Submitted to *Operations Research*
manuscript (Please, provide the manuscript number!)

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

Multistage Robust Mixed Integer Optimization with Adaptive Partitions

Dimitris Bertsimas, Iain Dunning
Operations Research Center, Massachusetts Institue of Technology, Cambridge, MA 02139, dbertsim@mit.edu, idunning@mit.edu

We present a new method for multistage adaptive mixed integer optimization (AMIO) problems that extends previous work on finite adaptability. The approach analyzes the optimal solution to a non-adaptive version of an AMIO problem to gain insight into which regions of the uncertainty set are restricting the objective function value. We use this information to construct partitions in the uncertainty set, leading to a finitely adaptable formulation of the AMIO problem. We repeat this process iteratively to further improve the objective until appropriate termination criteria are reached. We provide theoretical motivation for this method, and fully detail how to apply finite adaptability to multistage AMIO problems in a way that respects the natural nonanticipativity constraints. We provide lower and upper bounds on the solution quality that have implications for computational tractability. Finally we demonstrate in computational experiments that the method can provide substantial improvements over a non-adaptive solution and existing methods. In particular, we find that our method delivers high-quality solutions even as the problem scales in both number of time stages and in the number of decision variables.

Key words: adaptive optimization, robust optimization, integer optimization
History: This paper was first submitted on November 22, 2014.

1. Introduction

Robust optimization is a methodology for addressing uncertainty in optimization problems where *uncertain parameters* are modeled as belonging to *uncertainty sets*, which contrasts with the more traditional approach of modeling uncertainty with probability distributions. We solve robust optimization (RO) problems by optimizing with respect to the worst-case realization of the uncertain parameters over the uncertainty set Ξ,

$$\begin{aligned} \max_{\mathbf{x}} \ \min_{\boldsymbol{\xi}\in\Xi} \ & c(\boldsymbol{\xi},\mathbf{x}) \\ \text{subject to } & \mathbf{g}(\boldsymbol{\xi},\mathbf{x}) \le \mathbf{0} \qquad \forall \boldsymbol{\xi}\in\Xi \\ & \mathbf{x}\in\mathcal{X}, \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the vector of decision variables and $\boldsymbol{\xi}$ is a vector representing the uncertain parameters. The optimization problem (1) typically has an infinite number of constraints but is tractable for many uncertainty sets of interest by reformulating as a deterministic optimization problem of finite size (see, e.g. Bertsimas et al. (2011a), Ben-Tal et al. (2009) for a survey). Another approach is to only generate constraints as needed until all uncertain constraints are satisfied (e.g. Fischetti and Monaci (2012)), in the same spirit as other constraint generation methods like Benders' decomposition.

RO was introduced in the context of single-stage problems as a way to address uncertainty in problem data. Since then it has been shown to be a more general methodology to address multistage optimization problems, where the uncertain parameters are revealed over time. In most multistage problems our future *recourse* decisions can be made with knowledge of at least some of the values of the uncertain parameters - those realized by the time those decisions must be made. If we place no restrictions on the functional relationship between uncertain parameters and recourse decisions, which is the ideal case, then the resulting problem is generally intractable (Ben-Tal et al. 2004) (although there has been some success for specific problems, e.g. Bertsimas et al. (2013)). The goal then of adaptive optimization (AO) is to provide methods to approximate and approach this fully adaptable ideal and to quantify, if possible, how close our approximations are.

Perhaps the most popular approach in the AO literature is to restrict the recourse decisions to be simple functions of the uncertain parameters. Affine adaptability, also known as linear decision rules, was introduced to the RO setting in Ben-Tal et al. (2004) and has proven to be useful in a wide variety of settings including control problems (Goulart et al. 2006), supply chain problems (Ben-Tal et al. 2005), and project management (Chen et al. 2007). The choice of these linear decision rules is primarily one of computational practicality with no real expectation that they will be optimal (where optimality would be finding an affinely adaptive policy that performs as well as the fully adaptive policy). Many of the papers just cited demonstrate these shortcomings with both toy problems and more realistic experiments, with more in-depth characterizations provided in particular settings, e.g. Bertsimas et al. (2010), Bertsimas and Goyal (2012). More complicated approaches trade computational efficiency for optimality by using polynomial adaptability (Bertsimas et al. 2011b) and deflected linear decision rules (Chen et al. 2008). A major shortcoming of all these approaches is that they do not allow for discrete recourse decisions, significantly reducing their modeling power for adaptive mixed-integer optimization (AMIO) problems. This has been partially addressed recently by Bertsimas and Georghiou (2013) which proposes a cutting-plane method that allows for piecewise linear decision rules for continuous recourse decisions and piecewise constant decision rules for integer recourse decisions. This approach was shown to be able to solve small problems but has significant trouble with fractionality of the solution as the number of time stages and pieces grows. A recent extension to this work in Bertsimas and Georghiou (2014) for the case of binary recourse decisions that allows for a reformulation demonstrates improved performance on multistage problems.

An alternative approach is finite adaptability, where instead of focusing on the functional form of the recourse decisions we instead partition the uncertainty set and have different recourse decisions for each partition. This can be viewed as modeling the recourse decisions as piecewise constant functions of the uncertain parameters, with the domain of each piece (or equivalently the uncertainty set partitioning) either fixed, or decided as part of the optimization problem. One of the key benefits of this approach is that it handles discrete recourse decisions naturally, suggesting it is a good choice for AMIO. The quality of solutions obtained with a finite adaptability approach rests entirely on how the partitions are selected. For example, in Vayanos et al. (2011) the uncertainty set is partitioned ahead of time using hyper-rectangles. A bilinear optimization problem that decides the best two-partition recourse decision is proposed in Bertsimas and Caramanis (2010), and the gap between a fully adaptive solution and a finitely adaptive solution is bounded. Finally Hanasusanto et al. (2014) propose a general methodology for obtaining K-partition recourse decisions using an arbitrarily accurate mixed-integer approximation. They too can solve small two-stage AMIO problems but are negatively affected by the degree to which the problem structure is affected (in particular, the use of fractional big-M constraints). While this recent work represents great progress in AMIO, the size of AMIO problems that can be solved remains small and lags significantly behind the size of tractable continuous AO problems, particularly in the multistage case.

While finalizing our paper we became of aware of a recent paper (Postek and Den Hertog 2014) that presents an iterative finite partitioning approach similar to the idea we present in this paper. They too make the observation that it is not necessary to directly optimize the specific partitions, but that instead we can pick partitions heuristically and still realize significant improves over the static case. However, their method of constructing the partitions differs substantially from the method we present in this paper, as we detail in Section 2. We compare the computational aspects of the two approaches in Section 5.

In this paper we will consider multistage AMIO problems of the general form

$$\begin{aligned} z_{full} = \min_{\mathbf{x}} \max_{\boldsymbol{\xi}\in\Xi} & \sum_{t=1}^{T} \mathbf{c}^t(\boldsymbol{\xi}) \cdot \mathbf{x}^t\left(\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}\right) \\ \text{subject to} \quad & \sum_{t=1}^{T} \mathbf{A}^t(\boldsymbol{\xi}) \cdot \mathbf{x}^t\left(\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}\right) \le \mathbf{b}(\boldsymbol{\xi}) \quad \forall \boldsymbol{\xi} = \left(\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^T\right) \in \Xi \\ & \mathbf{x} \in \mathcal{X}, \end{aligned} \tag{2}$$

where $\boldsymbol{\xi}$ represents the uncertain parameters, $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$ define linear constraints and the objective respectively (all functions of $\boldsymbol{\xi}$), and $\mathbf{x}^t$ is the vector of decision variables for stage t, which can be a mixture of discrete and continuous variables as captured by the deterministic set $\mathcal{X}$. For $t=1$ we note that $\mathbf{x}^1$ is a non-adaptive here-and-now decision that does not depend on $\boldsymbol{\xi}$.

The key contribution of this paper is a new method for solving the multistage AMIO problem (2) that scales better to larger instances than alternative methods. We demonstrate this with computational evidence that our method produces high quality solutions in a reasonable time. The method builds on the basic finite adaptability approach to AMIO that utilizes the information obtained about the (possibly partitioned) uncertainty set when solving an AO problem to select a new partitioning scheme. This process of solving and updating the partitioning can be repeated until a termination criterion in terms of solution quality or computation time is reached. We note that affine adaptability can be incorporated into our approach, and that it contrasts with approaches that determine the optimal partitions simultaneously with the values of the decisions $\mathbf{x}$ (Bertsimas and Caramanis 2010, Hanasusanto et al. 2014) and methods that select a partitioning *a priori* (Vayanos et al. 2011). Finally, we give a detailed treatment to the interactions between finite adaptability and the need to enforce *non-anticipativity* - the natural notion that a decision at time t must be made without certain knowledge of uncertain parameters that will be revealed after time t.

We have structured the paper as follows:

- In Section 2, we present a theoretical result that demonstrates a property that a new partitioning of an uncertainty set must have to improve a solution of an AMIO over an existing partitioning. We use this result to propose a method of constructing partitions using the solution of AMIO problems that is inspired by Voronoi diagrams. We detail two variants of this method for the two-stage case that have different computational and theoretical properties and provide guidance on practical implementations of both variants.

- In Section 3, we generalize both variants to the multistage case, and in particular discuss in depth when nonanticipativity constraints must be enforced.

- In Section 4, we provide theoretical results for three bounds. First, we demonstrate a lower bound that improves with additional iterations of our method and enables a gap-based termination criterion. Second, we demonstrate that we can provide an upper bound (and feasible solution) for subsequent iterations of the variants of the methods, and discuss the implications for tractability of our method. Third, we demonstrate a bound on the most improvement possible from iteration-to-iteration for our method, which can aid tractability and also suggest whether we should terminate early.

