristics, we observe that the attractiveness of optimizing 1 weight arc per node per layer has reduced due to the increase in computational cost. However, the two local-search heuristics in Figures 4.14c and 4.14d do not only reduce the computational cost significantly, but they recover a better solution quality, a 20% relative increase, than the traditional gradient descent. One may argue that in the latter method, the weights in all layers are optimized only once (i.e., a single cycle) which explains the lower solution quality. This is true, but running a second optimization cycle ramps up the computational cost which is already unattractive in comparison with the local-search heuristics.



(a) All weight arcs (single layer-wise cycle)

(b) 1 weight arc per node per layer (multiple cycles)

(c) 1 weight arc per layer (multiple cycles)

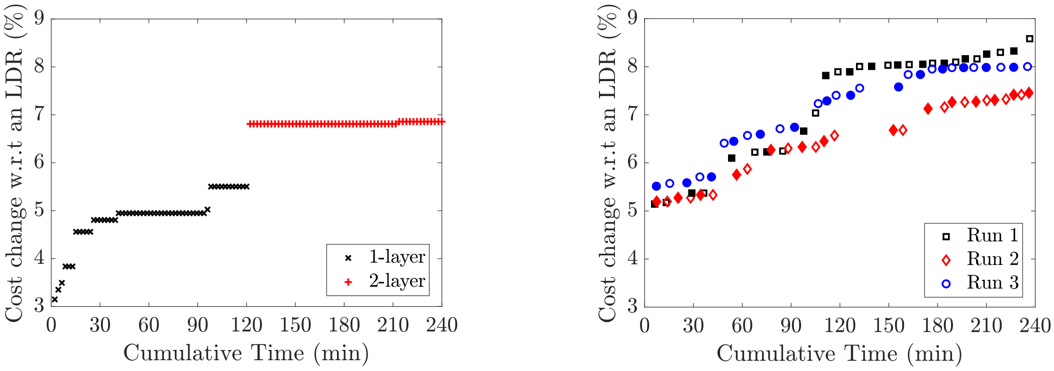
(d) 1 weight arc per network (no structured cycles)

Figure 4.14: Derivative-based solution for the medium dimensional two-stage airlift operations scheduling problem using a 3-layer deep lifting network via (a) a traditional gradient-descent, (b) a 1 weight arc per layer per node, (c) a 1 weight arc per layer and (d) a 1 weight arc per network local-search heuristics.

Large dimensional instance

Lastly, we solve a large dimensional instance with 10 aircraft types, 10 routes and $\boldsymbol{\xi} \in \mathbb{R}^{10}$. The computational setting is randomly generated using a similar magnitude to the medium instance. The objective of this experiment is to study the scaling capabilities of the proposed solution methods. First, we highlight the scenario-based dimensionality problem where the memory requirement for a set of 1.20e+4 scenarios (i.e., 2 samples per dimension) exceeds the allocated 16GB memory. For comparison, the total number of scenarios obtained from 3 samples per dimension is equal to $59,049$. We also found that the computational cost of a traditional gradient descent is prohibitive.

With this being said, we illustrate the training of a 2-layer network with 5 nodes per layer and 1 breakpoint at each node via a black-box optimizer in Figure 4.15a and a local-search heuristic, 1 weight arc per layer, in Figure 4.15b. Black-box training is performed layer-wise where the 1-layer weights are first optimized for 120 min, then they are fixed and the 2-layer weights are optimized for another 120 min. On the other hand, derivative-based optimization is done in multiple cycles up to 240 min. We observe that the local-search heuristic is more efficient in terms of solution quality and it scales well despite the lower solution quality improving rate when compared with the small and medium instances. We also note the black-box optimizer limited ability to find better weights in the second layer (i.e., red plus signs).



(a) Black-box optimization (layer-wise)

(b) 1 weight arc per layer heuristic (multiple cycles)

Figure 4.15: Black-box and local-search optimization of the large dimensional two-stage airlift operations scheduling problem using a 2-layer network with 5 nodes per layer and 1 breakpoint at each node.

4.7 Conclusions and future research directions

The quest to construct flexible decision rules for adaptive policies has attracted growing interests in the stochastic and robust optimization communities since the introduction of affine decision rules. As a part of the ongoing efforts, this work introduced an instrument to devise complex piecewise linear decision rules using a deep lifting network to solve two-stage stochastic optimization problems. The solution involves simultaneous training of the network. In this regard, we proposed efficient local-search heuristics that are stemmed from the stochastic gradient descent approach in deep neural networks. The heuristics recover similar solution quality to a traditional gradient descent with a significant computational cost reduction. The method is not restricted to two-stage adaptive optimization problems. It can be also extended to multi-stage adaptive optimization problems.

The cross-pollination of ideas from deep learning to stochastic adaptive optimization leads to some appealing future research directions. The first is related to the method used in computing gradient information. Indeed this was a major setback for training deep neural networks before the development of the back-propagation algorithm. We did not attempt to incorporate the algorithm within our methodology though we realized the increasing computational cost in computing the gradient using the finite difference method. Second, the operation at each node in all layers can be different from the current lifting. For example, a rectified linear activation unit, which has proven to be efficient in training deep neural networks, is an attractive candidate. Third, the success of artificial neural networks is due in no small part to their intrinsic parallel architecture that allows for parallel solution methods, we did not explore parallelism in this work. It would be quite interesting to better understand how our ideas can be parallelized and how new parallel approaches can be brought to bear on stochastic optimization problems that employ decision rules.

Lastly, being a methodological chapter, we have admittedly made a strong assumption in requiring the uncertainty set to be fully specified a priori, after which a decision rule is learned. In practice, it is not always feasible to assume that the data defining the uncertainty set can be collected and preprocessed in its entirety before then attempting to construct decision rules. It would be interesting to incorporate our ideas into an integrated procedure in which one interacts continuously with the data, and hence the form of the uncertainty set, while constructing decision rules.

Chapter 5

Norm Induced Polyhedral Uncertainty Set for Robust Linear Optimization

5.1 Introduction

Robust optimization (RO) has received lots of attention in recent years as a modeling framework for immunizing against uncertainty in mathematical optimization. It relies on appropriately defining an uncertainty set and solving a reformulated deterministic robust counterpart to ensure worst-case feasibility over the uncertainty set. Static robust optimization corresponds to a class of RO problems where all optimal robust decisions are deterministic (Ben-Tal et al. 2009). On the other hand, adaptive optimization, a different class of RO problems introduced by Ben-Tal et al. (2004), classifies decision into deterministic static decisions and adaptive uncertainty-dependent decisions that are implemented progressively as the uncertainty is revealed.

A general guideline for uncertainty set construction in robust optimization is that it should not lead to overly conservative or computationally challenging deterministic robust counterpart formulation. Traditionally, the uncertainty set has been modeled as a symmetric bounded set. For static robust linear optimization, Soyster (1973) introduced the interval-based box type uncertainty set and Ben-Tal & Nemirovski (2000) introduced the ellipsoidal type uncertainty set. For box and polytopic sets, the robust counterpart of a linear robust problem is linear programming (LP), while for ellipsoidal sets, the constructed counterpart of the same robust model is a second-order conic programming model (SOCP). Bertsimas & Sim (2004) modeled uncertainty using a budget uncertainty set which restricts the number of uncertain parameters allowed to reach the worst case values simultaneously to a pre-specified budget value. It is a flexible extension of a box uncertainty set with the ability to tune the solution conservatism while maintaining the same modeling feature of an LP robust counterpart. Bertsimas et al. (2004) introduced general norm induced uncertainty set for robust static linear optimization.

This is the most general type of symmetric bounded uncertainty set, which can lead to various forms under different norms. For example, ℓ_1-, ℓ_2- and ℓ_∞-norm induced sets are equivalent to a polyhedral, ellipsoidal and a box uncertainty sets, respectively. Gotoh & Uryasev (2016) studied the $CVaR$- and *Deltoidal*-norms and dual norms and derived their LP representations.

The increase in the robust optimization complexity limits the uncertainty set types that can be used to construct a tractable robust counterpart. For quadratically constrained quadratic programming and SOCP problems, a tractable semi-definite programming robust counterpart is derived only under a single ellipsoid uncertainty set (Ben-Tal et al. 2002). With a polytopic or an intersection of ellipsoidal uncertainty sets, the robust counterpart is found to be NP-hard (Ben-Tal et al. 2002). Further, the robust counterpart for semi-definite robust optimization problems is NP-hard, in general, for both polytopic and ellipsoidal uncertainty sets (Ben-Tal et al. 2000). A specific exception, where a tractable SDP robust counterpart exists, is introduced in Boyd et al. (1994) for unstructured norm-bounded uncertainty sets.

Deriving the robust counterparts for adaptive robust optimization problem is similar to the static RO case. After approximating the adaptive decisions to a certain function or rule (e.g., linear or quadratic), the robust tractable counterpart is derived based on the structure of the RO problem and the uncertainty set. For an adaptive robust optimization review, the reader is referred to the work of Yanıkoğlu et al. (2019).

Ideally, bounded constructed uncertainty sets in robust optimization must cover all possible instances of the uncertain parameters. As a result, the probability of constraint violation is zero or, equivalently, the probability of constraint satisfaction is one. The high level of immunity against risk comes at the expense of a very conservative solution. In practice, a decision-maker may tolerate a certain margin of constraint's violation in return for a better solution quality. Besides, uncertainties may follow an unbounded distribution that cannot be entirely represented via a finite bounded set. In both cases, the probabilistic guarantee of constraint satisfaction will be less than one. Ben-Tal & Nemirovski (2000) derived the probabilistic guarantee for interval+ellipsoid uncertainty sets. Bertsimas et al. (2004) addressed the guarantee for the interval+polyhedral uncertainty sets. Li et al. (2011) and Li, Tang & Floudas (2012) compared the robust counterpart formulations and probabilistic guarantees for various symmetric uncertainty sets which included box, ellipsoid, polyhedral, interval+ellipsoid and interval+polyhedral uncertainty sets. Guzman et al. (2016, 2017*a*,*b*) developed new probabilistic guarantees for linear robust counterparts using bounded/unbounded and symmetric/asymmetric independent uncertainty sets with different level of probability distribution information ranging from limited to exact information. Chen et al. (2007) introduced deviation measures to capture distributional asymmetry between the uncertain parameters without any correlation. Yuan et al. (2016) studied robust linear optimization under correlated uncertainty and demonstrated the advantage of introducing correlation information into the uncertainty set construction. Similarly, Zhang et al. (2018) illustrated, via a production scheduling problem, the significance of capturing correlation within the uncertainty in improving the robustness of a solution.

While there is no prior information on the uncertainty distribution, a data-driven uncertainty set construction method is a practical choice. Several statistical and analytical techniques have been implemented in constructing data-driven uncertainty sets. Dirichlet process mixtures model (DPMM) is a technique used to characterize distributional information from random data. Ning & You (2018*b*) used a DPMM and maximum likelihood estimation to construct data-driven multi-class uncertainty sets to solve stochastic robust optimization problems. Ning & You (2019*a*) constructed data-driven uncertainty set for wind forecast errors, in a unit-commitment problem, using the same technique. Likewise, Zhao, Zhong & Du (2019) implemented DPMM to construct data-driven uncertainty sets for steam systems in ethylene plants. Ning & You (2019*b*) reviewed data-driven optimization under uncertainty and the integration of machine learning and mathematical programming for decision-making under uncertainty.

Kernel density estimation method (KDEM) is yet another technique to estimate the density functions of random data. Ning & You (2017) implemented KDEM to construct data-driven uncertainty set to solve adaptive robust optimization problems. Ning & You (2018*a*) implemented KDEM and principal component analysis to construct data-driven uncertainty sets for control, scheduling and planning applications. Shen et al. (2020) employed KDEM to construct the uncertainty set for the optimization of an industrial steam system. Dai et al. (2020) used a similar principal component analysis and KDEM combination to construct the data-driven uncertainty set in a crude oil blending under uncertainty problem.

Clustering is another popular technique in constructing data-driven uncertainty sets. Shen et al. (2020) employed a generalized intersection kernel support vector clustering is to construct the uncertainty set in an energy system optimization problem. Shang et al. (2019) used a support vector clustering technique to characterize data-driven uncertainty set for an irrigation system problem couple with a conditional uncertainty set to describe the uncertainty dependence on other variables. Garuba et al. (2020) implemented K-means clustering and supervised learning techniques to construct data-driven discrete and polyhedral uncertainty for network design and expansion problems. Li et al. (2016) implemented an autoregressive integrated moving average (ARIMA) model and whitening transforms on correlated data for the aforementioned purpose to address robust unit-commitment problems.

In this chapter, we compared various norm-induced polyhedral uncertainty sets. We present a novel method for asymmetric uncertainty set construction based on the distributional information and sampling data of the uncertain parameters. Correlation between primitive uncertain parameters is captured in the proposed method. Deterministic robust counterpart formulation is derived using polyhedral-norms induced uncertainty set. We also derived robust counterparts where the uncertainty belongs to an intersection of two symmetric general-norms induced uncertainty sets and where the uncertainty belongs to an intersection of a symmetric and asymmetric general-norms induced uncertainty sets.

In the subsequent sections, we first present the general norm-induced symmetric uncertainty sets. Then, we present the proposed uncertainty set construction method and demonstrate some data-driven uncertainty set examples. The robust counterpart for asymmetric norm-induced uncertainty sets is constructed. Next, the robust counterpart is derived where the uncertainty belongs to an intersection of norm-induced uncertainty sets. Lastly, we compare the solution quality using symmetric and asymmetric norm-induced uncertainty sets emphasizing the added value by the latter via a numerical example. We highlight the significance of modeling the uncertainty as an intersection of uncertain parameter bounds and a norm-induced set from distributional information via a reactor design problem.

5.2 Polyhedral Norms

General norm. A general norm $\|\cdot\|$ in the space $\mathbb{R}^n$ is a function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ that satisfies the following properties:

- $\|\boldsymbol{x}\| \geq 0$ for any $\boldsymbol{x} \in \mathbb{R}^n$, $\|\boldsymbol{x}\| = 0$ if and only if $\boldsymbol{x} = 0$.
- $\|\lambda \boldsymbol{x}\| = |\lambda| \|\boldsymbol{x}\|$ for any $\boldsymbol{x} \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$.
- $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ for any $\boldsymbol{x},\ \boldsymbol{y} \in \mathbb{R}^n$.

For a given norm, the unit ball is defined as the set $\mathcal{B} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \|\boldsymbol{x}\| \leq 1\}$. It is symmetric with respect to the origin, convex, closed, bounded and has a non-empty interior.

Polyhedral norm. A general norm is a polyhedral norm if its unit ball $\mathcal{B}$ is a polytope. Stated differently, a norm is polyhedral if the unit ball norm admits a closed polyhedral representation $\mathcal{B} = \{\boldsymbol{x} \in \mathbb{R}^n : \exists \boldsymbol{y} \in \mathbb{R}^m,\ \boldsymbol{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{y} \leq \boldsymbol{d}\}$.

Dual norm. The dual of a general norm is given in (5.1).

$$\|\boldsymbol{x}\|^* = \max_{\|\mathbf{s}\| \leq 1} \mathbf{s}^\top \boldsymbol{x} \tag{5.1}$$

ℓ_1 and ℓ_∞ norm

For a vector $\boldsymbol{x}$ in the space $\mathbb{R}^n$, the standard ℓ_p-norm is defined in (5.2) where the parameter $p \geq 1$ (the norm properties are not satisfied for $0 < p < 1$).

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \tag{5.2}$$

Specifically, the ℓ_1-, ℓ_2- and ℓ_∞-norms defined in (5.3) are widely-used for defining the Manhattan distance, Euclidean distance and Chebyshev distance, respectively. Figure 5.1 illustrates the unit balls in $\mathbb{R}^2$ induced by the three norms.

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|, \quad \|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}, \quad \|\boldsymbol{x}\|_\infty = \max_i |x_i| \tag{5.3}$$

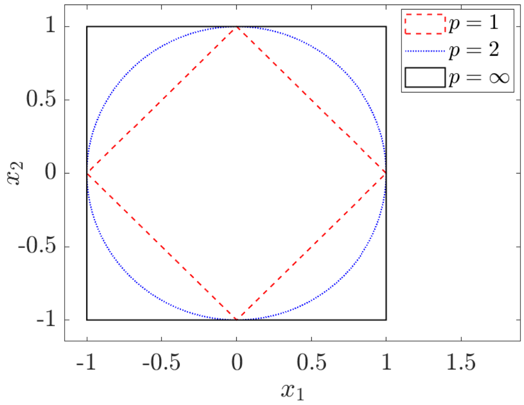


Figure 5.1: ℓ_p-norm induced unit balls

The dual of an ℓ_p-norm is given in (5.4). The dual of an ℓ_2-norm is also an ℓ_2-norm (i.e., $p = q = 2$). This special property is known as self-dual. The dual of an ℓ_1-norm is an ℓ_∞-norm and it is easily verified that the reverse is true.

$$\|\boldsymbol{x}\|_p^* = \|\boldsymbol{x}\|_q, \qquad \text{where } q = 1 + \frac{1}{p-1} \tag{5.4}$$

D-norm

For a vector $\boldsymbol{x}$ in the space $\mathbb{R}^n$, the D-norm is defined in (5.5) using a parameter Γ where N is a set of all indices $\{1, \cdots, n\}$ (Bertsimas et al. 2004). Parameter Γ takes a value between 0 (excluded) and n.

$$\|\boldsymbol{x}\|_\Gamma^D = \max_{\{S \cup \{t\} | S \subseteq N, |S| \leq \lfloor \Gamma \rfloor, t \in N \setminus S\}} \left\{ \sum_{i \in S} |x_i| + (\Gamma - \lfloor \Gamma \rfloor)|x_i| \right\} \tag{5.5}$$

A D-norm is proportional to ℓ_1- and ℓ_∞-norms for specific Γ values. For $0 < \Gamma \leq 1$, a D-norm is equal to an ℓ_∞-norm scaled by Γ. That is, $\|\boldsymbol{x}\|_\Gamma^D = \Gamma\|\boldsymbol{x}\|_\infty$. On the other hand, for $\Gamma = n$, a D-norm is equal to an ℓ_1-norm; $\|\boldsymbol{x}\|_{\Gamma=n}^D = \|\boldsymbol{x}\|_1$. Figure 5.2 shows the D-norm induced units ball for $\Gamma = \{1, 1.3, 1.5, 1.7, 2\}$. The dual of the D-norm, given in (5.6), is the maximum of an ℓ_∞-norm and an ℓ_1-norm scaled by the inverse of Γ. For a detailed derivation of the dual see Bertsimas et al. (2004).

$$\|\boldsymbol{x}\|_\Gamma^{D*} = \max \left\{ \|\boldsymbol{x}\|_\infty, \frac{1}{\Gamma}\|\boldsymbol{x}\|_1 \right\} \tag{5.6}$$

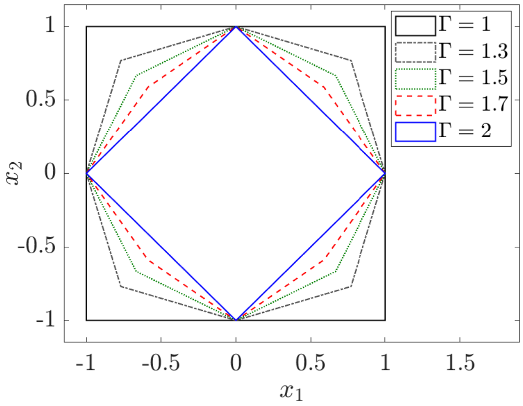


Figure 5.2: D-norm unit balls.

Remark. In the literature, the so-called large-k norm is a special case of a D-norm (Kusunoki & Tatsumi 2019). For a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and a parameter $k \in [1, n]$, a large-k norm is defined as: $\|\boldsymbol{x}\|_k = |x_{(1)}| + |x_{(2)}| + \cdots + |x_{(k)}|$, where $x_{(i)}$ is the element whose absolute value is the i^{th} largest in the n elements of $\boldsymbol{x}$, i.e., $|x_{(1)}| \geq |x_{(2)}| \geq \cdots \geq |x_{(n)}|$.

$CVaR$-norm

A conditional value at risk (i.e., $CVaR$) is equal to the expectation of the $1-\alpha$ tail-percentile of a distribution. For a vector $\boldsymbol{x}$ in the space $\mathbb{R}^n$, a $CVaR$-norm is defined in (5.7) where $0 \leq \alpha \leq 1$, $N = \{1, \cdots, n\}$ and $(y)^+ = \max(0, y)$ (Gotoh & Uryasev 2016). Figure 5.3 shows the $CVaR$-norm induced unit balls for a set of α values $\{0.5, 0.35, 0.25, 0.15, 0\}$.

$$\|\boldsymbol{x}\|_{\alpha}^{CVaR} = \min_{c} \left\{ n(1-\alpha)c + \sum_{i \in N} (|x_i| - c)^+ \right\} \tag{5.7}$$

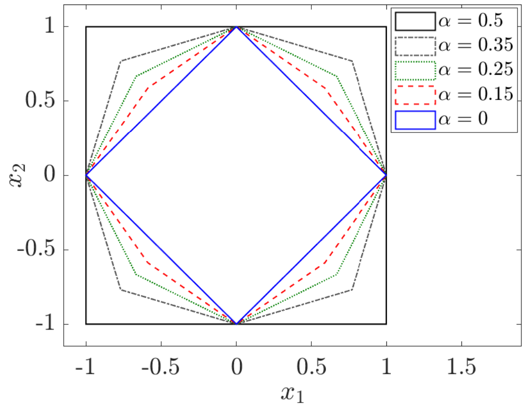


Figure 5.3: $CVaR$-norm induced unit balls

A $CVaR$-norm is proportional to a D-norm for specific values ranges of the parameter α. For $\alpha \in [0, \frac{n-1}{n}]$, a $CVaR$-norm is equal to a D-norm with the parameter Γ equal to $n(1-\alpha)$. Taking the extremal values of α as examples, we have $\|\boldsymbol{x}\|_{\alpha=0}^{CVaR} = \|x\|_{\Gamma=n}^{D} = \|\boldsymbol{x}\|_1$ and $\|\boldsymbol{x}\|_{\alpha=\frac{n-1}{n}}^{CVaR} = \|x\|_{\Gamma=1}^{D} = \|\boldsymbol{x}\|_\infty$. On the opposite end, for $\alpha \in (\frac{n-1}{n}, 1]$, a $CVaR$-norm is equal to a D-norm with $\Gamma = 1$ scaled by the value $n(1-\alpha)$. That is, $\|\boldsymbol{x}\|_{\alpha}^{CVaR} = n(1-\alpha)\|\boldsymbol{x}\|_{\Gamma=1}^{D} = n(1-\alpha)\|\boldsymbol{x}\|_\infty$.

The CVaR dual norm, given in (5.8), is the maximum of an ℓ_∞-norm and an ℓ_1-norm scaled by the inverse of $n(1-\alpha)$. For a detailed derivation of the dual see Gotoh & Uryasev (2016).

$$\|\boldsymbol{x}\|_{\alpha}^{CVaR*} = \max\left\{\|\boldsymbol{x}\|_\infty, \frac{1}{n(1-\alpha)}\|\boldsymbol{x}\|_1\right\} \tag{5.8}$$

Deltoidal-norm

For a vector $\boldsymbol{x}$ in the space $\mathbb{R}^n$, a *Deltoidal*-norm defined in (5.9) is a convex combination of ℓ_1- and ℓ_∞-norms where the parameter λ is between 0 and 1 (Gotoh & Uryasev 2016). Equivalently, a *Deltoidal*-norm may be defined as a convex combination of D- or $CVaR$-norms based on the equivalence properties previously discussed for these two norms. Figure 5.4 illustrates the *Deltoidal*-norm induced unit balls for a set of λ values $\{0, 0.1, 0.4, 0.8, 1\}$.

$$\|\boldsymbol{x}\|_{\lambda}^{Deltoidal} = (1-\lambda)\|\boldsymbol{x}\|_1 + \lambda\|\boldsymbol{x}\|_\infty \tag{5.9}$$

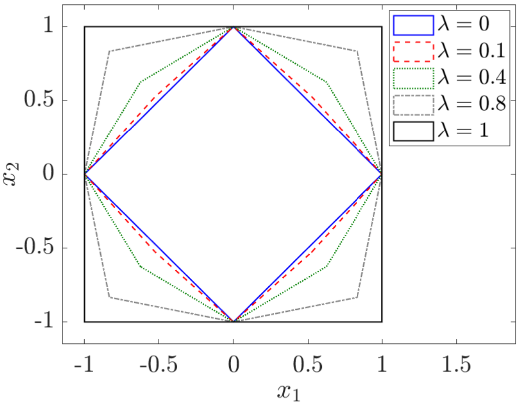


Figure 5.4: *Deltoidal*-norm induced unit balls

The dual of a *Deltoidal*-norm is given in (5.10) as maximum of scaled $CVaR$-norms. For a detailed derivation of the dual see Gotoh & Uryasev (2016).

$$\|\boldsymbol{x}\|_{\lambda}^{Deltoidal*} = \max\left\{\|\boldsymbol{x}\|_{\alpha=\frac{n-1}{n}}^{CVaR}, \frac{1}{2-\lambda}\|\boldsymbol{x}\|_{\alpha=\frac{n-2}{n}}^{CVaR}, \ldots, \frac{1}{n-(n-1)\lambda}\|\boldsymbol{x}\|_{\alpha=0}^{CVaR}\right\} \tag{5.10}$$

Polytope-norm

Given a polytope in the $\mathbb{R}^n$ space defined by the hyperplane $\{\boldsymbol{x} \in \mathbb{R}^n : \mathbf{a}_i^\top \boldsymbol{x} \leq b_i,\ i = 1, \ldots, m\}$ and an inner point $\boldsymbol{\mu}$, a *Polytope*-norm is defined in (5.11).

$$\|\boldsymbol{x}\|_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{Polytope} = \max_{i=1,\cdots,m} \frac{\mathbf{a}_i^\top \boldsymbol{x} - \mathbf{a}_i^\top \boldsymbol{\mu}}{b_i - \mathbf{a}_i^\top \boldsymbol{\mu}} \tag{5.11}$$

For example, consider the polytope $\boldsymbol{Ax} \leq \boldsymbol{b}$, where $\boldsymbol{A}$ = [-1 -1; 0 1; 1 0] and $\boldsymbol{b}$ = [-4; 3; 3], the scaled uncertainty sets $\{\boldsymbol{x} : \|\boldsymbol{x}\|_{A,b,\mu}^{Polytope} \leq \Delta\}$ induced by the *Polytope*-norm are illustrated in Figure 5.5.

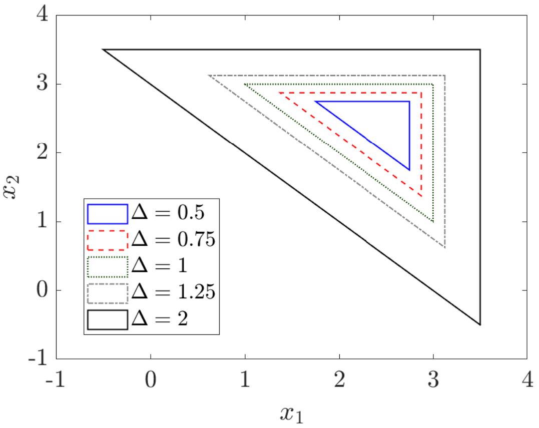


Figure 5.5: *Polytope*-norm induced uncertainty sets with different Δ.

The derivation of a dual *Polytope*-norm is illustrated here. First, we reformulate equation (5.11) as a linear programming model

$$\begin{aligned} \min_{t,\boldsymbol{x}} \quad & t \\ \text{s.t.} \quad & t(b_i - \mathbf{a}_i^\top \boldsymbol{\mu}) \geq \mathbf{a}_i^\top \boldsymbol{x} - \mathbf{a}_i^\top \boldsymbol{\mu} \quad i = 1, \cdots, m \\ & t \geq 0 \end{aligned}$$

Using the dual definition in (5.1), the LP dual norm representation is presented as

$$\begin{aligned} \max_{t,\mathbf{s}} \quad & \mathbf{s}^\top \boldsymbol{x} \\ \text{s.t.} \quad & t \leq 1 \\ & -t(b_i - \mathbf{a}_i^\top \boldsymbol{\mu}) + \mathbf{a}_i^\top \mathbf{s} \leq \mathbf{a}_i^\top \boldsymbol{\mu} \quad i = 1, \ldots, m \\ & t \geq 0 \end{aligned}$$

The dual problem is given below, where the dual variable θ corresponds to the first constraint and the dual variables λ_i for all i correspond to the set of constraints indexed i. Both primal and dual formulations are equivalent based on the strong duality property of LP problems.

$$\begin{aligned}\min_{\theta,\lambda_i} \quad & \theta + \sum_{i=1}^{m} (\boldsymbol{\mu}^\top \mathbf{a}_i)\lambda_i \\ \text{s.t.} \quad & \theta - \sum_{i=1}^{m} (b_i - \boldsymbol{\mu}^\top \mathbf{a}_i)\lambda_i = 0 \\ & \sum_{i=1}^{m} \mathbf{a}_i \lambda_i = \boldsymbol{x} \\ & \lambda_i \geq 0 \quad i = 1, \cdots, m\end{aligned}$$

The variable θ is removed using the information in the first equality constraint. The final polytope dual norm is given in (5.12) where it is independent from any interior point $\boldsymbol{\mu}$.

$$\begin{aligned}\min_{\lambda_i} \quad & \sum_{i=1}^{m} b_i \lambda_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} \mathbf{a}_i \lambda_i = \boldsymbol{x} \\ & \lambda_i \geq 0 \quad i = 1, \cdots, m\end{aligned} \tag{5.12}$$

A summary of the polyhedral norms and their duals is given in Tables 5.1 and 5.2, respectively.

Table 5.1: Polyhedral norm expressions.

Notation	Expression
$\lVert\boldsymbol{x}\rVert_1$	$\sum_{i=1}^{n} \lvert x_i\rvert$
$\lVert\boldsymbol{x}\rVert_\infty$	$\max_{\{i=1,\cdots,n\}} \lvert x_i\rvert$
$\lVert\boldsymbol{x}\rVert_\Gamma^D$	$\max_{\{S\cup\{t\}\mid S\subseteq N, \lvert S\rvert\leq\lfloor\Gamma\rfloor, t\in N\setminus S\}} \sum_{i\in S} \lvert x_i\rvert + (\Gamma - \lfloor\Gamma\rfloor)\lvert x_t\rvert$
$\lVert\boldsymbol{x}\rVert_\alpha^{CVaR}$	$\min_{\{c\}} n(1-\alpha)c + \sum_i (\lvert x_i\rvert - c)^+$
$\lVert\boldsymbol{x}\rVert_\lambda^{Deltoidal}$	$(1-\lambda)\lVert\boldsymbol{x}\rVert_1 + \lambda\lVert\boldsymbol{x}\rVert_\infty$
$\lVert\boldsymbol{x}\rVert_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{Polytope}$	$\min_{\{i=1,\cdots,m\}} \frac{\mathbf{a}_i^\top \boldsymbol{x} - \mathbf{a}_i^\top \boldsymbol{\mu}}{b_i - \mathbf{a}_i^\top \boldsymbol{\mu}}$

Table 5.2: Polyheral dual norm expressions.

Notation	Expression
$\Vert\boldsymbol{x}\Vert_1^*$	$\Vert\boldsymbol{x}\Vert_\infty$
$\Vert\boldsymbol{x}\Vert_\infty^*$	$\Vert\boldsymbol{x}\Vert_1$
$\Vert\boldsymbol{x}\Vert_\Gamma^{*D}$	$\max\left\{\Vert\boldsymbol{x}\Vert_\infty, \frac{1}{\Gamma}\Vert\boldsymbol{x}\Vert_1\right\}$
$\Vert\boldsymbol{x}\Vert_\alpha^{*CVaR}$	$\max\left\{\Vert\boldsymbol{x}\Vert_\infty, \frac{1}{n(1-\alpha)}\Vert\boldsymbol{x}\Vert_1\right\}$
$\Vert\boldsymbol{x}\Vert_\lambda^{*Deltoidal}$	$\max\left\{\Vert\boldsymbol{x}\Vert_{\alpha=\frac{n-1}{n}}^{CVaR}, \frac{1}{2-\lambda}\Vert\boldsymbol{x}\Vert_{\alpha=\frac{n-2}{n}}^{CVaR}, \cdots, \frac{1}{n-(n-1)\lambda}\Vert\boldsymbol{x}\Vert_{\alpha=0}^{CVaR}\right\}$
$\Vert\boldsymbol{x}\Vert_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{*Polytope}$	$\min\left\{\boldsymbol{b}^\top\boldsymbol{\lambda},\ \text{s.t. } \boldsymbol{A}^\top\boldsymbol{\lambda}=\boldsymbol{x},\ \boldsymbol{\lambda}\geq 0\right\}$

All polyhedral norms have linear programming representations (see Table 5.3). For an ℓ_1-norm, auxiliary variables u_i are introduced to model the absolute of x_i. For an ℓ_∞-norm, auxiliary variable v models the maximum of the absolute of x_i. The same notations are applied in *Deltoidal*-norm. For a *CVaR*-norm, auxiliary variables z_i are introduced to reformulate $(|x_i|-c)^+$. For a *D*-norm, its LP representation is derived as follows (Bertsimas et al. 2004)

$$\begin{aligned}
\|\boldsymbol{x}\|_\Gamma^D = \max_{u_i} \quad & \sum_{i=1}^n u_i|x_i| \\
\text{s.t.} \quad & \sum_{i=1}^n u_i \leq \Gamma \\
& 0 \leq u_i \leq 1 \quad i=1,\cdots,n
\end{aligned}$$

An equivalent LP formulation is further constructed from LP duality, after some reformulation.

$$\begin{aligned}
\|\boldsymbol{x}\|_\Gamma^D = \min_{r,t_i} \quad & \Gamma r + \sum_{i=1}^n t_i \\
\text{s.t.} \quad & r + t_i \geq x_i && i=1,\cdots,n \\
& r + t_i \geq -x_i && i=1,\cdots,n \\
& t_i \geq 0 && i=1,\cdots,n \\
& r \geq 0
\end{aligned}$$

Table 5.3: Polyhedral norms LP representations.

Polyhedral norm	LP representation		
$\lVert\boldsymbol{x}\rVert_1$	$\min\limits_{u_i}$	$\sum_{i=1}^n u_i$	
	s.t.	$u_i \geq x_i$	$i = 1, \cdots, n$
		$u_i \geq -x_i$	$i = 1, \cdots, n$
$\lVert\boldsymbol{x}\rVert_\infty$	$\min\limits_{v}$	v	
	s.t.	$v \geq x_i$	$i = 1, \cdots, n$
		$v \geq -x_i$	$i = 1, \cdots, n$
$\lVert\boldsymbol{x}\rVert_\Gamma^D$	$\min\limits_{r,\mathbf{t}}$	$\Gamma r + \sum_{i=1}^n t_i$	
	s.t.	$r + t_i \geq x_i$	$i = 1, \cdots, n$
		$r + t_i \geq -x_i$	$i = 1, \cdots, n$
		$t_i \geq 0$	$i = 1, \cdots, n$
		$r \geq 0$	
$\lVert\boldsymbol{x}\rVert_\alpha^{CVaR}$	$\min\limits_{c,z}$	$n(1-\alpha)c + \sum_{i=1}^n z_i$	
	s.t.	$z_i \geq x_i - c$	$i = 1, \cdots, n$
		$z_i \geq -x_i - c$	$i = 1, \cdots, n$
		$z_i \geq 0$	$i = 1, \cdots, n$
$\lVert\boldsymbol{x}\rVert_\lambda^{Deltoidal}$	$\min\limits_{u_i,v}$	$(1-\lambda)\sum_{i=1}^n u_i + \lambda v$	
	s.t.	$u_i \geq x_i$	$i = 1, \cdots, n$
		$u_i \geq -x_i$	$i = 1, \cdots, n$
		$v \geq x_i$	$i = 1, \cdots, n$
		$v \geq -x_i$	$i = 1, \cdots, n$
$\lVert\boldsymbol{x}\rVert_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{Polytope}$	$\min\limits_{t}$	t	
	s.t.	$t(b_i - \mathbf{a}_i^\top \boldsymbol{\mu}) \geq \mathbf{a}_i^\top \boldsymbol{x} - \mathbf{a}_i^\top \boldsymbol{\mu}$	$i = 1, \cdots, m$

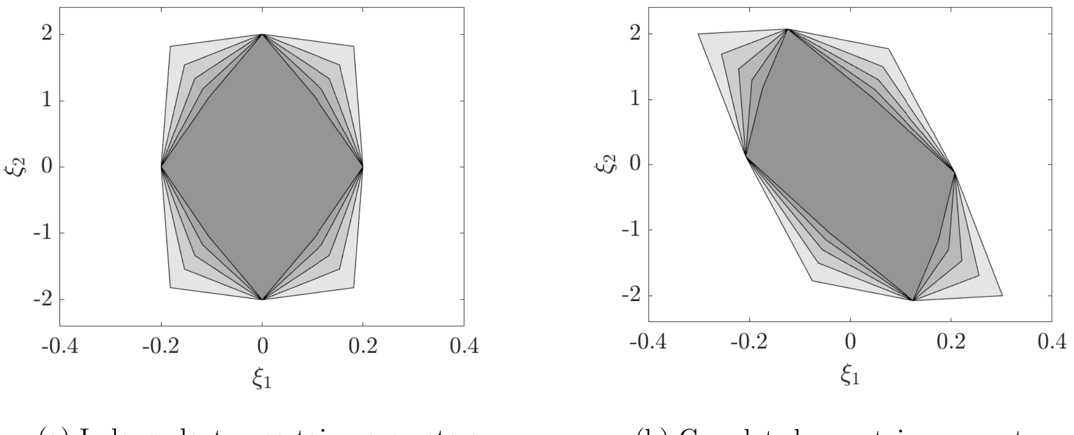
5.3 Norm-induced symmetric uncertainty sets

A general norm-induced symmetric uncertainty set is given in (5.13) where $\mathbf{M} \in \mathbb{R}^{n\times n}$ is an invertible matrix, $\bar{\boldsymbol{\xi}}$ is the uncertain nominal value and Δ is a set size parameter (Bertsimas et al. 2004).

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \;\middle|\; \lVert \mathbf{M}(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}) \rVert \leq \Delta \right\} \tag{5.13}$$

Example.

Consider an uncertain vector $\boldsymbol{\xi} \in [\bar{\boldsymbol{\xi}} - \boldsymbol{d}, \bar{\boldsymbol{\xi}} + \boldsymbol{d}]$ in the $\mathbb{R}^2$ space. Figure 5.6 shows the D-norm induced symmetric uncertainty sets with $\bar{\xi} = [0, 0]$, $\Delta = 1$ and various Γ values $\{1.1, 1.3, 1.5, 1.7, 1.9\}$. The left figure is generated with a matrix $\mathbf{M} = [5, 0; 0, 0.5]$ which corresponds to independent uncertain parameters with $d_1 = 0.2$ and $d_2 = 2$, whereas the right figure is generated using a matrix $\mathbf{M} = [5, 0.3; 0.3, 0.5]$ which corresponds to correlated uncertain parameters through a covariance matrix $\mathbf{\Sigma} = [0.451, -0.066; -0.066, 1.439]$. In both figures, the most outert and inner uncertainty sets are obtained at $\Gamma = 1.1$ and $\Gamma = 1.9$, respectively.



(a) Independent uncertain parameters

(b) Correlated uncertain parameters

Figure 5.6: Symmetric D-norm induced uncertainty sets with a set size $\Delta = 1$ and $\Gamma = \{1.1, 1.3, 1.5, 1.7, 1.9\}$.

Choice of M. Matrix $\mathbf{M}$ in (5.13) scales the uncertain parameters. We give the following two ways for determining $\mathbf{M}$

- If ξ_i is bounded with $\xi_i \in [\bar{\xi_i} - d_i, \bar{\xi_i} + d_i]$ for all i, we define $\mathbf{M}$ as a diagonal matrix with the following diagonal entries $\left\{\frac{1}{d_1}, \cdots, \frac{1}{d_n}\right\}$.
- If ξ_i is unbounded for all i with known variance information, we compute matrix $\mathbf{M}$ using the covariance matrix $\mathbf{M} = \mathbf{\Sigma}^{-\frac{1}{2}}$.

5.4 Norm-induced asymmetric uncertainty sets

5.4.1 Independent primitive uncertainty

Consider an uncertain vector $\boldsymbol{\xi} = (\xi_1, \cdots, \xi_n) \in \mathbb{R}^n$ with independent uncertain elements and a nominal value vector $\bar{\boldsymbol{\xi}}$. The positive and negative perturbation parts of $\boldsymbol{\xi}$ are defined as $\boldsymbol{\xi}^+ = \max\{\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}, 0\}$ and $\boldsymbol{\xi}^- = \max\{-(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}), 0\}$, respectively. An equivalence property is given in (5.14) which is easily verified.

$$\boldsymbol{\xi} - \bar{\boldsymbol{\xi}} = \boldsymbol{\xi}^+ - \boldsymbol{\xi}^- \tag{5.14}$$

If if $\boldsymbol{\xi} - \bar{\boldsymbol{\xi}} \geq 0$, then $\boldsymbol{\xi}^+ - \boldsymbol{\xi}^- = \max\{\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}, 0\} - \max\{-(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}), 0\} = \boldsymbol{\xi} - \bar{\boldsymbol{\xi}} - 0 = \boldsymbol{\xi} - \bar{\boldsymbol{\xi}}$; else $\boldsymbol{\xi} - \bar{\boldsymbol{\xi}} < 0$, then $\boldsymbol{\xi}^+ - \boldsymbol{\xi}^- = \max\{\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}, 0\} - \max\{-(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}), 0\} = 0 - (-(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}})) = \boldsymbol{\xi} - \bar{\boldsymbol{\xi}}$.

A norm-induced asymmetric uncertainty set is modeled in (5.15) using the positive and negative uncertain perturbations and the equivalence property in (5.14). The matrices $\mathbf{P}$ and $\mathbf{Q}$ are in the $\mathbb{R}^{n\times n}$ and $\|\cdot\|$ is a general norm.

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \,\middle|\, \begin{array}{l} \boldsymbol{\xi} = \bar{\boldsymbol{\xi}} + (\boldsymbol{\xi}^+ - \boldsymbol{\xi}^-), \\ \boldsymbol{\xi}^+ \geq 0,\ \boldsymbol{\xi}^- \geq 0, \\ \|\mathbf{P}\boldsymbol{\xi}^+ + \mathbf{Q}\boldsymbol{\xi}^-\| \leq \Delta \end{array} \right\} \tag{5.15}$$

Choice of matrices P and Q. The matrices **P** and **Q** are used for scaling the positive and negative uncertain perturbations, respectively. We give the following two ways for determining **P** and **Q**

- If ξ_i is bounded within the interval $[\bar{\xi}_i - d_i^-, \bar{\xi}_i + d_i^+]$ for all i, the bounds of the perturbations are given as $0 \leq \xi_i^+ \leq d_i^+$ and $0 \leq \xi_i^- \leq d_i^-$ for all $i = \{1, \cdots, n\}$. We select **P** and **Q** to be diagonal matrices where the diagonal entries are $\left\{\frac{1}{d_1^+}, \cdots, \frac{1}{d_n^+}\right\}$ and $\left\{\frac{1}{d_1^-}, \cdots, \frac{1}{d_n^-}\right\}$, respectively.

- If ξ_i is unbounded for all i and the variance information of ξ_i^+ and ξ_i^- are known, we define the standard deviations for ξ_i^+ and ξ_i^- as σ_i^+ and σ_i^-, respectively. We select **P** and **Q** to be diagonal matrices where the diagonal entries are $\left\{\frac{1}{\sigma_1^+}, \cdots, \frac{1}{\sigma_n^+}\right\}$ and $\left\{\frac{1}{\sigma_1^-}, \cdots, \frac{1}{\sigma_n^-}\right\}$, respectively.

5.4.2 Correlated primitive uncertainty

Given an uncertain vector $\boldsymbol{\xi}$ in the $\mathbb{R}^n$ space, a nominal value vector $\bar{\boldsymbol{\xi}}$ and a covariance matrix $\boldsymbol{\Sigma}$, we define a new random vector $\boldsymbol{\eta}$ which defines the deviation of $\boldsymbol{\xi}$ from its nominal value ($\boldsymbol{\eta} = \boldsymbol{\xi} - \bar{\boldsymbol{\xi}}$). The vector $\boldsymbol{\eta}$ also has the same covariance matrix $\boldsymbol{\Sigma}$. Using the unitary matrix of eigenvectors $\boldsymbol{\Phi}$ of the covariance matrix $\boldsymbol{\Sigma}$, we define a new random vector $\boldsymbol{\mu} = \boldsymbol{\Phi}^\top \boldsymbol{\eta}$. The aim of this step is to de-correlate the elements of the random vector $\boldsymbol{\eta}$. The generated random elements μ_i are independent and the covariance matrix is a diagonal matrix with the eigenvalues of $\boldsymbol{\Sigma}$ as the diagonal entries.

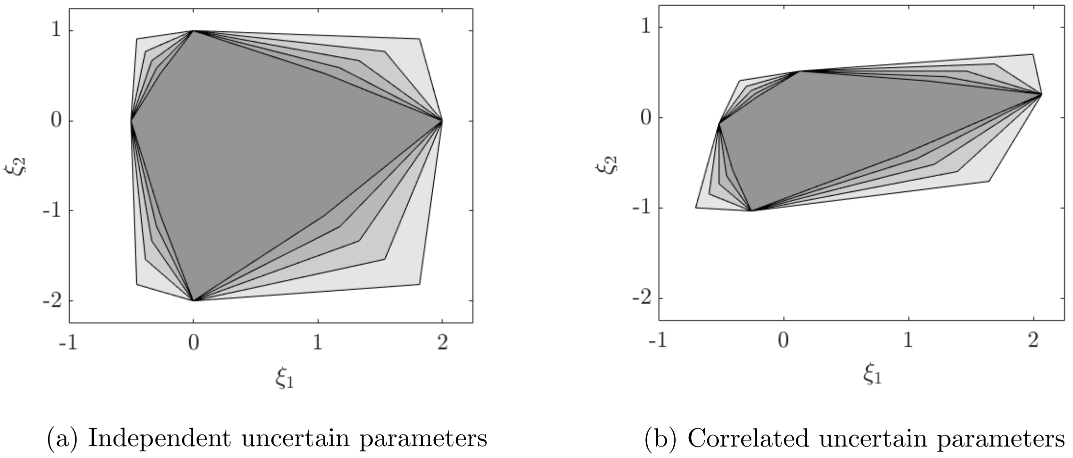
The correlated asymmetric uncertainty set is in given (5.16) where the matrices **P** and **Q** are in the $\mathbb{R}^{n \times n}$ space and $\|\cdot\|$ is a general norm. The vectors $\boldsymbol{\mu}^+$ and $\boldsymbol{\mu}^-$ are the positive (i.e., $\max\{\boldsymbol{\mu}, 0\}$) and negative (i.e., $\max\{-\boldsymbol{\mu}, 0\}$) perturbations of the independent vector $\boldsymbol{\mu}$, respectively.

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \,\middle|\, \begin{array}{l} \boldsymbol{\Phi}^\top(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}) = \boldsymbol{\mu}^+ - \boldsymbol{\mu}^-, \\ \boldsymbol{\mu}^+ \geq 0,\ \boldsymbol{\mu}^- \geq 0, \\ \|\mathbf{P}\boldsymbol{\mu}^+ + \mathbf{Q}\boldsymbol{\mu}^-\| \leq \Delta \end{array} \right\} \tag{5.16}$$

The matrices **P** and **Q** are selected using the same procedure as for the independent asymmetric uncertainty set in (5.15). In the special case where there is no correlation among the elements of $\boldsymbol{\xi}$, the de-correlation matrix $\boldsymbol{\Phi}$ is set as the identity matrix and formulation (5.16) reduces to (5.15).

Example.

Consider an uncertain vector $\boldsymbol{\xi}$ in the $\mathbb{R}^2$ space. Figure 5.7 shows the D-norm induced asymmetric uncertainty sets with a set size $\Delta = 1$ and different Γ values $\{1.1, 1.3, 1.5, 1.7, 1.9\}$. The left figure corresponds to an independent uncertain vector where $\mathbf{\Phi} = \mathbf{I}$ and the perturbations bounds are $d_1^+ = 2,\ d_2^+ = 1,\ d_1^- = 0.5$ and $d_2^- = 2$. The right figure corresponds to a correlated uncertain elements where $\mathbf{\Phi} = [1, -0.25; -0.25, 2]$ and the perturbations variance information are $\sigma_1^+ = 2,\ \sigma_2^+ = 1,\ \sigma_1^- = 0.5$ and $\sigma_2^- = 2$. In both figures, the most outer and inner uncertainty sets are obtained at $\Gamma = 1.1$ and $\Gamma = 1.9$, respectively.



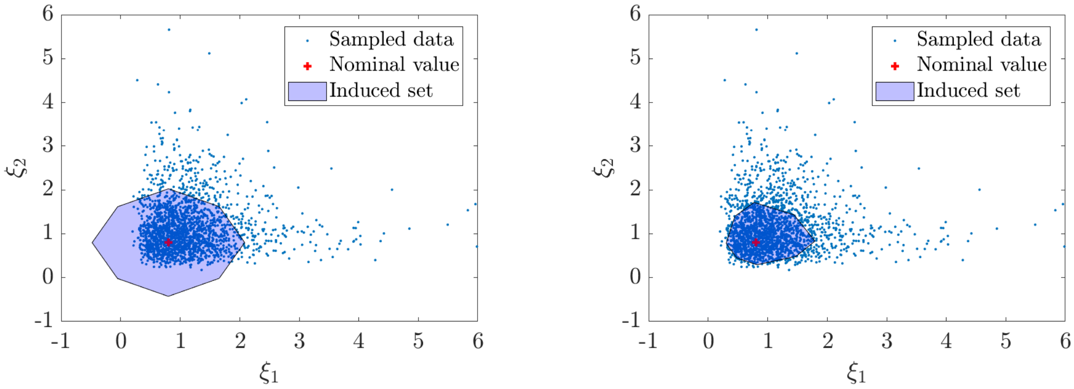
(a) Independent uncertain parameters

(b) Correlated uncertain parameters

Figure 5.7: Asymmetric D-norm induced uncertainty sets with a set size $\Delta = 1$ and $\Gamma = \{1.1, 1.3, 1.5, 1.7, 1.9\}$.

Numerical illustrations

In this section, we illustrate the symmetric and asymmetric uncertainty set construction for independent and correlated uncertain parameters given uncertainty distributional information via two numerical examples. In the first example, we generate 2000 scenarios for two uncertain parameters using independent lognormal distribution $log\mathcal{N}(0, 0.25)$, variance $(\sigma_1^2, \sigma_2^2) = (0.4097, 0.3759)$ and nominal values $\bar{\boldsymbol{\xi}} = [0.8, 0.8]$. Figure 5.8 shows the D-norm induced symmetric and asymmetric uncertainty sets where Γ is equal to 1.5 and the set size Δ is equal to 2.0. For the symmetric set, we have $\mathbf{M} = [1.5624, 0; 0, 1.6310]$ and for the asymmetric set, we have $\mathbf{\Phi} = [0.0322, -0.9995; -0.9995, -0.0322]$, $\mathbf{P} = [3.9441, 0; 0, 1.1424]$ and $\mathbf{Q} = [2.1868, 0; 0, 4.0069]$.



(a) Symmetric uncertainty set

(b) Asymmetric uncertainty set

Figure 5.8: Data-driven D-norm symmetric and asymmetric induced uncertainty sets for independent uncertainty following marginal lognormal distributions $log\mathcal{N}(0, 0.25)$.

The symmetric set of size $\Gamma = 1.5$ above has a volume value 2.35, which covers 1368 out of the 2000 (68.40%) samples. For the asymmetric set of the same size $\Gamma = 1.5$, the volume is only 0.8, and 875 out of 2000 (43.75%) samples are covered. To further investigate the relation between the set volume and the sample coverage, we change the set size as Γ=[0.5, 1.0, 2.0, 3.0, 4.0, 5.0] and obtain the plot shown in Figure 5.9. It is seen from the plot that to cover the same number of samples, the symmetric set has a larger volume than the asymmetric set. In other words, the symmetric set is more conservative.

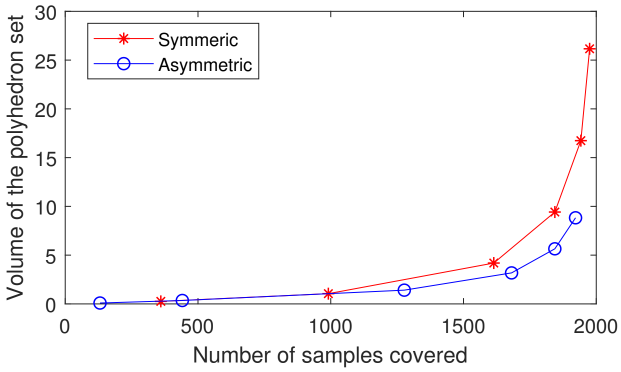
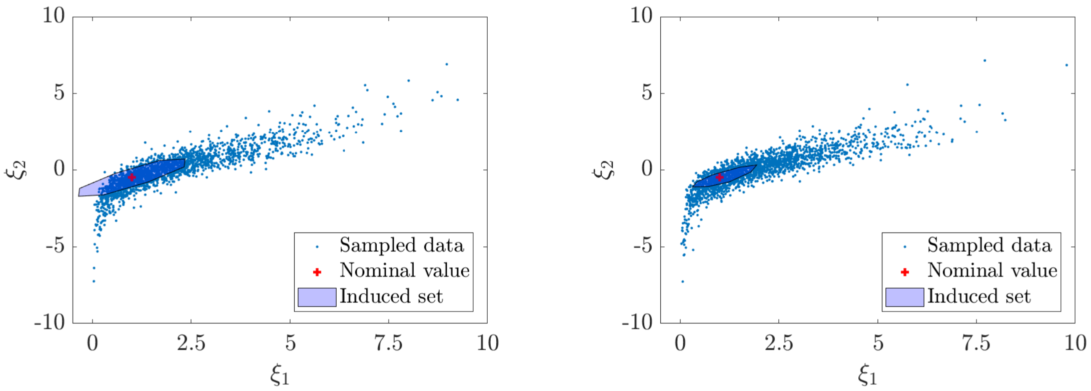


Figure 5.9: Volume of the polyhedron set and number of samples covered from different set size Γ=[0.5, 1.0, 2.0, 3.0, 4.0, 5.0].

In the second example, we consider 2000 scenarios of correlated uncertain parameters where ξ_1 follows a Gamma-distribution $\Gamma(2, 1)$ and ξ_2 follows a t-distribution with 5 degrees of freedom. The nominal values are $\bar{\boldsymbol{\xi}} = [1, -0.5]$ and the covariance matrix is $\boldsymbol{\Sigma} = [1.9363, 1.492; 1.4492, 1.4978]$. Figure 5.10 illustrates the D-norm induced symmetric and asymmetric uncertainty sets where Γ is equal to 1.5 and the set size Δ is equal to 1.0. For the symmetric set, we have $\mathbf{M} = [1.1702, -0.7088; -0.7088, 1.3847]$ and for the asymmetric set, we have $\boldsymbol{\Phi} = [0.6521, -0.7582; -0.7582, -0.6521]$, $\mathbf{P} = [2.8930, 0; 0, 1.1424]$ and $\mathbf{Q} = [4.0, 0; 0, 0.8158]$.



(a) Symmetric uncertainty set

(b) Asymmetric uncertainty set

Figure 5.10: Data-driven D-norm, with $\Gamma = 1.5$, induced symmetric and asymmetric uncertainty sets where correlated ξ_1 and ξ_2 follow a Gamma- and t-distributions, respectively.

In the two examples, the polyhedra represent the constructed uncertainty sets and the sampling data is displayed to show the density of the distribution. The symmetric and asymmetric constructed uncertainty sets capture the correlation between the uncertain parameters in the second example. It is also observed that using the same set size and Γ parameter for a D-norm, the symmetric set is larger in both cases. The symmetric uncertainty set unnecessarily covers some low density region, while the asymmetric uncertainty set fits better to the joint distribution.

5.5 Robust linear optimization under norm induced uncertainty sets

Consider an uncertain vector $\boldsymbol{\xi} \in \mathbb{R}^n$ in all linear constraints of a robust optimization problem. After re-arrangement, and introducing auxiliary variables if necessary, we get a set of general linear uncertain constraints $y_0^i + \sum_{k=1}^{n} \xi_k y_k^i \leq 0$, for all $i = 1, 2, \cdots, m$. An example of such reformulation is given in the case studies section. The constraints are rewritten in a vector form in (5.17) where $\boldsymbol{y}^i = [y_1^i, \cdots, y_n^i]^\top$ and $\boldsymbol{\xi} = [\xi_1, \cdots, \xi_n]^\top$.

$$y_0^i + \boldsymbol{\xi}^\top \boldsymbol{y}^i \leq 0 \quad i = 1, 2, \cdots, m \tag{5.17}$$

5.5.1 Robust counterpart under symmetric uncertainty sets

Property 1 *Under the symmetric general norm-induced uncertainty set in* (5.13)*, the robust counterpart of* (5.17) *is equivalent to*

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|^* \leq 0 \quad i = 1, 2, \cdots, m \tag{5.18}$$

Proof. Let $\mathbf{s} = \frac{\mathbf{M}(\boldsymbol{\xi}-\bar{\boldsymbol{\xi}})}{\Delta}$. The uncertainty is rewritten as

$$\boldsymbol{\xi} = \Delta \mathbf{M}^{-1}\mathbf{s} + \bar{\boldsymbol{\xi}}$$

where the variable s belongs to a unit ball uncertainty set defined as

$$S = \{\mathbf{s} \in \mathbb{R}^n | \; \|\mathbf{s}\| \leq 1\}$$

The robust counterpart of equation (5.17) is equal to

$$y_0^i + \max_{\boldsymbol{\xi} \in U} \; \boldsymbol{\xi}^\top \boldsymbol{y}^i \leq 0 \quad i = 1, 2, \ldots, m$$

Introducing the variable s

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \max_{\|\mathbf{s}\| \leq 1} \Delta(\mathbf{M}^{-1}\mathbf{s})^\top \boldsymbol{y}^i \leq 0 \quad i = 1, 2, \ldots, m$$

Since $\Delta > 0$, we have

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \max_{\|\mathbf{s}\| \leq 1} \mathbf{s}^\top(\mathbf{M}^{-\top}\boldsymbol{y}^i) \leq 0 \quad i = 1, 2, \ldots, m$$

The maximization problem is equal to the dual norm definition of a general norm in equation (5.1). The final form of the robust counterpart is

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{M}^{-\top}\boldsymbol{y}^i\|^* \leq 0 \quad i = 1, 2, \ldots, m$$

□

Example 5.1.1: ℓ_1-norm: The robust counterpart using an ℓ_1-norm induced uncertainty set is given as

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{M}^{-\top}\boldsymbol{y}^i\|_1^* \leq 0 \qquad \Rightarrow \qquad y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{M}^{-\top}\boldsymbol{y}^i\|_\infty \leq 0$$

The equivalent LP representation is

$$\left.\begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot z^i \leq 0 \\ -z^i \leq (\mathbf{M}^{-\top}\boldsymbol{y}^i)_j \leq z^i \quad j = 1, \cdots, n \end{array}\right\} \; i = 1, \cdots, m$$

Example 5.1.2: ℓ_∞-norm: The robust counterpart using an ℓ_∞-norm induced uncertainty set is given as

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{M}^{-\top}\boldsymbol{y}^i\|_\infty^* \leq 0 \qquad \Rightarrow \qquad y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{M}^{-\top}\boldsymbol{y}^i\|_1 \leq 0$$

The equivalent LP representation is

$$\left.\begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \mathbf{1}^\top \boldsymbol{z}^i \leq 0 \\ -\boldsymbol{z}^i \leq \mathbf{M}^{-\top} \boldsymbol{y}^i \leq \boldsymbol{z}^i \end{array}\right\} \quad i = 1, \cdots, m$$

where the vector $\mathbf{1}^\top$ is a unit-vector in the $\mathbb{R}^n$ space.

Example 5.1.3: D-norm: The robust counterpart using a D-norm induced uncertainty set is given as

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_\Gamma^{D*} \leq 0$$

which is equivalent to

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \max \left\{ \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_\infty, \frac{1}{\Gamma} \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_1 \right\} \leq 0$$

The equivalent LP presentation is derived in (5.19) by adding an auxiliary variable z^i to model the maximization and introducing the LP representations of both ℓ_1- and ℓ_∞-norms.

$$\left.\begin{array}{ll} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta z^i \leq 0 & \\ z^i \geq u_j^i & j = 1, \cdots, n \\ z^i \geq \frac{1}{\Gamma} \cdot \mathbf{1}^\top \boldsymbol{u}^i & \\ -\boldsymbol{u}^i \leq \mathbf{M}^{-\top} \boldsymbol{y}^i \leq \boldsymbol{u}^i & \end{array}\right\} \quad i = 1, \ldots, m \tag{5.19}$$

Example 5.1.4: $CVaR$-norm: The dual $CVaR$-norm in (5.8) is identical to the dual D-norm in (5.6) except for the parameter Γ which is substituted for $n(1-\alpha)$. Consequently, a robust counterpart under a $CVaR$-norm induced uncertainty set is equal formulation (5.19) while replacing Γ with $n(1-\alpha)$.

Example 5.1.5: *Deltoidal*-norm: The robust counterpart in (5.18) under a *Deltoidal*-norm induced uncertainty set is given as

$$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \max \left\{ \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_{\alpha=\frac{n-1}{n}}^{CVaR}, \frac{1}{2-\lambda} \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_{\alpha=\frac{n-2}{n}}^{CVaR}, \ldots, \frac{1}{n-(n-1)\lambda} \|\mathbf{M}^{-\top} \boldsymbol{y}^i\|_{\alpha=0}^{CVaR} \right\} \leq 0$$

The dual *Deltoidal*-norm includes a maximization of a set $n - CVaR$-norms which are LP representable. An auxiliary variable z^i is introduced to model the maximization operator

$$\begin{aligned} z^i &\geq \min \ \frac{1}{k-(k-1)\lambda} \left[n\left(1 - \frac{n-k}{n}\right) c_k + \sum_{j=1}^n \left(\left|(\mathbf{M}^{-\top} \boldsymbol{y}^i)_j\right| - c_k \right)^+ \right] \\ &\geq \min \ \frac{1}{k-(k-1)\lambda} \left[k c_k + \sum_{j=1}^n \left(\left|(\mathbf{M}^{-\top} \boldsymbol{y}^i)_j\right| - c_k \right)^+ \right] \qquad k = 1, \ldots, n \end{aligned} \tag{5.20}$$

The minimization over c_k is amended to the minimization of the objective function value without affecting optimality and feasibility of equation (5.20). Consequently, the minimization operator in (5.20) is dropped and c_k is still modelled as an optimization variable.

Additional auxiliary variables are introduced to eliminate non-linearities. The reformulated LP robust counterpart under a *Deltoidal*-norm induced uncertainty set is equal to

$$\left.\begin{array}{ll} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot z^i \leq 0 & \\ z^i \geq \frac{1}{k-(k-1)\lambda}\left[kc_k + \mathbf{1}^\top \boldsymbol{v}_k^i\right] & k = 1, \cdots, n \\ \boldsymbol{v}_k^i \geq \mathbf{M}^{-\top}\boldsymbol{y}^i - c_k.\mathbf{1} & k = 1, \cdots, n \\ \boldsymbol{v}_k^i \geq -\mathbf{M}^{-\top}\boldsymbol{y}^i - c_k.\mathbf{1} & k = 1, \cdots, n \\ \boldsymbol{v}_k^i \geq 0 & k = 1, \cdots, n \end{array}\right\} i = 1, \cdots, m$$

Example 5.1.6: *Polytope*-norm: The dual *Polytope*-norm is already in an LP representable form as shown in (5.12). The robust counterpart in (5.18) under a *Polytope*-norm induced uncertainty set is obtained after dropping the minimization operator from the *Polytope* dual-norm.

$$\left.\begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot \mathbf{b}^\top \boldsymbol{\lambda}^i \leq 0 \\ \boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{M}^{-\top}\mathbf{y}^i \\ \boldsymbol{\lambda}^i \geq 0 \end{array}\right\} i = 1, \ldots, m$$

A summary of the robust counterpart LP representation of a general linear uncertain constraint under various polyhedral-norms induced symmetric uncertainty sets is given in Table 5.4. A comparison between the number of variables and constraints in each robust counterpart is given in Table 5.5.

Table 5.4: Robust counterpart LP-representation of the uncertain constraint $y_0^i + \max_{\boldsymbol{\xi} \in U} \boldsymbol{\xi}^\top \mathbf{y}^i \leq 0$, where U is a norm induced symmetric uncertainty set given in (5.13).

Polyehdral norms	Robust counterpart
ℓ_1-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0$ $-z^i \leq (\mathbf{M}^{-\top} \boldsymbol{y}^i)_j \leq z^i \quad j = 1, ..., n$
ℓ_∞-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot \mathbf{1}^\top \mathbf{z}^i \leq 0$ $-\mathbf{z}^i \leq \mathbf{M}^{-\top} \mathbf{y}^i \leq \mathbf{z}^i$
D-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0$ $z^i \geq u_j^i \qquad j = 1, ..., n$ $z^i \geq \frac{1}{\Gamma} \cdot \mathbf{1}^\top \boldsymbol{u}^i$ $-\boldsymbol{u}^i \leq \mathbf{M}^{-\top} \mathbf{y}^i \leq \boldsymbol{u}^i$
$CVaR$-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0$ $z^i \geq u_j^i \qquad j = 1, ..., n$ $z^i \geq \frac{1}{n(1-\alpha)} \cdot \mathbf{1}^\top \boldsymbol{u}^i$ $-\boldsymbol{u}^i \leq \mathbf{M}^{-\top} \mathbf{y}^i \leq \boldsymbol{u}^i$
Deltoidal-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0$ $z^i \geq \frac{1}{k-(k-1)\lambda} \big(k c_k + \mathbf{1}^\top \boldsymbol{v}_k^i\big) \qquad k = 1, \ldots, n$ $-(\boldsymbol{v}_k^i + c_k.\mathbf{1}) \leq \mathbf{M}^{-\top} \boldsymbol{y}^i \leq (\boldsymbol{v}_k^i + c_k.\mathbf{1}) \quad k = 1, \ldots, n$ $\boldsymbol{v}_k^i \geq 0 \qquad k = 1, \ldots, n$
Polytope-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \mathbf{b}^\top \boldsymbol{\lambda}^i \leq 0$ $\boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{M}^{-\top} \mathbf{y}^i$ $\boldsymbol{\lambda}^i \geq 0$

Table 5.5: Number of variables and constraints of robust counterparts in Table 5.4. Size of $\boldsymbol{\xi}$ is given by n; number of hyperplanes in a polytope is given by m'.