- In Section 5, we provide results of computational experiments that show that our method is practical and can solve problems that were previously unsolvable by other methods. In particular, we solve instances of a capital budgeting problem Hanasusanto et al. (2014), a multistage lot sizing problem Bertsimas and Georghiou (2013), and a multistage lot sizing problem in a network setting.
- In the final Section 6, we provide some concluding remarks.

Notation

We use lowercase, non-bold face symbols for scalar quantities (e.g., $z \in \mathbb{R}$) and a bold face for vectors (e.g., $\mathbf{x} \in \mathbb{R}^n$). Matrices are denoted by bold face uppercase symbols, e.g., $\mathbf{A} \in \mathbb{R}^{m \times n}$. We use Ξ to denote an uncertainty set and $\boldsymbol{\xi}$ to denote uncertain parameters. Our uncertain parameters are typically a vector-of-vectors, i.e.,

$$\boldsymbol{\xi} = \left[\boldsymbol{\xi}^1, \dots, \boldsymbol{\xi}^t, \dots, \boldsymbol{\xi}^T\right],$$

where $\boldsymbol{\xi}^t$ is the vector of uncertain parameters for stage t. The jth component of the uncertain parameters for time stage t is expressed with a subscript, e.g., ξ^t_j. We also often refer to *samples*, or particular realizations of uncertain parameters, from the uncertainty set. These are denoted with a hat and may be indexed by subscripts. For example $\hat{\boldsymbol{\xi}}_i$ is the ith sample, $\hat{\boldsymbol{\xi}}^t_i$ is the vector of uncertain parameters for sample i for stage t. Finally we occasionally need to refer to a particular component of $\hat{\boldsymbol{\xi}}^t_i$, for which we employ a second subscript: the jth component of the vector $\hat{\boldsymbol{\xi}}^t_i$ is the scalar $\hat{\xi}^t_{i,j}$.

2. Iterative improvement method for adapting partitions

In this section, we describe an iterative improvement method for a finite adaptability approach to two-stage AMIO. In particular, we consider the problem

$$\min_{\mathbf{x},z} z \tag{3}$$

$$
\begin{aligned}
\text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{c}^2(\boldsymbol{\xi}) \cdot \mathbf{x}^2(\boldsymbol{\xi}) \le z && \forall \boldsymbol{\xi} \in \Xi \\
& \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}^2(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi} \in \Xi,\ i \in \{1,\ldots,m\} \\
& \mathbf{x} \in \mathcal{X},
\end{aligned}
$$

which has m linear uncertain constraints and the objective expressed in epigraph form. This is a special case of the multistage problem (2) where $T = 2$. We first demonstrate some properties of AO solutions that motivate the design of our proposed method, then discuss algorithmic and implementation details.

2.1. Cutting planes and Voronoi diagrams

Two main approaches exist in the literature to solve RO problems. The first uses duality theory to reformulate the robust problem into a deterministic problem of finite size, and is by far the most commonly discussed approach in the literature. The use of reformulation introduces new variables and constraints and can change the problem class: for example, a robust linear problem with an ellipsoidal uncertainty set will become a second-order cone problem (Bertsimas et al. 2011a, Ben-Tal et al. 2009). The other approach, which we refer to as the cutting plane method, iteratively solves a deterministic relaxation of the RO problem with a finite subset of the possible constraints (the *master problem*) and adds violated constraints from the full RO problem as needed until the solution of the master problem is feasible with respect to all uncertain parameters in the uncertainty set. It has been shown (Bertsimas et al. 2014, Fischetti and Monaci 2012) that one approach does not dominate the other in performance and the best approach for a given problem depends on the uncertainty set, the structure of the uncertainty in the problem, and the problem class. For example, a purely binary robust optimization problem will remain purely binary using a cutting plane method but will become a mixed-integer optimization problem using reformulation, with possible impacts on the fractionality of relaxed solutions. On the other hand, for some problems a very large number of constraints may need to be added to reach feasibility with respect to the uncertain parameters, which is not only time-consuming but may negatively affect the sparsity patterns of a problem. The method described in this paper works with both reformulation and the cutting plane method, but we will use the cutting plane method to aid the exposition.

Consider the state of the cutting plane method's master problem at optimality. For every uncertain constraint in the original RO problem we may have many generated constraints in the master problem, each of which corresponds to a realization of the uncertain parameters $\boldsymbol{\xi} \in \Xi$. Each of these generated constraints will have some slack $s \geq 0$, with some possibly having no slack (the constraint is said to be active or tight). For each set of generated constraints we define the active uncertain parameters (or active samples) as the value of the uncertain parameters for the constraint with minimum slack (selecting arbitrarily if there a multiple uncertain parameters with the same slack). If we were to remove all generated constraints that correspond to uncertain parameters that are not active at optimality, we are left with a reduced set of constraints that is sufficient to constrain the solution of the master problem to be no better than the solution to the original RO problem. These remaining constraints, and the values of the active uncertain parameters that give rise to them, offer an insight into how the uncertainty set structure relates to the optimal solution of the RO problem.

We now to turn to finite adaptability (e.g., Bertsimas and Caramanis (2010)), which is a computationally tractable way of allowing wait-and-see decisions to depend on the uncertain parameters. It can be viewed as a restriction of the decisions $\mathbf{x}^2(\boldsymbol{\xi})$ in (3) to the class of piecewise constant policies, i.e.

$$\mathbf{x}^2(\boldsymbol{\xi}) = \begin{cases} \mathbf{x}^2_1, & \boldsymbol{\xi} \in \Xi_1, \\ \vdots & \\ \mathbf{x}^2_K, & \boldsymbol{\xi} \in \Xi_K, \end{cases}$$

where $\Xi_k, k \in \{1, \ldots, K\}$ defines a partitioning of Ξ and $\mathbf{x}^2_k$ is the value of $\mathbf{x}^2$ if the realized value of $\boldsymbol{\xi}$ is in the partition Ξ_k. In the limiting case of many small partitions we approach the fully adaptable solution, but this is unlikely to be computationally tractable due to the resulting increase in problem size. Our method then is primarily addressing the question of how to select a partitioning scheme in such a way as to deliver an improvement of the solution without sacrificing too much computational tractability.

If we solve a static (non-adaptive) version of the two-stage problem (3), i.e.,

$$
\begin{aligned}
z_{static} = \min_{\mathbf{x},z} \ & z \\
\text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2 \leq z \\
& \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}^2 \leq b_i(\boldsymbol{\xi}) \quad \forall \boldsymbol{\xi} \in \Xi,\ i \in \{1,\dots,m\} \\
& \mathbf{x} \in \mathcal{X},
\end{aligned} \tag{4}
$$

with the cutting-plane method then we can define $\hat{\Xi}$ to be the set of active uncertain parameters $\hat{\boldsymbol{\xi}}$ from the uncertainty set Ξ that correspond to the generated constraints with minimum slack in the master problem. If we were to now arbitrarily partition the uncertainty set into two sets Ξ_1 and Ξ_2, we could construct a finitely adaptable approximation to (3), i.e.,

$$
\begin{aligned}
z_{partition} = \min_{\mathbf{x},z} \ & z \\
\text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}_1^2 \leq z & \forall \boldsymbol{\xi} \in \Xi_1 \\
& \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}_2^2 \leq z & \forall \boldsymbol{\xi} \in \Xi_2 \\
& \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}_1^2 \leq b_i(\boldsymbol{\xi}) & \forall \boldsymbol{\xi} \in \Xi_1,\ i \in \{1,\dots,m\} \\
& \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}_2^2 \leq b_i(\boldsymbol{\xi}) & \forall \boldsymbol{\xi} \in \Xi_2,\ i \in \{1,\dots,m\} \\
& \mathbf{x} \in \mathcal{X},
\end{aligned}
$$

which we would hope improves over the static problem soultion. We now consider what properties Ξ_1 and Ξ_2 need to have such that this improvement is possible.

THEOREM 1. *If $\hat{\Xi}$, the set of active uncertain parameters for the static problem, is either $\hat{\Xi} \subset \Xi_1$ or $\hat{\Xi} \subset \Xi_2$ then $z^*_{partition} = z^*_{static}$. Otherwise $z^*_{partition} \leq z^*_{static}$.*

Proof Consider the problem

$$\begin{aligned} z_{halfpart} = \min_{\mathbf{x},z} \; & z \\ \text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}_1^2 \le z && \forall \boldsymbol{\xi} \in \Xi_1 \\ & \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}_1^2 \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi} \in \Xi_1, \; i \in \{1,\ldots,m\} \\ & \mathbf{x} \in \mathcal{X}, \end{aligned} \tag{5}$$

and suppose that the set of active uncertain parameters $\hat{\Xi}$ is a subset of Ξ_1. This means that $z_{halfpart} = z_{static}$, as every deterministic constraint with minimal slack in (4) will also be a valid constraint for (5), so no improvement is possible. The equivalent problem to (5) for the other partition Ξ_2 may have an objective function value better than z_{static}, but as $z_{partition}$ is limited by the worst objective function value of the two partitions there will be no improvement of objective function value overall (i.e., $z_{partition} = z_{static}$). On the other hand, if we now suppose that $\hat{\Xi} \not\subset \Xi_1$ and $\hat{\Xi} \not\subset \Xi_2$ then the possibility of improvement exists as the solution to (5) is not restricted to be the same solution as (4).

The above result shows we must partition the uncertainty set in such a way as to guarantee the uncertain parameters $\hat{\boldsymbol{\xi}}$ for the active constraints do not all lie in one partition, at the very least. This naturally leads us to consider a partitioning where every $\hat{\boldsymbol{\xi}}$ lies in its own partition. There are many ways to do this, but an efficiently computable and generalizeable method is to use *Voronoi diagrams* (Aurenhammer 1991). Given a set of N points $\hat{\boldsymbol{\xi}}_1, \ldots, \hat{\boldsymbol{\xi}}_N \in \Xi$ the Voronoi diagram associated these points defines a partition for each $\hat{\boldsymbol{\xi}}_i$ that contains only the $\boldsymbol{\xi} \in \Xi$ such that the Euclidean distance between $\hat{\boldsymbol{\xi}}_i$ and $\boldsymbol{\xi}$ is less than or equal to the distance to any other given point $\hat{\boldsymbol{\xi}}_j$.