Polyhedral norms	Variables	Constraints
ℓ_1-norm	$2 + n$	$1 + 2n$
ℓ_∞-norm	$1 + 2n$	$1 + 2n$
D-norm	$2 + 2n$	$2 + 3n$
$CVaR$-norm	$2 + 2n$	$2 + 3n$
Deltoidal-norm	$2 + n + n(1 + n)$	$1 + n(1 + 3n)$
Polytope-norm	$1 + n + m'$	$1 + n + m'$

5.5.2 Robust counterpart under asymmetric uncertainty sets

Property 2 *Under the general asymmetric norm-induced uncertainty set* (5.16)*, the robust counterpart of* (5.17) *is given in* (5.21) *where* $\|.\|^*$ *is the dual norm.*

$$\left.\begin{array}{l} y_0^i + (\boldsymbol{y}^i)^\top \bar{\boldsymbol{\xi}} + \Delta \|\mathbf{t}^i\|^* \leq 0 \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i \\ \mathbf{t}^i \geq 0 \end{array}\right\} \; i = 1, \ldots, m \tag{5.21}$$

Proof. Consider the inner maximization problem in (5.17) where $\boldsymbol{\xi}$ belongs to the uncertainty set U in (5.16).

$$\begin{array}{ll} \max\limits_{\boldsymbol{\xi}, \boldsymbol{\mu}^+, \boldsymbol{\mu}^-} & \boldsymbol{\xi}^\top \mathbf{y}^i \\ \text{s.t.} & \boldsymbol{\Phi}^\top (\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}) = \boldsymbol{\mu}^+ - \boldsymbol{\mu}^- \\ & \|\mathbf{P}\boldsymbol{\mu}^+ + \mathbf{Q}\boldsymbol{\mu}^-\| \leq \Delta \\ & \boldsymbol{\mu}^+ \geq 0, \; \boldsymbol{\mu}^- \geq 0 \end{array}$$

The model is simplified by removing the variable $\boldsymbol{\xi}$ using the equality constraint $\boldsymbol{\xi} = \bar{\boldsymbol{\xi}} + \boldsymbol{\Phi}^{-\top}(\boldsymbol{\mu}^+ - \boldsymbol{\mu}^-)$,

$$\begin{array}{ll} \max\limits_{\boldsymbol{\mu}^+, \boldsymbol{\mu}^-} & (\boldsymbol{y}^i)^\top \bar{\boldsymbol{\xi}} + (\boldsymbol{y}^i)^\top \boldsymbol{\Phi}^{-\top} \boldsymbol{\mu}^+ - (\boldsymbol{y}^i)^\top \boldsymbol{\Phi}^{-\top} \boldsymbol{\mu}^- \\ \text{s.t.} & \|\mathbf{P}\boldsymbol{\mu}^+ + \mathbf{Q}\boldsymbol{\mu}^-\| \leq \Delta \\ & \boldsymbol{\mu}^+ \geq 0, \boldsymbol{\mu}^- \geq 0 \end{array}$$

Define new variables $\boldsymbol{v} = \mathbf{P}\boldsymbol{\mu}^+$ and $\boldsymbol{w} = \mathbf{Q}\boldsymbol{\mu}^-$,

$$\begin{array}{ll} \max\limits_{\boldsymbol{v}, \boldsymbol{w}} & (\boldsymbol{y}^i)^\top \bar{\boldsymbol{\xi}} + (\boldsymbol{y}^i)^\top \boldsymbol{\Phi}^{-\top} \mathbf{P}^{-1} \boldsymbol{v} - (\boldsymbol{y}^i)^\top \boldsymbol{\Phi}^{-\top} \mathbf{Q}^{-1} \boldsymbol{w} \\ \text{s.t.} & \|\boldsymbol{v} + \boldsymbol{w}\| \leq \Delta \\ & \boldsymbol{v} \geq 0, \boldsymbol{w} \geq 0 \end{array}$$

or

$$\begin{array}{ll} \max\limits_{\boldsymbol{v}, \boldsymbol{w}} & (\boldsymbol{y}^i)^\top \bar{\boldsymbol{\xi}} + (\mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i)^\top \boldsymbol{v} - (\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i)^\top \boldsymbol{w} \\ \text{s.t.} & \|\boldsymbol{v} + \boldsymbol{w}\| \leq \Delta \\ & \boldsymbol{v} \geq 0, \boldsymbol{w} \geq 0 \end{array}$$

the optimal objective is $(\boldsymbol{y}^i)^\top \bar{\boldsymbol{\xi}} + \Delta \|\mathbf{t}^i\|^*$ (Chen et al. 2007), with

$$t_j^i = \max\{(\mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i)_j, (-\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\boldsymbol{y}^i)_j, 0\}$$

The max operator can be represented by three linear inequalities, and the overall LP representation of the robust counterpart is given in (5.21). □

Example 5.2.1: ℓ_1-norm: Applying the robust counterpart formulation in (5.21), we get $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_1^* \leq 0$, which is equivalent to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_\infty \leq 0$. Implementing the LP representation of $\|\mathbf{t}^i\|_\infty$, the robust counterpart LP formulation under ℓ_1-norm induced asymmetric uncertainty set is obtained where $|\mathbf{t}^i| = \mathbf{t}^i$ because the vector $\mathbf{t}^i$ is non-negative.

$$\left.\begin{array}{ll} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta z^i \leq 0 & \\ z^i \geq t_j^i & j = 1, \ldots, n \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq 0 & \end{array}\right\} i = 1, \ldots, m$$

Example 5.2.2: ℓ_∞-norm: The robust counterpart $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_\infty^* \leq 0$ is equivalent to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_1 \leq 0$. Substituting $\|\mathbf{t}^i\|_1$ by its LP representation, the robust counterpart LP formulation under ℓ_∞-norm induced asymmetric uncertainty set is obtained where $|\mathbf{t}^i| = \mathbf{t}^i$ because the vector $\mathbf{t}^i$ is non-negative.

$$\left.\begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \mathbf{1}^\top \mathbf{t}^i \leq 0 \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i \\ \mathbf{t}^i \geq 0 \end{array}\right\} i = 1, \ldots, m$$

Example 5.2.3: D-norm: The robust counterpart $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_\Gamma^{D*} \leq 0$ is equivalent to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \max\{\frac{1}{\Gamma}\|\mathbf{t}^i\|_1, \|\mathbf{t}^i\|_\infty\} \leq 0$. Both $\|\mathbf{t}^i\|_1$ and $\|\mathbf{t}^i\|_\infty$ are LP-representable. The robust counterpart LP formulation under D-norm induced asymmetric uncertainty set is obtained where $|\mathbf{t}^i| = \mathbf{t}^i$ because the vector $\mathbf{t}^i$ is non-negative.

$$\left.\begin{array}{ll} y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0 & \\ z^i \geq t_j^i & j = 1, \ldots, n \\ z^i \geq \frac{1}{\Gamma} \cdot \mathbf{1}^\top \mathbf{t}^i & \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq 0 & \end{array}\right\} i = 1, \ldots, m \tag{5.26}$$

Example 5.2.4: $CVaR$-norm: The robust counterpart LP formulation under a $CVaR$-norm induced asymmetric uncertainty set is identical to the formulation in (5.26) except for the parameter $\frac{1}{\Gamma}$ which is replaced by $\frac{1}{n(1-\alpha)}$.

Example 5.2.5: $Deltoidal$-norm: The robust counterpart is equal to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta\|\mathbf{t}^i\|_\lambda^{Deltoidal*}$

≤ 0 which is equivalent to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \max\limits_{k=1,\ldots,n} \left\{ \frac{1}{k-(k-1)\lambda} \|\mathbf{t}^i\|_{\alpha=\frac{n-k}{n}}^{CVaR} \right\} \leq 0$. The robust counterpart LP formulation under *Deltoidal*-norm induced asymmetric uncertainty set is equal to

$$\left.\begin{array}{ll} y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot z^i \leq 0 & \\ z^i \geq \frac{1}{k-(k-1)\lambda}[kc_k + \mathbf{1}^\top \boldsymbol{v}_k^i] & k = 1, \ldots, n \\ \boldsymbol{v}_k^i \geq \mathbf{t}^i - c_k.\mathbf{1} & k = 1, \ldots, n \\ \boldsymbol{v}_k^i \geq -\mathbf{t}^i - c_k.\mathbf{1} & k = 1, \ldots, n \\ \boldsymbol{v}_k^i \geq 0 & k = 1, \ldots, n \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\mathbf{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\mathbf{\Phi}^{-1}\mathbf{y}^i & \\ \mathbf{t}^i \geq 0 & \end{array}\right\} i = 1, \ldots, m$$

Since $\boldsymbol{v}_k^i \geq \mathbf{t}^i - c_k \cdot \mathbf{1}$ and $\mathbf{t}^i \geq 0$, the inequality $\boldsymbol{v}_k^i \geq -\mathbf{t}^i - c_k \cdot \mathbf{1}$ is redundant.

Example 5.2.6: *Polytope*-norm: The robust counterpart is equal to $y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \|\mathbf{t}\|^{Polytope*} \leq 0$ where the dual norm is LP-representable. The robust counterpart LP formulation under a *Polytope*-norm is obtained after dropping the minimization operator.

$$\left.\begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \mathbf{b}^\top \boldsymbol{\lambda}^i \leq 0 \\ \boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{t}^i \\ \boldsymbol{\lambda}^i \geq 0 \\ \mathbf{t}^i \geq \mathbf{P}^{-1}\mathbf{\Phi}^{-1}\mathbf{y}^i \\ \mathbf{t}^i \geq -\mathbf{Q}^{-1}\mathbf{\Phi}^{-1}\mathbf{y}^i \\ \mathbf{t}^i \geq 0 \end{array}\right\} i = 1, \ldots, m$$

A summary of the robust counterpart LP representation of a general linear uncertain constraint under various polyhedral-norms induced asymmetric uncertainty sets is given in Table 5.6. A comparison between the number of variables and constraints in each robust counterpart is given in Table 5.7.

Table 5.6: Robust counterpart LP-representation of the uncertain constraint $y_0^i + \max_{\boldsymbol{\xi}\in U} \boldsymbol{\xi}^\top \mathbf{y}^i \leq 0$, where U is a norm induced asymmetric uncertainty set given in (5.16).

Polyhedral norms	Robust counterparts
ℓ_1-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot z^i \leq 0$ $z^i \geq t_j^i \qquad j = 1, \ldots, n$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$
ℓ_∞-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \mathbf{1}^\top \mathbf{t}^i \leq 0$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$
D-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta z^i \leq 0$ $z^i \geq t_j^i \qquad j = 1, \ldots, n$ $z^i \geq \frac{1}{\Gamma} \cdot \mathbf{1}^\top \mathbf{t}^i$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$
$CVaR$-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot z^i \leq 0$ $z^i \geq t_j^i \qquad j = 1, \ldots, n$ $z^i \geq \frac{1}{n(1-\alpha)} \cdot \mathbf{1}^\top \mathbf{t}^i$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$
Deltoidal-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \mathbf{y}^i + \Delta \cdot z^i \leq 0$ $z^i \geq \frac{1}{k-(k-1)\lambda}[c_k + \mathbf{1}^\top \boldsymbol{v}_k^i] \quad k = 1, \ldots, n$ $\boldsymbol{v}_k^i \geq \mathbf{t}^i - c_k.\mathbf{1} \quad k = 1, \ldots, n$ $\boldsymbol{v}_k^i \geq 0 \quad k = 1, \ldots, n$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$
Polytope-norm	$y_0^i + \bar{\boldsymbol{\xi}}^\top \boldsymbol{y}^i + \Delta \cdot \mathbf{b}^\top \boldsymbol{\lambda}^i \leq 0$ $\boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{t}^i$ $\boldsymbol{\lambda}^i \geq 0$ $\mathbf{t}^i \geq \mathbf{P}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq -\mathbf{Q}^{-1}\boldsymbol{\Phi}^{-1}\mathbf{y}^i$ $\mathbf{t}^i \geq 0$

Table 5.7: Number of variables and constraints of robust counterparts in Table 5.6. Size of $\boldsymbol{\xi}$ is given by n; number of hyperplanes in a polytope is given by m'.

Polyhedral norms	Number of Variables	Number of Constraints
ℓ_1-norm	$2+2n$	$1+4n$
ℓ_∞-norm	$1+2n$	$1+3n$
D-norm	$2+2n$	$2+4n$
$CVaR$-norm	$2+2n$	$2+4n$
Deltoidal-norm	$2+2n+n(1+n)$	$1+3n+n(1+2n)$
Polytope-norm	$1+2n+m'$	$1+4n+m'$

5.6 Robust linear optimization with an intersection of norm-induced uncertainty sets

In some instances, the uncertainty space is defined as an intersection of more than one norm-induced uncertainty set. The robust counterparts previously derived are not applicable. In this section, we derive a robust counterpart formulation for a general uncertain constraint in (5.17) where the uncertainty belongs to (i) an intersection of two norms induced symmetric uncertainty sets and (ii) an intersection of asymmetric and asymetric norms induced uncertainty sets. The results are supported by proofs and two examples for the former case.

Property 3 *Let the uncertainty set be the intersection of two general norm-induced symmetric sets as in* (5.27) *where* $\bar{\boldsymbol{\xi}}_1$ and $\bar{\boldsymbol{\xi}}_2$ *are the centers of the two uncertainty sets.*

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \middle| \; \|\mathbf{M}_1(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1)\|_A \leq \Delta_1, \; \|\mathbf{M}_2(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_2)\|_B \leq \Delta_2 \right\} \tag{5.27}$$

The robust counterpart of (5.17) *for* $\boldsymbol{\xi} \in U$ *is equal to*

$$y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \left\| \mathbf{M}_1^{-\top}(\mathbf{r}^i - \boldsymbol{y}^i) \right\|_A^* + \Delta_2 \left\| \mathbf{M}_2^{-\top} \mathbf{r}^i \right\|_B^* \leq 0 \; \forall i = 1, \ldots, m \tag{5.28}$$

where $\mathbf{r}^i = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}^i \in \mathbb{R}^n, \; \boldsymbol{w}^i \in \mathbb{R}^n$ *and* $\|.\|^*$ *is the dual norm.*

Proof. The proof is based on applying conic duality to the inner maximization problem (Matthews et al. 2018, section 3)

$$\max_{\boldsymbol{\xi}} \left\{ \boldsymbol{\xi}^\top \boldsymbol{y}^i : \; \|\mathbf{M}_1(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1)\|_A \leq \Delta_1, \; \|\mathbf{M}_2(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_2)\|_B \leq \Delta_2 \right\}$$

Given the general norms properties: non-negativity, absolute homogeneity, sub-additivity and point separating, it is verified that a norm ball of radius r and center x_c, given by $\{x \mid \|x - x_c\| \leq r\}$, is associated with the norm cone $K = \{(x,t) \subset \mathbb{R}^{n+1} \mid \|x\| \leq t\}$ (Boyd et al. 2004). This property is used to construct the problem's conic representation, after reversing the sense of the optimization problem, as follows

$$\min_{\boldsymbol{\xi}} \Big\{ -(\boldsymbol{y}^i)^\top \boldsymbol{\xi} : \ \mathbf{P}_1\boldsymbol{\xi} - \mathbf{p}_1 \in K_1, \ \mathbf{P}_2\boldsymbol{\xi} - \mathbf{p}_2 \in K_2 \Big\}$$

where the parameters and cones are defined as

$$\mathbf{P}_1 = \begin{bmatrix} \mathbf{M}_{1(n\times n)} \\ 0_{(1\times n)} \end{bmatrix}, \mathbf{p}_1 = \begin{bmatrix} \mathbf{M}_{1(n\times n)}\bar{\boldsymbol{\xi}}_{1(n\times 1)} \\ -\Delta_1 \end{bmatrix}; \ K_1 = \Big\{ (\boldsymbol{\theta}_{(n\times 1)}; t) \Big| \ \|\boldsymbol{\theta}\|_A \le t \Big\}.$$

$$\mathbf{P}_2 = \begin{bmatrix} \mathbf{M}_{2(n\times n)} \\ 0_{(1\times n)} \end{bmatrix}, \mathbf{p}_2 = \begin{bmatrix} \mathbf{M}_{2(n\times n)}\bar{\boldsymbol{\xi}}_{2(n\times 1)} \\ -\Delta_2 \end{bmatrix}; \ \ K_2 = \Big\{ (\boldsymbol{\theta}_{(n\times 1)}; t) \Big| \ \|\boldsymbol{\theta}\|_B \le t \Big\}.$$

Applying conic duality Ben-Tal et al. (2009, Appendix A), the dual problem is derived as

$$\max_{\mathbf{s}^1,\mathbf{s}^2} \Big\{ \mathbf{p}_1^\top \mathbf{s}^1 + \mathbf{p}_2^\top \mathbf{s}^2 : \ \mathbf{M}_1^\top \boldsymbol{w}_1 + \mathbf{M}_2^\top \boldsymbol{w}_2 = -\boldsymbol{y}^i, \ \mathbf{s}^1 \in K_1^*, \ \mathbf{s}^2 \in K_2^* \Big\}$$

where the dual variables and dual cones are equal to

$$\mathbf{s}_1 = \big(\boldsymbol{w}_{1(n\times 1)}; \tau_1\big) \in K_1^* \text{ where } K_1^* = \Big\{ (\boldsymbol{w}_1; \tau_1) \Big| \ \|\boldsymbol{w}_1\|_A^* \le \tau_1 \Big\}.$$

$$\mathbf{s}_2 = \big(\boldsymbol{w}_{2(n\times 1)}; \tau_2\big) \in K_2^* \text{ where } K_2^* = \Big\{ (\boldsymbol{w}_2; \tau_1) \Big| \ \|\boldsymbol{w}_2\|_B^* \le \tau_2 \Big\}.$$

Substituting for the definition of the dual variables and dual cones, we get

$$\max_{\boldsymbol{w}_1,\boldsymbol{w}_2,\tau_1,\tau_2} \left\{ \begin{array}{l} \bar{\boldsymbol{\xi}}_1^\top \mathbf{M}_1^\top \boldsymbol{w}_1 - \Delta_1\tau_1 + \bar{\boldsymbol{\xi}}_2^\top \mathbf{M}_2^\top \boldsymbol{w}_2 - \Delta_2\tau_2 : \\ \mathbf{M}_1^\top \boldsymbol{w}_1 + \mathbf{M}_2^\top \boldsymbol{w}_2 = -\boldsymbol{y}^i, \\ \|\boldsymbol{w}_1\|_A^* \le \tau_1, \ \|\boldsymbol{w}_2\|_B^* \le \tau_2, \\ \boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}^n \ \tau_1, \tau_2 \in \mathbb{R} \end{array} \right\}$$

The variables τ_1 and τ_2 are substituted for their lower bounds in the objective function

$$\max_{\boldsymbol{w}_1,\boldsymbol{w}_2} \Big\{ \ \bar{\boldsymbol{\xi}}_1^\top \mathbf{M}_1^\top \boldsymbol{w}_1 - \Delta_1\|\boldsymbol{w}_1\|_A^* + \bar{\boldsymbol{\xi}}_2^\top \mathbf{M}_2^\top \boldsymbol{w}_2 - \Delta_2\|\boldsymbol{w}_2\|_B^* : \ \mathbf{M}_1^\top \boldsymbol{w}_1 + \mathbf{M}_2^\top \boldsymbol{w}_2 = -\boldsymbol{y}^i \ \Big\}$$

Next, we replace the variable $\boldsymbol{w}_2 = -\mathbf{M}_2^{-\top}(\boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1)$ in the objective function and remove the equality constraint.

$$\max_{\boldsymbol{w}_1} \Big\{ \bar{\boldsymbol{\xi}}_1^\top \mathbf{M}_1^\top \boldsymbol{w}_1 - \bar{\boldsymbol{\xi}}_2^\top (\boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1) - \Delta_1\|\boldsymbol{w}_1\|_A^* - \Delta_2 \Big\| -\mathbf{M}_2^{-\top}(\boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1) \Big\|_B^* \Big\}$$

We introduce an auxiliary variable $\mathbf{r} = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1$, the problem becomes

$$\max_{\boldsymbol{w}_1,\mathbf{r}} \left\{ \begin{array}{l} -\bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i - (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r} - \Delta_1 \Big\| \mathbf{M}_1^{-\top}(\mathbf{r} - \boldsymbol{y}^i) \Big\|_A^* - |-1|\Delta_2 \Big\| \mathbf{M}_2^{-\top} \mathbf{r} \Big\|_B^* \\ \mathbf{r} = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1 \end{array} \right\}$$

The optimization sense is reformulated to a minimization. The robust counterpart becomes equal to

$$\begin{aligned} y_0^i + \min_{\boldsymbol{w}_1,\mathbf{r}} \quad & \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r} + \Delta_1 \left\| \mathbf{M}_1^{-\top} (\mathbf{r} - \boldsymbol{y}^i) \right\|_A^* + \Delta_2 \left\| \mathbf{M}_2^{-\top} \mathbf{r} \right\|_B^* \leq 0 \\ \text{s.t.} \quad & \mathbf{r} = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}_1 \end{aligned}$$

Lastly, the final form in (5.28) is obtained after dropping the minimization operator and setting $\boldsymbol{w}_1 \to \boldsymbol{w}^i$ and $\mathbf{r} \to \mathbf{r}^i$. □

Example 6.1. Let U be an intersection of ℓ_∞- and D-norm induced uncertainty sets as follows

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \middle| \; \left\| \mathbf{M}_1(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1) \right\|_\infty \leq \Delta_1, \; \left\| \mathbf{M}_2(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_2) \right\|_\Gamma^D \leq \Delta_2 \right\}$$

the robust counterpart of (5.17) for $\boldsymbol{\xi} \in U$ is given as

$$y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \left\| \mathbf{M}_1^{-T} (\mathbf{r}^i - \boldsymbol{y}^i) \right\|_\infty^* + \Delta_2 \left\| \mathbf{M}_2^{-T} \mathbf{r}^i \right\|_\Gamma^{D*} \leq 0, \quad \forall i = 1, \ldots, m$$

which is equivalent to

$$\begin{aligned} y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \left\| \mathbf{M}_1^{-\top} (\mathbf{r}^i - \boldsymbol{y}^i) \right\|_1 & \\ + \Delta_2 \max_{\mathbf{r}^i} \left\{ \frac{1}{\Gamma} \left\| \mathbf{M}_2^{-\top} \mathbf{r}^i \right\|_1, \left\| \mathbf{M}_2^{-\top} \mathbf{r}^i \right\|_\infty \right\} & \leq 0, \quad \forall i = 1, \ldots, m \end{aligned}$$

where $\mathbf{r}^i = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}^i \in \mathbb{R}^n$, and $\boldsymbol{w}^i \in \mathbb{R}^n$ for all i. Using the LP representations of ℓ_1- and ℓ_∞-norms, the robust counterpart LP representation is equal to

$$\left. \begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \mathbf{1}^\top \boldsymbol{v}^i + \Delta_2 z^i \leq 0 \\ z^i \geq u_j^i, \quad j = 1, \ldots, n \\ z^i \geq \frac{1}{\Gamma} \cdot \mathbf{1}^\top \boldsymbol{u}^i \\ -\boldsymbol{u}^i \leq \mathbf{M}_2^{-\top} \mathbf{r}^i \leq \boldsymbol{u}^i \\ -\boldsymbol{v}^i \leq \mathbf{M}_1^{-\top} (\mathbf{r}^i - \boldsymbol{y}^i) \leq \boldsymbol{v}^i \\ \mathbf{r}^i = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}^i \\ \boldsymbol{w}^i, \mathbf{r}^i, \boldsymbol{v}^i, \boldsymbol{u}^i \in \mathbb{R}^n, z^i \in \mathbb{R} \end{array} \right\} \; i = 1, \ldots, m$$

Example 6.2. Let U be an intersection between ℓ_1- and polytope-norm induced uncertainty sets as follows

$$U = \left\{ \boldsymbol{\xi} \in \mathbb{R}^n \middle| \; \left\| \mathbf{M}_1(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1) \right\|_1 \leq \Delta_1, \left\| \mathbf{M}_2(\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_2) \right\|_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{Polytope} \leq \Delta_2 \right\}$$

The robust counterpart of (5.17) for $\xi \in U$ is equal to

$$y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \left\| \mathbf{M}_1^{-\top} (\mathbf{r}^i - \boldsymbol{y}^i) \right\|_1^* - \Delta_2 \left\| \mathbf{M}_2^{-\top} \mathbf{r}^i \right\|_{\boldsymbol{A},\boldsymbol{b},\boldsymbol{\mu}}^{Polytope*} \leq 0, \quad \forall i = 1, \ldots, m$$

which is equivalent to

$$\begin{aligned} y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 \left\| \mathbf{M}_1^{-\top} (\mathbf{r}^i - \boldsymbol{y}^i) \right\|_\infty & \\ - \Delta_2 \min_{\boldsymbol{\lambda}^i \geq 0} \left\{ -\boldsymbol{b}^\top \boldsymbol{\lambda}^i \middle| \ \boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{M}_1^{-\top} \mathbf{r}^i \right\} & \leq 0, \quad \forall i = 1, \ldots, m \end{aligned}$$

where $\mathbf{r}^i = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}^i \in \mathbb{R}^n$ and $\boldsymbol{w}^i \in \mathbb{R}^n$. Using the ℓ_∞-norm LP representation, the robust counterpart LP representation is equal to

$$\left. \begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_2 - \bar{\boldsymbol{\xi}}_1)^\top \mathbf{r}^i + \Delta_1 z^i + \Delta_2 \boldsymbol{b}^\top \boldsymbol{\lambda}^i \leq 0 \\ -z^i \leq (\mathbf{M}_1^{-\top} (\mathbf{r}^i - \boldsymbol{y}^i))_j \leq z^i, \quad j = 1, \ldots, n \\ \boldsymbol{A}^\top \boldsymbol{\lambda}^i = \mathbf{M}_2^{-\top} \mathbf{r}^i \\ \mathbf{r}^i = \boldsymbol{y}^i + \mathbf{M}_1^\top \boldsymbol{w}^i \\ \boldsymbol{\lambda}^i \geq 0 \\ \mathbf{r}^i, \boldsymbol{w}^i \in \mathbb{R}^n, z^i \in \mathbb{R} \end{array} \right\} i = 1, \ldots, m$$

Property 4 *Let the uncertainty set be the intersection of a general norm-induced symmetric set and a general norm-induced asymmetric set as in* (5.29) *where* $\bar{\boldsymbol{\xi}}_1$ *and* $\bar{\boldsymbol{\xi}}_2$ *are the center of the two uncertainty sets, respectively.*

$$U = \left\{ \boldsymbol{\xi}, \boldsymbol{\mu}^+, \boldsymbol{\mu}^- \in \mathbb{R}^n \middle| \begin{array}{l} \left\| \mathbf{M}_1 (\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1) \right\|_A \leq \Delta_1, \\ \boldsymbol{\Phi}^\top (\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_2) = \boldsymbol{\mu}^+ - \boldsymbol{\mu}^-, \\ \| \mathbf{P} \boldsymbol{\mu}^+ + \mathbf{Q} \boldsymbol{\mu}^- \|_B \leq \Delta_2, \\ \boldsymbol{\mu}^+, \boldsymbol{\mu}^- \geq 0 \end{array} \right\} \tag{5.29}$$

The robust counterpart of (5.17) *for* $\boldsymbol{\xi} \in U$ *is equal to*

$$\left. \begin{array}{l} y_0^i + \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_1 - \bar{\boldsymbol{\xi}}_2)^\top \boldsymbol{\Phi} \boldsymbol{z}^i + \Delta_1 \| \mathbf{M}_1^{-\top} (\boldsymbol{\Phi} \boldsymbol{z}^i + \boldsymbol{y}^i) \|_A^* + \Delta_2 \| \boldsymbol{w}^i \|_B^* \leq 0 \\ \mathbf{P}^\top \boldsymbol{w}^i - \boldsymbol{z}^i \leq 0 \\ \mathbf{Q}^\top \boldsymbol{w}^i + \boldsymbol{z}^i \leq 0 \\ \boldsymbol{w}^i, \boldsymbol{z}^i \in \mathbb{R}^n \end{array} \right\} i = 1, \ldots, m \tag{5.30}$$

Proof. We first introduce a max operator, in constraint (5.17), which ensures feasibility for any realization of the uncertainty

$$\begin{array}{rl} \displaystyle\max_{\boldsymbol{\xi}, \boldsymbol{\mu}^+, \boldsymbol{\mu}^-} & \boldsymbol{\xi}^\top \boldsymbol{y}^i \\ \text{s.t} & \left\| \mathbf{M}_1 (\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_1) \right\|_A \leq \Delta_1, \\ & \| \mathbf{P} \boldsymbol{\mu}^+ + \mathbf{Q} \boldsymbol{\mu}^- \|_B \leq \Delta_2 \\ & \boldsymbol{\mu}^+, \boldsymbol{\mu}^- \geq 0 \\ & \boldsymbol{\Phi}^\top \boldsymbol{\xi} - \boldsymbol{\mu}^+ + \boldsymbol{\mu}^- = \boldsymbol{\Phi}^\top \bar{\boldsymbol{\xi}}_2 \\ & \boldsymbol{\xi}, \boldsymbol{\mu}^+, \boldsymbol{\mu}^- \in \mathbb{R}^n \end{array}$$

It is shown in the previous proof that norm is associated with a cone K. We change the sense of the optimization problem to a minimization and reformulate a conic programming problem

$$\begin{array}{lll}\min\limits_{\boldsymbol{\xi},\boldsymbol{\mu}^+,\boldsymbol{\mu}^-} & \mathbf{c}^\top[\boldsymbol{\xi};\boldsymbol{\mu}^+;\boldsymbol{\mu}^-] & \\ \text{s.t} & \mathbf{P}_1\boldsymbol{\xi}-\mathbf{p}_1 & \in K_1 \\ & \mathbf{P}_2[\boldsymbol{\mu}^+;\boldsymbol{\mu}^-]-\mathbf{p}_2 & \in K_2 \\ & \mathbf{P}_3\boldsymbol{\mu}^+ & \in K_3 \\ & \mathbf{P}_4\boldsymbol{\mu}^- & \in K_4 \\ & \boldsymbol{\Phi}^\top\boldsymbol{\xi}-\boldsymbol{\mu}^++\boldsymbol{\mu}^- & =\boldsymbol{\Phi}^\top\bar{\boldsymbol{\xi}}_2 \\ & \boldsymbol{\xi},\boldsymbol{\mu}^+,\boldsymbol{\mu}^-\in\mathbb{R}^n & \end{array}$$

where the cost vector $\mathbf{c}=\begin{bmatrix}-\boldsymbol{y}^i\\ 0_{(n\times1)}\\ 0_{(n\times1)}\end{bmatrix}$ and the remaining parameters and cones are equal to

$$\mathbf{P}_1=\begin{bmatrix}\mathbf{M}_{1(n\times n)}\\ 0_{(1\times n)}\end{bmatrix},\ \mathbf{p}_1=\begin{bmatrix}\mathbf{M}_{1(n\times n)}\bar{\boldsymbol{\xi}}_{1(n\times1)}\\ -\Delta_1\end{bmatrix};\ K_1=\Big\{(\boldsymbol{\theta}_{(n\times1)};t)\ \Big|\ \|\boldsymbol{\theta}\|_A\le t\Big\}.$$

$$\mathbf{P}_2=\begin{bmatrix}\mathbf{P}_{(n\times n)} & \mathbf{Q}_{(n\times n)}\\ 0_{(1\times n)} & 0_{(1\times n)}\end{bmatrix},\ \mathbf{p}_2=\begin{bmatrix}0_{n\times1}\\ -\Delta_2\end{bmatrix};\ K_2=\Big\{(\boldsymbol{\theta}_{(n\times1)};t)\ \Big|\ \|\boldsymbol{\theta}\|_B\le t\Big\}.$$

$$\mathbf{P}_3=\mathbf{I}_{(n\times n)};\ K_3=\big\{\boldsymbol{\theta}_{(n\times1)}\ \big|\ \boldsymbol{\theta}\ge0\big\}\equiv\mathbb{R}^+_n.$$

$$\mathbf{P}_4=\mathbf{I}_{(n\times n)};\ K_4=\big\{\boldsymbol{\theta}_{(n\times1)}\ \big|\ \boldsymbol{\theta}\ge0\big\}\equiv\mathbb{R}^+_n.$$

where $\mathbf{I}_{(n\times n)}$ is the identity matrix. Applying conic duality Ben-Tal et al. (2009, Appendix A), the dual problem is equal to

$$\begin{array}{lll}\max\limits_{\mathbf{s}_1,\mathbf{s}_2,\mathbf{s}_3,\mathbf{s}_4,\boldsymbol{z}} & \mathbf{p}_1^\top\mathbf{s}_1+\mathbf{p}_2^\top\mathbf{s}_2+(\boldsymbol{\Phi}^\top\bar{\boldsymbol{\xi}}_2)^\top\boldsymbol{z} & \\ \text{s.t} & \mathbf{M}_1^\top\boldsymbol{w}_1+\boldsymbol{\Phi}\boldsymbol{z}=-\boldsymbol{y}^i & \\ & \mathbf{P}^\top\boldsymbol{w}_2+\mathbf{P}_3^\top\mathbf{s}_3-\boldsymbol{z}=0 & \\ & \mathbf{Q}^\top\boldsymbol{w}_2+\mathbf{P}_4^\top\mathbf{s}_4+\boldsymbol{z}=0 & \\ & \mathbf{s}_i\in K_i^* & i=1,2,3,4 \\ & \boldsymbol{z}\in\mathbb{R}^n & \end{array}$$

The dual variables are

$$\mathbf{s}_1=\big(\boldsymbol{w}_{1(n\times1)};\tau_1\big)\in K_1^*\text{ where }K_1^*=\Big\{(\boldsymbol{w}_1;\tau_1)\Big|\ \|\boldsymbol{w}_1\|_A^*\le\tau_1\Big\}.$$

$$\mathbf{s}_2=\big(\boldsymbol{w}_{2(n\times1)};\tau_2\big)\in K_2^*\text{ where }K_2^*=\Big\{(\boldsymbol{w}_2;\tau_2)\Big|\ \|\boldsymbol{w}_2\|_B^*\le\tau_2\Big\}.$$

$$\mathbf{s}_3\in K_3^*\text{ where }K_3^*\equiv K_3=\Big\{\boldsymbol{\theta}_{(n\times1)}\ \Big|\ \theta\ge0\Big\}.$$

$$\mathbf{s}_4\in K_4^*\text{ where }K_4^*\equiv K_4=\Big\{\boldsymbol{\theta}_{(n\times1)}\ \Big|\ \theta\ge0\Big\}.$$

$$\boldsymbol{z}\in\mathbb{R}^n$$

Note that a non-negative orthant cone (i.e., K_3/K_4) is a self-dual cone by definition. We substitute the dual variables and cones by their definitions

$$\begin{aligned}
\max_{\substack{\boldsymbol{w}_1,\tau_1,\boldsymbol{w}_2;\\ \tau_2,\mathbf{s}_3,\mathbf{s}_4,\boldsymbol{z}}} \quad & \bar{\boldsymbol{\xi}}_1^\top \mathbf{M}_1^\top \boldsymbol{w}_1 - \Delta_1\tau_1 - \Delta_2\tau_2 + (\boldsymbol{\Phi}^\top \bar{\boldsymbol{\xi}}_2)^\top \boldsymbol{z} && (5.31a)\\
\text{s.t} \quad & \mathbf{M}_1^\top \boldsymbol{w}_1 + \boldsymbol{\Phi}\boldsymbol{z} = -\boldsymbol{y}^i && (5.31b)\\
& \mathbf{P}^\top \boldsymbol{w}_2 + \mathbf{s}_3 - \boldsymbol{z} = 0 && (5.31c)\\
& \mathbf{Q}^\top \boldsymbol{w}_2 + \mathbf{s}_4 + \boldsymbol{z} = 0 && (5.31d)\\
& \|\boldsymbol{w}_1\|_A^* \leq \tau_1 && (5.31e)\\
& \|\boldsymbol{w}_2\|_B^* \leq \tau_2 && (5.31f)\\
& \mathbf{s}_3, \mathbf{s}_4 \in \mathbb{R}_+^n && (5.31g)\\
& \tau_1, \tau_2 \in \mathbb{R} && (5.31h)\\
& \boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{z} \in \mathbb{R}^n && (5.31i)
\end{aligned}$$

The variables τ_1 and τ_2 are substituted in the objective function by their lower bounds in (5.31e) and (5.31f). The constraints (5.31d) and (5.31c) are reformulated knowing that $\mathbf{s}_3$ and $\mathbf{s}_4$ belong to the non-negative orthant.