We can express the partition induced by $\hat{\boldsymbol{\xi}}_i$ as the set

$$\Xi\left(\hat{\boldsymbol{\xi}}_i\right) = \Xi \cap \left\{ \boldsymbol{\xi} \,\middle|\, \left\|\hat{\boldsymbol{\xi}}_i - \boldsymbol{\xi}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j - \boldsymbol{\xi}\right\|_2 \quad \forall j \in \{1,\ldots,N\}, \; i \neq j \right\}$$

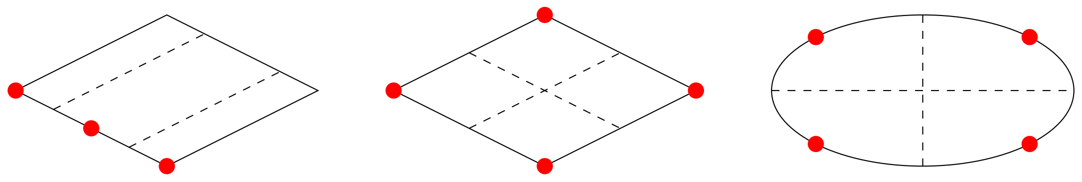

Figure 1 Voronoi diagrams embedded in simple uncertainty sets. The first two diagrams use the polyhedral uncertainty set $\Xi_P = \left\{ \xi \middle| \left\| \left[\frac{1}{2}\xi_1,\ \xi_2 \right] \right\|_1 \le 1 \right\}$, and the third uses an ellipsoidal set $\Xi_U = \left\{ \xi \middle| \left\| \left[\frac{1}{2}\xi_1,\ \xi_2 \right] \right\|_2 \le 1 \right\}$. The bold points represent the active samples $\hat{\boldsymbol{\xi}}$.

$$= \Xi \cap \left\{ \boldsymbol{\xi} \middle| \sum_k \left(\hat{\xi}_{i,k} - \xi_k \right)^2 \le \sum_k \left(\hat{\xi}_{j,k} - \xi_k \right)^2 \quad \forall j \in \{1,\dots,N\},\ i \ne j \right\}$$
$$= \Xi \cap \left\{ \boldsymbol{\xi} \middle| 2\sum_k \left(\hat{\xi}_{i,k} - \hat{\xi}_{j,k} \right) \xi_k \ge \sum_k \left(\left(\hat{\xi}_{i,k} \right)^2 - \left(\hat{\xi}_{j,k} \right)^2 \right) \quad \forall j \in \{1,\dots,N\},\ i \ne j \right\},$$

which is a set defined by Ξ and $N-1$ additional linear constraints (and if Ξ is polyhedral then $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ will also be polyhedral). In Figure 1 we demonstrate the partitions induced in a polyhedral uncertainty set for two different sets of points, as well as an ellipsoidal set. When Voronoi diagrams are discussed in the literature the focus is often on the computational complexity of enumerating all extreme points of the partitions of the Voronoi diagram which becomes increasingly difficult in higher dimensions, with asymptotic complexity of $\mathcal{O}\left(N^{\lceil d/2 \rceil}\right)$ where d is the dimension. This may suggest that they are not appropriate for our needs, but we note that in our application we are generating new constraints by optimizing affine functions of $\boldsymbol{\xi}$ over a partition $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$, for which we do not need to emumerate all the extreme points of $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$. Computational experience in the literature with linear optimization thus suggests that, if Ξ is polyhedral, this is a substantially more tractable computation than enumeration of extreme points.

After an initial solve of an AO problen that uses finite adaptability with the Voronoi diagram partitions, we will have a new set of active uncertain parameters which can be used to improve the solution further. We propose two variants for iteratively partitioning which we analyse with both theoretical bounds and computational results. The first variant we propose, the *non-nested* variant, uses all active samples $\hat{\boldsymbol{\xi}}$ obtained so far to construct a new partitioning scheme from scratch at each iteration of the method (Section 2.2). We will refer to the second variant as the *nested* variant which, unlike the non-nested variant, retains an association between samples $\hat{\boldsymbol{\xi}}$ and the partition they were obtained from (Section 2.3). We use this information to create *nested* partitions inside the partitions formed at the previous iteration. We will describe the non-nested variant first as it is more conceptually and notationally simpler, before moving to the nested variant. Additionally we restrict ourselves to the two-stage case of (3) for simplicity of exposition, but we generalize the results to the multistage case in Section 3.

2.2. Non-nested Variant

We now present the first of the two variants of our method to iteratively improve the partitions in a finite adaptability approach to two-stage AMIO using the understanding developed above. One of the key operations in both variants is to determine the set of active samples $\hat{\Xi}$ from the solution of a RO problem. Given a solution $(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2)$ and a set $\bar{\Xi}$ we can define the set of active samples as

$$\hat{\Xi}\left(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \bar{\Xi}\right) = \left\{ \hat{\boldsymbol{\xi}} \,\middle|\, \hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi} \in \bar{\Xi}} \left\{ b_i(\boldsymbol{\xi}) - \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \bar{\mathbf{x}}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \bar{\mathbf{x}}^2 \right\} \quad \forall i \in \{1, \dots, m\} \right\}$$
$$\cup \left\{ \hat{\boldsymbol{\xi}} \,\middle|\, \hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi} \in \bar{\Xi}} \left\{ \bar{z} - \mathbf{c}^1(\boldsymbol{\xi}) \cdot \bar{\mathbf{x}}^1 - \mathbf{c}^2(\boldsymbol{\xi}) \cdot \bar{\mathbf{x}}^2 \right\} \right\}.$$

In the non-nested variant we create a partition for every sample seen at any previous iteration, thus we simply maintain a set $\mathcal{F}^k$ of all samples observed up to but not including iteration k.

ALGORITHM 1. Non-nested variant of iterative partitioning method.

1. **Initialization**. Define $\mathcal{F}^1$ to be the initially empty set of samples, and set iteration counter $k \leftarrow 1$.

2. **Initial Solve**. Solve the following unpartitioned, non-adaptive version of (3)

$$\begin{aligned} z_{nonnested}\left(\mathcal{F}^1\right) = \min_{\mathbf{x}, z} \;& z \\ \text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{c}^2(\boldsymbol{\xi}) \cdot \mathbf{x}^2 \le z && \forall \boldsymbol{\xi} \in \Xi \\ & \mathbf{a}_i^1(\boldsymbol{\xi}) \cdot \mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi}) \cdot \mathbf{x}^2 \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi} \in \Xi,\ i \in \{1, \dots, m\} \\ & \mathbf{x} \in \mathcal{X}, \end{aligned}$$

with solution $(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2)$. Set $\mathcal{F}^2 \leftarrow \hat{\Xi}(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \Xi)$ and $k \leftarrow 2$.

3. **Partitioned Problem**. Solve the following partitioned version of (3), with one partition for every $\hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k$:

$$\begin{aligned}
z_{nonnested}\left(\mathcal{F}^k\right) = \min_{\mathbf{x},z} \;\; & z && (6)\\
\text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{c}^2(\boldsymbol{\xi})\cdot\mathbf{x}_j^2 \le z && \forall \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall\hat{\boldsymbol{\xi}}_j\in\mathcal{F}^k\\
& \mathbf{a}_i^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{a}_i^2(\boldsymbol{\xi})\cdot\mathbf{x}_j^2(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall\hat{\boldsymbol{\xi}}_j\in\mathcal{F}^k,\ i\in\{1,\dots,m\}\\
& \mathbf{x}\in\mathcal{X}
\end{aligned}$$

with solution $(\bar{z},\bar{\mathbf{x}}^1,\bar{\mathbf{x}}_1^2,\bar{\mathbf{x}}_2^2,\dots)$, where

$$\Xi\left(\hat{\boldsymbol{\xi}}_i\right) = \Xi \cap \left\{\boldsymbol{\xi} \;\middle|\; \left\|\hat{\boldsymbol{\xi}}_i - \boldsymbol{\xi}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j - \boldsymbol{\xi}\right\|_2 \quad \forall\hat{\boldsymbol{\xi}}_j\in\mathcal{F}^k\right\}$$

and $\mathbf{x}_j^2$ is the set of recourse decisions corresponding to the partition induced by sample $\hat{\boldsymbol{\xi}}_j$.

4. **Grow set**. Let

$$\mathcal{F}^{k+1} \leftarrow \mathcal{F}^k \cup \left\{\bigcup_j \hat{\Xi}\left(\bar{z},\bar{\mathbf{x}}^1,\bar{\mathbf{x}}_j^2,\Xi\left(\hat{\boldsymbol{\xi}}_j\right)\right)\right\}$$

and $k \leftarrow k+1$. Terminate if termination criteria are reached otherwise and go to 3.

We will defer discussion of termination criteria until Section 4. We will demonstrate this method with a simple example problem.