$$\begin{aligned}
\max_{\boldsymbol{w}_1,\boldsymbol{w}_2,\boldsymbol{z}} \quad & \bar{\boldsymbol{\xi}}_1^\top \mathbf{M}_1^\top \boldsymbol{w}_1 - \Delta_1\|\boldsymbol{w}_1\|_A^* - \Delta_2\|\boldsymbol{w}_2\|_B^* + \bar{\boldsymbol{\xi}}_2^\top \boldsymbol{\Phi}\boldsymbol{z} && (5.32a)\\
\text{s.t} \quad & \mathbf{M}_1^\top \boldsymbol{w}_1 + \boldsymbol{\Phi}\boldsymbol{z} = -\boldsymbol{y}^i && (5.32b)\\
& \mathbf{P}^\top \boldsymbol{w}_2 - \boldsymbol{z} \leq 0 && (5.32c)\\
& \mathbf{Q}^\top \boldsymbol{w}_2 + \boldsymbol{z} \leq 0 && (5.32d)\\
& \boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{z} \in \mathbb{R}^n && (5.32e)
\end{aligned}$$

The variable $\boldsymbol{w}_1$ is replaced in the objective function by $-\mathbf{M}_1^{-\top}(\boldsymbol{\Phi}\boldsymbol{z} + \boldsymbol{y}^i)$ using (5.32b)

$$\begin{aligned}
\max_{\boldsymbol{w}_2,\boldsymbol{z}} \quad & -\bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i - (\bar{\boldsymbol{\xi}}_1 - \bar{\boldsymbol{\xi}}_2)^\top \boldsymbol{\Phi}\boldsymbol{z} - \Delta_1\|-\mathbf{M}_1^{-\top}(\boldsymbol{\Phi}\boldsymbol{z} + \boldsymbol{y}^i)\|_A^* - \Delta_2\|\boldsymbol{w}_2\|_B^*\\
\text{s.t.} \quad & \mathbf{P}^\top \boldsymbol{w}_2 - \boldsymbol{z} \leq 0\\
& \mathbf{Q}^\top \boldsymbol{w}_2 + \boldsymbol{z} \leq 0\\
& \boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{z} \in \mathbb{R}^n
\end{aligned}$$

The sense of the optimization problem is changed to a minimization and the robust linear constraint becomes

$$
\begin{aligned}
y_0^i + \min_{\boldsymbol{w}_2, \boldsymbol{z}} \; & \bar{\boldsymbol{\xi}}_1^\top \boldsymbol{y}^i + (\bar{\boldsymbol{\xi}}_1 - \bar{\boldsymbol{\xi}}_2)^\top \boldsymbol{\Phi} \boldsymbol{z} + \Delta_1 | - 1 | \|\mathbf{M}_1^{-\top} (\boldsymbol{\Phi} \boldsymbol{z} + \boldsymbol{y}^i)\|_A^* + \Delta_2 \|\boldsymbol{w}_2\|_B^* \leq 0 \\
\text{s.t.} \; & \mathbf{P}^\top \boldsymbol{w}_2 - \boldsymbol{z} \leq 0 \\
& \mathbf{Q}^\top \boldsymbol{w}_2 + \boldsymbol{z} \leq 0 \\
& \boldsymbol{w}_2, \boldsymbol{z} \in \mathbb{R}^n
\end{aligned}
$$

Finally, the robust counterpart in (5.30) is obtained after dropping the minimization operator and setting $\boldsymbol{w}_2 \to \boldsymbol{w}^i$ and $\boldsymbol{z} \to \boldsymbol{z}^i$. □

5.7 Case studies

5.7.1 Numerical example

In this section, we consider the following numerical example with decision variables x_1 and x_2, two uncertain parameters ξ_1 and ξ_2 and two uncertain constraints.

$$
\begin{aligned}
\min \quad & 2x_1 + 3x_2 \\
\text{s.t.} \quad & (2 + \xi_1)x_1 + 6x_2 \geq 180 \\
& 3x_1 + (3.4 - \xi_2)6x_2 \geq 162 \\
& x_1 + x_2 \leq 100 \\
& x_1 \geq 0, \; x_2 \geq 0
\end{aligned}
$$

The uncertain constraints are rearranged as $y_0^i + \sum\limits_{k=1}^{2} \xi_k y_k^i \leq 0$ for $i = 1, 2$ with the intercept and slope expressions as follow

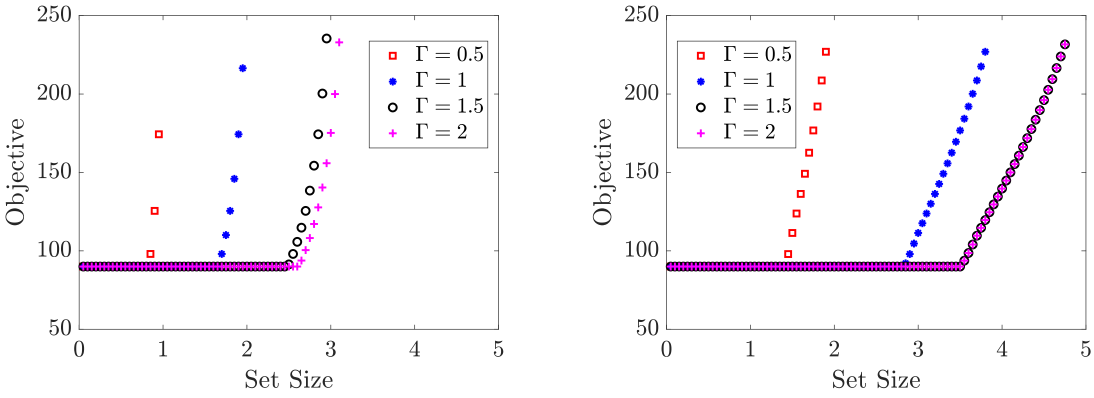
$$
\begin{cases}
y_0^{(1)} = 180 - 2x_1 - 6x_2, & y_1^{(1)} = -x_1, \quad y_2^{(1)} = 0 \\
y_0^{(2)} = 162 - 3x_1 - 20.4x_2, & y_1^{(2)} = 0, \quad\;\;\; y_2^{(2)} = 6x_2
\end{cases}
$$

Assume the first uncertain parameter ξ_1 follows a Gamma-distribution $\Gamma(2, 1)$, the second uncertain parameter ξ_2 follows a t-distribution with 5 degrees of freedom subject to correlation. We use the same data set generated and the same distributional information computed in Figure 5.10.

We model the uncertainty using symmetric and asymmetric *D*- and *Deltoidal*-norm induced uncertainty sets for different set sizes, Γ and λ values, respectively. We do not consider ℓ_1- and ℓ_∞-norms as they are equivalent to specific instances of the *D*- and *Deltoidal*-norms. We also did not consider the *CVaR*-norm induced uncertainty set because there is a one-to-one correspondence between the robust counterparts constructed using *D*- and *CVaR*-norms induced uncertainty sets.

Figure 5.11 illustrates the objective function value using symmetric and asymmetric D-norms induced uncertainty sets. For both types, using a higher value Γ leads to a less conservative solution as it decreases the extent at which the uncertain parameters take their worst-case values simultaneously. The ℓ_1- and ℓ_∞-norm induced uncertainty sets are equal to the D-norm induced uncertainty set with $\Gamma = 2$ and $\Gamma = 1$, respectively. In general, under the same Γ parameter and the uncertainty set size, we observe that the asymmetric uncertainty sets leads to better solution (smaller objective) than the symmetric uncertainty set. The use of the uncertain parameter correlation information in the asymmetric set leads to a better uncertainty set construction.

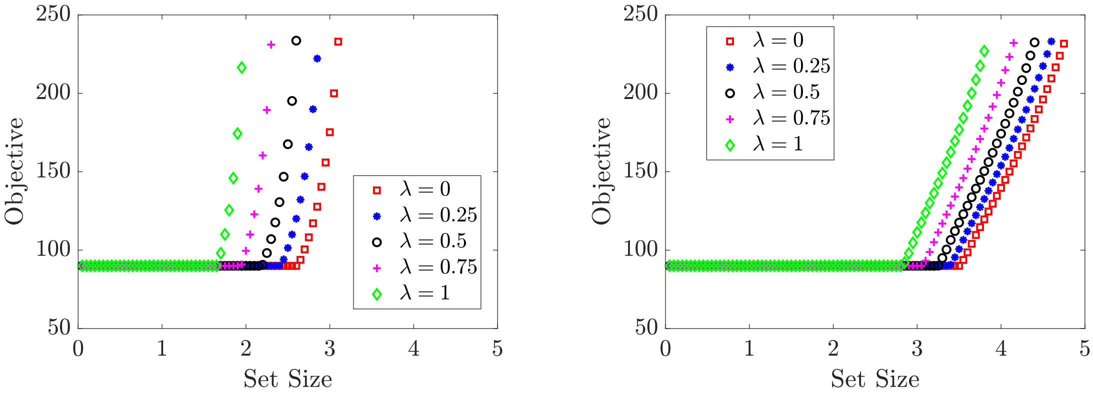
Figure 5.12 depicts the objective function value using symmetric and asymmetric *Deltoidal*-norm induced uncertainty sets. The ℓ_1- and ℓ_∞-norm induced uncertainty sets are equal to the two *Deltoidal*-norm induced uncertainty set extremes at $\lambda = 0$ and $\lambda = 1$, respectively. Similar to Figure 5.11, the asymmetry incorporated in the uncertainty set construction via the correlation information reduces conservatism and improves the robustness of the solution at a fixed λ and set size.



(a) Symmetric sets

(b) Asymmetric sets

Figure 5.11: Numerical example optimal objective value using symmetric and asymmetric D-norm induced uncertainty sets with different Γ values.



(a) Symmetric sets

(b) Asymmetric sets

Figure 5.12: Numerical example optimal objective value using symmetric and asymmetric *Deltoidal*-norm induced uncertainty sets with different λ values.

At the end of this example, we note two contrasts between D- and *Deltoidal*-norms based results. First, a D-norm representation is able to model scaled ℓ_∞-norms for $0 < \Gamma \leq 1$, while a *Deltoidal*-norm representation is not. This is helpful in testing the solution of a problem using different uncertain parameters bounds. Second, the size of a D-norm induced robust counterpart for a single uncertain constraint is linear in the uncertain parameters dimension n, whereas the size of the same robust counterpart under a *Deltoidal*-norm uncertainty set is quadratic in n. Overall, the use of a D-norm for uncertainty set construction is more favourable due to its flexibility and modeling features.

5.7.2 Reactor design problem

In this section, we consider a reaction-separation process shown in Figure 5.13. Material A is fed into a reactor where it reacts to products B and C at an uncertain conversion ratio k. The products B and C are then separated to satisfy product demand D_B and D_C, which are both uncertain also. The nominal value for uncertain parameter $\boldsymbol{\xi} = [k, D_B, D_c]$ is $\bar{\boldsymbol{\xi}} = [0.6, 7, 4]$. Assume the conversion ratio k follows an independent normal distribution $\mathcal{N}(0.6, 0.01)$, products demand D_B and D_C follow a correlated distribution with marginal lognormal distribution $log\mathcal{N}(7, 0.01)$ and $log\mathcal{N}(4, 0.01)$, respectively.

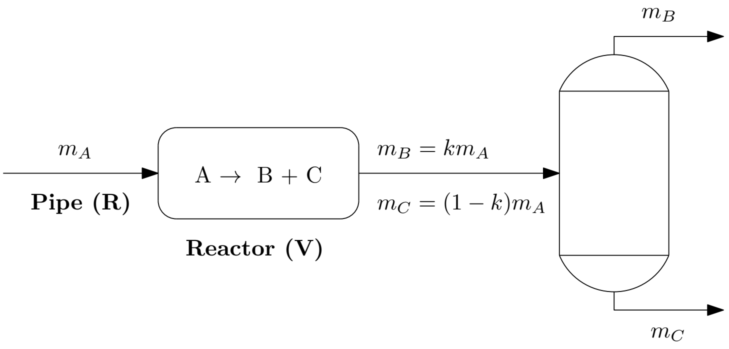


Figure 5.13: Reactor-separator process

The reactor design problem is formulated in equation (5.34) where the first three constraints are enforced for flow conditions and the last two constraints are enforced for product demands. The objective is to minimize the cost.

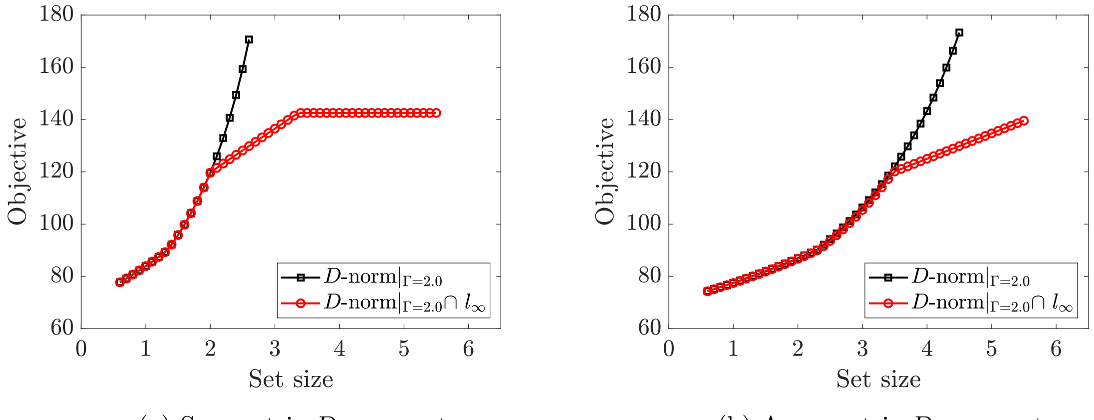
$$
\begin{aligned}
\min_{m_A,V,R>0} \quad & 5V + R \\
\text{s.t.} \quad & 0.2V \leq m_A \leq V \\
& m_A - R \leq 0 \\
& -30 - m_A + 0.8V + 1.2R \leq 0 \\
& -km_A + D_B \leq 0 \\
& -(1-k)m_A + D_C \leq 0
\end{aligned} \tag{5.34}
$$

Knowledge about the uncertain parameter bounds is more often available and can be used to improve the solution quality of the robust optimization problem. To generate this synthetic case study, an unbounded distribution is used to generate the data, while the bounds are used to reflect possible knowledge about the actual limits of the parameters. For example, boundaries can be obtained from actual operations or created from statistical methods. Finally, the uncertainty set is constructed according to both the data and the bounds. This is also a motivation for us to study the intersection type of uncertainty set. In this problem, the bounds incorporated into the problem as a symmetric l_∞-norm induced uncertainty set with $\Delta = 1$ and the nominal values being the midpoint for all uncertain elements. For this example, the uncertain parameters lower and upper bounds are equal to $[0.4, 5.5, 3.25]$ and $[0.8, 8.5, 4.75]$, respectively. The robust counterparts are derived for an intersection between a symmetric/asymmetric D-norm induced set, constructed from distributional information, and a symmetric l_∞-norm induced set constructed from the uncertain bounds information.

The distributional information is computed using 5000 generated scenarios. The covariance matrix is $\mathbf{\Sigma} = [0.0102, 0.0012, 0.0004; 0.0012, 0.4869, 0.1948; 0.0004, 0.1948, 0.1556]$. For the symmetric D-norm induced uncertainty set, we have $\mathbf{M} = \mathbf{\Sigma}^{-0.5} = [9.9171, -0.0218, 0.0028;$ $-0.0218, 1.7724, -0.9860; 0.0028, -0.9860, 3.4496]$. For the asymmetric D-norm induced uncertainty set, the decorrelation matrix is $\mathbf{\Phi} = [0.0102, 0.0012, 0.0004; 0.0012, 0.4869, 0.1948; 0.0004, 0.1948, 0.1556]$, $\mathbf{P}$ and $\mathbf{Q}$ are diagonal matrices with the diagonal entries $(17.0081, 6.8245, 2.1254)$ and $(17.0468, 6.5666, 2.3997)$, respectively.

Figure 5.14 illustrates the solution for $\Gamma = 2$ and set sizes $0.5 \le \Delta \le 5.5$. We also included the solution where the uncertainty belongs to a D-norm induced uncertainty set only. We first observe that incorporating the uncertainty bounds into the solution acts as a safeguard against infeasibility. For symmetric and asymmetric D-norm induced uncertainty, the solution becomes infeasible at a set size value less than 5.5. Moreover, the solution curve corresponding to the intersection of the two sets follows a pattern. First, the solution is the same as that for D-norm induced set only. This region corresponds to the case when the D-norm induced set is a subset of the ℓ_∞-norm induced set as shown in Figures 5.15a and 5.15b. Second, the robust solution starts deviating from that of D-norm induced set only. In this region the conservatism is reduced and the robust solution lies in the intersection of both D- and ℓ_∞-norm sets as illustrated in Figures 5.15c, 5.15d and 5.15f. Third, at a specific set size value, the solution plateaus as shown for the symmetric D-norm induced set case. In this region, the robust solution lies on a vertex of the ℓ_∞-norm induced set which is also an interior point of the D-norm induced set as shown in Figure 5.15e.

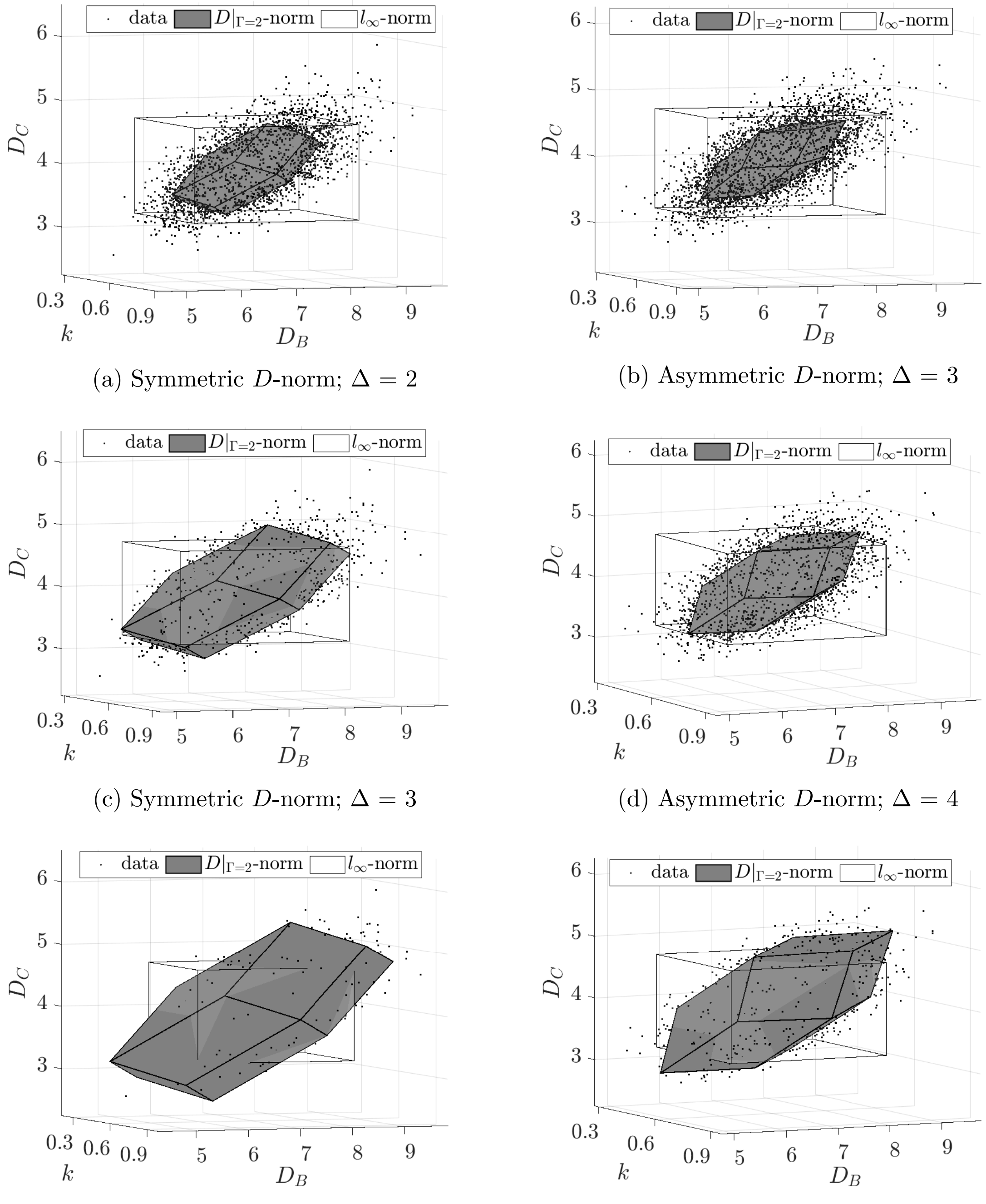
Under the same 5000 data samples, Figures 5.15 show the ℓ_∞-norm induced bounds set and D-norm induced symmetric and asymmetric uncertainty sets using $\Gamma = 2$ and various Δ values. The asymmetric constructed uncertainty sets better fit the uncertain data which in turn leads to better solution quality as previously shown. This observation is reaffirmed in Figure 5.16 where we produce the robust solution quality where the uncertainty belongs to an intersection asymmetric/symmetric D-norm induced uncertainty set and uncertain bounds for various values of Γ.



(a) Symmetric D-norm set

(b) Asymmetric D-norm set

Figure 5.14: Comparison between a D- and an intersection of a D- and an ℓ_∞-norms induced uncertainty sets robust reactor design solution.



(a) Symmetric D-norm; $\Delta = 2$

(b) Asymmetric D-norm; $\Delta = 3$

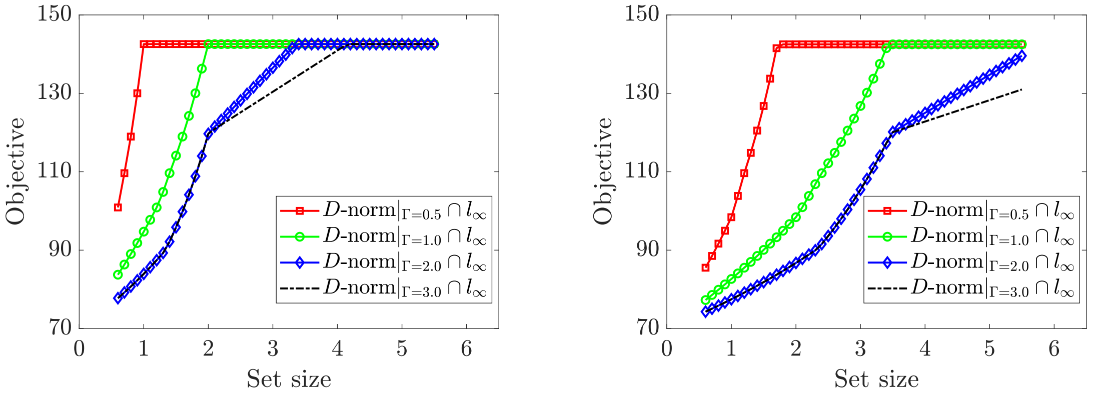
(c) Symmetric D-norm; $\Delta = 3$

(d) Asymmetric D-norm; $\Delta = 4$

(e) Symmetric D-norm; $\Delta = 4$

(f) Asymmetric D-norm; $\Delta = 5$

Figure 5.15: Uncertain data, ℓ_∞- and symmetric/asymmetric D-norm induced uncertainty sets for the reactor design problem at various set size values.



(a) Symmetric D-norm sets

(b) Asymmetric D-norm sets

Figure 5.16: Robust reactor design solution for an intersection of a D- and an l_∞-norms induced uncertainty sets as a function of D-norm set size.

5.8 Conclusions

Uncertainty set construction is a key step in solving robust optimization problems. The quality of the constructed uncertainty set dictates the solution conservatism and robustness. In this work, we propose a novel uncertainty set construction method based on various polyhedral norms. We empirically show the advantage of capturing the asymmetry in the uncertainty set for independent and correlated uncertain parameters. Indeed, asymmetric uncertainty sets are shown to better fit data-driven uncertainty, avoid low-density regions, reduced the solution conservatism. We derive the robust counterpart for a general uncertain linear constraint for various polyhedral norms induced symmetric and asymmetric uncertainty sets. We illustrate the equivalence properties among the different polyhedral norms and point out the favorable modeling features of D-norm induced uncertainty sets.

We also derive the robust counterpart where the uncertain data belongs to (i) an intersection of two general norm symmetric induced sets and (ii) an intersection of general norm symmetric and asymmetric sets. The latter features can be used while integrating distributional information and data into the construction of the data-driven uncertainty sets. Future extensions of this work will include integrating the framework with statistical and data-mining techniques to obtain information required for the construction of the uncertainty sets.

Chapter 6

Decision Rule Methods for Multiparametric Linear Programming

6.1 Introduction

Decision-making under parametric uncertainty is inevitable in the process systems engineering community where values of parameters such as reaction kinetics, heat and mass transfer coefficients and other design configurations exhibit a degree of variability. Multiparametric programming is a community that develops theory and applications for optimization problems with parametric uncertainties.

The theory of solving multiparametric linear programming (mpLP) was first developed by Gal & Nedoma (1972). The solution of mpLPs is obtained by defining a set of critical regions or partitions that cover the uncertain parameters space. In each critical region, optimal decisions and cost value are defined as affine functions of the uncertain parameters. The mpLP solution follows a bottom-up strategy in constructing the critical regions based on three main basis: optimal invariancy property (Gal & Greenberg 2012), support set invariancy property (Hadigheh et al. 2007, Hadigheh & Terlaky 2006) and optimal partition invariancy property (Hadigheh, Terlaky 2006). For convex multiparametric quadratic programming problems (mpQPs), the solution presented in Bemporad, Bozinis, Dua, Morari & Pistikopoulos (2000), Bemporad, Morari, Dua & Pistikopoulos (2000), Pistikopoulos et al. (2000) required a new theoretical framework, namely the basic sensitivity theorem Fiacco (1976). In particular, the work of Bemporad et al. (2002) introduced a geometric approach to construct the optimal critical regions by exploiting the facet-to-facet properties of the polytopic sets. Similar to mpLP, each critical region is a polytope and exhibits optimal decisions that are affine in the uncertain parameters.

A wide variety of decision-making problems in process system engineering include non-linearities in their model. Hence research work in developing multiparametric nonlinear programming (mpNLP) solutions soon followed. On one hand, the solution technique for convex mpNLPS is based on constructing a linear (Acevedo & Salgueiro 2003, Dua & Pistikopoulos 1999) or quadratic (Domínguez & Pistikopoulos 2013) approximations coupled with mpLP solution techniques. In case the maximum error does not meet a prescribed tolerance, the uncertainty space is partitioned and the procedure reiterated. On the other hand, spatial branch and bound (Dua et al. 2004) and decomposition (Fotiou, Rostalski, Parrilo & Morari 2006,?, Fotiou, Rostalski, Sturmfels & Morari 2006) Fotiou et al. (2005) solution methods are developed for non-convex mpNLP.

Multiparametric programming has been widely adopted in explicit model predictive control applications (Grancharova & Johansen 2012, Pistikopoulos et al. 2015). Parametric programming has seen its way to scheduling problems (Li & Ierapetritou 2007) and integrated scheduling and control systems (Baldea & Harjunkoski 2014, Subramanian et al. 2013). The merit of multiparametric programming stems from two main attributes. It substitutes the need for frequent computationally demanding on-line optimization with an algebraic evaluation of optimal control/scheduling decisions obtained off-line and it offers valuable insights regarding the feasible uncertain parameter space which corresponds to the operability boundaries of a system. However, its main limitation is scalability. Multiparametric programming is not efficient for complex problems with a large number of uncertain parameters and decisions. It becomes limited by the exponential growth in the number of critical regions required to cover the entire parameter space. This leads up to exacerbate memory storage and solution querying requirements.

These limitations are not present in decision rule solution-based methods or adaptive robust optimization. The framework was presented bu Ben-Tal et al. (2004) and it addresses optimization problems with parametric uncertainty. The key is to approximate the uncertainty-dependent adaptive decisions with a specific function- linear (Ben-Tal et al. 2004), piecewise-linear (Georghiou et al. 2015), quadratic (Kuhn et al. 2011) or polynomial (Bertsimas et al. 2011)- coupled with the strong duality theorem for convex optimization. Unlike parametric programming, adaptive robust optimization provides an approximate solution and it is not limited by scalability. Tejeda-Iglesias et al. (2019) present a recent work along these lines for explicit model predictive control. Our contributions in this work are:

1. We propose a decision rule solution-based method to approximate the solution of the multiparametric linear programming problem. We highlight its benefits via small, medium and large dimensional instances, in particular, we illustrate the computational ease of the proposed algorithm with scalability.

2. We propose a novel decision rule generated by the complexity provided via rectilinear activation units (ReLUs). The latter instrument has proven to add value to the machine learning community which motivates us to incorporate it in the adaptive robust optimization community. The concept is based on (1) generating new uncertain parameters (i.e., features) using ReLU units on affine images of the original uncertain parameters and (ii)

augmenting these features to a linear decision rule of the original uncertain parameters.

3. We propose a branching algorithm to refine the approximation quality of the optimal ReLU decision rule approximation of multiparametric linear programming problems. We empirically illustrate the attractiveness of the algorithm in terms of solution quality and computational cost.

The remaining of this chapter is divided into five sections. Section 2 introduces the problem statement which is the focus of our work. Section 3 describes the concept of decision rule solution-based methods and rectilinear activation units. Section 4 presents the approximate solution algorithm and the branching scheme followed by the computational experiments in section 5. Section 6 concludes the chapter with final remarks.

6.2 Problem statement

In this work we approximate the solution of a multi-parametric linear programming problem with right-hand-uncertainty given in equation (6.1). The cost coefficient vector $\mathbf{c}$ is assumed to be constant. The left-hand-side matrix is $\mathbf{A} \in \mathbb{R}^{m \times n_x}$, the right-hand-side constant vector is $\mathbf{b} \in \mathbb{R}^m$ and the uncertainty matrix is $\mathbf{F} \in \mathbb{R}^{m \times n_p}$. The uncertainty set Θ is a hyper-rectangle where $\underline{\theta} \leq \theta \leq \bar{\theta}$ and $\theta \in \mathbb{R}^{n_p}$.

$$\min_{\mathbf{x}} \quad \mathbf{c}^\top \mathbf{x} \tag{6.1a}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} + \mathbf{F}\theta \quad \theta \in \Theta \tag{6.1b}$$

$$\mathbf{x} \in \mathbb{R}^{n_x} \tag{6.1c}$$

Multiparametric programming solution method constructs the exact optimal decisions of equation (6.1) over the uncertainty set using the concept of critical regions or, stated differently, partitions. It follows a down-up approach in constructing the critical regions. In each critical region, the optimal decisions $\mathbf{x}$ are affine functions of the uncertain parameters θ. The solution method's main limitation is scalability. Indeed, for a large number of uncertain parameters and variables, it struggles in finding the solution over the entire uncertainty space. Additionally, as the number of critical regions grows substantially, the memory storage and the solution query requirements reduce the attractiveness of the parametric solution method.

On the other hand, decision rule solution based-method provides an approximate optimal solution for equation (6.1). It does not suffer from scalability and addresses high uncertain parameters and variables dimension with more computational ease.

6.3 Decision rule method

In this section, we revisit the definition of a linear decision rule and the use of the strong duality theorem to construct a robust tractable counterpart. Then, we introduce our proposed ReLU decision rule which elevates the approximation quality of linear decision rules by augmenting additional features.

6.3.1 Linear decision rule

Linear decision rule defines a decision variable as an affine function of the uncertain parameters $\mathbf{x}(\theta) = \mathbf{x}^0 + \mathbf{X}^1\theta$ where $\mathbf{x}^0 \in \mathbb{R}^{n_x}$ is the intercept vector and $\mathbf{X}^1 \in \mathbb{R}^{n_x \times n_p}$ is the slope matrix. The presence of the uncertainty in the original model leads to semi-infinite constraints (i.e., constraints that have to be satisfied over a set) which are computationally intractable. Strong duality theorem for linear programming is exploited to circumvent this issue and to derive the linear robust counterpart. For example, consider the i^{th} constraint in equation (6.1)

$$\sum_{j=1}^{n_x} a_{ij}(x_j^0 + \mathbf{X}_j^1\theta) \leq b_i + \mathbf{F}_i\theta, \quad \forall\theta \in \Theta$$

where $\mathbf{X}_j^1$ and $\mathbf{F}_i$ is the i^{th} row of the matrices $\mathbf{X}$ and $\mathbf{F}$, respectively. It is equivalent to

$$-\sum_{j=1}^{n_x} a_{ij}x_j^0 + b_i + \left\{ \begin{array}{ll} \min\limits_{\theta} & \left(\mathbf{F}_i - \sum\limits_{j=1}^{n_p} a_{ij}\mathbf{X}_j^1\right)\theta \\ \text{s.t.} & \mathbf{Q}\theta \geq \mathbf{p} \end{array} \right\} \geq 0 \tag{6.2}$$

The dual of the inner minimization problem is constructed as follows

$$-\sum_{j=1}^{n_x} a_{ij}x_j^0 + b_i + \left\{ \begin{array}{ll} \max\limits_{\lambda} & \mathbf{p}^\top\lambda \\ \text{s.t.} & \mathbf{Q}^\top\lambda = \left(\mathbf{F}_i - \sum\limits_{j=1}^{n_p} a_{ij}\mathbf{X}_j^1\right)^\top \\ & \lambda \geq 0 \end{array} \right\} \geq 0 \tag{6.3}$$

The linear robust counterpart is obtained after dropping the redundant maximum operator. The overall robust counterpart is derived by performing the same procedure for all semi-infinite constraints.