Example 1. Two-stage inventory problem, non-nested variant. In this problem, we must order stock to meet future, unknown demand. There are three ways to order stock: ordering any amount now at a unit cost of 50, ordering a fixed lot of size 25 at a unit cost of 60, and ordering a fixed lot of size 25 at a unit cost of 75. Only one lot of each size may be ordered, but the decision to order them can be made after the demand is known. We must meet all the demand, and we pay a unit cost of 65 for any stock left over. We formulate this as the two-stage AMIO problem

$$\begin{aligned}
\min\ & z && (7)\\
\text{subject to } & 50x^1 + 65I^2(\xi) + 1500y_A^2(\xi) + 1875y_B^2(\xi) \le z && \forall\xi\in\Xi\\
& I^2(\xi) \ge 0 && \forall\xi\in\Xi\\
& x^1 \ge 0 &&\\
& y_A^2(\xi),\ y_B^2(\xi) \in \{0,1\} && \forall\xi\in\Xi
\end{aligned}$$

where x^1 is the here-and-now continuous ordering decision, $y_A^2(\xi)$ and $y_B^2(\xi)$ are the wait-and-see binary ordering decisions, $I^2(\xi) = x^1 - \xi + 25y_A^2(\xi) + 25y_B^2(\xi)$, and ξ is the uncertain demand that is drawn from the uncertainty set $\Xi = \{\xi \,|5 \leq \xi \leq 95\}$. We solve (7) intially with no partitions (step 2) to obtain the solution $z = 10600$, $x^1 = 95$, $y_A^2 = 0$, $y_B^2 = 0$. The worst-case for the objective is when the demand is low ($\hat{\xi} = 5$, leaving 90 units of stock) and the worst-case for the non-negative inventory constraint is when the demand is high ($\hat{\xi} = 95$, leaving no stock), thus as the start of iteration 2 (step 3) we have $\mathcal{F}^2 = \left\{\hat{\xi}_1 = 5,\ \hat{\xi}_2 = 95\right\}$. We use this information to construct and solve the partitioned problem

$$\begin{aligned}
\min\ & z \\
\text{subject to}\ & 50x^1 + 65I_j^2(\xi) + 1500y_{A,j}^2(\xi) + 1875y_{B,j}^2(\xi) \leq z && \forall \xi \in \Xi\left(\hat{\xi}_j\right),\ j \in \{1,2\} \\
& I_j^2(\xi) \geq 0 && \forall \xi \in \Xi\left(\hat{\xi}_j\right),\ j \in \{1,2\} \\
& y_{A,j}^2(\xi),\ y_{B,j}^2(\xi) \in \{0,1\} && \forall \xi \in \Xi\left(\hat{\xi}_j\right),\ j \in \{1,2\} \\
& x^1 \geq 0,
\end{aligned}$$

where

$$\begin{aligned}
\Xi\left(\hat{\xi}_1 = 5\right) &= \{\xi \,|5 \leq \xi \leq 95\} \cap \{\xi \,| \|5 - \xi\|_2 \leq \|95 - \xi\|_2\} \\
&= \{\xi \,|5 \leq \xi \leq 50\}
\end{aligned}$$

and $\Xi\left(\hat{\xi}_2 = 95\right) = \{\xi \,|50 \leq \xi \leq 95\}$. The solution to this problem is $z = 7926$, $x^1 = 70$, $y_{A,1}^2 = 0$, $y_{B,1}^2 = 0$, $y_{A,2}^2 = 1$, $y_{B,2}^2 = 0$, where we have reduced the objective value to 75% of the non-adaptive solution. We now grow the set (step 4) to $\mathcal{F}^3 = \{5,95\} \cup \{5,50\} \cup \{50,95\} = \{5,50,95\}$. We solve the partitioned problem at iteration $k = 3$ and obtain a solution $z = 7575$, $x^1 = 45$, $\left(y_{A,1}^2,\ y_{B,1}^2\right) = (0,0)$ for $\hat{\xi}_1 = 5$, $\left(y_{A,2}^2,\ y_{B,2}^2\right) = (1,0)$ for $\hat{\xi}_2 = 50$, $\left(y_{A,3}^2,\ y_{B,3}^2\right) = (1,1)$ for $\hat{\xi}_3 = 95$. We can also express this solution as piecewise constant policies, that is

$$y_A^2(\xi) = \begin{cases} 0, & 5 \leq \xi < 35, \\ 1, & 35 \leq \xi \leq 95, \end{cases}$$

and

$$y_B^2(\xi) = \begin{cases} 0, & 5 \le \xi < 35, \\ 1, & 65 \le \xi \le 95. \end{cases}$$

This third iteration solution represents a reduction in objective value to 71% of the non-adaptive solution.

The above example demonstrated the non-nested variant, which was able to make a substantial improvement over a non-adaptive solution. We now describe the nested variant and will revisit this same example to contrast the two variants.

2.3. Nested Variant

The key distinction between the nested and non-nested variants is that, in the nested variant, partitions are constructed inside the partitions of the previous iteration, which we can think of as nested Voronoi diagrams. From an algorithmic standpoint this means that we need to maintain a hierarchy or a *tree* (in the computer science sense) of samples, rather than a simple set. To describe how the partitions are constructed from a sample tree $\mathcal{T}$ we require the following operations: $Leaves(\mathcal{T})$ is the set of leaves of the tree, $Children\left(\hat{\boldsymbol{\xi}}\right)$ is the set of children of the sample $\hat{\boldsymbol{\xi}}$ in the tree, $Parent\left(\hat{\boldsymbol{\xi}}\right)$ is the parent sample of sample $\hat{\boldsymbol{\xi}}$ in the tree, and $Siblings\left(\hat{\boldsymbol{\xi}}\right)$ is the set of children of the parent sample of $\hat{\boldsymbol{\xi}}$, i.e.

$$Siblings\left(\hat{\boldsymbol{\xi}}\right) = Children\left(Parent\left(\hat{\boldsymbol{\xi}}\right)\right).$$

Each leaf of $\mathcal{T}$ corresponds to a partition of the uncertainty set Ξ and each level of the tree corresponds to an iteration of the method. At each iteration we construct a partition for a leaf sample $\hat{\boldsymbol{\xi}}_i$ as an intersection of partitions

$$\begin{aligned} \Xi\left(\hat{\boldsymbol{\xi}}_i\right) = & \left\{ \boldsymbol{\xi} \,\middle|\, \left\|\hat{\boldsymbol{\xi}}_i - \boldsymbol{\xi}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j - \boldsymbol{\xi}\right\|_2 \quad \forall \hat{\boldsymbol{\xi}}_j \in Siblings\left(\hat{\boldsymbol{\xi}}_i\right) \right\} \\ & \cap \left\{ \boldsymbol{\xi} \,\middle|\, \left\|Parent\left(\hat{\boldsymbol{\xi}}_i\right) - \boldsymbol{\xi}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j - \boldsymbol{\xi}\right\|_2 \quad \forall \hat{\boldsymbol{\xi}}_j \in Siblings\left(Parent\left(\hat{\boldsymbol{\xi}}_i\right)\right) \right\} \\ & \vdots \\ & \cap \Xi, \end{aligned} \tag{8}$$

which terminates when the parent is the root node, which has no siblings and thus is equivalent to the entire uncertainty set. At each iteration we add the active samples for partition $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ as leaves of $\hat{\boldsymbol{\xi}}_i$, which subdivides $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ and has no effect on the other partitions. A visual comparison of the two variants is provided in Figure 2 to aid in understanding the difference. We now formally state the nested variant of the method:

Algorithm 2. Nested variant of iterative partitioning method.

1. **Initialization**. Define $\mathcal{T}^1$ to be the initial tree of samples, consisting of one root node (any $\hat{\boldsymbol{\xi}} \in \Xi$). Set iteration counter $k \leftarrow 1$

2. **Partitioned Problem**. Solve the following partitioned version of (3), with one partition for every $\hat{\boldsymbol{\xi}}_j \in \mathit{Leaves}\left(\mathcal{T}^k\right)$:

$$\begin{aligned} z_{nested}\left(\mathcal{T}^k\right) = \min_{\mathbf{x},z} \quad & z && (9)\\ \text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{c}^2(\boldsymbol{\xi})\cdot\mathbf{x}^2_i \le z && \forall \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall\hat{\boldsymbol{\xi}}_j \in \mathit{Leaves}\left(\mathcal{T}^k\right)\\ & \mathbf{a}^1_i(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{a}^2_i(\boldsymbol{\xi})\cdot\mathbf{x}^2_j(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall\hat{\boldsymbol{\xi}}_j \in \mathit{Leaves}\left(\mathcal{T}^k\right), i\in I\\ & \mathbf{x}\in\mathcal{X} \end{aligned}$$

with solution $(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2_1, \bar{\mathbf{x}}^2_2, \dots)$, where $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ is the set defined in equation 8, $I = \{1,\dots,m\}$ and $\mathbf{x}^2_j$ is the set of recourse decisions corresponding to the partition induced by sample $\hat{\boldsymbol{\xi}}_j$.

(a) **Grow tree**. Initialize $\mathcal{T}^{k+1} \leftarrow \mathcal{T}^k$. For every leaf $\hat{\boldsymbol{\xi}}_j \in \mathit{Leaves}\left(\mathcal{T}^{k+1}\right)$ add the samples

$$\hat{\Xi}\left(\bar{z}, \bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2_j, \Xi\left(\hat{\boldsymbol{\xi}}_j\right)\right)$$

as children of that leaf. Terminate if termination criteria are reached otherwise set $k \leftarrow k+1$ and go to 2.

We now demonstrate this variant on the same problem as example 1.

Example 2. Two-stage inventory problem with the nested variant. We consider the same problem described in 1. We will initalize our tree $\mathcal{T}^1$ with a root sample $\hat{\xi} = 50$, which means we have no partitions initially and thus the same solution as the non-nested variant at iteration 1. We grow the tree (step 3) to obtain $\mathcal{T}^2 = \{50 : \{5, 95\}\}$ and use this to construct and solve the partitioned problem for iteration 2. The partitions are $\Xi\left(\hat{\xi}_1 = 5\right) = \{\xi \,|5 \leq \xi \leq 50\}$ and $\Xi\left(\hat{\xi}_2 = 95\right) = \{\xi \,|50 \leq \xi \leq 95\}$, which are the same as in the non-nested variant and thus we again have the same solution. The key difference is that we associate the active samples for $\Xi\left(\hat{\xi}_1 = 5\right)$ and $\Xi\left(\hat{\xi}_1 = 95\right)$ with their respective partitions, leading to the tree $\mathcal{T}^3 = \{50 : \{5 : \{5, 50\}, 95 : \{50, 95\}\}\}$. We have 4 leaf samples and thus 4 partitions in iteration 3, against the non-nested variant's 3 partitions. The calculation of the partition for the leaf $\hat{\xi} = 50$ that is a child sample of $\hat{\xi} = 5$ is

$$\begin{aligned}
\Xi\left(\hat{\xi} = 50\right) &= \{\xi \,|\|50 - \xi\|_2 \leq \|5 - \xi\|_2\} \cap \{\boldsymbol{\xi} \,|\|5 - \xi\|_2 \leq \|95 - \xi\|_2\} \cap \Xi \\
&= \{\xi \,|\xi \geq 27.5\} \cap \{\xi \,|\xi \leq 50\} \cap \Xi \\
&= \{\xi \,|27.5 \leq \xi \leq 50\},
\end{aligned}$$

which we repeat for the other three leaves. When we solve the resulting partitioned problem we obtain the solution $z = 7375$, $x^1 = 47.5$, $\left(y^2_{A,1},\ y^2_{B,1}\right) = (0, 0)$, $\left(y^2_{A,2},\ y^2_{B,2}\right) = (1, 0)$, $\left(y^2_{A,3},\ y^2_{B,3}\right) = (1, 0)$, and $\left(y^2_{A,4},\ y^2_{B,4}\right) = (1, 1)$ which is a 70% reduction over the non-adaptive solution, slightly better than the non-nested variant ($z = 7575$) and slightly worse than the fully adaptive solution ($z = 7250$).

2.4. Affine Adaptability

In the introduction we discussed the popularity and success of affine adaptability, also known as linear decisions rules. We can express this type of adaptability in the context of our two-stage problem (3) with the substitution

$$\mathbf{x}^2(\boldsymbol{\xi}) = \mathbf{F}\boldsymbol{\xi} + \mathbf{g}, \tag{10}$$

where $\mathbf{F}$ and $\mathbf{g}$ are now the decision variables instead of $\mathbf{x}^2$. There are two problems with substituting (10) directly into (3). The first is that if $\mathbf{a}^2_i(\boldsymbol{\xi})$ and $\mathbf{c}^2(\boldsymbol{\xi})$ are not constant with respect to the uncertain parameters then we will have products of uncertain parameters, which is not generally tractable (Ben-Tal et al. 2004). The second is that if a variable $x^2_j(\boldsymbol{\xi})$ is integer then $\mathbf{F}_j$ must

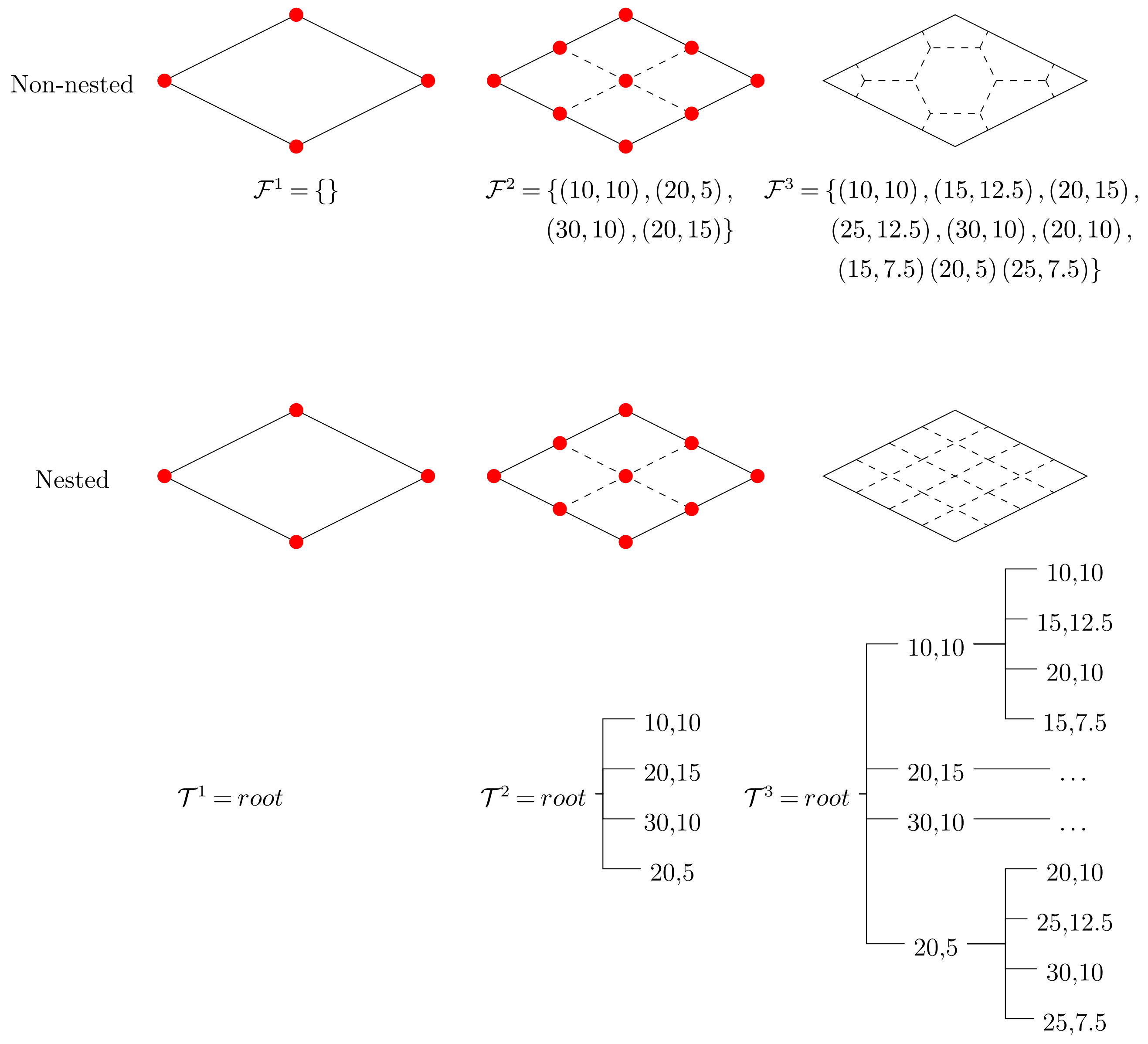

Figure 2 Comparison of the partitions and sample sets created by the non-nested variant (top) and nested variant (bottom) of the iterative partitioning method over three iterations. The uncertainty set displayed is $\Xi = \left\{ (\xi_1, \xi_2) \,\middle|\, \frac{|\xi_1 - 20|}{10} + \frac{|\xi_1 - 10|}{5} \leq 1 \right\}$, and we assume that at each iteration all the extreme points of the set/partitions are active samples. The red points denote the active samples in the solved partitioned problem for each iteration. As expected the two variants have the same partitions at iterations 1 and 2, but diverge in iteration 3.

necessarily be $\mathbf{0}^T$ and there is no adaptability. Given these problems we consider the following variant of (3)

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y},z} \quad & z && (11)\\ \text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{c}^2_x\cdot\mathbf{x}^2(\boldsymbol{\xi}) + \mathbf{c}^2_y(\boldsymbol{\xi})\cdot\mathbf{y}^2(\boldsymbol{\xi}) \le z && \forall \boldsymbol{\xi}\in\Xi\\ & \mathbf{a}^1_i(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{a}^2_{x,i}\cdot\mathbf{x}^2(\boldsymbol{\xi}) + \mathbf{a}^2_{y,i}(\boldsymbol{\xi})\cdot\mathbf{y}^2(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi}\in\Xi,\ i\in\{1,\dots,m\}\\ & \mathbf{x}^2\in\mathbb{R}^{n_x},\ \mathbf{y}\in\mathbb{Z}^{n_y}, \end{aligned}$$

where we assume that the deterministic constraints $\mathcal{X}$ are captured by the other constraints and variable type restrictions. We have restricted the continuous second stage decisions $\mathbf{x}^2$ to be certain which enables us to substitute (10) into (11) to obtain the problem

$$\begin{aligned} \min_{\mathbf{F},\mathbf{g},\mathbf{y},z} \quad & z && (12)\\ \text{subject to} \quad & \mathbf{c}^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{c}^2_x\cdot(\mathbf{F}\boldsymbol{\xi}+\mathbf{g}) + \mathbf{c}^2_y(\boldsymbol{\xi})\cdot\mathbf{y}^2(\boldsymbol{\xi}) \le z && \forall \boldsymbol{\xi}\in\Xi\\ & \mathbf{a}^1_i(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{a}^2_{x,i}\cdot(\mathbf{F}\boldsymbol{\xi}+\mathbf{g}) + \mathbf{a}^2_{y,i}(\boldsymbol{\xi})\cdot\mathbf{y}^2(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi}\in\Xi,\ i\in\{1,\dots,m\}\\ & \mathbf{F}\in\mathbb{R}^{n_x\times n_\xi},\ \mathbf{g}\in\mathbb{R}^{n_x},\ \mathbf{y}\in\mathbb{Z}^{n_y} \end{aligned}$$

which has a similar form to the original problem two-stage problem (3). The key observation with respect to our finite partitioning method is that $\mathbf{F}$ and $\mathbf{g}$ are themselves second-stage variables, so can be functions of $\boldsymbol{\xi}$. This allows use to combine finite adaptability and affine adaptability, as we can have a different affine policy $(\mathbf{F},\mathbf{g})$ for each partition. More formally, we will have the piecewise linear policy

$$\mathbf{x}^2(\boldsymbol{\xi}) = \begin{cases} \mathbf{F}_1\boldsymbol{\xi}+\mathbf{g}_2, & \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_1\right),\\ \mathbf{F}_2\boldsymbol{\xi}+\mathbf{g}_2, & \boldsymbol{\xi}\in\Xi\left(\hat{\boldsymbol{\xi}}_2\right),\\ \vdots & \vdots \end{cases}$$

for continuous second stage decisions, and the piecewise constant policy

$$\mathbf{y}^2(\boldsymbol{\xi}) = \begin{cases} \mathbf{y}^2_1, & \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_1\right), \\ \mathbf{y}^2_2, & \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_2\right), \\ \vdots & \vdots \end{cases}$$

for integer decisions. We demonstrate combining affine adaptability with finite adaptability in our mulitstage lot sizing example (Section 5.2).