6.3.2 ReLU decision rule

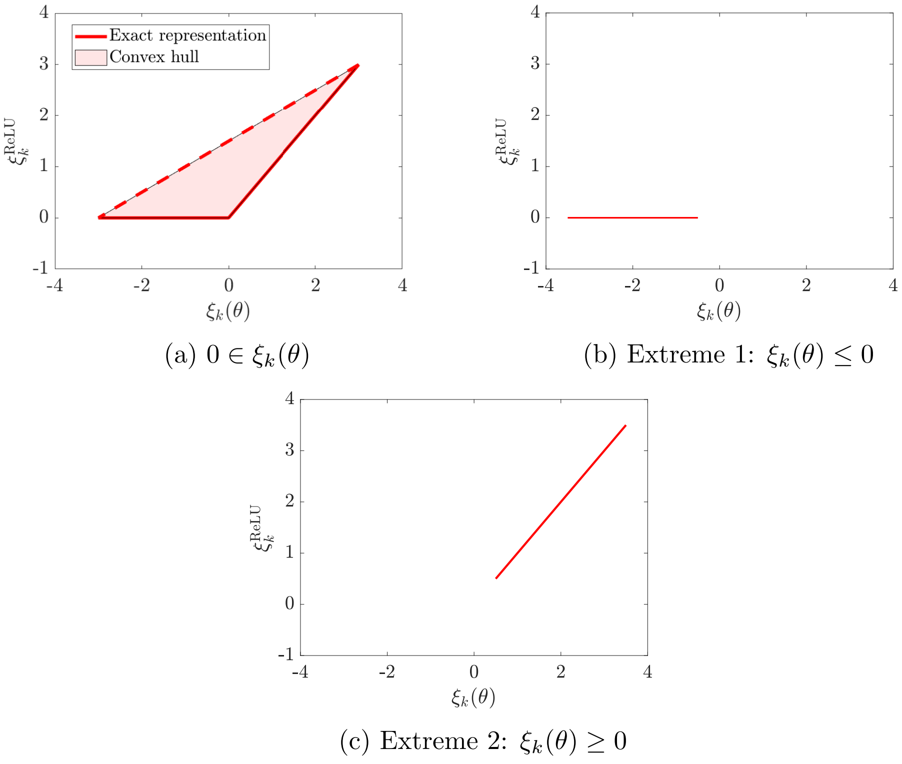
The motivation behind the ReLU decision rule is the wide success found in the machine learning community when using rectilinear activation units in extracting complex features. The new augmented uncertain parameters are obtained by implementing the rectilinear activation unit on the affine mappings of the original uncertain parameters at given ReLU nodes. The image ξ_k of the affine mapping at a ReLU node k is given in equation (6.4) where w_k^0 and $w_{k,i}^1$ are the intercept and slope coefficients, respectively.

$$\xi_k = w_k^0 + \sum_{i'=1}^{n_p} w_{k,i}^1\theta_i, \quad k = 1, \ldots, N \tag{6.4}$$

The definition of a ReLU function at a node k is given in equation (6.5).

$$\xi_k^{\text{ReLU}} = \max\{0, \xi_k\}, \quad k = 1, \dots, N \tag{6.5}$$

The output of a ReLU function has two extreme cases illustrated in Figure 6.1. Extreme case 1 materializes when the maximum of ξ_k is negative and as a consequence ξ_k^{ReLU} is null. In principle, the augmented parameter under this case does not exist. Extreme case 2 materializes when the minimum of ξ_k is positive and hence ξ_k^{ReLU} is a bounded linear segment. In this case, non-linearity is not introduced by the augmented feature and there is no improvement. The norm case which adds value to the ReLU decision rule is when ξ_k^{ReLU} exhibits a two piece-wise linear form with a breakpoint at zero as illustrated in Figure 6.1a.



(a) $0 \in \xi_k(\theta)$

(b) Extreme 1: $\xi_k(\theta) \leq 0$

(c) Extreme 2: $\xi_k(\theta) \geq 0$

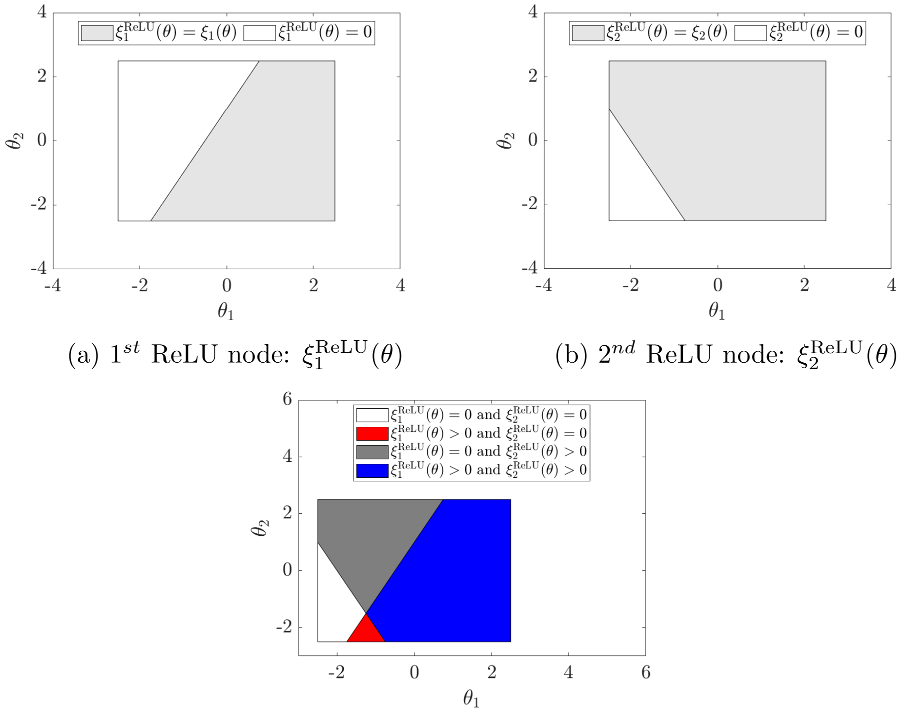
Figure 6.1: Three different outputs of a ReLU node uncertain parameter.

To visualize the added value by the norm ReLU output case, consider an original square uncertainty set $(\theta_1, \theta_2) = [-2.5, 2.5] \times [-2.5, 2.5]$ and the following two ReLU nodes

$$\xi_1(\theta) = 1 + 2\theta_1 - \theta_2 \tag{6.6}$$

$$\xi_2(\theta) = 2 + \theta_1 + 0.5\theta_2 \tag{6.7}$$

Figures 6.2a and 6.2b shows the adjustment introduced to the decision rule policy by the two features ξ_1^{ReLU} and ξ_2^{ReLU}, respectively. Both features partition the uncertainty set between active (i.e., shaded) and not active (i.e., clear) partitions. The superposition of the two features leads to four partitions where in each partition a decision variable may exhibit a unique affine function in θ_1 and θ_2.



(a) 1^{st} ReLU node: $\xi_1^{\text{ReLU}}(\theta)$

(b) 2^{nd} ReLU node: $\xi_2^{\text{ReLU}}(\theta)$

(c) Overall visualization

Figure 6.2: Two augmented ReLU-based uncertain parameters under the norm case.

ReLU-based uncertainty set

The first element of the overall ReLU-based uncertainty set Ψ is given by the affine mapping from θ to ξ as in equation (6.8) where $\mathbf{W} \in \mathbb{R}^{N \times n_p}$ is the weight matrix.

$$\xi = \mathbf{W}\theta \tag{6.8}$$

The second element of Ψ is the definition of the uncertainty at each ReLU node based on the three cases illustrated in Figure 6.1. In all cases, the uncertainty set is modelled as in equation (6.9) where the matrix and right-hand-side vector are defined in Table 6.1..

$$\mathbf{Q}_k^{\text{ReLU}}(\xi_k; \xi_k^{\text{ReLU}}) \geq \mathbf{p}_k^{\text{ReLU}}, \quad k = 1, \ldots, N \tag{6.9}$$

Table 6.1: Uncertainty set $\mathbf{Q}_k^{\text{ReLU}}(\xi_k; \xi_k^{\text{ReLU}}) \geq \mathbf{p}_k^{\text{ReLU}}$ at a ReLU node k where $\xi_k \in [lb_k, ub_k]$.

Case	Condition	Uncertainty set definition
Norm	$0 \in [lb_k, ub_k]$	$\begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 0 & 1 \\ -1 & 1 \\ \frac{ub_k}{ub_k - lb_k} & -1 \end{bmatrix} \begin{bmatrix} \xi_k \\ \xi_k^{\text{ReLU}} \end{bmatrix} \geq \begin{bmatrix} -ub_k \\ lb_k \\ 0 \\ 0 \\ \frac{ub_k lb_k}{ub_k - lb_k} \end{bmatrix}$
Extreme 1	$ub_k \leq 0$	$\begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \xi_k \\ \xi_k^{\text{ReLU}} \end{bmatrix} \geq \begin{bmatrix} -ub_k \\ lb_k \\ 0 \\ 0 \end{bmatrix}$
Extreme 2	$lb_k \geq 0$	$\begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} \xi_k \\ \xi_k^{\text{ReLU}} \end{bmatrix} \geq \begin{bmatrix} -ub_k \\ lb_k \\ 0 \\ 0 \end{bmatrix}$

Let the overall uncertain primitive vector for N ReLU nodes be $\psi = (\theta; \xi_1; \xi_1^{\text{ReLU}}, \ldots, \xi_N; \xi_N^{\text{ReLU}})$, a ReLU decision rule is given in equation (6.10) where $\mathbf{x}^0 \in \mathbb{R}^{n_x}$ is the intercept vector and $\mathbf{X}^1 \in \mathbb{R}^{n_x \times (n_p + 2N)}$ is the slope matrix.

$$\mathbf{x}(\psi) = \mathbf{x}^0 + \mathbf{X}^{1\prime}\psi, \quad \psi \in \mathbf{\Psi} \tag{6.10}$$

The ReLU-based uncertainty set Ψ is given in equation (6.11).

$$\Psi := \left\{ \psi = (\theta; \xi; \xi^{\text{ReLU}}) \;\middle|\; \begin{array}{l} \xi = \mathbf{W}\theta \\ \mathbf{T}\psi \geq \mathbf{q} \end{array} \right\} \tag{6.11}$$

The matrix $\mathbf{T}$ and the right-hand-side vector $\mathbf{q}$ are equal to

$$\mathbf{T} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_1^{\text{ReLU}} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{Q}_N^{\text{ReLU}} \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} \mathbf{p} \\ \mathbf{p}_1^{\text{ReLU}} \\ \vdots \\ \mathbf{p}_N^{\text{ReLU}} \end{bmatrix}$$

Given that the ReLU decision rule is affine in ψ and the overall uncertainty set is a polytope, the ReLU-based robust counterpart is constructed similarly to the linear robust counterpart using the strong duality theorem. The overall ReLU robust counterpart is still a linear programming problem.

6.4 Solution algorithm

6.4.1 Mathematical model

We propose model (6.12) to approximate a multi-parametric linear programming problem. First, the decision vector $\mathbf{x}$ is assumed to be an adaptive decision vector with an affine dependence on the ReLU-based uncertain vector ψ. Second, the objective function is assumed to be a summation of the base-model objective function (i.e., equation (6.1a)) at randomly generated pseudo-samples. The pseudo-samples are assumed to act as "pins" that will orient the approximation of the exact objective function. Third, the right-hand-matrix is reformulated such that $\mathbf{F}'\psi = \mathbf{F}\theta$.

$$\min_{\mathbf{x}^0, \mathbf{X}^{1\prime}} \quad \sum_{s \in \mathcal{S}} \mathbf{c}^\top (\mathbf{x}^0 + \mathbf{X}^{1\prime} \psi_s) \tag{6.12a}$$

$$\text{s.t.} \quad \boldsymbol{A}(\mathbf{x}^0 + \mathbf{X}^{1\prime}\psi) \leq \mathbf{b} + \mathbf{F}'\psi \qquad \forall\, \psi \in \Psi \tag{6.12b}$$

6.4.2 Evaluation criterion

The approximation of the optimal ReLU decision rule is evaluated using a testing sample set different than the pseudo-samples used in the solution of equation (6.12). The criterion used is the root mean squared error (RMSE) between the approximated cost value and the exact cost given in equation (6.13). The exact cost value $z^{\text{exact}}(\theta_i)$ is obtained from solving equation (6.1) with the uncertain parameter fixed at θ_i, while the approximated cost $z^{\text{ReLU}}(\theta_i) = z^{\text{ReLU}}(\psi_i) = \mathbf{c}^\top(\mathbf{x}^0 + \mathbf{X}^{1\prime}\psi_i)$ is computed using the optimal ReLU decision rule and the weight matrix which maps θ to ξ.

$$\text{RMSE} = \sqrt{\frac{\sum_{i \in \mathcal{S}^{\text{test}}} (z^{\text{exact}}(\theta_i) - z^{\text{ReLU}}(\theta_i))^2}{N^{\text{samples}}}} \tag{6.13}$$

6.4.3 Solution algorithm

The solution of equation (6.12) provides an approximation of the parametric solution which then is evaluated via the RMSE criterion. The approximation quality can be improved through the optimization parameters: number of ReLU nodes, weight matrix and the number of random pseudo-samples in the objective function. A higher number of nodes improve the approximation, however this is unfavorable for high dimensional instances. A "good" weight matrix is hard to search for and random searching is not guaranteed to be efficient.

One way to improve the approximation quality of the optimal ReLU decision rule is through branching the original uncertainty space and re-solving equation (6.12) for the child nodes. Two questions come with any branching scheme: How to branch? When to stop branching? Heuristics are introduced to answer these questions using the so-called branching parameters. Algorithm 1 describes the flow of the ReLU decision rule approximation algorithm.

In the proposed branching scheme, a branching number dictates the stopping criterion. The samples per branch number (n^{spb}) corresponds to the random sampling number within a node at each branching step. For each sample, the exact cost value is computed via solving equation (6.1) or via exploiting critical region information from previous deterministic optimization solution (see Appendix D.10). Then, the absolute error at each sample is measured using the optimal ReLU based decision at the current node. The sample with the highest absolute error is identified as the branching sample. The dimension at which we axial branch at the latter sample is defined as the dimension that is closest to the midpoint of its bounds (see Appendix D.2). The logic behind the latter heuristic is to reduce the volume difference between the two created child nodes.

After branching, each child node inherits the same weight matrix as the parent node and the ReLU node number. The samples and their exact cost value are passed to the child nodes accordingly. In the case a child node inherits samples less than the minimum node samples number (n^{int}), the algorithm will add samples to complement this requirement. At this point, and using mixed-integer programming terminology, we are faced with two options: strong branching or weak branching. Strong branching indicates that the two created child nodes have to be both processed sequentially, while weak branching does not mandate that. We adopt the latter where all the nodes (i.e., not only the new two child nodes) are sorted based on the maximum absolute error of the respective interior samples. The top node in the stack is given priority and equation (6.12) is solved for that using a newly generated set of n^{psn} pseudo-samples. Also, note that we do not follow a specific breadth-first or depth-first branching strategy.

At the end of the last branching step, RMSE is evaluated using a testing sample set. The cost value is computed using a parametric solution if a sample θ_i belongs to a critical region constructed in the branching algorithm. Otherwise, the node at which θ_i belongs to is first identified and the approximated cost value is computed using the optimal ReLU decision rule belonging to that node.

Algorithm 1: ReLU decision rule approximation algorithm

```
Data: Linear programming problem with right-hand-uncertainty and a testing sample
      set
Result: ReLU decision rule approximation and the corresponding root mean square
        error
Set the optimization parameters: ReLU nodes number, weight matrix and
  pseudo-samples number ($n^{\text{psn}}$)
Set the branching parameters: Branching number ($n^{\text{bn}}$), samples per branch number
  ($n^{\text{spb}}$) and minimum node samples number ($n^{\text{int}}$)
Initialize an empty list of critical regions ($\mathcal{C}$)
Add the root node to the stack and set it as the parent node
Solve a ReLU decision rule optimization problem at the parent node
for $i$=$1$:$n^{\text{bn}}$ do
    Generate $n^{\text{spb}}$ random samples within the parent node
    for each sample: $\theta_k$ do
        if $\theta_k \in \mathcal{C}$ then
            Compute the exact cost value using the available parametric solution
        else
            Solve a deterministic problem at $\theta_k \rightarrow$ to obtain the exact cost value
            Save the parametric solution
            Add the constructed critical region to the list $\mathcal{C}$
        end
        Compute the approximated cost value using the ReLU decision rule in the parent
          node
    end
    Identify the branching sample which has the maximum absolute approximation error
    Identify the branching dimension which is the closest to the midpoint of its bounds
    Create two child nodes that inherit the parent node weight matrix
    for each child node do
        Set the uncertainty set obtained after branching.
        Identify the inherited interior samples and set their number to $p$
        if  $p < n^{int}$ then
            Generate random samples to reach $n^{int}$
            Compute the exact true cost values using the procedure explained above
        end
    end
    Delete and add the parent and the two child nodes to the stack, respectively
    Sort all nodes in the stack in descending maximum absolute approximation error
    Set the node at the top of the stack as the new parent node
    Solve a ReLU decision rule optimization problem at the parent node
end
Evaluate the ReLU decision rule approximation quality via the root mean squared error
```

6.5 Computational experiments

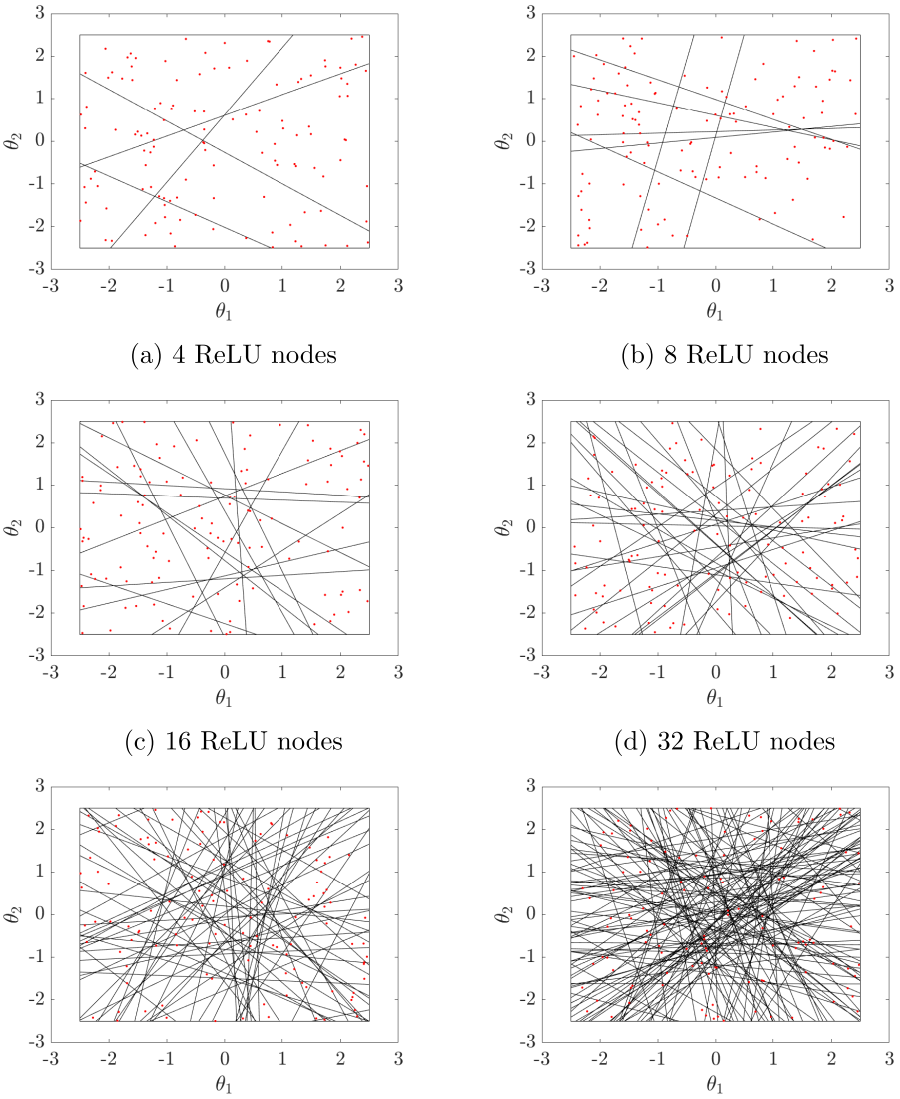
In this section, we first illustrate the different steps of the ReLU decision rule approximation algorithm via a toy example. Then, we highlight the contribution by the proposed method using higher dimension instances. In particular, its ability to generate a feasible approximated policy at a reduced computational cost and or in cases parametric solution struggle to do so. The parametric solution is computed using a MATLAB-based Multi-Parametric Toolbox (i.e., MPT3). The ReLU decision rule approximation is implemented in Julia/JuMP using the extension JumPeR. We used Gurobi 9.0 to solve the ReLU linear robust counterpart.

6.5.1 Illustrative instance

The illustrative instance is a numerical example given in equation (6.14). There are 6 variables, 2 uncertain parameters and 15 constraints and the uncertainty set is a square given by $\Theta := [-2.5, 2.5] \times [-2.5, 2.5]$

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & x_1 + x_2 + x_3 + x_4 && && \text{(6.14a)}\\
\text{s.t.} \quad & -x_1 + x_5 \leq 0 && && \text{(6.14b)}\\
& -x_1 - x_5 \leq 0 && && \text{(6.14c)}\\
& -x_2 + x_6 \leq 0 && && \text{(6.14d)}\\
& -x_2 - x_6 \leq 0 && && \text{(6.14e)}\\
& -x_3 \leq \theta_1 + \theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14f)}\\
& -x_3 - x_5 \leq \theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14g)}\\
& -x_3 \leq -\theta_1 - \theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14h)}\\
& -x_3 + x_5 \leq -\theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14i)}\\
& -x_4 - x_5 \leq \theta_1 + 2\theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14j)}\\
& -x_4 - x_5 - x_6 \leq \theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14k)}\\
& -x_4 + x_5 \leq -\theta_1 - 2\theta_2 && (\theta_1, \theta_2) \in \Theta && \text{(6.14l)}\\
& x_5 \leq 1 && && \text{(6.14m)}\\
& -x_5 \leq 1 && && \text{(6.14n)}\\
& x_6 \leq 1 && && \text{(6.14o)}\\
& -x_6 \leq 1 && && \text{(6.14p)}
\end{aligned}
$$

The parametric solution is computed, by MPT3 toolbox, in 1.2 seconds and reveals 13 critical regions. We solve the ReLU decision rule optimization problem using 2, 4, 8, 16, 32, 64 and 128 ReLU nodes. The increase in the partitioning complexity with the increase in the ReLU node number is shown in Figure 6.3. The partitions are also known as "linear regions" in the machine learning community. The red dots correspond to 121 random pseudo-samples used in the definition of the objective function in equation (6.12a). Each partition has the potential to approximate a decision variable with a unique policy. The optimal "assembly" of the partitions is determined by the optimizer and the red dots acts as "pins" that orient the fitting of those approximated partitions to the exact cost function.



(a) 4 ReLU nodes

(b) 8 ReLU nodes

(c) 16 ReLU nodes

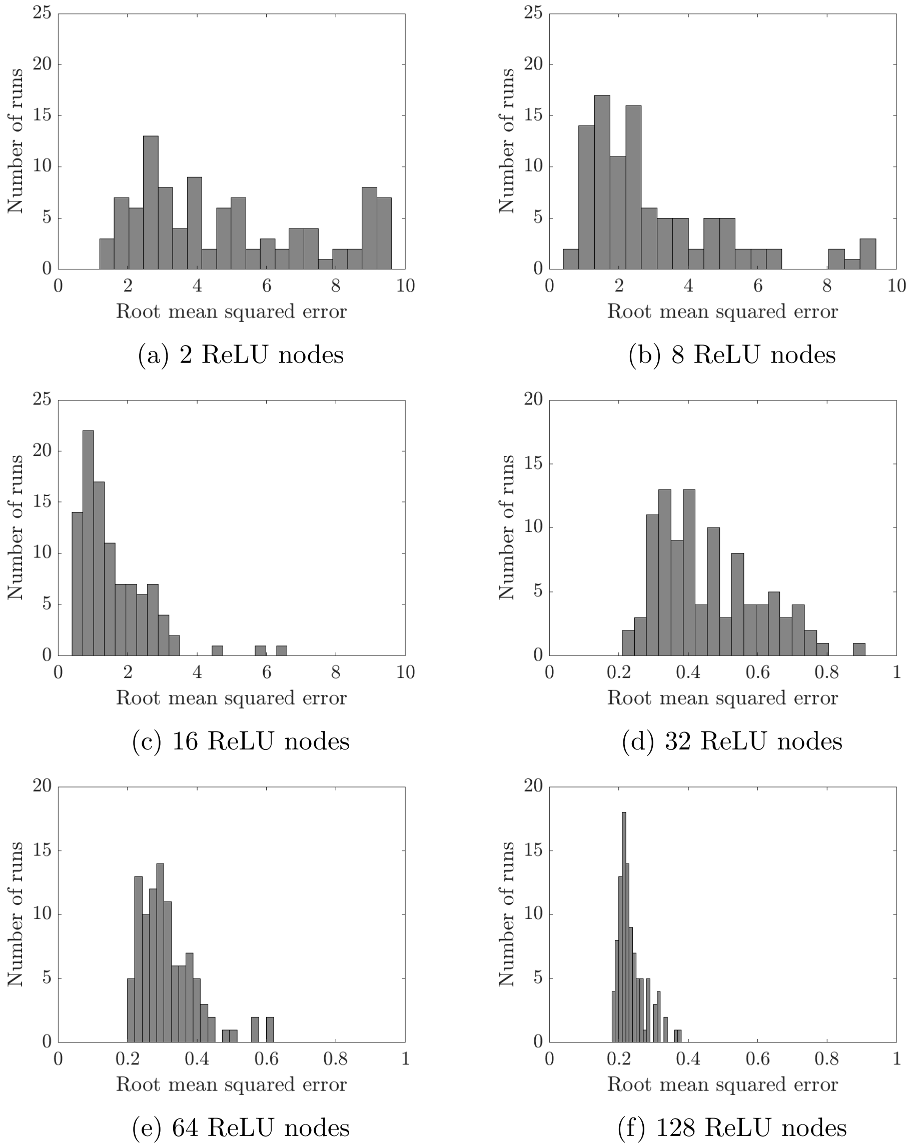
(d) 32 ReLU nodes

(e) 64 ReLU nodes

(f) 128 ReLU nodes

Figure 6.3: Complex uncertainty set partitioning induced by increasing ReLU nodes (red dots represent the pinning pseudo-samples).

The first experiment involves the solution of 100 ReLU decision rule optimization problems with different weight matrices and no branching. The root mean square error histogram for each ReLU node setting (except for 4 ReLU nodes) is shown in Figure 6.4. As expected, the approximation quality increases with the increase in the ReLU nodes number. Further, we show that increasing the ReLU nodes number reduces the histogram dispersion which indicates a reduced impact of the weight matrix random selection on the approximation quality.



(a) 2 ReLU nodes

(b) 8 ReLU nodes

(c) 16 ReLU nodes

(d) 32 ReLU nodes

(e) 64 ReLU nodes

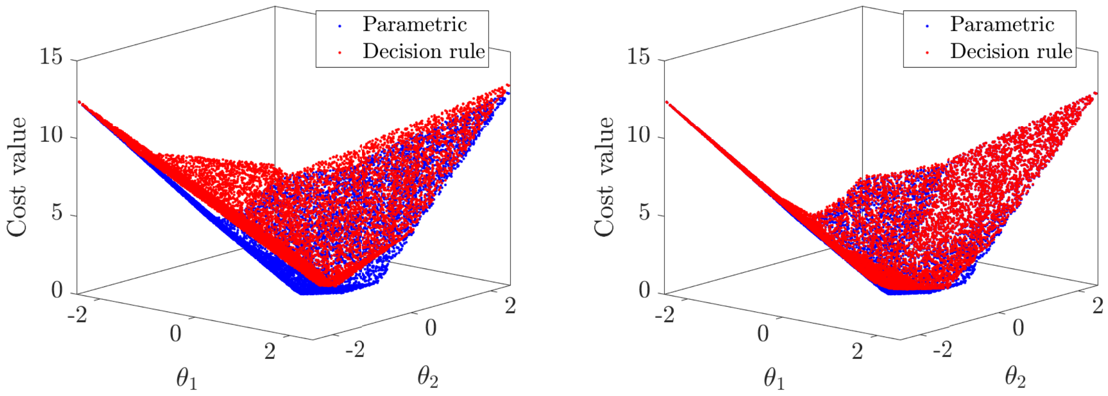
(f) 128 ReLU nodes

Figure 6.4: Root mean square error distribution using 100 different ReLU decision rules and ReLU nodes number.

The best-found solution quality (i.e., lowest RMSE) exhibited by the ReLU decision rule solution at each node setting is shown in Table 6.2. The solution time for the relative optimization problem is shown as well. The model-based cost at each node setting is computed with different randomly generated pseudo-samples. We observe that the rate of decrease in the RMSE value reduces beyond the 32 ReLU nodes setting. We also point out the solution time magnitude difference between the parametric solution and the ReLU-based approximate solution. For example, the computational cost of the 32 ReLU nodes optimization problem is 41 folds less than that of MPT3 (i.e., 0.029 sec vs 1.2 sec). Figure 6.5 illustrates the ReLU decision rule- and the parametric-based cost function over the uncertainty set for the best attained 2- and 32- ReLU nodes weight matrices.

Table 6.2: Optimal model-based cost, best-attained root mean square error from 100 random ReLU decision rules at various ReLU nodes number .

ReLU nodes	2	4	8	16	32	64	128
Optimal cost	643.4359	610.7068	582.1570	605.9716	557.1610	603.0500	599.3509
RMSE	1.3190	0.7782	0.4950	0.3116	0.2174	0.2011	0.1842
Time (sec)	0.0022	0.0030	0.0040	0.0112	0.0290	0.0859	0.3122

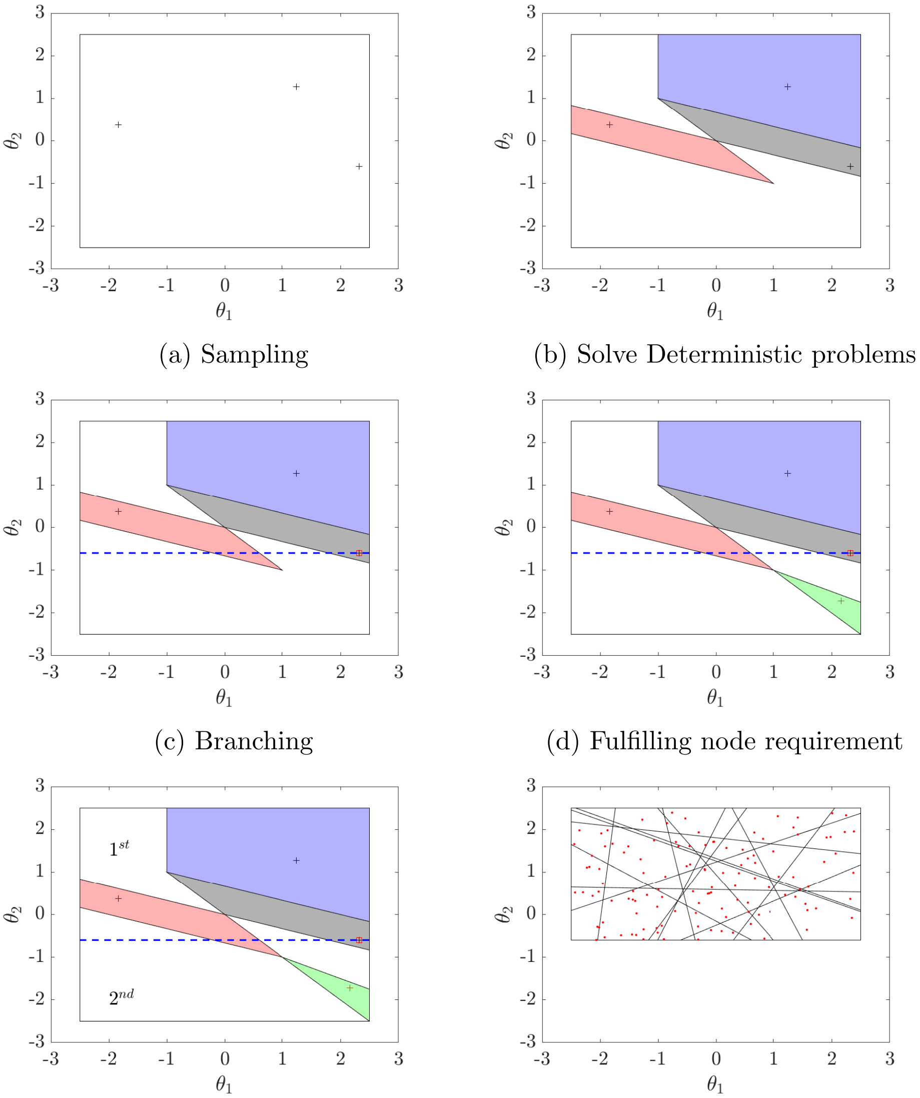


(a) 2 ReLU nodes; RMSE = 1.3190

(b) 32 ReLU nodes; RMSE = 0.2174

Figure 6.5: Parametric and ReLU-based approximate cost functions.

An alternative strategy to refine the ReLU decision rule approximation quality is using branching. Figure 6.6 describes the sequential phases in a single branching step where the parent node is assumed to be the root node with the uncertainty set Θ. We first generate 3 random samples (i.e., $n^{\text{spb}} = 3$) and compute their exact cost value. Then the branching sample is identified based on the maximum approximation error, and the branching dimension is identified as the one closest to the midpoint of its bounds (i.e., θ_2). The lower child node does not include any interior sample, hence we generate 1 (i.e., $n^{\text{int}} = 1$) sample inside that node and we compute its exact cost value. After that, we delete the parent node from the stack and add the two child nodes to it. All nodes in the stack are sorted based on descending sample-based maximum approximation error and the ReLU decision rule optimization is solved for the node at the top of the stack. If the branching number is still not satisfied, the latter node will undergo a similar procedure as described earlier, otherwise the algorithm terminates.



(a) Sampling

(b) Solve Deterministic problems

(c) Branching

(d) Fulfilling node requirement

(e) Sorting the stack

(f) Solving ReLU-based model

Figure 6.6: Steps of a single branching iteration.

In our illustrative branching experiment, we set the ReLU nodes number, the branching number, the samples per branch number and minimum samples in a node to 16, 2, 3 and 1, respectively. We solve 100 ReLU decision rule optimization problems with randomly generated weight matrices. The histogram demonstrating the reduction in the root mean square error to the RMSE at the root node is shown in Figure 6.7. The improvement in the approximation quality comes at the expense of increased computational cost. However, as will be emphasized later, the increase in the computational cost does not outweigh the added value.

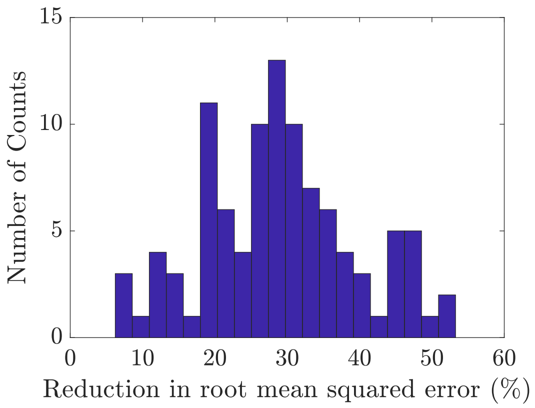


Figure 6.7: Root mean square error reduction, to the root node, of 100 different ReLU decision rules after 2 branching iterations.