2.5. Implementation Considerations

We have described two variants of our iterative partitioning method and demonstrated them on an example two-stage AMIO problem. We now present minor alterations which could be considered when implementing either variant to explore the trade-off between computation time and quality of solution. Consider re-writing (6) as

$$\begin{aligned} \min_{\mathbf{x},z,\tilde{\mathbf{z}}} \quad & z + \epsilon \mathbf{e}\cdot\tilde{\mathbf{z}} \\ \text{subject to} \quad & \tilde{z}_j \le z && \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k \\ & \mathbf{c}^1(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{c}^2(\boldsymbol{\xi})\cdot\mathbf{x}^2_j \le \tilde{z}_j && \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k \\ & \mathbf{a}^1_i(\boldsymbol{\xi})\cdot\mathbf{x}^1 + \mathbf{a}^2_i(\boldsymbol{\xi})\cdot\mathbf{x}^2_j(\boldsymbol{\xi}) \le b_i(\boldsymbol{\xi}) && \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k,\ i \in \{1,\dots,m\} \end{aligned}$$

where ϵ is a very small number and $\mathbf{e}$ is the vector of ones. This epigraph formulation creates an auxiliary variable for the objective value for each partition and attempts to force them individually as low as possible without affecting z. At optimality one or more *active partitions* will have an objective value $\tilde{z}^*_j$ equal to the objective function value z^*. If a partition is not active, further partitioning it may not deliver as much of an improvement to the objective as one that is. Therefore we the change to the method would be to only add samples to $\mathcal{F}^k$ (or $\mathcal{T}^k$) if they come from an active partition. This will reduce the number of partitions at the next iteration, improving computational efficiency. The possibility of applying this arises in Example (2) (nested variant) as at the end of iteration 3 the objectives for the four partitions $\Xi\left(\hat{\xi}_1 = 5\right)$ to $\Xi\left(\hat{\xi}_4 = 95\right)$ are 5137.5, 6800, 5337.5, and 7375 respectively, and it is simple to check that adding more partitions at the lower end of the uncertain parameter's range does not further improve the solution.

A similar alteration would be to only take active samples from generated constraints with no slack. The reasoning behind this alteration is that constraints with slack may not constrain the solution even with additional partitions, so we can reduce our partition count further. This may not be applicable in all cases: for example, a purely binary AO problem is very likely to have some slack in most of its constraints if there is fractional data in the problem.

2.6. Contrast with Alternative Partitioning Proposal

The method presented in Postek and Den Hertog (2014) shares some similarities with the algorithm we propose. Both describe an iterative partitioning method for AMIO that uses active uncertain parameters to inform which partitions to use. The key difference is that we use Voronoi diagrams to construct the partitioning, which results in multiple partitions being added at each iteration, whereas multiple heuristics are presented in Postek and Den Hertog (2014) that only add one partition at each iteration. This results in both qualitative differences in the partitioning scheme and quantitative differences, as shown in Section 5.

3. Multistage Partitioning

We now generalize our partitioning method described above from two-stage AMIO to the full multi-stage case (2). The order of events, for the sake of clarity, is that the here-and-now decision $\mathbf{x}^1$ is made, then the first stage uncertain parameters $\boldsymbol{\xi}^1$ are revealed. We make the decision $\mathbf{x}^2$ with the benefit of knowing $\boldsymbol{\xi}^1$, but no other knowledge of the uncertainty parameters except what is implied by knowing $\boldsymbol{\xi}^1$. This then proceeds until we make the decision $\mathbf{x}^T$ knowing $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^{T-1}$ and the final uncertain parameters $\boldsymbol{\xi}^T$ are revealed. Modeling these nonanticipativity restrictions - that a decision now cannot depend on exact future knowledge - is the primary complication that we address in this section. Enforcing these restrictions is trivial for some classes of adaptability, such as affinely adaptive policies, as we simply restrict the policy to use only the correct components of $\boldsymbol{\xi}$. It is less obvious how to proceed with finite adaptability: the following example demonstrates how a naive application of the methods in Section 2 will either not produce valid solutions or will produce needlessly conservative solutions.

Example 3. Consider a three-stage ($T = 3$) version of the inventory problem of Examples 1 and 2. We make a continuous ordering decision x^1 before any demand is known, as before. The demand ξ^1 then arrives, which we must satisfy with a combination of the stock ordered beforehand and the fixed lots $y_A^2(\xi^1)$ and $y_B^2(\xi^1)$. Any stock remaining at this point $I^2(\xi^1)$ incurs a holding cost. We then make another continuous ordering decision $x^2(\xi^1)$ before a second round of demand arives, ξ^2. This demand may be satisfied with a combination of left-over inventory $I^2(\xi^1)$, $x^2(\xi^1)$, and fixed lots $y_A^3(\xi^1,\xi^2)$ and $y_B^3(\xi^1,\xi^2)$. Finally we incur a holding cost penalty on any remaining inventory. The three-stage AMIO is formulated (using the same parameters as before) as

$$
\begin{aligned}
& \min\ z && (13)\\
\text{subject to } & 50x^1 + 65I^2\left(\xi^1\right) + 1500y_A^2\left(\xi^1\right) + 1875y_B^2\left(\xi^1\right) && \\
& \quad + 50x^2\left(\xi^1\right) + 65I^3\left(\xi^1,\xi^2\right) + 1500y_A^3\left(\xi^1,\xi^2\right) + 1875y_B^3\left(\xi^1,\xi^2\right) \le z && \forall\left(\xi^1,\xi^2\right)\in\Xi\\
& I^2\left(\xi^1\right),\ I^3\left(\xi^1,\xi^2\right) \ge 0 && \forall\left(\xi^1,\xi^2\right)\in\Xi\\
& x^1,\ x^2\left(\xi^1\right) \ge 0 && \forall\left(\xi^1,\xi^2\right)\in\Xi\\
& y_A^2\left(\xi^1\right),\ y_B^2\left(\xi^1\right),\ y_A^3\left(\xi^1,\xi^2\right),\ y_B^3\left(\xi^1,\xi^2\right) \in \{0,1\} && \forall\left(\xi^1,\xi^2\right)\in\Xi
\end{aligned}
$$

where

$$I^2\left(\xi^1\right) = x^1 - \xi + 25y_A^2\left(\xi^1\right) + 25y_B^2\left(\xi^1\right)$$

and

$$I^3\left(\xi^1,\xi^2\right) = I^2\left(\xi^1\right) + x^2 - \xi^2 + 25y_A^3\left(\xi^1,\xi^2\right) + 25y_B^3\left(\xi^1,\xi^2\right).$$

We will use the simple box uncertainty set $\Xi = \{5 \le \xi^1, \xi^2 \le 95\}$. If we solve this problem with no partitions we obtain the three active samples $\hat{\boldsymbol{\xi}}_1 = (5,5)$ (for the objective constraint), $\hat{\boldsymbol{\xi}}_2 = (95,5)$

(for the non-negativity constraint on $I^2(\xi^1)$) and $\hat{\boldsymbol{\xi}}_3 = (95, 95)$ (for the non-negativity constraint on $I^3(\xi^1, \xi^2)$). The objective value of the unpartitioned problem is 27050. We now construct partitions for each of these samples exactly as we did before, resulting in the three partitions

$$\Xi\left(\hat{\boldsymbol{\xi}}_1 = (5,5)\right) = \left\{\xi^1, \xi^2 \,\middle|\, 5 \le \xi^1 \le 50,\ \xi^1 + \xi^2 \le 100\right\},$$

$$\Xi\left(\hat{\boldsymbol{\xi}}_2 = (95,5)\right) = \left\{\xi^1, \xi^2 \,\middle|\, 50 \le \xi^1 \le 95,\ 5 \le \xi^2 \le 50\right\},$$

$$\Xi\left(\hat{\boldsymbol{\xi}}_3 = (95,95)\right) = \left\{\xi^1, \xi^2 \,\middle|\, 5 \le \xi^1 \le 95,\ \xi^1 + \xi^2 \ge 100\right\},$$

which are displayed in Figure 3. As this is a multistage problem we must now consider nonanticipativity. In particular for each partition we will create new variables, for example x^2_1, x^2_2, and x^2_3 for $x^2(\xi^1)$. The intention is that if the realized value of ξ^1 is in $\Xi\left(\hat{\boldsymbol{\xi}}_1\right)$, then we select x^2_1 as the implemented decision, and similarly for the other partitions. However consider the case when ξ^1 is 10. This value of ξ^1 is in both $\Xi\left(\hat{\boldsymbol{\xi}}_1\right)$ and $\Xi\left(\hat{\boldsymbol{\xi}}_3\right)$, thus we must add the restriction that $x^2_1 = x^2_3$ as there is insufficient information at that time to distinguish between them. If we now consider $\xi^1 = 90$, which is in both $\Xi\left(\hat{\boldsymbol{\xi}}_2\right)$ and $\Xi\left(\hat{\boldsymbol{\xi}}_3\right)$, we know we must also add the restriction that $x^2_2 = x^2_3$, leaving us with no adaptability in our second-stage decisions. The third-stage variables are unaffected by these problems as at that time we know both ξ^1 and ξ^2, which allows us to fully identify the partition (and decision) we implement. The solution to this partitioned problem is $x^1 = 95$, $y^2_{A,1} = y^2_{B,1} = y^2_{A,2} = y^2_{B,2} = y^2_{A,3} = y^2_{B,3} = 0$, $x^2_1 = x^2_2 = x^2_3 = 45$, $y^3_{A,1} = y^3_{B,1} = y^3_{A,2} = 0$, $y^3_{B,2} = y^3_{A,3} = y^3_{B,3} = 1$ with objective 22075, which is 82% of the unpartitioned solution. The high initial order of stock suggests there is room for improvement in stage 2.

This example demonstrates that unless we take appropriate care to respect non-anticipitavity we will lose the benefits of partitioning that we hoped to gain and were demonstrated in the examples in Section 2. We now present generalizations of the two variants of Section 2 that allow us to benefit from the partitioning while respecting nonanticipativity.

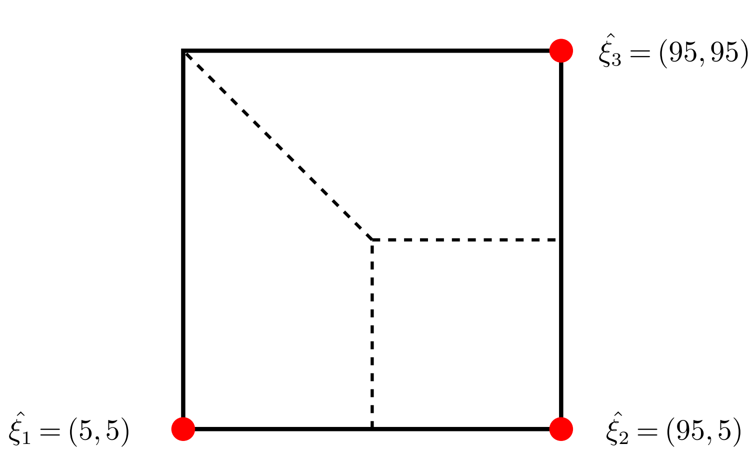

Figure 3 Partitioning of the uncertainty set in Example 3 using the two-stage non-nested variant case (Algorithm 1) without any adjustments for the multistage case, producing poor partitions.

3.1. Multistage Non-nested Variant

The multistage non-nested variant is essentially the same as the two-stage case, except with a more complex partition construction procedure. We do need this added complexity to ensure that nonanticipativity is enforceable and does not cut the number of effective partitions too much (as in Example 3, where we lost all adaptability in the second stage). We propose using only a subset of the uncertain parameters to construct the partition. In particular, when determining the dividing hyperplane between the partitions for a sample $\hat{\boldsymbol{\xi}}_i$ and a sample $\hat{\boldsymbol{\xi}}_j$ we first compare them to determine which components $\hat{\boldsymbol{\xi}}^t$ they share. For example, in Example 3 samples $\hat{\boldsymbol{\xi}}_2$ and $\hat{\boldsymbol{\xi}}_3$ are identical in the first stage, but different in the second. Samples $\hat{\boldsymbol{\xi}}_1$ and $\hat{\boldsymbol{\xi}}_3$ were different in both the first and second stages. We then construct the hyperplane that defines the boundary between between two partitions using only the uncertain parameters for the first time stage where there is a difference. Formally, we can express the partition for a sample $\hat{\boldsymbol{\xi}}_i$ as

$$\Xi\left(\hat{\boldsymbol{\xi}}_i\right) = \Xi \cap \left\{ \boldsymbol{\xi} \,\middle|\, \left\| \hat{\boldsymbol{\xi}}_i^{t_{i,j}} - \boldsymbol{\xi}^{t_{i,j}} \right\|_2 \le \left\| \hat{\boldsymbol{\xi}}_j^{t_{i,j}} - \boldsymbol{\xi}^{t_{i,j}} \right\|_2 \quad \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k \right\} \tag{14}$$

where $\mathcal{F}^k$ is the set of samples (as in Algorithm 1) and $t_{i,j}$ is the minimum t such that $\hat{\boldsymbol{\xi}}_i^t \neq \hat{\boldsymbol{\xi}}_j^t$. We then claim the following result regarding non-anticipitavity:

PROPOSITION 1. If $\mathbf{x}_i^t$ and $\mathbf{x}_j^t$ are the decisions at time stage $t \geq 2$ corresponding to the partitions $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ and $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ respectively, then constraining $\mathbf{x}_i^t = \mathbf{x}_j^t$ iff $\hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1} = \hat{\boldsymbol{\xi}}_j^{1,\ldots,t-1}$ ensures nonanticipativity.

Proof $\mathbf{x}^t$ is a function of the uncertain parameters $\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}$, thus to "ensure nonanticipativity" means that it's value may not depend on $\boldsymbol{\xi}^t,\ldots,\boldsymbol{\xi}^T$. If $t \leq t_{i,j}$ (as defined in Equation 14), then $\hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1} = \hat{\boldsymbol{\xi}}_j^{1,\ldots,t-1}$ so the hyperplane between $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ and $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ does not not involve any $\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}$ and thus any realization of those uncertain parameters that falls within $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ could be in $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$. Thus, we must enforce the nonanticipativity constraints. If $t > t_{i,j}$ then a realization of the uncertain parameters $\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}$ can be uniquely assigned to either one of the partitions (or neither) as the separating hyperplane involves the components $\boldsymbol{\xi}^{t_{i,j}}$. Thus, we do not need to enforce the nonanticipativity constraints.

We again define the set of active samples for a partition or set given a solution as

$$\begin{aligned}\hat{\Xi}\left(\bar{z},\bar{\mathbf{x}},\bar{\Xi}\right) = &\left\{\hat{\boldsymbol{\xi}} \;\middle|\; \hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi}\in\bar{\Xi}} \left\{ b_i\left(\boldsymbol{\xi}\right) - \sum_{t=1}^{T} \mathbf{a}_i^t\left(\boldsymbol{\xi}\right)\cdot\bar{\mathbf{x}}^t \right\} \quad \forall i \in \{1,\ldots,m\} \right\} \\ &\cup \left\{\hat{\boldsymbol{\xi}} \;\middle|\; \hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi}\in\bar{\Xi}} \left\{ \bar{z} - \sum_{t=1}^{T} \mathbf{c}^t\left(\boldsymbol{\xi}\right)\cdot\bar{\mathbf{x}}^t \right\} \right\},\end{aligned} \tag{15}$$

which is the same as in Algorithm 1 with the natural generalization to multiple stages.

ALGORITHM 3. Non-nested variant of iterative partitioning algorithm (multistage).

1. **Initialization**. Define $\mathcal{F}^1$ to be the initial empty set of samples, and iteration counter $k \leftarrow 1$
2. **Initial Solve**. Solve the following unpartitioned, non-adaptive version of (2)

$$\begin{aligned} z_{nonested}\left(\mathcal{F}^1\right) = \min_{\mathbf{x},z} \; & z \\ \text{subject to} \; & \sum_{t=1}^{T} \mathbf{c}^t\left(\boldsymbol{\xi}\right)\cdot\mathbf{x}^t \leq z && \forall \boldsymbol{\xi} \in \Xi \\ & \sum_{t=1}^{T} \mathbf{a}_i^t\left(\boldsymbol{\xi}\right)\cdot\mathbf{x}^t \leq b_i\left(\boldsymbol{\xi}\right) && \forall \boldsymbol{\xi} \in \Xi,\; i \in \{1,\ldots,m\} \\ & \mathbf{x} \in \mathcal{X}, \end{aligned}$$

with solution $(\bar{z},\bar{\mathbf{x}})$. Set $\mathcal{F}^2 \leftarrow \hat{\Xi}\left(\bar{z},\bar{\mathbf{x}},\Xi\right)$ and $k \leftarrow 2$.

3. **Partitioned Problem**. Solve the following partitioned version of (2), with one partition for every $\hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k$:

$$\begin{aligned} z_{nonnested}\left(\mathcal{F}^k\right) = \min_{\mathbf{x},z} \quad & z & (16)\\ \text{subject to} \quad & \sum_{t=1}^{T} \mathbf{c}^t(\boldsymbol{\xi}) \cdot \mathbf{x}_j^t \le z & \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k \\ & \sum_{t=1}^{T} \mathbf{a}_i^t(\boldsymbol{\xi}) \cdot \mathbf{x}_j^t \le b_i(\boldsymbol{\xi}) & \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k,\ i \in \{1,\dots,m\} \\ & \mathbf{x}_i^t = \mathbf{x}_j^t & \forall \hat{\boldsymbol{\xi}}_i, \hat{\boldsymbol{\xi}}_j \in \mathcal{F}^k \text{ where } \hat{\boldsymbol{\xi}}_i^{1,\dots,t-1} = \hat{\boldsymbol{\xi}}_j^{1,\dots,t-1},\ \forall t \in \{2,\dots,T\} \\ & \mathbf{x} \in \mathcal{X} \end{aligned}$$

with solution $(\bar{z}, \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \dots)$ and partitions $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ as defined in Equation 14.

4. **Grow set**. Let

$$\mathcal{F}^{k+1} \leftarrow \mathcal{F}^k \cup \left\{ \bigcup_j \hat{\Xi}\left(\bar{z}, \bar{\mathbf{x}}_j, \Xi\left(\hat{\boldsymbol{\xi}}_j\right)\right) \right\}$$

where $\hat{\Xi}$ is defined in Equation 15, and set $k \leftarrow k+1$. Terminate if termination criteria are reached otherwise and go to 3.

The partitions we create and the nonanticipativity constraints $\mathbf{x}_i^t = \mathbf{x}_j^t$ are naturally related. A partition is created using only the uncertain parameters for the first stage where they differ. As a result the variables corresponding to two different samples must take the same values for all stages up to but not including that first different stage. We now demonstrate the multistage non-nested variant with the same three-stage problem as Example 3.