6.5.2 Medium dimension instances

The exact and approximated solution of ten medium-size instances is shown in Table 6.3. The exact solution is obtained via MPT3 and is represented by the number of critical regions and solution time. The approximated solution is represented by the root mean square error and solution time of an optimal ReLU decision rule with 32 nodes and no branching. Table 6.3 highlights one of the advantages of the proposed approximation solution technique in its last column. Indeed, the significant computational cost difference between the two solution techniques motivates the implementation of branching to refine the ReLU decision rule approximate solution.

Table 6.3: Parametric and ReLU decision rule solution for 10 medium dimensional linear programming instances.

	Dimensionality			MPT		32-ReLU nodes		
Index	n_x	n_p	n^{const}	Critical regions	Time (sec)	RMSE	Time (sec)	$\frac{\text{MPT Time}}{\text{ReLU Time}}$
1	5	3	70	80	2.9950	2.0163	1.3367	2.2405
2	5	4	70	106	3.9820	1.7085	1.6192	2.4592
3	6	4	72	179	9.4490	4.7444	2.9234	3.2322
4	6	3	72	214	12.1570	2.0200	3.2108	3.7863
5	4	6	68	207	18.5610	5.0947	2.1148	8.7766
6	8	5	76	396	25.4820	6.0336	3.1070	8.2014
7	6	5	72	462	26.2510	10.9588	2.8915	9.0788
8	8	4	76	659	33.4570	5.9712	2.4693	13.5493
9	8	4	76	890	45.1890	6.5420	2.5626	17.6339
10	8	6	76	2074	187.6000	5.2887	6.2941	29.8059

Figure 6.8 illustrates the decrease in RMSE and the cumulative solution time up to five

branching iterations for five different runs of the 10^{10} instance in Table 6.3. Each run signifies a different weight matrix assigned to the root node and inherited to all its child nodes. The ReLU nodes number in the experiment is 32, the number of samples in each branching step is 6, the minimum samples in a node is 2, the randomly generated pseudo-samples in the objective function is 10^3 and the size of the testing samples set is 10^5. The reduction of the RMSE to that in the root node (i.e., branching step 0) is nearly 50% and the solution time is less than 27% the computational cost of the MPT solution. This outcome empirically illustrates the potential of the branching technique to improve the approximate solution at an attractive cost and with less dependence on the randomly selected weight matrices or the complexity introduced by high ReLU nodes number.

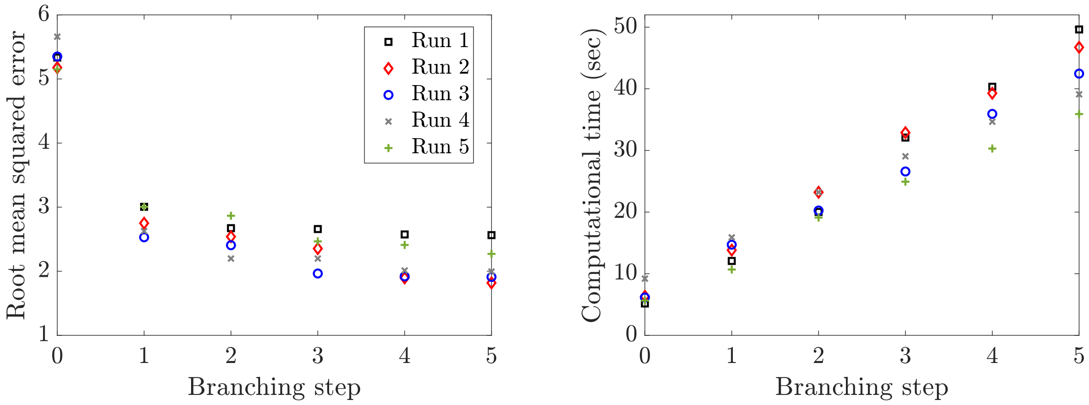


Figure 6.8: Root mean square error and cumulative time for five different ReLU decision rules at 5 branching steps for the 10^{th} medium dimension instance.

6.5.3 Large dimension instance

We now consider an instance with a large number of variables and uncertain parameters where the parametric solution struggles to construct the exact solution over the entire uncertainty space using the down-up approach. For an instance with 8 uncertain parameters, 16 variables and 106 constraints, the MPT algorithm terminated based on the 10000 maximum critical regions criterion in 42 min. Stated differently, a complete exact policy for this instance was not possible in 42 min. Alternatively, the approximation algorithm generated a feasible complete policy using 32 ReLU nodes decision rule in 0.5 min. Though the approximation quality at the root node is not expected to be satisfactory (e.g., RMSE =11.09), the significant difference in computational cost paves the way for the creation of many refining branches. Figure 6.9 illustrates the decrease in RMSE and the cumulative solution time up to five branching steps for five different ReLU decision rules. The number of samples in each branching step is 20, the minimum sample number in a node is 5, the randomly generated pseudo-samples in the objective function is 10^3 and the size of the testing samples set is 10^5.

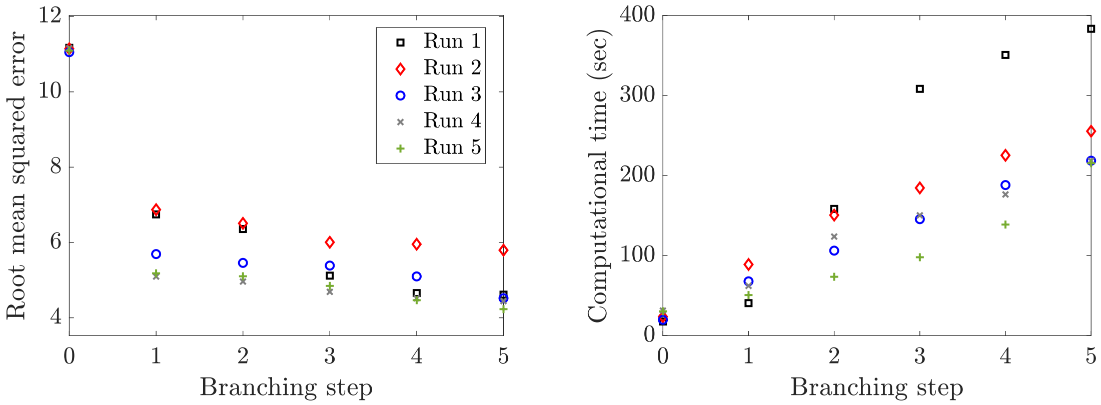


Figure 6.9: Root mean square error and cumulative time profiles for five different ReLU decision rules at 5 branching steps for a large dimension instance.

6.6 Conclusions

We present in this work a decision rule solution-based method to approximate multiparametric linear programming problems. In contrast to parametric programming, the proposed methodology is not limited by scalability and is an attractive candidate for large-scale optimization problems. We also propose a branching scheme to refine the approximated solution. We illustrate that despite the additional computational work required for the branching steps, the proposed algorithm is less computationally expensive than parametric programming. The proposed methodology can be extended to multiparametric quadratic and nonlinear programming problems with minimal modifications.

Chapter 7

Conclusions and future research directions

The thesis contributes to the recent advances of decision rules in addressing optimization under uncertainty. The thesis introduces methodological and algorithmic development to improve decision rules flexibility and solution quality. It also motivates and extends the use of decision rules in the process system engineering and process control communities.

In Chapter 2, we first provide an unexpected result in which a less complex decision rule (e.g., linear decision rule) is *superior* to a more complex decision rule (e.g., piecewise linear decision rule with a single breakpoint) when assessed within a given simulator. This shows that a model may not be reliable in predicting the actual improvement in solution quality by a complex decision rule and it highlights the need for assessing the quality of the look-ahead model policies within a given simulator, instead of just relying on the look-ahead model's objective function value. The latter being commonly overlooked in the literature, unfortunately. Then, we emphasize the concept of implementing hybrid decision rules (HDRs) as a promising direction to mitigate the increased computational burden in high-dimensional multistage adaptive optimization problems. The main aspect of HDRs explored is the lifting strategies or the axial combinations of the LDR and PLDRs where it is empirically illustrated that having higher uncertainty resolution (i.e., more linear pieces) in early stages is more important than having it in late stages.

Chapter 3 introduces reactive and proactive scheduling methods for the steel-making and continuous-casting process under processing time uncertainty via adaptive robust optimization. The scheduling model is represented using the unit-specific event-based continuous-time formulation. We used mathematical tools named "logical sequencing constraints" to infer temporal precedence of the uncertain parameters in different units and to derive linear decision rules that satisfy the non-anticipativity property. To the best of our knowledge, no previous attempt to extend a decision rule method for this specific class of time formulation was made. Hence, the novelty in this chapter. Adaptive robust hybrid scheduling approach is introduced in this chapter as a trade-off between deterministic reactive and robust proactive scheduling filling an existing gap in the literature.

Chapter 4 cross-pollinates ideas from the deep learning community to the two-stage adaptive optimization community. In particular, the concept of extracting complex features/approximations from successive simple operations via a network. We proposed a deep lifting network that devises a flexible piecewise-linear decision rule capable of approximating complex adaptive policies. Further, we developed local-search heuristics inspired by the stochastic-gradient concept also found in deep learning. The heuristics are empirically shown to outperform the traditional gradient-descent method in training the deep lifting network in terms of solution quality and computational cost.

Chapter 5 addresses an assumption made in the methodological works in the previous three chapters. That is the assumption that uncertainty follows a set-based representation. Chapter 5 presents a novel uncertainty set construction method based on various polyhedral norms. In particular, we highlight the benefits of capturing the asymmetry in the uncertainty set for independent and correlated uncertain parameters using the uncertainty distribution information. Asymmetric uncertainty sets are shown to better fit uncertainty distribution, avoid low-density regions, reduce the solution conservatism and improve the solution robustness. We demonstrate the advantage of presenting the uncertainty as an intersection of symmetric and asymmetric uncertainty sets in terms of safeguarding against infeasibility.

Chapter 6 extends the use of decision rules to approximate linear parametric programming solutions. The latter community has extensive application in explicit model predictive control. The motivation behind the attempt is the ability of decision rules to address high-dimensional instances with relatively computational ease; in contrast to, linear parametric programming which is known to be limited by scalability. We developed a branching scheme to refine the parametric approximation and we illustrated the attractiveness of the algorithm for medium and high dimensional instances in terms of computational time and solution quality.

7.1 Future research directions

7.1.1 Optimization under uncertainty and machine learning

The prospects of integrating machine learning techniques with optimization under uncertainty can not be missed, in particular, in an era where data is becoming available more than ever. The idea has already attracted research interests in the process system engineering, process control and operations research communities in the works.

Machine learning techniques such as Bayesian reasoning are used to capture the correlation, asymmetry and multi-mode of the uncertain data which in turn reduces the conservativeness of multistage linear adaptive robust decision problems (Ning & You 2019*c*). Neural networks are used in the prediction of external uncertain factors as in the case of electricity prices (Biel 2019). A task-based end-to-end approach that is rooted in the works of Bengio (1997) and discussed by Elmachtoub & Grigas (2017) in the context of linear programming is an attractive future research direction. In contrary to the common practice, where a probabilistic model is trained for high prediction accuracy. The authors argue that this may not necessarily translate to high solution quality in the operational decision-making problem. The reason being that all models simply do errors. Instead, the authors introduced an approach to evaluate the trade-off between the prediction in the machine learning setting and the solution quality within the decision-making setting. Oroojlooyjadid et al. (2020) implemented a similar approach by integrating the forecasting and inventory-optimization steps in the multi-feature newsvendor problem. In their approach, the uncertain demand, whose probability is unknown is defined through a set of features that are used to train a deep neural network to output the optimal order quantities.

Though the thesis's main focus is on optimization under uncertainty, it borrows some concepts from the deep/machine learning community. It brings attention to the benefits of such cross-pollination and motivates a more extensive integration between the two tasks (i.e., prediction and optimization). An integration. that despite its prospects, is still in its early stages.

7.1.2 Multistage conditional value at risk adaptive optimization

Several open questions are ripe for additional research in the multistage adaptive optimization community. In addition to the points mentioned at the end of Chapter 1, a key endeavor remains which is how to address multistage adaptive optimization using conditional value at risk measure. The flexibility provided by the aforementioned risk measure via the parameter α in quantifying the value of risk is not to be overestimated. It presents a monetary instrument to weigh the risk information value. The benefits of being immune 95% of the time rather than 100% of the time will be quantified which provides better support to choose which set of decisions to take. Still, an obstacle remains in the way of implementing conditional value at risk multistage adaptive optimization and that is time inconsistency. A subject that is valuable in its right to pursue and address through theoretical and methodological frameworks.

7.1.3 Integration of decision rules and parametric programming

In chapter 6, the proposed decision rule algorithm is shown to be efficient in approximating parametric linear solutions. The framework can be naturally extended to approximate quadratic and nonlinear convex optimization (with linear constraints) problems. The performance of the algorithm is not expected to be as computationally smooth, nonetheless, rooms for improvement in the branching heuristics provide a potential for enhancing approximation refinement. Further, an integration between parametric programming and decision rule approximation stands up as a promising direction. As for the former, its main benefit is solution quality (i.e., exact), though it is limited by scalability. For the latter, its advantage is computational ease with scalability, while its less favorable feature is the approximation quality. Integration between the two techniques allows parametric programming to exploit the uncertainty space up to a time limit set by the decision-maker and then apply decision rule approximation for the rest of unexploited space to provide a feasible policy over the entire uncertainty space.

Bibliography

Acevedo, J. & Salgueiro, M. (2003), 'An efficient algorithm for convex multiparametric nonlinear programming problems', *Industrial & Engineering Chemistry Research* **42**(23), 5883–5890.

Ang, M., Lim, Y. F. & Sim, M. (2012), 'Robust storage assignment in unit-load warehouses', *Management Science* **58**(11), 2114–2130.

Apap, R. M. & Grossmann, I. E. (2017), 'Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties', *Computers & Chemical Engineering* **103**, 233–274.

Ariyawansa, K. & Felt, A. J. (2004), 'On a new collection of stochastic linear programming test problems', *INFORMS Journal on Computing* **16**(3), 291–299.

Arslan, A. N. & Papageorgiou, D. J. (2017), 'Bulk ship fleet renewal and deployment under uncertainty: A multi-stage stochastic programming approach', *Transportation Research Part E: Logistics and Transportation Review* **97**, 69–96.

Artzner, P., Delbaen, F., Eber, J.-M. & Heath, D. (1999), 'Coherent measures of risk', *Mathematical finance* **9**(3), 203–228.

Bachouch, A., Huré, C., Langrené, N. & Pham, H. (2018), 'Deep neural networks algorithms for stochastic control problems on finite horizon, part 2: Numerical applications', *arXiv preprint arXiv:1812.05916* .

Bakkehaug, R., Eidem, E. S., Fagerholt, K. & Hvattum, L. M. (2014), 'A stochastic programming formulation for strategic fleet renewal in shipping', *Transportation Research Part E: Logistics and Transportation Review* **72**, 60–76.

Baldea, M. & Harjunkoski, I. (2014), 'Integrated production scheduling and process control: A systematic review', *Computers & Chemical Engineering* **71**, 377–390.

Bampou, D. & Kuhn, D. (2011), Scenario-free stochastic programming with polynomial decision rules, *in* 'Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on', IEEE, pp. 7806–7812.

Baptista, S., Gomes, M. I. & Barbosa-Povoa, A. P. (2012), A two-stage stochastic model for the design and planning of a multi-product closed loop supply chain, *in* 'Computer Aided Chemical Engineering', Vol. 30, Elsevier, pp. 412–416.

Bayat, F., Johansen, T. A. & Jalali, A. A. (2012), 'Flexible piecewise function evaluation methods based on truncated binary search trees and lattice representation in explicit mpc', *- IEEE Transactions on Control Systems Technology* **20**(3), 632–640.

Bemporad, A., Borrelli, F. & Morari, M. (2000), Piecewise linear optimal controllers for hybrid systems, *in* 'Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)', Vol. 2, IEEE, pp. 1190–1194.

Bemporad, A., Bozinis, N. A., Dua, V., Morari, M. & Pistikopoulos, E. N. (2000), 'Model predictive control: A multi-parametric programming approach', *Computer Aided Chemical Engineering* **8**, 301–306.

Bemporad, A., Morari, M., Dua, V. & Pistikopoulos, E. N. (2000), The explicit solution of model predictive control via multiparametric quadratic programming, *in* 'Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)', Vol. 2, IEEE, pp. 872–876.

Bemporad, A., Morari, M., Dua, V. & Pistikopoulos, E. N. (2002), 'The explicit linear quadratic regulator for constrained systems', *Automatica* **38**(1), 3–20.

Ben-Tal, A. & Den Hertog, D. (2011), Immunizing conic quadratic optimization problems against implementation errors, Technical report, Tilburg Univeristy.

Ben-Tal, A., El Ghaoui, L. & Nemirovski, A. (2009), *Robust optimization*, Princeton University Press.

Ben-Tal, A., El Housni, O. & Goyal, V. (2018), 'A tractable approach for designing piecewise affine policies in two-stage adjustable robust optimization', *Mathematical Programming* pp. 1–46.

Ben-Tal, A., Ghaoui, L. E. & Nemirovski, A. (2000), *Robustness*, Handbook of semidefinite programming, Springer, pp. 139–162.

Ben-Tal, A., Golany, B., Nemirovski, A. & Vial, J.-P. (2005), 'Retailer-supplier flexible commitments contracts: A robust optimization approach', *Manufacturing & Service Operations Management* **7**(3), 248–271.

Ben-Tal, A., Goryashko, A., Guslitzer, E. & Nemirovski, A. (2004), 'Adjustable robust solutions of uncertain linear programs', *Mathematical Programming* **99**(2), 351–376.

Ben-Tal, A. & Nemirovski, A. (2000), 'Robust solutions of linear programming problems contaminated with uncertain data', *Mathematical Programming* **88**(3), 411–424.

Ben-Tal, A., Nemirovski, A. & Roos, C. (2002), 'Robust solutions of uncertain quadratic and conic-quadratic problems', *SIAM Journal on Optimization* **13**(2), 535–560.

Bengio, Y. (1997), 'Using a financial training criterion rather than a prediction criterion', *International journal of neural systems* **8**(04), 433–443.

Bertsimas, D. & Georghiou, A. (2015), 'Design of near optimal decision rules in multistage adaptive mixed-integer optimization', *Operations Research* **63**(3), 610–627.

Bertsimas, D., Iancu, D. A. & Parrilo, P. A. (2010), 'Optimality of affine policies in multistage robust optimization', *Mathematics of Operations Research* **35**(2), 363–394.

Bertsimas, D., Iancu, D. A. & Parrilo, P. A. (2011), 'A hierarchy of near-optimal policies for multistage adaptive optimization', *IEEE Transactions on Automatic Control* **56**(12), 2809–2824.

Bertsimas, D., Pachamanova, D. & Sim, M. (2004), 'Robust linear optimization under general norms', *Operations Research Letters* **32**(6), 510–516.

Bertsimas, D. & Sim, M. (2004), 'The price of robustness', *Operations Research* **52**(1), 35–53.

Bertsimas, D. & Thiele, A. (2006), 'Robust and data-driven optimization: Modern decision-making under uncertainty', *INFORMS tutorials in operations research: models, methods, and applications for innovative decision making* **3**.

Bertsimas, D. & Tsitsiklis, J. N. (1997), *Introduction to linear optimization*, Vol. 6, Athena Scientific Belmont, MA.

Beuchat, P. N., Warrington, J., Summers, T. H. & Morari, M. (2016), 'Performance bounds for look-ahead power system dispatch using generalized multistage policies', *IEEE Transactions on Power Systems* **31**(1), 474–484.

Biel, M. (2019), 'Optimal day-ahead orders using stochastic programming and noise-driven rnns', *arXiv preprint arXiv:1910.04510* .

Birge, J. R. & Louveaux, F. (2011), *Introduction to stochastic programming*, Springer Science & Business Media.

Bodur, M. & Luedtke, J. R. (2018), 'Two-stage linear decision rules for multi-stage stochastic programming', *Mathematical Programming* pp. 1–34.

Boyd, S., Boyd, S. P. & Vandenberghe, L. (2004), *Convex optimization*, Cambridge university press.

Boyd, S., Ghaoui, L. E., Feron, E. & Balakrishnan, V. (1994), *Linear matrix inequalities in system and control theory*, SIAM.

Braathen, J. & Eriksrud, A. L. (2013), Hydropower bidding using linear decision rules, Master's thesis, Institutt for industriell økonomi og teknologiledelse.

Bruni, M. E., Beraldi, P. & Conforti, D. (2015), 'A stochastic programming approach for operating theatre scheduling under uncertainty', *IMA Journal of Management Mathematics* **26**(1), 99–119.

Chaari, T., Chaabane, S., Aissani, N. & Trentesaux, D. (2014), Scheduling under uncertainty: Survey and research directions, *in* '2014 International conference on advanced logistics and transport (ICALT)', IEEE, pp. 229–234.

Chen, X., Sim, M. & Sun, P. (2007), 'A robust optimization perspective on stochastic programming', *Operations Research* **55**(6), 1058–1071.

Chen, X., Sim, M., Sun, P. & Zhang, J. (2008), 'A linear decision-based approximation approach to stochastic programming', *Operations Research* **56**(2), 344–357.

Chen, X. & Zhang, Y. (2009), 'Uncertain linear programs: Extended affinely adjustable robust counterparts', *Operations Research* **57**(6), 1469–1482.

Chen, Z., Sim, M. & Xiong, P. (2020), 'Robust stochastic optimization made easy with rsome', *Management Science* .

Dai, X., Wang, X., He, R., Du, W., Zhong, W., Zhao, L. & Qian, F. (2020), 'Data-driven robust optimization for crude oil blending under uncertainty', *Computers & Chemical Engineering* **136**, 106595.

Daryalal, M., Bodur, M. & Luedtke, J. R. (2020), 'Lagrangian dual decision rules for multistage stochastic mixed integer programming', *arXiv preprint arXiv:2001.00761* .

de Ruiter, F. J., Brekelmans, R. C. & den Hertog, D. (2016), 'The impact of the existence of multiple adjustable robust solutions', *Mathematical Programming* **160**(1-2), 531–545.

Dillon, M., Oliveira, F. & Abbasi, B. (2017), 'A two-stage stochastic programming model for inventory management in the blood supply chain', *International Journal of Production Economics* **187**(Supplement C), 27–41. ID: 271692.

Domínguez, L. F. & Pistikopoulos, E. N. (2013), 'A quadratic approximation-based algorithm for the solution of multiparametric mixed-integer nonlinear programming problems', *AIChE Journal* **59**(2), 483–495.

Donti, P., Amos, B. & Kolter, J. Z. (2017), Task-based end-to-end model learning in stochastic optimization, *in* 'Advances in Neural Information Processing Systems', pp. 5484–5494.

Dorigo, M., Maniezzo, V. & Colorni, A. (1996), 'Ant system: optimization by a colony of cooperating agents', *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **26**(1), 29–41.

Dua, V., Papalexandri, K. P. & Pistikopoulos, E. N. (2004), 'Global optimization issues in multiparametric continuous and mixed-integer optimization problems', *Journal of Global Optimization* **30**(1), 59–89.

Dua, V. & Pistikopoulos, E. N. (1999), 'Algorithms for the solution of multiparametric mixed-integer nonlinear optimization problems', *Industrial & Engineering Chemistry Research* **38**(10), 3976–3987.

Dunning, I., Huchette, J. & Lubin, M. (2017), 'Jump: A modeling language for mathematical optimization', *SIAM Review* **59**(2), 295–320.

Dunning, I. R. (2016), Advances in robust and adaptive optimization: algorithms, software, and insights, PhD thesis, Massachusetts Institute of Technology.

Elmachtoub, A. N. & Grigas, P. (2017), 'Smart" predict, then optimize"', *arXiv preprint arXiv:1710.08005* .

Fiacco, A. V. (1976), 'Sensitivity analysis for nonlinear programming using penalty methods', *Mathematical Programming* **10**(1), 287–311.

Fisher, M., Ramdas, K. & Zheng, Y.-S. (2001), 'Ending inventory valuation in multiperiod production scheduling', *Management Science* **47**(5), 679–692.

Fotiou, I. A., Beccuti, A. G., Papafotiou, G. & Morari, M. (2006), Optimal control of piece-wise polynomial hybrid systems using cylindrical algebraic decomposition, *in* 'International Workshop on Hybrid Systems: Computation and Control', Springer, pp. 227–241.

Fotiou, I. A., Parrilo, P. A. & Morari, M. (2005), Nonlinear parametric optimization using cylindrical algebraic decomposition, *in* '- Proceedings of the 44th IEEE Conference on Decision and Control', pp. 3735–3740.

Fotiou, I. A., Rostalski, P., Parrilo, P. A. & Morari, M. (2006), 'Parametric optimization and optimal control using algebraic geometry methods', *null* **79**(11), 1340–1358.

Fotiou, I. A., Rostalski, P., Sturmfels, B. & Morari, M. (2006), An algebraic geometry approach to nonlinear parametric optimization in control, *in* '2006 American Control Conference', IEEE, p. 6 pp.

Gal, T. & Greenberg, H. J. (2012), *Advances in sensitivity analysis and parametric programming*, Vol. 6, Springer Science & Business Media.

Gal, T. & Nedoma, J. (1972), 'Multiparametric linear programming', *Management Science* **18**(7), 406–422.

GAMS Development Corporation (2018), 'General algebraic modeling system (gams)'. Release 25.0.3, Fairfax, VA, USA.

Gao, X., Shang, C., Jiang, Y., Huang, D. & Chen, T. (2014), 'Refinery scheduling with varying crude: A deep belief network classification and multimodel approach', *AIChE Journal* **60**(7), 2525–2532.

Garstka, S. J. & Wets, R. J.-B. (1974), 'On decision rules in stochastic programming', *Mathematical Programming* **7**(1), 117–143.

Garuba, F., Goerigk, M. & Jacko, P. (2020), 'A comparison of data-driven uncertainty sets for robust network design', *arXiv preprint arXiv:2003.10507* .

Gauvin, C., Delage, E. & Gendreau, M. (2017), 'Decision rule approximations for the risk averse reservoir management problem', *European Journal of Operational Research* **261**(1), 317 – 336.

Gawehn, E., Hiss, J. A. & Schneider, G. (2016), 'Deep learning in drug discovery', *Molecular informatics* **35**(1), 3–14.

Georghiou, A., Kuhn, D. & Wiesemann, W. (2019), 'The decision rule approach to optimization under uncertainty: methodology and applications', *Computational Management Science* **16**(4), 545–576.

Georghiou, A., Wiesemann, W. & Kuhn, D. (2015), 'Generalized decision rule approximations for stochastic programming via liftings', *Mathematical Programming* **152**(1-2), 301–338.

Gerardi, D., Marlin, T. E. & Swartz, C. L. E. (2013), 'Optimization of primary steelmaking purchasing and operation under raw material uncertainty', *Industrial & Engineering Chemistry Research* **52**(35), 12383–12398.

Goel, V. & Grossmann, I. E. (2004), 'A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves', *Computers and Chemical Engineering* **28**(8), 1409–1429.

Goel, V. & Grossmann, I. E. (2006), 'A class of stochastic programs with decision dependent uncertainty', *Mathematical Programming* **108**(2), 355–394.

Goh, J. & Sim, M. (2010), 'Distributionally robust optimization and its tractable approximations', *Operations research* **58**, 902–917.

Goh, J. & Sim, M. (2011), 'Robust optimization made easy with rome', *Operations Research* **59**(4), 973–985.

Gotoh, J. & Uryasev, S. (2016), 'Two pairs of families of polyhedral norms versus ℓ_p-norms: proximity and applications in optimization', *Mathematical Programming* **156**(1-2), 391–431.

Grancharova, A. & Johansen, T. A. (2012), *Explicit nonlinear model predictive control: Theory and applications*, Vol. 429, Springer Science & Business Media.

Grass, E. & Fischer, K. (2016), 'Two-stage stochastic programming in disaster management: A literature survey', *Surveys in Operations Research and Management Science* **21**(2), 85–100. ID: 280252.

Grossmann, I. E., Apap, R. M., Calfa, B. A., García-Herreros, P. & Zhang, Q. (2016), 'Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty', *Computers & Chemical Engineering* **91**, 3–14.

Guigues, V., Juditski, A. & Nemirovski, A. (2020), 'Constant depth decision rules for multistage optimization under uncertainty', *arXiv preprint arXiv:2005.13387* .

Gupta, A., Bhartiya, S. & Nataraj, P. (2011), 'A novel approach to multiparametric quadratic programming', *Automatica* **47**(9), 2112–2117.

Gupta, D. & Maravelias, C. T. (2016), 'On deterministic online scheduling: Major considerations, paradoxes and remedies', *Computers & Chemical Engineering* **94**, 312–330.

Gupta, D., Maravelias, C. T. & Wassick, J. M. (2016), 'From rescheduling to online scheduling', *Chemical Engineering Research and Design* **116**, 83–97.

Gupta, V. & Grossmann, I. E. (2011), 'Solution strategies for multistage stochastic programming with endogenous uncertainties', *Computers and Chemical Engineering* **35**(11), 2235–2247.

Guzman, Y. A., Matthews, L. R. & Floudas, C. A. (2016), 'New a priori and a posteriori probabilistic bounds for robust counterpart optimization: I. unknown probability distributions', *Computers & Chemical Engineering* **84**, 568–598.

Guzman, Y. A., Matthews, L. R. & Floudas, C. A. (2017*a*), 'New a priori and a posteriori probabilistic bounds for robust counterpart optimization: II. a priori bounds for known symmetric and asymmetric probability distributions', *Computers & Chemical Engineering* **101**, 279–311.

Guzman, Y. A., Matthews, L. R. & Floudas, C. A. (2017*b*), 'New a priori and a posteriori probabilistic bounds for robust counterpart optimization: III. exact and near-exact a posteriori expressions for known probability distributions', *Computers & Chemical Engineering* **103**, 116–143.

Hadigheh, A. G., Mirnia, K. & Terlaky, T. (2007), 'Active constraint set invariancy sensitivity analysis in linear optimization', *Journal of Optimization Theory and Applications* **133**(3), 303–315.

Hadigheh, A. G. & Terlaky, T. (2006), 'Generalized support set invariancy sensitivity analysis in linear optimization', *Journal of Industrial & Management Optimization* **2**(1), 1.

Han, J. (2016), 'Deep learning approximation for stochastic control problems', *arXiv preprint arXiv:1611.07422* .

Hanasusanto, G. A., Kuhn, D. & Wiesemann, W. (2015), 'K-adaptability in two-stage robust binary programming', *Operations Research* **63**(4), 877–891.

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G. & Wassick, J. (2014), 'Scope for industrial applications of production scheduling models and solution methods', *Computers and Chemical Engineering* **62**, 161–193.

Holland, J. H. et al. (1992), *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT press.

Hong, Y. & Wang, X. (2014), Robust operation optimization in cold rolling production process, *in* 'The 26th Chinese Control and Decision Conference (2014 CCDC)', IEEE, pp. 1365–1370.

Huré, C., Pham, H., Bachouch, A. & Langrené, N. (2018), 'Deep neural networks algorithms for stochastic control problems on finite horizon, part i: convergence analysis', *arXiv preprint arXiv:1812.04300* .

Iancu, D. A. & Trichakis, N. (2013), 'Pareto efficiency in robust optimization', *Management Science* **60**(1), 130–147.

Iglesias Escudero, M., Villanueva Balsera, J., Ortega Fernandez, F. & Rodriguez Montequín, V. (2019), 'Planning and scheduling with uncertainty in the steel sector: A review', *Applied Sciences* **9**(13).

Janak, S. L. & Floudas, C. A. (2008), 'Improving unit-specific event based continuous-time approaches for batch processes: Integrality gap and task splitting', *Computers & Chemical Engineering* **32**(4), 913 – 955.

Jiang, S. L., Liu, M., Lin, J. H. & Zhong, H. X. (2016), 'A prediction-based online soft scheduling algorithm for the real-world steelmaking-continuous casting production', *Knowledge-Based Systems* **111**, 159–172.

Jiang, S., Liu, M. & Hao, J. (2017), 'A two-phase soft optimization method for the uncertain scheduling problem in the steelmaking industry', *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **47**(3), 416–431.

Jin, J. & Xu, Y. (2018), Segregated linear decision rules for distributionally robust control with linear dynamics and quadratic cost, *in* '2018 Annual American Control Conference (ACC)', IEEE, pp. 2687–2694.

Johansen, T. A. (2002), On multi-parametric nonlinear programming and explicit nonlinear model predictive control, *in* 'Proceedings of the 41st IEEE Conference on Decision and Control, 2002.', Vol. 3, IEEE, pp. 2768–2773.

Johansen, T. A. (2003), 'Reduced explicit constrained linear quadratic regulators', *IEEE transactions on automatic control* **48**(5), 823–829.

Johnson, S. G. (2020), *The NLopt nonlinear-optimization package*, http://github.com/stevengj/nlopt.

Jonsbraten, T. W., Wets, R. J. & Woodruff, D. L. (1998), 'A class of stochastic programs with decision dependent random elements', *Annals of Operations Research* **82**, 83–106.

Kammammettu, S. & Li, Z. (2018), 'Multistage adaptive optimization for steelmaking and continuous casting scheduling under processing time uncertainty', *IFAC-PapersOnLine* **51**(21), 262–267.

Kennedy, J. & Eberhart, R. (1995), Particle swarm optimization, *in* 'Proceedings of ICNN'95-International Conference on Neural Networks', Vol. 4, IEEE, pp. 1942–1948.

Krokhmal, P., Palmquist, J. & Uryasev, S. (2002), 'Portfolio optimization with conditional value-at-risk objective and constraints', *Journal of risk* **4**, 43–68.

Kuhn, D., Wiesemann, W. & Georghiou, A. (2011), 'Primal and dual linear decision rules in stochastic and robust optimization', *Mathematical Programming* **130**(1), 177–209.

Kumar Roy, S., Mhammedi, Z. & Harandi, M. (2018), Geometry aware constrained optimization techniques for deep learning, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 4460–4469.

Kusunoki, Y. & Tatsumi, K. (2019), Scalarization for approximate multiobjective multiclass support vector machine using the large-k norm, *in* '2019 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (EUSFLAT 2019)', Atlantis Press.

Lappas, N. H. & Gounaris, C. E. (2016), 'Multi-stage adjustable robust optimization for process scheduling under uncertainty', *AIChE Journal* **62**(5), 1646–1667.

Lappas, N. H. & Gounaris, C. E. (2018), 'Robust optimization for decision-making under endogenous uncertainty', *Computers and Chemical Engineering* **111**, 252–266.

LeCun, Y., Bengio, Y. & Hinton, G. (2015), 'Deep learning', *nature* **521**(7553), 436–444.

Lee, J. H. & Lee, J. M. (2006), 'Approximate dynamic programming based approach to process control and scheduling', *Computers & chemical engineering* **30**(10-12), 1603–1618.

Lee, J. H., Shin, J. & Realff, M. J. (2018), 'Machine learning: Overview of the recent progresses and implications for the process systems engineering field', *Computers & Chemical Engineering* **114**, 111–121.

Lee, J. H. & Wong, W. (2010), 'Approximate dynamic programming approach for process control', *Journal of Process Control* **20**(9), 1038–1048.

Li, C., Zhao, J., Zheng, T. & Litvinov, E. (2016), Data-driven uncertainty sets: Robust optimization with temporally and spatially correlated data, *in* '2016 IEEE Power and Energy Society General Meeting (PESGM)', IEEE, pp. 1–5.

Li, J., Misener, R. & Floudas, C. A. (2012), 'Scheduling of crude oil operations under demand uncertainty: A robust optimization framework coupled with global optimization', *AIChE Journal* **58**(8), 2373–2396.

Li, J., Xiao, X., Tang, Q. & Floudas, C. A. (2012), 'Production scheduling of a large-scale steelmaking continuous casting process via unit-specific event-based continuous-time models: Short-term and medium-term scheduling', *Industrial & Engineering Chemistry Research* **51**(21), 7300–7319.

Li, Z., Ding, R. & Floudas, C. A. (2011), 'A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization', *Industrial & Engineering Chemistry Research* **50**(18), 10567–10603.

Li, Z. & Ierapetritou, M. (2008), 'Process scheduling under uncertainty: Review and challenges', *Computers and Chemical Engineering* **32**(4-5), 715–727.

Li, Z. & Ierapetritou, M. G. (2007), 'Process scheduling under uncertainty using multiparametric programming', *AIChE Journal* **53**(12), 3183–3203.

Li, Z., Tang, Q. & Floudas, C. A. (2012), 'A comparative theoretical and computational study on robust counterpart optimization: Ii. probabilistic guarantees on constraint satisfaction', *Industrial & Engineering Chemistry Research* **51**(19), 6769–6788.

Liepins, G. E. & Hilliard, M. (1989), 'Genetic algorithms: foundations and applications', *Annals of operations research* **21**(1), 31–57.

Linderoth, J., Shapiro, A. & Wright, S. (2006), 'The empirical behavior of sampling methods for stochastic programming', *Annals of Operations Research* **142**(1), 215–241.

Liu, D. C. & Nocedal, J. (1989), 'On the limited memory bfgs method for large scale optimization', *Mathematical programming* **45**(1-3), 503–528.

Lorca, A., Sun, X. A., Litvinov, E. & Zheng, T. (2016), 'Multistage adaptive robust optimization for the unit commitment problem', *Operations Research* **64**(1), 32–51.

Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J. & Tolson, B. A. (2019), 'Introductory overview: Optimization using evolutionary algorithms and other metaheuristics', *Environmental modelling & software* **114**, 195–213.

Matthews, L. R., Guzman, Y. A. & Floudas, C. A. (2018), 'Generalized robust counterparts for constraints with bounded and unbounded uncertain parameters', *Computers & Chemical Engineering* **116**, 451–467. Multi-scale Systems Engineering – in memory & honor of Professor C.A. Floudas.

Mogensen, P. K. & Riseth, A. N. (2018), 'Optim: A mathematical optimization package for Julia', *Journal of Open Source Software* **3**(24), 615.

Mori, J. & Mahalec, V. (2017), 'Planning and scheduling of steel plates production. Part II: Scheduling of continuous casting', *Computers & Chemical Engineering* **101**, 312–325.

Munoz-Alvarez, D., Bitar, E., Tong, L. & Wang, J. (2014), Piecewise affine dispatch policies for economic dispatch under uncertainty, *in* 'PES General Meeting— Conference & Exposition, 2014 IEEE', IEEE, pp. 1–5.

Niederreiter, H. (1992), *Random number generation and quasi-Monte Carlo methods*, SIAM.

Ning, C. & You, F. (2017), 'A data-driven multistage adaptive robust optimization framework for planning and scheduling under uncertainty', *AIChE Journal* **63**(10), 4343–4369.

Ning, C. & You, F. (2018*a*), 'Data-driven decision making under uncertainty integrating robust optimization with principal component analysis and kernel smoothing methods', *Computers & Chemical Engineering* **112**, 190–210.

Ning, C. & You, F. (2018*b*), 'Data-driven stochastic robust optimization: General computational framework and algorithm leveraging machine learning for optimization under uncertainty in the big data era', *Computers & Chemical Engineering* **111**, 115–133.

Ning, C. & You, F. (2019*a*), 'Data-driven adaptive robust unit commitment under wind power uncertainty: A bayesian nonparametric approach', *IEEE Transactions on Power Systems* **34**(3), 2409–2418.

Ning, C. & You, F. (2019*b*), 'Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming', *Computers & Chemical Engineering* **125**, 434–448.

Ning, C. & You, F. (2019*c*), 'Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming', *Computers & Chemical Engineering* **125**, 434–448.

Nohadani, O. & Sharma, K. (2018), 'Optimization under decision-dependent uncertainty', *SIAM Journal on Optimization* **28**(2), 1773–1795.

Noshadravan, A., Gaustad, G., Kirchain, R. & Olivetti, E. (2017), 'Operational strategies for increasing secondary materials in metals production under uncertainty', *Journal of Sustainable Metallurgy* **3**(2), 350–361.

Oroojlooyjadid, A., Snyder, L. V. & Takác, M. (2020), 'Applying deep learning to the newsvendor problem', *IISE Transactions* **52**(4), 444–463.

Ouelhadj, D. & Petrovic, S. (2009), 'A survey of dynamic scheduling in manufacturing systems', *Journal of Scheduling* **12**(4), 417–431.

Pacciarelli, D. & Pranzo, M. (2004), 'Production scheduling in a steelmaking-continuous casting plant', *Computers and Chemical Engineering* **28**(12), 2823–2835.

Pan, L., Housh, M., Liu, P., Cai, X. & Chen, X. (2015), 'Robust stochastic optimization for reservoir operation', *Water Resources Research* **51**(1), 409–429.

Peidro, D., Mula, J., Poler, R. & Lario, F.-C. (2009), 'Quantitative models for supply chain planning under uncertainty: a review', *The International Journal of Advanced Manufacturing Technology* **43**(3-4), 400–420.

Pereira, M. V. & Pinto, L. M. (1991), 'Multi-stage stochastic optimization applied to energy planning', *Mathematical programming* **52**(1-3), 359–375.

Pistikopoulos, E. (1995), 'Uncertainty in process design and operations', *Computers & Chemical Engineering* **19**, 553–563.

Pistikopoulos, E. N., Diangelakis, N. A., Oberdieck, R., Papathanasiou, M. M., Nascu, I. & Sun, M. (2015), 'Paroc—an integrated framework and software platform for the optimisation and advanced model-based control of process systems', *Chemical Engineering Science* **136**, 115–138.

Pistikopoulos, E. N., Dua, V., Bozinis, N. A., Bemporad, A. & Morari, M. (2000), 'On-line optimization via off-line parametric optimization tools', *Computers & Chemical Engineering* **24**(2), 183–188.

Poss, M. (2014), 'Robust combinatorial optimization with variable cost uncertainty', *European Journal of Operational Research* **237**(3), 836–845.

Powell, W. B. (2009), 'What you should know about approximate dynamic programming', *Naval Research Logistics (NRL)* **56**(3), 239–249.

Powell, W. B. (2011), *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, Vol. 842, John Wiley & Sons.

Powell, W. B. (2014), *Clearing the jungle of stochastic optimization*, Informs, chapter Chapter 4, pp. 109–137.

Rahal, S., Li, Z. & Papageorgiou, D. J. (2020), 'Proactive and reactive scheduling of the steelmaking and continuous casting process through adaptive robust optimization', *Computers & Chemical Engineering* **133**, 106658.

Rahal, S., Papageorgiou, D. J. & Li, Z. (2021), 'Hybrid strategies using linear and piecewise-linear decision rules for multistage adaptive linear optimization', *European Journal of Operational Research* **290**(3), 1014–1030.

Revelle, C., Joeres, E. & Kirby, W. (1969), 'The linear decision rule in reservoir management and design: 1, development of the stochastic model', *Water Resources Research* **5**(4), 767–777.

Rios, L. M. & Sahinidis, N. V. (2013), 'Derivative-free optimization: a review of algorithms and comparison of software implementations', *Journal of Global Optimization* **56**(3), 1247–1293.

Rocha, P. & Kuhn, D. (2012), 'Multistage stochastic portfolio optimisation in deregulated electricity markets using linear decision rules', *European Journal of Operational Research* **216**(2), 397–408.

Rockafellar, R. T. (2007), 'Coherent approaches to risk in optimization under uncertainty', *Tutorials in Operations Research* pp. 38–61.

Rockafellar, R. T. & Uryasev, S. (2000), 'Optimization of conditional value-at-risk', *Journal of risk* **2**, 21–42.

Romero, J., Puigjaner, L., Holczinger, T. & Friedler, F. (2004), 'Scheduling intermediate storage multipurpose batch plants using the s-graph', *American Institute of Chemical Engineers. AIChE Journal* **50**(2), 403–417.

Rong, A. & Lahdelma, R. (2008), 'Fuzzy chance constrained linear programming model for optimizing the scrap charge in steel production', *European Journal of Operational Research* **186**(3), 953–964.

Ross, S. M. (2014), *Introduction to stochastic dynamic programming*, Academic press.

Rossiter, J. A. & Grieder, P. (2005), 'Using interpolation to improve efficiency of multiparametric predictive control', *Automatica* **41**(4), 637–643.

Roy, S. K., Mhammedi, Z. & Harandi, M. (2018), Geometry aware constrained optimization techniques for deep learning, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 4460–4469.

Ruiz, R. & Vázquez-Rodríguez, J. A. (2010), 'The hybrid flow shop scheduling problem', *European Journal of Operational Research* **205**(1), 1–18.

Sahinidis, N. V. (2004), 'Optimization under uncertainty: state-of-the-art and opportunities', *Computers & Chemical Engineering* **28**(6-7), 971–983.

Schmidhuber, J. (2015), 'Deep learning in neural networks: An overview', *Neural networks* **61**, 85–117.

See, C.-T. & Sim, M. (2010), 'Robust approximation to multiperiod inventory management', *Operations research* **58**(3), 583–594.

Shaik, M. A., Janak, S. L. & Floudas, C. A. (2006), 'Continuous-time models for short-term scheduling of multipurpose batch plants: A comparative study', *Industrial & Engineering Chemistry Research* **45**(18), 6190–6209.

Shang, C., Chen, W.-H. & You, F. (2019), Robust constrained model predictive control of irrigation systems based on data-driven uncertainty set constructions, *in* '2019 American Control Conference (ACC)', IEEE, pp. 1–6.

Shang, C., Yang, F., Huang, D. & Lyu, W. (2014), 'Data-driven soft sensor development based on deep learning technique', *Journal of Process Control* **24**(3), 223–233.

Shapiro, A. (2011), 'Analysis of stochastic dual dynamic programming method', *European Journal of Operational Research* **209**(1), 63–72.

Shapiro, A., Dentcheva, D. & Ruszczynski, A. (2009), 'Lectures on stochastic programming', *MPS-SIAM series on optimization* **9**, 3.

Shapiro, A. & Nemirovski, A. (2005), 'On complexity of stochastic programming problems', *Continuous optimization* pp. 111–146.

Shen, F., Zhao, L., Du, W., Zhong, W. & Qian, F. (2020), 'Large-scale industrial energy systems optimization under uncertainty: A data-driven robust optimization approach', *Applied Energy* **259**, 114199.

Soyster, A. L. (1973), 'Convex programming with set-inclusive constraints and applications to inexact linear programming', *Operations research* **21**(5), 1154–1157.

Subramanian, K., Rawlings, J. B., Maravelias, C. T., Flores-Cerrillo, J. & Megan, L. (2013), 'Integration of control theory and scheduling methods for supply chain management', *Computers & Chemical Engineering* **51**, 4–20.

Sun, L., Luan, F. & Pian, J. (2015), 'An effective approach for the scheduling of refining process with uncertain iterations in steel-making and continuous casting process', *IFAC PapersOnLine* **48**(3), 1966–1972.

Tejeda-Iglesias, M., Lappas, N. H., Gounaris, C. E. & Ricardez-Sandoval, L. (2019), 'Explicit model predictive controller under uncertainty: An adjustable robust optimization approach', *Journal of Process Control* **84**, 115–132.

Tøndel, P., Johansen, T. A. & Bemporad, A. (2003), 'Evaluation of piecewise affine control via binary search tree', *Automatica* **39**(5), 945–950.

Vayanos, P., Kuhn, D. & Rustem, B. (2011), Decision rules for information discovery in multi-stage stochastic programming, *in* '2011 50th IEEE Conference on Decision and Control and European Control Conference', pp. 7368–7373.

Venkatasubramanian, V. (2019), 'The promise of artificial intelligence in chemical engineering: Is it here, finally', *AIChE J* **65**(2), 466–478.

Verderame, P. M., Elia, J. A., Li, J. & Floudas, C. A. (2010), 'Planning and scheduling under uncertainty: a review across multiple sectors', *Industrial & engineering chemistry research* **49**(9), 3993–4017.

Verderame, P. M. & Floudas, C. A. (2009), 'Operational planning of large-scale industrial batch plants under demand due date and amount uncertainty. i. robust optimization framework', *Industrial and Engineering Chemistry Research* **48**(15), 7214–7231.

Wang, D. J., Liu, F. & Jin, Y. (2017), 'A proactive scheduling approach to steel rolling process with stochastic machine breakdown', *Natural Computing* .

Wiesler, S., Richard, A., Schlüter, R. & Ney, H. (2014), Mean-normalized stochastic gradient for large-scale deep learning, *in* '2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', IEEE, pp. 180–184.

Worapradya, K. & Thanakijkasem, P. (2010), Worst case performance scheduling facing uncertain disruption in a continuous casting process, *in* '2010 IEEE International Conference on Industrial Engineering and Engineering Management', IEEE, pp. 291–295.

Yanıkoğlu, I., Gorissen, B. L. & den Hertog, D. (2019), 'A survey of adjustable robust optimization', *European Journal of Operational Research* **277**(3), 799–813.

Ye, Y., Li, Z., Li, J., Tang, Q., Xiao, X. & Floudas, C. A. (2014), 'Robust optimization and stochastic programming approaches for medium-term production scheduling of a large-scale steelmaking continuous casting process under demand uncertainty', *Computers and Chemical Engineering* **66**, 165–185.

Yu, S. (2013), 'A prediction method for abnormal condition of scheduling plan with operation time delay in steelmaking and continuous casting production process', *ISIJ International* **53**(6), 1028–1041.

Yu, S. P., Pang, X. F., Chai, T. Y. & Zheng, B. L. (2009), 'Research on production scheduling for steelmaking and continuous casting with processing time uncertainty [J]', *Control and Decision* **24**(10), 1467–1472.

Yuan, Y., Li, Z. & Huang, B. (2016), 'Robust optimization under correlated uncertainty: Formulations and computational study', *Computers & Chemical Engineering* **85**, 58–71.

Zhang, Q., Morari, M. F., Grossmann, I. E., Sundaramoorthy, A. & Pinto, J. M. (2016), 'An adjustable robust optimization approach to scheduling of continuous industrial processes providing interruptible load', *Computers & Chemical Engineering* **86**, 106–119.

Zhang, X., Georghiou, A. & Lygeros, J. (2015), Convex approximation of chance-constrained MPC through piecewise affine policies using randomized and robust optimization, *in* 'Decision and Control (CDC), 2015 IEEE 54th Annual Conference on', IEEE, pp. 3038–3043.

Zhang, Y., Jin, X., Feng, Y. & Rong, G. (2018), 'Data-driven robust optimization under correlated uncertainty: A case study of production scheduling in ethylene plant', *Computers & Chemical Engineering* **109**, 48–67.

Zhang, Z. & Zhao, J. (2017), ‘A deep belief network based fault diagnosis model for complex chemical processes’, *Computers & Chemical Engineering* **107**, 395–407.

Zhao, L., Ning, C. & You, F. (2019), ‘Operational optimization of industrial steam systems under uncertainty using data-driven adaptive robust optimization’, *AIChE Journal* **65**(7), e16500.

Zhao, L., Zhong, W. & Du, W. (2019), ‘Data-driven robust optimization for steam systems in ethylene plants under uncertainty’, *Processes* **7**(10), 744.

Zhen, J., den Hertog, D. & Sim, M. (2018), ‘Adjustable robust optimization via fourier–motzkin elimination’, *Operations Research* **66**(4), 1086–1100.

Zhu, W., Ma, Y., Benton, M. G., Romagnoli, J. A. & Zhan, Y. (2019), ‘Deep learning for pyrolysis reactor monitoring: From thermal imaging toward smart monitoring system’, *AIChE Journal* **65**(2), 582–591.

Zou, J., Ahmed, S. & Sun, X. A. (2019), ‘Stochastic dual dynamic integer programming’, *Mathematical Programming* **175**(1-2), 461–502.

Zugno, M., Morales, J. M. & Madsen, H. (2016), ‘Commitment and dispatch of heat and power units via affinely adjustable robust optimization’, *Computers & Operations Research* **75**, 191 – 201.

Appendix A

Chapter 2 supplementary materials

A.1 Nomenclature

General

	Definitions
Sets	
$\mathcal{T}$	Time stages
$\mathcal{T}_{-t}$	Time stages excluding stage t
Ξ	Uncertainty set
Ξ'_t	Lifted uncertainty in stage t
Ξ'	Overall Lifted uncertainty set
$\bar{\Xi}'$	Outer approximation of conv Ξ'
Parameters	
ξ_t	Primitive uncertainty in stage t
$\hat{\xi}_t$	Scenarios approximating ξ_t
$\boldsymbol{W}$	Matrix of Ξ
$\boldsymbol{h}$	Right hand side vector of Ξ'
$\boldsymbol{A}_t$	Matrix of conv Ξ'_t
$\boldsymbol{b}_t$	Right-hand side vector of conv Ξ'_t
$\boldsymbol{A}^1$	Matrix of $\bar{\Xi}'$
$\boldsymbol{b}^1$	Right-hand side vector of $\bar{\Xi}'$
$\boldsymbol{w}_i$	The i^{th} column vector of $\boldsymbol{W}$
$\boldsymbol{V}_t$	Observation matrix in stage t
$\boldsymbol{e}_t$	Vector with a value 1 at the t index, and 0 otherwise
$\boldsymbol{e}'_t$	Vector with a value of 1 from the index $\sum_1^{t-1} r_t + 1$ to $\sum_1^t r_t$, and 0 otherwise

Unless otherwise stated, the superscript " $'$ " refers to the same variable definition, but in the lifted uncertainty space.

Newsvendor Model

	Definitions
Variables	
x_t	Ordered units in stage t
I_t $[s_t^+]$ (s_t^-)	Balance [holding] (Backlog) of units in stage t
p_t^0	Intercept of LDR in stage t where $p = \{x, I, s^+, s^-\}$
$\mathbf{P}_t^1$	Slope of LDR at stage t where $\mathbf{P} = \{\boldsymbol{X}, \boldsymbol{I}, \boldsymbol{S}^+, \boldsymbol{S}^-\}$
Parameters	
d_t	Demand in stage t
r_t	Number of breakpoints for d_t
z_j^i	The j^{th} breakpoint in d_i
d'_{tj}	The j^{th} lifted element of d_t
U^{x}	Ordering amount limit
C_t $[H_t]$ (B_t)	Purchasing [Holding] (Backlogging) cost in stage t
I_1	Initial inventory
l_t u_t	Lower and upper bound of d_t

Transportation Model

	Definitions
Sets	
$\mathcal{I}$	Set of suppliers
$\mathcal{J}$	Set of customers
$\mathcal{Z}_{\text{base}}$	Base set of potential breakpoints
Variables	
x_{it} $[I_{it}]$	Produced [Inventory] units by supplier i in stage t
y_{ijt}	Transported items from supplier i to customer j in stage t
p_t^0	Intercept of LDR at stage t where $p := \{x_i, I_i, y_{ij}\}$
$\mathbf{P}_t^1$	Slope of LDR at stage t where $\mathbf{P} := \{\boldsymbol{X}_i, \boldsymbol{I}_i, \boldsymbol{y}_{ij}\}$
Parameters	
R_{jt}	Unit revenue of customer j in stage t
T_{ijt}	Unit transportation cost along (i, j) arc in stage t
C_{it} $[H_{it}]$	Unit production [Holding] cost of supplier i in stage t
S_i	Unit salvage value for supplier i
$U_i^{\max}$	Production limit of supplier i
D_{jt}	Customer demand j in stage t
D_{jt}^0 $[D_{jt}^1]$	Intercept [Slope] of customers j's linear demand function in stage t

A.2 Multistage newsvendor problem

The multistage newsvendor problem is given as

$$\min_{\substack{x_t(\cdot),I_t(\cdot)\\ s_t^+(\cdot),s_t^-(\cdot)}} \quad \mathbb{E}\left[\sum_{t=1}^{T-1} C_t x_t(\boldsymbol{d}_{[t]}) + \sum_{t=2}^{T}(H_t s_t^+(\boldsymbol{d}_{[t]}) + B_t s_t^-(\boldsymbol{d}_{[t]}))\right] \tag{A.2.1a}$$

$$\text{s.t.} \quad I_t(\boldsymbol{d}_{[t]}) = I_{t-1}(\boldsymbol{d}_{[t-1]}) + x_{t-1}(\boldsymbol{d}_{[t-1]}) - d_t \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.1b}$$

$$s_t^+(\boldsymbol{d}_{[t]}) \geq I_t(\boldsymbol{d}_{[t]}) \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.1c}$$

$$s_t^-(\boldsymbol{d}_{[t]}) \geq -I_t(\boldsymbol{d}_{[t]}) \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.1d}$$

$$0 \leq x_t(\boldsymbol{d}_{[t]}) \leq U^x \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-T} \tag{A.2.1e}$$

$$s_t^+(\boldsymbol{d}_{[t]}),\ s_t^-(\boldsymbol{d}_{[t]}) \geq 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.1f}$$

where $\boldsymbol{d}_{[t]} = [d_2, \cdots, d_t]$, $x_1(\boldsymbol{d}_{[1]}) \equiv x_1$ is the first-stage ordering decision, $I_1(\boldsymbol{d}_{[1]}) \equiv I_1$, and Ξ is the uncertainty set for demand. The expectation is computed with respect to the distribution of $\boldsymbol{d}$ (e.g., $\boldsymbol{d}_{[T]}$).

A.2.1 Newsvendor problem linear adaptive stochastic counterpart

For $x_t(\boldsymbol{d}_{[t]})$, $I_t(\boldsymbol{d}_{[t]})$, $s_t^+(\boldsymbol{d}_{[t]})$ and $s_t^-(\boldsymbol{d}_{[t]})$, consider the following LDRs

$$x_t(\boldsymbol{d}_{[t]}) = x_t^0 + \boldsymbol{X}_t^1 \boldsymbol{V}_t \boldsymbol{d} \qquad \forall t \in \mathcal{T}_{-T} \tag{A.2.2a}$$

$$I_t(\boldsymbol{d}_{[t]}) = I_t^0 + \boldsymbol{I}_t^1 \boldsymbol{V}_t \boldsymbol{d} \qquad \forall t \in \mathcal{T} \tag{A.2.2b}$$

$$s_t^+(\boldsymbol{d}_{[t]}) = s_t^{0+} + \boldsymbol{S}_t^{1+} \boldsymbol{V}_t \boldsymbol{d} \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.2c}$$

$$s_t^-(\boldsymbol{d}_{[t]}) = s_t^{0-} + \boldsymbol{S}_t^{1-} \boldsymbol{V}_t \boldsymbol{d} \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.2d}$$

where x_t^0, I_t^0, s_t^{0+}, s_t^{0-} are the intercepts and $\boldsymbol{X}_t^1, \boldsymbol{S}_t^{1+}$, $\boldsymbol{S}_t^{1-}$, $\boldsymbol{I}_t^1 \in \mathbb{R}^{1\times(T-1)}$ are the slopes. We let $x_1^0 = x_1$, $I_1^0 = I_1$ and $\boldsymbol{X}_1^1 = \boldsymbol{I}_1^1 = \boldsymbol{0}$.

Implementing LDRs in the set of constraints of Model (A.2.1), we obtain

$$I_t^0 + \boldsymbol{I}_t^1 \boldsymbol{V}_t \boldsymbol{d} = I_{t-1}^0 + \boldsymbol{I}_{t-1}^1 \boldsymbol{V}_{t-1} \boldsymbol{d} + x_{t-1}^0 + \boldsymbol{X}_{t-1}^1 \boldsymbol{V}_{t-1} \boldsymbol{d} + d_t \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.3a}$$

$$s_t^{0+} + \boldsymbol{S}_t^{1+} \boldsymbol{V}_t \boldsymbol{d} \geq I_t^0 + \boldsymbol{I}_t^1 \boldsymbol{V}_t \boldsymbol{d} \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.3b}$$

$$s_t^{0-} + \boldsymbol{S}_t^{1-} \boldsymbol{V}_t \boldsymbol{d} \geq -I_t^0 - \boldsymbol{I}_t^1 \boldsymbol{V}_t \boldsymbol{d} \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.3c}$$

$$x_t^0 + \boldsymbol{X}_t^1 \boldsymbol{V}_t \boldsymbol{d} \geq 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-T} \tag{A.2.3d}$$

$$x_t^0 + \boldsymbol{X}_t^1 \boldsymbol{V}_t \boldsymbol{d} \leq U^x \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-T} \tag{A.2.3e}$$

$$s_t^{0+} + \boldsymbol{S}_t^{1+} \boldsymbol{V}_t \boldsymbol{d} \geq 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.3f}$$

$$s_t^{0-} + \boldsymbol{S}_t^{1-} \boldsymbol{V}_t \boldsymbol{d} \geq 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.3g}$$

Recall the polyhedral uncertainty set $\Xi := \left\{\boldsymbol{d} \in \mathbb{R}^{T-1} |\ \boldsymbol{W}\boldsymbol{d} \geq \boldsymbol{h}\right\}$ where $\boldsymbol{W} \in \mathbb{R}^{m\times(T-1)}$ and $\boldsymbol{h} \in \mathbb{R}^m$. As a mean of example, we will derive the stochastic counterpart for the semi-infinite constraint in eq. (A.2.3b). First, it is rearranged as follows

$$s_t^{0+} - I_t^0 + \left\{ \begin{array}{ll} \min\limits_{\boldsymbol{d}} & \left(\boldsymbol{S}_t^{1+}\boldsymbol{V}_t - \boldsymbol{I}_t^1\boldsymbol{V}_t\right)\boldsymbol{d} \\ \text{s.t.} & \boldsymbol{W}\boldsymbol{d} \geq \boldsymbol{h} \end{array} \right\} \geq 0 \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.4}$$

Introducing the $\min_{\boldsymbol{d}}(\cdot)$ operator does not affect the solution. When eq. (A.2.4) is satisfied, it follows that eq. (A.2.3b) is satisfied for all $\boldsymbol{d} \in \Xi$. The dual of the inner minimization problem is derived as

$$s_t^{0+} - I_t^0 + \left\{ \begin{array}{ll} \max\limits_{\boldsymbol{\tau}_t} & \boldsymbol{h}^\top \boldsymbol{\tau}_t \\ \text{s.t.} & \boldsymbol{W}^\top \boldsymbol{\tau}_t = \left(\boldsymbol{S}_t^{1+}\boldsymbol{V}_t - \boldsymbol{I}_t^1\boldsymbol{V}_t\right)^\top \\ & \boldsymbol{\tau}_t \in \mathbb{R}_+^m \end{array} \right\} \geq 0 \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.5}$$

where $\boldsymbol{\tau}_t$ is the dual variable. The $\max_{\boldsymbol{\tau}_t}(\cdot)$ operator can be removed without affecting the optimal solution. The final form of the stochastic counterpart of eq. (A.2.3b) is equivalent to

$$s_t^{0+} - I_t^0 + \boldsymbol{h}^\top \boldsymbol{\tau}_t \geq 0 \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.6a}$$

$$\boldsymbol{W}^\top \boldsymbol{\tau}_t = \left(\boldsymbol{S}_t^{1+}\boldsymbol{V}_t - \boldsymbol{I}_t^1\boldsymbol{V}_t\right)^\top \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.6b}$$

$$\boldsymbol{\tau}_t \in \mathbb{R}_+^m \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.6c}$$

The stochastic counterparts of the remaining semi-infinite inequality constraints are derived following the same procedure. Equation (A.2.3a) is the only semi-infinite equality constraint, and its tractable counterpart is derived by forcing the intercept and the slope of the constraint to be equal to zero. To illustrate, eq. (A.2.3a) is equivalently rewritten as

$$I_t^0 - I_{t-1}^0 - x_{t-1}^0 + (\boldsymbol{I}_t^1\boldsymbol{V}_t - \boldsymbol{I}_{t-1}^1\boldsymbol{V}_{t-1} - \boldsymbol{X}_{t-1}^1\boldsymbol{V}_{t-1} + \boldsymbol{e}_{t-1}^\top)\boldsymbol{d} = 0 \qquad \forall \boldsymbol{d} \in \Xi,\ t \in \mathcal{T}_{-1} \tag{A.2.7}$$

The vector $\boldsymbol{e}_t \in \mathbb{R}^{1\times(T-1)}$ has a value of 1 at the t index and a value of 0 elsewhere. It is satisfied for all $\boldsymbol{d} \in \Xi$ if and only if the intercept and slope is equal to zero

$$I_t^0 - I_{t-1}^0 - x_{t-1}^0 = 0 \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.8a}$$

$$\boldsymbol{I}_t^1\boldsymbol{V}_t - \boldsymbol{I}_{t-1}^1\boldsymbol{V}_{t-1} - \boldsymbol{X}_{t-1}^1\boldsymbol{V}_{t-1} + \boldsymbol{e}_{t-1}^\top = \boldsymbol{0} \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.8b}$$

Hence, equation (A.2.8) is considered the stochastic counterpart of equation (A.2.3a).

The linear adaptive stochastic counterpart (LASC) of Model (A.2.3) is formulated as

$$
\begin{aligned}
& I_t^0 - I_{t-1}^0 - x_{t-1}^0 = 0 && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9a)}\\
& \boldsymbol{I}_t^1 \boldsymbol{V}_t - \boldsymbol{I}_{t-1}^1 \boldsymbol{V}_{t-1} - \boldsymbol{X}_{t-1}^1 \boldsymbol{V}_{t-1} + \boldsymbol{e}_{t-1}^\top = \boldsymbol{0} && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9b)}\\
& s_t^{0+} - I_t^0 + \boldsymbol{h}^\top \boldsymbol{\tau}_t \geq 0 && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9c)}\\
& \boldsymbol{W}^\top \boldsymbol{\tau}_t = (\boldsymbol{S}_t^{1+} \boldsymbol{V}_t - \boldsymbol{I}_t^1 \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9d)}\\
& s_t^{0-} + I_t^0 + \boldsymbol{h}^\top \boldsymbol{\alpha}_t \geq 0 && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9e)}\\
& \boldsymbol{W}^\top \boldsymbol{\alpha}_t = (\boldsymbol{S}_t^{1-} \boldsymbol{V}_t + \boldsymbol{I}_t^1 \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9f)}\\
& x_t^0 + \boldsymbol{h}^\top \boldsymbol{\gamma}_t \geq 0 && \forall t \in \mathcal{T}_{-T} && \text{(A.2.9g)}\\
& \boldsymbol{W}^\top \boldsymbol{\gamma}_t \geq (\boldsymbol{X}_t^1 \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-T} && \text{(A.2.9h)}\\
& U^x - x_t^0 + \boldsymbol{h}^\top \boldsymbol{\delta}_t \geq 0 && \forall t \in \mathcal{T}_{-T} && \text{(A.2.9i)}\\
& \boldsymbol{W}^\top \boldsymbol{\delta}_t = -(\boldsymbol{X}_t^1 \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-T} && \text{(A.2.9j)}\\
& s_t^{0+} + \boldsymbol{h}^\top \boldsymbol{\lambda}_t \geq 0 && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9k)}\\
& \boldsymbol{W}^\top \boldsymbol{\lambda}_t = (\boldsymbol{S}_t^{1+} \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9l)}\\
& s_t^{0-} + \boldsymbol{h}^\top \boldsymbol{\mu}_t \geq 0 && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9m)}\\
& \boldsymbol{W}^\top \boldsymbol{\mu}_t = (\boldsymbol{S}_t^{1-} \boldsymbol{V}_t)^\top && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9n)}\\
& \boldsymbol{u}_t,\ \boldsymbol{v}_t,\ \boldsymbol{\lambda}_t,\ \boldsymbol{\mu}_t \in \mathbb{R}_+^m && \forall t \in \mathcal{T}_{-1} && \text{(A.2.9o)}\\
& \boldsymbol{\gamma}_t,\ \boldsymbol{\delta}_t \in \mathbb{R}_+^m && \forall t \in \mathcal{T}_{-T} && \text{(A.2.9p)}
\end{aligned}
$$

where $\boldsymbol{\tau}_t,\ \boldsymbol{\alpha}_t,\ \boldsymbol{\lambda}_t,\ \boldsymbol{\mu}_t,\ \boldsymbol{\gamma}_t,\ \boldsymbol{\delta}_t$ are dual variables. We let $\boldsymbol{\gamma}_1 = \boldsymbol{\delta}_1 = \boldsymbol{0}$.

Defining the adaptive decisions in the objective function with the corresponding LDRs, the newsvendor problem's LASC becomes

$$
\begin{aligned}
\min\ & \sum_{t=1}^{T-1} C_t(x_t^0 + \boldsymbol{X}_t^1 \boldsymbol{V}_t \mathbb{E}[\boldsymbol{d}]) + \sum_{t=2}^{T} \left[H_t(s_t^{0+} + \boldsymbol{S}_t^{1+} \boldsymbol{V}_t \mathbb{E}[\boldsymbol{d}]) + B_t(s_t^{0-} + \boldsymbol{S}_t^{1-} \boldsymbol{V}_t \mathbb{E}[\boldsymbol{d}]) \right] && \text{(A.2.10a)}\\
\text{s.t.}\ & \text{eqs.(A.2.9a)} - \text{(A.2.9p)} && \text{(A.2.10b)}
\end{aligned}
$$

where $\mathbb{E}[\boldsymbol{d}]$ is the mean vector of the uncertain demand.

A.2.2 Newsvendor problem piecewise linear adaptive stochastic counterpart

The PLDRs, which are LDRs in the lifted uncertainty space, are defined similarly to eq. (A.2.2).

$$
\begin{aligned}
x_t(\boldsymbol{d}'_{[t]}) =&\ x_t'^0 + \boldsymbol{X}_t'^1 \boldsymbol{V}_t' \boldsymbol{d}' && \forall t \in \mathcal{T}_{-T} && \text{(A.2.11a)}\\
I_t(\boldsymbol{d}'_{[t]}) =&\ I_t'^0 + \boldsymbol{I}_t'^1 \boldsymbol{V}_t' \boldsymbol{d}' && \forall t \in \mathcal{T} && \text{(A.2.11b)}\\
s_t^+(\boldsymbol{d}'_{[t]}) =&\ s_t'^{0+} + \boldsymbol{S}_t'^{1+} \boldsymbol{V}_t' \boldsymbol{d}' && \forall t \in \mathcal{T}_{-1} && \text{(A.2.11c)}\\
s_t^-(\boldsymbol{d}'_{[t]}) =&\ s_t'^{0-} + \boldsymbol{S}_t'^{1-} \boldsymbol{V}_t' \boldsymbol{d}' && \forall t \in \mathcal{T}_{-1} && \text{(A.2.11d)}
\end{aligned}
$$

where the intercepts are given by $x_t'^0$, $I_t'^0$, $s_t'^{0+}$, $s_t'^{0-} \in \mathbb{R}$ and the slopes are defined as $\boldsymbol{X}_t'^1$, $\boldsymbol{I}_t'^1$, $\boldsymbol{S}_t'^{1+}$, $\boldsymbol{S}_t'^{1-} \in \mathbb{R}^{1\times k'}$. We let $x_1'^0 = x_1$, $I_1'^0 = I_1$ and $\boldsymbol{X}_1'^1 = \boldsymbol{I}_1'^1 = \boldsymbol{0}$.

The convex hull of the lifted line-segment uncertainty set Ξ_t' is derived by Georghiou et al. 2015, and introduced here with slight reformulation

$$\text{conv } \Xi_t' := \{\boldsymbol{d}_t' \in \mathbb{R}^{r_t} |\ \boldsymbol{A}_t'\boldsymbol{d}_t' \geq \boldsymbol{b}_t'\} \qquad \forall t \in \mathcal{T}_{-1} \tag{A.2.12}$$

where $\boldsymbol{A}_t' \in \mathbb{R}^{(r_t+1)\times(r_t)}$, $\boldsymbol{b}_t' \in \mathbb{R}^{r_t+1}$ are equal to

$$\boldsymbol{A}_t' = \begin{bmatrix} -\frac{1}{z_1^t - l_t} & & & & \\ \frac{1}{z_1^t - l_t} & -\frac{1}{z_2^t - z_1^t} & & & \\ & \frac{1}{z_2^t - z_1^t} & \ddots & & \\ & & \ddots & -\frac{1}{z_{r_t-1}^t - z_{r_t-2}^t} & \\ & & & \frac{1}{z_{r_t-1}^t - z_{r_t-2}^t} & -\frac{1}{u_t - z_{r_t-1}^t} \\ & & & & \frac{1}{u_t - z_{r_t-1}^t} \end{bmatrix}, \quad \boldsymbol{b}_t' = \begin{bmatrix} -\frac{z_1^t}{z_1^t - l_t} \\ \frac{l_t}{z_1^t - l_t} \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

For a generic polytope Ξ,the convex hull of the lifted uncertainty set is not tractable. Georghiou et al. 2015 presented a tractable outer approximation of the convex hull which is given below

$$\begin{aligned} \bar{\Xi}' : &= \{\boldsymbol{d}' \in \mathbb{R}^{k'} |\ \mathbf{W}\ R\boldsymbol{d}' \geq \mathbf{h},\ \boldsymbol{A}_t'\boldsymbol{d}_t' \geq \boldsymbol{b}_t' \quad t \in \mathcal{T}_{-1}\} \\ &= \{\boldsymbol{d}' \in \mathbb{R}^{k'} |\ \boldsymbol{W}'\boldsymbol{d}' \geq \mathbf{h},\ \boldsymbol{A}_t'\boldsymbol{d}_t' \geq \boldsymbol{b}_t' \quad t \in \mathcal{T}_{-1}\} \\ &= \{\boldsymbol{d}' \in \mathbb{R}^{k'} |\ \boldsymbol{A}^l\boldsymbol{d}' \geq \boldsymbol{b}^l\} \quad \supseteq \text{conv } \Xi' \end{aligned} \tag{A.2.13}$$

The matrix $\boldsymbol{W}' \in \mathbb{R}^{T-1\times k'}$ is defined as

$$\boldsymbol{W}' = \left[\boldsymbol{w}_1\boldsymbol{1}_{r_2}, \boldsymbol{w}_2\boldsymbol{1}_{r_3}, \cdots, \boldsymbol{w}_{T-1}\boldsymbol{1}_{r_T}\right]$$

where $\boldsymbol{w}_i \in \mathbb{R}^{T-1}$ is the i^{th} column of $\boldsymbol{W}$, and $\boldsymbol{1}_{r_i} \in \mathbb{R}^{1\times r_1}$ is a unit row vector. The matrix $\boldsymbol{A}^l \in \mathbb{R}^{(m+m')\times k'}$ and vector $\boldsymbol{b}^l \in \mathbb{R}^{m+m'}$, where $m' = \sum_{t=2}^{T}(r_t + 1)$, are formulated as

$$\boldsymbol{A}^l = \begin{bmatrix} & \boldsymbol{W}' & & \\ \boldsymbol{A}_2' & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_3' & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{A}_T' \end{bmatrix}, \quad \boldsymbol{b}^l = \begin{bmatrix} \boldsymbol{h} \\ \boldsymbol{b}_2' \\ \boldsymbol{b}_3' \\ \vdots \\ \boldsymbol{b}_T' \end{bmatrix}$$

Since both uncertainty sets Ξ and $\bar{\Xi}'$ are polytopes, the procedure of constructing the Piecewise linear adaptive stochastic counterpart (PWLASC) is similar to that of LASC with a few changes in parameters and dimensions. The formulation of the multistage newsvendor's

PWLASC is given as

$$\begin{aligned}
\min \quad & \sum_{t=1}^{T-1} C_t(x_t'^0 + \boldsymbol{X}_t'^1 \boldsymbol{V}_t' \mathbb{E}[\boldsymbol{d}']) + \sum_{t=2}^{T} (H_t(s_t'^{0+} + \boldsymbol{S}_t'^{1+} \boldsymbol{V}_t' \mathbb{E}[\boldsymbol{d}']) + B_t(s_t'^{0-} + \boldsymbol{S}_t'^{1-} \boldsymbol{V}_t' \mathbb{E}[\boldsymbol{d}'])) \\
\text{s.t.} \quad & \text{eqs. (A.2.9a)} - \text{(A.2.9p)} \qquad \text{(A.2.14a)}
\end{aligned}$$

The change in parameters are $\boldsymbol{W} \to \boldsymbol{A}^1,\ \boldsymbol{h} \to \boldsymbol{b}^1,\ \boldsymbol{V}_t \to \boldsymbol{V}_t',\ \boldsymbol{e}_{t-1} = \boldsymbol{e}_{t-1}' and\ m \to m + m'$. The notataion “a” → “b” means that b is replaced by a. The mean vector of the lifted uncertain demand is given by $\mathbb{E}[\boldsymbol{d}']$ and The row vector $\boldsymbol{e}_t' \in \mathbb{R}^{k'}$ has a value of 1 from index $\sum_1^{t-1} r_t + 1$ to $\sum_1^t r_t$ and a value of 0 elsewhere.

A.2.3 Optimal policies of multistage newsvendor illustrative example

The ordering, holding and backlog optimal policies for the illustrative multistage stochastic newsvendor example are depicted in Table A.4 where $T = 4,\ U^x = 8,\ I_1 = 4, d_t \sim \mathcal{U}(0, 1)\ t \in \mathcal{T}_{-1}$.

Table A.4: Optimal adaptive policies for a multistage stochastic newsvendor problem using an LDR, a PLDR-1 ($\mathbb{E}[d_t]$) and a PLDR-1 (U^x).

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ s_2^+ \\ s_3^+ \\ s_4^+ \\ s_2^- \\ s_3^- \\ s_4^- \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0.8 & 0 & 0 \\ 0 & 0.8 & 0 \\ -1 & 0 & 0 \\ -0.2 & -1 & 0 \\ 0 & -0.2 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.2 & 0 & 0 \end{bmatrix} \begin{bmatrix} d_2 \\ d_3 \\ d_4 \end{bmatrix} + \begin{bmatrix} 8 \\ 0 \\ 0 \\ 12 \\ 12 \\ 12 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

(a) LDR

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ s_2^+ \\ s_3^+ \\ s_4^+ \\ s_2^- \\ s_3^- \\ s_4^- \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 \\ -0.4 & 0 & -1 & -0.6 & 0 & 0 \\ -0.4 & 0 & -0.6 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d'_{21} \\ d'_{22} \\ d'_{31} \\ d'_{32} \\ d'_{41} \\ d'_{42} \end{bmatrix} + \begin{bmatrix} 6 \\ 0 \\ 0 \\ 10 \\ 10 \\ 10 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

(b) PLDR-1 ($\mathbb{E}[d_t]$)

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ s_2^+ \\ s_3^+ \\ s_4^+ \\ s_2^- \\ s_3^- \\ s_4^- \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} d'_{21} \\ d'_{22} \\ d'_{31} \\ d'_{32} \\ d'_{41} \\ d'_{42} \end{bmatrix} + \begin{bmatrix} 4 \\ 0 \\ 0 \\ 8 \\ 8 \\ 8 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

(c) PLDR-1 (U^{x})

Appendix B

Chapter 3 supplementary materials

B.1 Nomenclature

Sets	
$\mathcal{S}$	Set of processing stages s
$\mathcal{N}$	Set of event points n
$\mathcal{J}$	Set of units j
$\mathcal{I}$	Set of ladles i
$\mathcal{C}$	Set of casts c
$\mathcal{J}_s$, $\mathcal{J}_c$	Set of units associated with processing stage s and cast c
$\mathcal{J}^{\mathrm{p}}$	Set of processing units
$\mathcal{I}_c^{\text{first}}$	Set of first ladle i in a cast c
$\mathcal{I}_c^{\text{last}}$	Set of last ladle i in a cast c
$\mathcal{S}^{\text{ZW}}$	Set of processing stages with “zero wait” policy
$\mathcal{S}^{\text{perish}}$	Set of storage unit prior to CC
$\mathcal{C}_j$	Set of casts processed by unit j
$l_{c(i)}^{\text{first}}$	Absolute index of the first ladle in a cast c
Binary Variables	
$x_{c,n}$	Assignment of a cast c to an event point n
$y_{c,j}$	Assignment of a cast c to a processing unit j
$z_{c,j,n}$	Assignment of a cast c in a unit j and an event point n
Continuous Variables	
$t_{i,s}^{\mathrm{s}}(t_{i,s}^{\mathrm{f}})$	Start (finish) time of ladle i in processing stage s
$t_{j,n}^{\mathrm{s}}(t_{j,n}^{\mathrm{f}})$	Start (finish) time of event point n in processing unit j
$v_{c,c',j,j',n,n'}$	Auxiliary variable
$u_{j',n,n',c,c'}$	Auxiliary variable
MS	Total makespan
τ_i	Slack variable to control the casting speed of ladle i
Parameters	
n^{EAF}	Number of EAF units
n^{T}	Number of storage units prior to AOD
n^{AOD}	Number of AOD units
n^{CC}	Number of CC units
P_j^{setup}	Setup time of processing unit j
P_j^{perish}	Maximum storage duration in $s \in \mathcal{S}^{\text{perish}}$

$\overline{PT}_{i,j}$	Nominal processing time of ladle i in processing unit j
p_i	Processing order of a ladle i within cast $c(i)$
s_j	Processing stage of unit j
L_c	Total number of ladles in a cast c
$\xi_{i,j}$	Processing time uncertainty of ladle i in processing unit j
Δ_j	Maximum delay of all ladles in processing unit j
Δ_j^{tr}	Maximum delay of all ladles in EAF and AOD units in truncated set
η	Auxiliary parameter
μ	Mean of a given distribution
M	Positive scalar value equal to 4320 (arbitrary upper bound to MS)
M_1	Positive scalar value equal to 80

B.2 Materialization constraints

For $t^{\mathrm{s}}_{i,s}(\boldsymbol{\xi})$ LDR

1. For the uncertain parameters inferred using LSC 1/LSC 2:

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{i,s,i',j',n'} \leq M z_{c(i),j',n} z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{i,s,i',j',n'} \geq -M z_{c(i),j',n} z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall s \in \mathcal{S},\ c, c' \in \mathcal{C}, c' \neq,\ i \in \mathcal{I}_{\mathrm{c}},\ i' \in \mathcal{I}_{\mathrm{c}'}, \\ s' < s,\ j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}},\ n \in \mathcal{N},\ n' < n \end{array} \tag{B.2.1}$$

Equation (B.2.1) first identifies the unit and event point at which cast $c(i)$ starts processing in $s' < s$ (i.e., $z_{c(i),j',n} = 1$), then it identifies the casts c' that are processed in past events in the same unit (i.e., $z_{c'(i'),j',n'} = 1$).

2. For the uncertain parameters inferred using LSC 2/LSC 1:

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{i,s,i',j,n'} \leq M z_{c(i),j,n} z_{c'(i'),j,n'} z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{i,s,i',j,n'} \geq -M z_{c(i),j,n} z_{c'(i'),j,n'} z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall c, c' \in \mathcal{C}, c' \neq c,\ i \in \mathcal{I}_{\mathrm{c}},\ i' \in \mathcal{I}_{\mathrm{c}'}, s \in \mathcal{S}, \\ j \in \mathcal{J}_{s} \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}},\ s' < s, \\ j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}},\ n \in \mathcal{N},\ n' < n \end{array} \tag{B.2.2}$$

Equation (B.2.2) first identifies the unit and event point at which cast $c(i)$ is processed (i.e., $z_{c(i),j,n} = 1$), then it identifies the ladles i' within $c'(i')$ that are processed in the same unit in past events (i.e., $z_{c'(i'),j,n'} = 1$), then it identifies the units where ladles i' are processed in past processing stages (i.e., $z_{c'(i'),j',n'} = 1$).

For this problem the trilinear term in equation (B.2.2) is simplified to the bilinear term $z_{c(i),j,n}.z_{c'(i'),j',n'}$ in equation (B.2.3). If $z_{c'(i'),j',n'} = 1$, then $z_{c'(i'),j,n'} = 1$.

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{i,s,i',j',n'} \leq M z_{c(i),j,n} z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{i,s,i',j',n'} \geq -M z_{c(i),j,n} z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall c, c' \in \mathcal{C}, c' \neq c,\ i \in \mathcal{I}_{\mathrm{c}}, s \leq \mathcal{S},\ i' \in \mathcal{I}_{\mathrm{c}'}, \\ j \in \mathcal{J}_{s} \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}},\ s' < s, \\ j' \in \mathcal{J}_{s'} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}},\ n \in \mathcal{N},\ n' < n \end{array} \tag{B.2.3}$$

3. For the uncertain parameters inferred using NIS policy in current processing stage s:

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{i,s,i',j',n'} \leq M z_{c(i),j,n} z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{i,s,i',j',n'} \geq -M z_{c(i),j,n} z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall c, c' \in \mathcal{C}, c' \neq c,\ i \in \mathcal{I}_{\mathrm{c}},\ i' \in \mathcal{I}_{\mathrm{c}'}, s \in \mathcal{S}, \\ j \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}},\ 1 \leq k \leq L_c - 1, \\ i' \leq l^{\text{first}}_{c'(i)} + L_c - k - 1,\ n \in \mathcal{N},\ n' < n, \\ j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.4a}$$

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{i,s,i',j',n} \leq M z_{c(i'),j',n} \\ t^{1,\mathrm{s}}_{i,s,i',j',n} \geq -M z_{c(i'),j',n} \end{array}\right\} \begin{array}{l} \forall s \in \mathcal{S},\ c \in \mathcal{C},\ i, i' \in \mathcal{I}_{\mathrm{c}},\ 1 \leq k \leq p(i) - 2, \\ i' \leq l^{\text{first}}_{c(i)} + p(i) - k - 2,\ j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c(i')} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.4b}$$

The bilinear terms in equations (B.2.1),(B.2.3) and (B.2.4a) are linearized using the same auxiliary variable in equation (3.8) but with the relative set of indexes.

For $t^{\mathrm{s}}_{j,n}(\boldsymbol{\xi})$ and $t^{\mathrm{f}}_{j,n}(\boldsymbol{\xi})$ LDRs

1. For the uncertain parameters inferred using LSC 2/LSC 1, the simplified form is given as

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{j,n,i',j',n'} \leq M_1 z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{j,n,i',j',n'} \geq -M_1 z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall j \in \mathcal{J},\ n \in \mathcal{N},\ i' \in \mathcal{I}_{\mathrm{c}}, \\ n' < n,\ s < s^j,\ j' \in \mathcal{J}_s \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.5}$$

   Note that $z_{c'(i'),j',n'} = 1$ implicitly implies that $z_{c'(i'),j,n'} = 1$.

2. For the uncertain parameters inferred using LSC 1/LSC 2, the simplified bilinear constraints

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{j,n,i',j',n'} \leq M_1 z_{c(i),j',n} z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{j,n,i',j',n'} \geq -M_1 z_{c(i),j',n} z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall j \in \mathcal{J},\ n \in \mathcal{N},\ c, c' \in \mathcal{C},\ c' \neq c,\ i \in \mathcal{I}_{\mathrm{c}}, \\ i' \in \mathcal{I}_{\mathrm{c}'},\ n' < n,\ s < s^j,\ j' \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.6}$$

   Equation (B.2.6) first identifies the cast and units in previous processing stages in event n (i.e., $z_{c(i),j',n} = 1$). Then, it identifies the ladles processed in past events n' in unit j' (i.e., $z_{c'(i'),j',n'} = 1$).

   The bilinear term is linearized using the an auxiliary variable $u_{j',n,n',c(i),c'(i')}$ and the following set of constraints

$$\left.\begin{array}{l} u_{j',n,n',c(i),c'(i')} \leq z_{c(i),j',n} \\ u_{j',n,n',c(i),c'(i')} \leq z_{c'(i'),j',n'} \\ u_{j',n,n',c(i),c'(i')} \geq z_{c(i),j',n} + z_{c'(i'),j',n'} - 1 \\ u_{j',n,n',c(i),c'(i')} \in \{0, 1\} \end{array}\right\} \begin{array}{l} \forall\ n \in \mathcal{N},\ c, c' \in \mathcal{C},\ c' \neq c, \\ i \in \mathcal{I}_{\mathrm{c}},\ i' \in \mathcal{I}_{\mathrm{c}'},\ n' < n,\ s < s^j, \\ j' \in \mathcal{J}_s \cap \mathcal{J}_{c(i)} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.7}$$

3. For uncertain parameters inferred using the NIS policy in processing unit j

$$\left.\begin{array}{l} t^{1,\mathrm{s}}_{j,n,i',j',n'} \leq M_1 z_{c'(i'),j',n'} \\ t^{1,\mathrm{s}}_{j,n,i',j',n'} \geq -M_1 z_{c'(i'),j',n'} \end{array}\right\} \begin{array}{l} \forall j \in \mathcal{J},\ n \in \mathcal{N},\ n' \leq n - 1 + \eta,\ i' \in \mathcal{I}_{\mathrm{c}'},\ 1 \leq k \leq L_{c'} - 1, \\ i' \leq l^{\text{first}}_{c(i)} + L'_c - k - 1,\ j' \in \mathcal{J}_{s+k} \cap \mathcal{J}_{c'(i')} \cap \mathcal{J}^{\mathrm{p}} \end{array} \tag{B.2.8}$$

B.3 Computational parameters for the case studies

The production orders describing the ladle to cast assignment and the ladles' processing time in each unit for the 3-cast, 5-cast and 8-cast instances are shown in Table B.3.1. The 3-cast and 5-cast instances correspond to the first 3 and 5 casts respectively. For the process structure with no parallel units, EAF1, AOD1 and CC1 are only used.

Table B.3.1: Production order for the steel-making and continuous-casting problem in Li, Xiao, Tang & Floudas (2012) for the 3-cast, 5-cast and 8-cast instances.

Cast	Ladle		Processing units							
	Number	Index	EAF1	EAF2	AOD1	AOD2	LF1	LF2	CC1	CC2
1	1	1	70	75	75	80	15	15	56	70
2	3	2-4	70	75	80	75	15	15	51	65
3	4	5-8	70	75	75	80	15	15	56	64
4	5	9-13	70	75	75	80	15	15	56	68
5	5	14-18	70	75	75	70	15	15	70	60
6	1	19	70	75	75	80	15	15	56	62
7	2	20-21	70	75	75	70	15	15	56	63
8	3	22-24	70	75	75	70	15	15	56	65

Appendix C

Chapter 4 supplementary materials

C.1 Nomenclature

General

	Definitions
Parameters	
$\boldsymbol{\xi}$	Original uncertain primitive vector
Ξ	Original uncertainty set
$\boldsymbol{\xi}^k$	Uncertain vector in layer k (note, $\boldsymbol{\xi}^0 \equiv \boldsymbol{\xi}$)
$\boldsymbol{\alpha}^k$	Lifted uncertain vector in layer k
Ξ^{deep}	Overall deep lifted uncertainty set
Ξ_k^{deep}	Deep lifted uncertainty set in layer k (note, $\Xi_0^{\text{deep}} \equiv \Xi$)
$\mathbf{W}^k$	Weight matrix in layer k
$\mathbf{z}^k$	Breakpoints vector in layer k for $\boldsymbol{\xi}^k$
$\mathbf{Q}^k(\mathbf{z}^k)$	Matrix of the polyhedral lifted uncertainty set in layer k
$\mathbf{p}^k(\mathbf{z}^k)$	Right-hand-side-vector of the polyhedral lifted uncertainty set in layer k
n^{node}	Number of nodes in each layer
n^{brkp}	Number of breakpoints at each node
$w^{\text{lb}}, w^{\text{ub}}$	Lower and upper bounds of weight values
P_s	Probability of a discrete scenario s

Transportation model

	Definitions
Sets	
$\mathcal{I}$	Set of suppliers
$\mathcal{J}$	Set of consumers
Variables	
x_i	Produced units by supplier i
$y_{i,j}(\boldsymbol{\xi})$	Distributed units from supplier i to consumer j
Parameters	
$R_{i,j}$	Revenue of a produced unit by supplier i from consumer j

$T_{i,j}$	Per unit distribution cost along (i,j) arc
C_i	Per unit production cost by supplier i
$D_j(\boldsymbol{\xi})$	Uncertain consumer demand j

Airlift operations scheduling model

	Definitions
Sets	
$\mathcal{I}$	Set of aircraft types
$\mathcal{J}$	Set of routes
Variables	
F_i	Maximum number of flight hours by aircraft type i
$x_{i,j}^{\text{orig}}$	Number of flights originally planned for route j using aircraft type i
$x_{i,j,k}(\boldsymbol{\xi})$	Number of flights planned for switched from route j to k using aircraft type i
$y_j^{\text{com}}(\boldsymbol{\xi})$	Commercially contracted capacity for route j
$y_j^{\text{emp}}(\boldsymbol{\xi})$	Unused capacity for route j
Parameters	
ξ_j	Uncertain capacity of route j
$c_{i,j}$	Cost per flight for route j using aircraft type i
$c_{i,j,k}$	Increase in cost per flight when switching from route j to k using aircraft type i
$a_{i,j}$	Flying hours per flight for route j using aircraft type i
$a_{i,j,k}$	Increase in flying hours when switching from route j to k using aircraft type i
c_j^{com}	Per unit cost of commercially contracted capacity for route j
p_j	Per unit penalty of unused capacity for route j
s_j	Scaling factor of the uncertain parameter ξ_j

C.2 Tractable robust counterpart constraints derivation

The affine stochastic counterpart of the two-stage transportation problem given by (4.4) is constructed by deriving the robust counterpart of each semi-infinite constraint satisfied for all $\boldsymbol{\xi}$. As an illustration, the procedure of deriving the tractable counterpart for equation (4.4b) is shown. Introducing a $\min_{\boldsymbol{\xi}}(\cdot)$ operator does not affect the solution. When equation (C.2.1) is satisfied, it follows that equation (4.4b) is satisfied for all $\boldsymbol{\xi}$.

$$x_i + \left\{ \begin{array}{ll} \min\limits_{\boldsymbol{\xi}} & -\sum\limits_{j\in\mathcal{J}} \mathbf{Y}_{i,j}\boldsymbol{\xi} \\ \text{s.t.} & \boldsymbol{A\xi} \geq \boldsymbol{b} \end{array} \right\} \geq 0 \qquad \forall i \in \mathcal{I} \tag{C.2.1}$$

The dual of the inner minimization problem is derived as

$$x_i + \left\{ \begin{array}{ll} \max\limits_{\boldsymbol{\tau}_i} & \boldsymbol{b}^\top \boldsymbol{\tau}_i \\ \text{s.t.} & \boldsymbol{A}^\top \boldsymbol{\tau}_i = -\sum\limits_{j\in\mathcal{J}} \mathbf{Y}_{i,j}^\top \\ & \boldsymbol{\tau}_i \in \mathbb{R}_+^m \end{array} \right\} \geq 0 \qquad \forall i \in \mathcal{I} \tag{C.2.2}$$

where $\boldsymbol{\tau}_i$ is the dual vector and m is the number of hyperplanes in the polyhedral uncertainty set. The $\max_{\boldsymbol{\tau}_i}(\cdot)$ operator is removed without affecting the optimal solution of (4.4) and the counterpart final form is given in (C.2.3).

$$x_i + \boldsymbol{b}^\top \boldsymbol{\tau}_i \geq 0 \qquad \forall i \in \mathcal{I} \tag{C.2.3a}$$

$$\boldsymbol{A}^\top \boldsymbol{\tau}_i = -\sum_{j \in \mathcal{J}} \mathbf{Y}_{i,j}^\top \quad \forall i \in \mathcal{I} \tag{C.2.3b}$$

$$\boldsymbol{\tau}_i \in \mathbb{R}_+^m \qquad \forall i \in \mathcal{I} \tag{C.2.3c}$$

C.3 Lifted uncertainty sets representation

The representation of a lifted uncertainty set is introduced using the deep lifting network notation in section 4.4.1. A similar representation can be derived using the nomenclature in section 4.3.2, but is not shown here for brevity.

For a generic polyhedral uncertainty set at layer k, the exact convex hull of the lifted uncertainty set is found to be intractable by Georghiou et al. (2015). Alternatively, an outer approximation of the convex hull is proposed and is restated here in equation (C.3.4).

$$\begin{aligned} \bar{\Xi}'_k :&= \{\boldsymbol{\alpha}^k \in \mathbb{R}^q | \ \mathbf{F}^k \ \mathbf{R}^k \boldsymbol{\alpha}^k \geq \mathbf{g}^k, \ \boldsymbol{A}_i^k \boldsymbol{\alpha}_i^k \geq \boldsymbol{b}_i^k \quad \forall i = 1, \dots, n^{\text{node}} \ \} \\ &= \{\boldsymbol{\alpha}^k \in \mathbb{R}^q | \ \mathbf{Q}^k(\boldsymbol{z}^k)\boldsymbol{\alpha}^k \geq \mathbf{p}^k(\boldsymbol{z}^k)\} \quad \supseteq \text{conv } \Xi'_k \qquad \forall k = 1, \dots, L \end{aligned} \tag{C.3.4}$$

where $q = \sum_{i=1}^{n^{\text{node}}} r_i^k$. $\mathbf{F}^k$ and $\mathbf{g}^k$ are the matrix and right-hand-side vector defining the original polyhedral uncertainty set (i.e., for $\boldsymbol{\xi}^k = \mathbf{R}^k \boldsymbol{\alpha}^k$) at layer k, respectively. $\mathbf{Q}^k(\boldsymbol{z}^k)$ is a block diagonal matrix where the block matrices are $\mathbf{F}^k\mathbf{R}^k$ and $\boldsymbol{A}_i^k$ for all nodes i. Similarly, $\mathbf{p}^k(\boldsymbol{z}^k)$ is obtained through the vertical concatenation of $\mathbf{g}^k$ and $\boldsymbol{b}_i^k$ for all nodes i. The parameters $\boldsymbol{A}_i^k \in \mathbb{R}^{(r_i^k+1)\times(r_i^k)}$ and $\boldsymbol{b}_i^k \in \mathbb{R}^{r_i^k+1}$ define the convex hull of the parameter-independent lifted uncertainty set generated by lifting ξ_i^k. They are equal to

$$\boldsymbol{A}_i^k = \begin{bmatrix} -\frac{1}{z_{i,1}^k - l_i^k} & & & & \\ \frac{1}{z_{i,1}^k - l_i^k} & -\frac{1}{z_{i,2}^k - z_{i,1}^k} & & & \\ & \frac{1}{z_{i,2}^k - z_{i,1}^k} & \ddots & & \\ & & \ddots & -\frac{1}{z_{i,r_i^k-1}^k - z_{i,r_i^k-2}^k} & \\ & & & \frac{1}{z_{i,r_i^k-1}^k - z_{i,r_i^k-2}^k} & -\frac{1}{u_i^k - z_{i,r_i^k-1}^k} \\ & & & & \frac{1}{u_i^k - z_{i,r_i^k-1}^k} \end{bmatrix}, \quad \boldsymbol{b}_i^k = \begin{bmatrix} -\frac{z_{i,1}^k}{z_{i,1}^k - l_i^k} \\ \frac{l_i^k}{z_{i,1}^k - l_i^k} \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

C.4 Computational setting of the medium dimensional airlift operation scheduling instance

The computational setting, including cost and uncertainty related parameters, for the medium dimensional two-stage airlift operations scheduling problem is given in Table C.4.4.

Table C.4.4: Computational setting for the medium dimensional two-stage airlift operations scheduling instance.

	Flying hours per flight ($a_{i,j}$)			
i ↓, j →	1	2	3	4
1	24	14	49	24
2	49	29	14	20
3	24	20	49	14
4	32	21	38	20

	Serviced capacity per flight ($b_{i,j}$)			
i ↓, j →	1	2	3	4
1	50	20	45	32
2	20	75	20	38
3	38	35	32	75
4	36	44	33	49

	Cost per flight ($c_{i,j}$)			
i ↓, j →	1	2	3	4
1	7200	6000	6500	5200
2	5200	7200	6500	5000
3	7200	5000	5000	6500
4	6500	6000	6000	5500

	Auxiliary cost	
j	c_j^{com}	p_j
1	500	0
2	300	0
3	450	0
4	420	0

Flying hours	
F_1	3600
F_2	3600
F_3	3600
F_4	3600

	Increase in flying hours											
k	$a_{k,1,2}$	$a_{k,1,3}$	$a_{k,1,4}$	$a_{k,2,1}$	$a_{k,2,3}$	$a_{k,2,4}$	$a_{k,3,1}$	$a_{k,3,2}$	$a_{k,3,4}$	$a_{k,4,1}$	$a_{k,4,2}$	$a_{k,4,3}$
1	19	29	25	29	24	28	19	24	27	24	30	26
2	18	28	30	28	23	22	18	23	27	23	25	18
3	36	56	40	56	46	18	36	46	22	20	25	30
4	21.6	33.6	24	33.6	27.6	10.8	21.6	27.6	13.2	12	15	18

	Cost per switched flight											
k	$c_{k,1,2}$	$c_{k,1,3}$	$c_{k,1,4}$	$c_{k,2,1}$	$c_{k,2,3}$	$c_{k,2,4}$	$c_{k,3,1}$	$c_{k,3,2}$	$c_{k,3,4}$	$c_{k,4,1}$	$c_{k,4,2}$	$c_{k,4,3}$
1	7000	8200	7500	8200	7600	6000	7000	7600	8000	6000	8000	6500
2	5800	6100	7000	7200	5900	6500	8300	6400	7000	6700	7500	7000
3	5500	7100	6500	4200	6300	5500	5300	6400	7000	6000	7200	6800
4	4400	5680	5200	3360	5040	4400	4240	5120	5600	4800	5760	5440

	Beta-distribution parameters				
j	α_j	β_j	l_j^0	u_j^0	s_j
1	8	2	0	1	1250
2	6	4	0	1	2500
3	7	3	0	1	2500
4	3	7	0	1	2800

Appendix D

Chapter 6 supplementary materials

D.1 Exploiting deterministic solution information

In the sampling phase in each branching step, the parametric information obtained from a single deterministic solution is exploited to be used for the future samples. Equation (6.1) for a sample $\boldsymbol{\theta}_i$ can be reformulated to the standard linear programming problem form in equation (D.1.1) where the new decision variables $\boldsymbol{x}$ include positive slack variables to enforce equality constraints and auxiliary positive values to model free variables.

$$\min_{\boldsymbol{x} \geq 0} \quad \boldsymbol{c}^\top \boldsymbol{x} \tag{D.1.1a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} + \boldsymbol{F}\boldsymbol{\theta}_i \tag{D.1.1b}$$

Let the matrix $\boldsymbol{A}_B$ consist of the columns in $\boldsymbol{A}$ corresponding to the basic variables and the vector $\boldsymbol{c}_B$ corresponds to the basic variables cost value in $\boldsymbol{c}$. The critical region (i.e., partition in the uncertainty space) definition is obtained from the optimality condition in equation (D.1.2b) and the parametric optimal cost solution is given in equation (D.1.2a).

$$z(\boldsymbol{\theta}) = \quad \boldsymbol{c}_B \boldsymbol{A}_B^{-1} \mathbf{b} + \boldsymbol{c}_B \boldsymbol{A}_B^{-1} \boldsymbol{F}\boldsymbol{\theta} \tag{D.1.2a}$$

$$\text{s.t.} \quad \boldsymbol{A}_B^{-1} \boldsymbol{F}\boldsymbol{\theta} \geq -\boldsymbol{A}_B^{-1} \boldsymbol{b} \tag{D.1.2b}$$

D.2 Branching dimension heuristic

The dimension at which we axial branch is defined as the dimension that is closest to the midpoint of its bounds. Let $\boldsymbol{\theta}^{\text{branch}}$ be the branching sample, $\bar{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{\theta}}$ be the upper and lower uncertainty bounds , respectively. The branching dimension is the minimum value index of the vector $\boldsymbol{r}$ given in equation (D.2.3). For example, if the index value is 3, we branch along θ_3.

$$\boldsymbol{r} = \left| \mathbf{1} - \frac{\bar{\boldsymbol{\theta}} - \boldsymbol{\theta}^{\text{branch}}}{\boldsymbol{\theta}^{\text{branch}} - \underline{\boldsymbol{\theta}}} \right| \tag{D.2.3}$$