Example 4. Consider the three-stage problem of Example 3. After solving the unpartitioned problem we had three active samples $\hat{\boldsymbol{\xi}}_1 = (5,5)$, $\hat{\boldsymbol{\xi}}_2 = (95,5)$ and $\hat{\boldsymbol{\xi}}_3 = (95,95)$ and an objective of 27050. We now consider building the partition for the second sample. As the second sample and third sample first differ in the second stage we set $t_{2,3} = 2$, and as the second and first samples differ in the first stage we set $t_{2,1} = 1$. We can now construct the partition for the second sample as

$$\Xi\left(\hat{\boldsymbol{\xi}}_2 = (95,5)\right) = \Xi \cap \left\{ \boldsymbol{\xi} \,\middle|\, \left\| 95 - \xi^1 \right\|_2 \le \left\| 5 - \xi^1 \right\|_2 \right\}$$

$$\cap \left\{ \boldsymbol{\xi} \,\middle|\, \left\|5-\xi^2\right\|_2 \le \left\|95-\xi^2\right\|_2 \right\}$$

$$= \left\{ \boldsymbol{\xi} \,\middle|\, 50 \le \xi^1 \le 95,\ 5 \le \xi^2 \le 50 \right\},$$

and likewise for the other partitions

$$\Xi\left(\hat{\boldsymbol{\xi}}_1 = (5,5)\right) = \left\{ \boldsymbol{\xi} \,\middle|\, 5 \le \xi^1 \le 50 \right\}$$

and

$$\Xi\left(\hat{\boldsymbol{\xi}}_3 = (95,95)\right) = \left\{ \boldsymbol{\xi} \,\middle|\, 50 \le \xi^1 \le 95,\ 50 \le \xi^2 \le 95 \right\}.$$

These partitions are displayed in Figure 4. We also impose the restriction that $x_2^2 = x_3^2$ as they share the same first stage value. If we now check this construction like we did before, we note that $\xi^1 = 10$ is unambiguously in $\Xi\left(\hat{\boldsymbol{\xi}}_1\right)$ and we have no nonanticipativity constraint for x_1^2. On the other hand $\xi^2 = 90$ could be in either $\Xi\left(\hat{\boldsymbol{\xi}}_2\right)$ or $\Xi\left(\hat{\boldsymbol{\xi}}_3\right)$, but we have already taken account of that in the construction of the partitioned problem. Thus we have used the three samples to construct three partitions, with two distinguishable partitions for the first stage uncertain parameters - an improvement on the lack of adaptability before. The solution to this partitioned problem is $x^1 = 50$, $y_{A,1}^2 = y_{B,1}^2 = 0$, $y_{A,2}^2 = y_{B,2}^2 = y_{A,3}^2 = y_{B,3}^2 = 1$, $x_1^2 = 95$, $x_2^2 = x_3^2 = 65$, $y_{A,1}^3 = y_{B,1}^3 = y_{A,2}^3 = y_{B,2}^3 = y_{B,3}^3 = 0$, $y_{A,3}^3 = 1$ with objective 19725, which is 73% of the unpartitioned solution and an improvement on the incorrectly partitioned solution of Example 3 which only improved to 82% due to a lack of adaptability in the second stage.

3.2. Multistage Nested Variant

As the two-stage nested variant extended the non-nested variant, the multistage nested variant extends the multistage non-nested variant. We again have a partition for each leaf of the sample tree, and we will use the same tree operators and terminology we established previously. There is

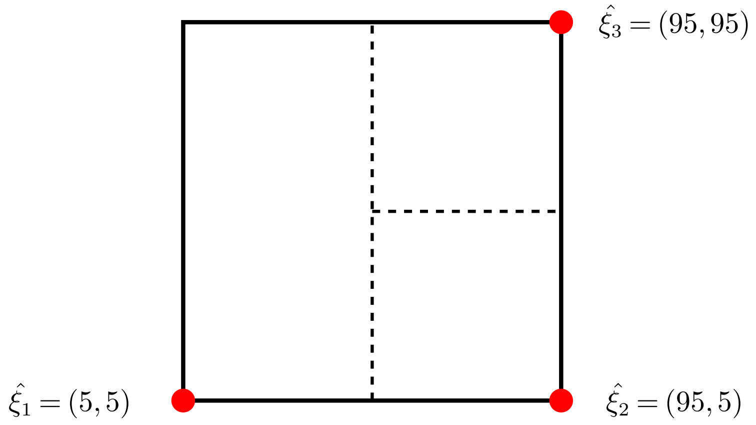

Figure 4 Partitioning of the uncertainty set in Example 4 using the multistage non-nested variant (Algorithm 3).

no change in how the samples are stored in the tree, only in how they are converted into partitions. The partitioning scheme we use is

$$\begin{aligned}\Xi\left(\hat{\boldsymbol{\xi}}_i\right) = & \left\{\boldsymbol{\xi} \,\middle|\, \left\|\hat{\boldsymbol{\xi}}_i^{t_{i,j}} - \boldsymbol{\xi}^{t_{i,j}}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j^{t_{i,j}} - \boldsymbol{\xi}^{t_{i,j}}\right\|_2 \quad \forall \hat{\boldsymbol{\xi}}_j \in \mathit{Siblings}\left(\hat{\boldsymbol{\xi}}_i\right)\right\} \\ & \cap \left\{\boldsymbol{\xi} \,\middle|\, \left\|\mathit{Parent}\left(\hat{\boldsymbol{\xi}}_i\right)^{t'_{i,j}} - \boldsymbol{\xi}^{t'_{i,j}}\right\|_2 \le \left\|\hat{\boldsymbol{\xi}}_j^{t'_{i,j}} - \boldsymbol{\xi}^{t'_{i,j}}\right\|_2 \quad \forall \hat{\boldsymbol{\xi}}_j \in \mathit{Siblings}\left(\mathit{Parent}\left(\hat{\boldsymbol{\xi}}_i\right)\right)\right\} \\ & \vdots \\ & \cap \Xi,\end{aligned} \tag{17}$$

where $t_{i,j}$ is the minimum t such that $\hat{\boldsymbol{\xi}}_i^t \neq \hat{\boldsymbol{\xi}}_j^t \in \mathit{Siblings}\left(\hat{\boldsymbol{\xi}}_i\right)$, $t'_{i.j}$ is the minimum t such that $\mathit{Parent}\left(\hat{\boldsymbol{\xi}}_i\right)^t \neq \hat{\boldsymbol{\xi}}_j^t \in \mathit{Siblings}\left(\mathit{Parent}\left(\hat{\boldsymbol{\xi}}_i\right)\right)$, and so on. This is the same nesting scheme as in Algorithm 2, with the selective use of components seen in (14).

Enforcement of the nonanticipativity constraints is more complex than in Algorithm 3. We propose a method to determine whether, for two samples $\hat{\boldsymbol{\xi}}_i$ and $\hat{\boldsymbol{\xi}}_j$, we need to enforce the nonanticipativity constraints $\mathbf{x}_i^t = \mathbf{x}_j^t$. The rule is as follows:

1. Let $\hat{\boldsymbol{\xi}}_A \leftarrow \hat{\boldsymbol{\xi}}_i$ and $\hat{\boldsymbol{\xi}}_B \leftarrow \hat{\boldsymbol{\xi}}_j$.
2. If $\hat{\boldsymbol{\xi}}_A$ and $\hat{\boldsymbol{\xi}}_B$ are siblings, then let $t_{A,B} = \arg\min_s \left\{\hat{\boldsymbol{\xi}}_A^s \neq \hat{\boldsymbol{\xi}}_B^s\right\}$. If $t > t^{A,B}$, then we do not need to enforce the nonanticipativity constraints. If $t \le t^{A,B}$, then we must.
3. If they are not siblings, let $\hat{\boldsymbol{\xi}}_A \leftarrow \mathit{Parent}\left(\hat{\boldsymbol{\xi}}_A\right)$ and $\hat{\boldsymbol{\xi}}_B \leftarrow \mathit{Parent}\left(\hat{\boldsymbol{\xi}}_B\right)$ and go to Step 2.

We first seek to demonstrate that this decision rule is sufficient to enforce nonanticipativity, but we will then show by example that it is conservative: it will enforce nonanticipativity constraints needlessly in some cases.

Proposition 2. If $\mathbf{x}_i^t$ and $\mathbf{x}_j^t$ are the decisions at time stage $t \geq 2$ corresponding to the partitions $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ and $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ respectively, then constraining $\mathbf{x}_i^t = \mathbf{x}_j^t$ according to the above decision rule ensures nonanticipativity.

Proof If $\hat{\boldsymbol{\xi}}_i$ and $\hat{\boldsymbol{\xi}}_j$ are siblings, then the rule is equivalent to the situation in Proposition 1. If not, we let $\hat{\boldsymbol{\xi}}_A \leftarrow Parent\left(\hat{\boldsymbol{\xi}}_i\right)$ and $\hat{\boldsymbol{\xi}}_B \leftarrow Parent\left(\hat{\boldsymbol{\xi}}_j\right)$, and $t_{A,B} = \arg\min_s \left\{\hat{\boldsymbol{\xi}}_A^s \neq \hat{\boldsymbol{\xi}}_B^s\right\}$. If they are siblings, then we know that all child partitions are separated by a constraint involving the terms $\boldsymbol{\xi}^{t_{A,B}}$. If $t \leq t_{A,B}$ then the hyperplane induced by $\hat{\boldsymbol{\xi}}_A$ and $\hat{\boldsymbol{\xi}}_B$ does not not involve any $\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}$ and thus any realization of those uncertain parameters could fall in a child partition of $\hat{\boldsymbol{\xi}}_A$ or $\hat{\boldsymbol{\xi}}_B$, and we must enforce the nonanticipativity constraints. If $t > t_{i,j}$ then a realization of the uncertain parameters $\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^{t-1}$ can be uniquely assigned to one of child partitions of either $\hat{\boldsymbol{\xi}}_A$ or $\hat{\boldsymbol{\xi}}_B$ (or neither) as the separating hyperplane involves the components $\boldsymbol{\xi}^{t_{A,B}}$, so we do not need to enforce the nonanticipativity constraints. If they are not siblings, then we set $\hat{\boldsymbol{\xi}}_A \leftarrow Parent\left(\hat{\boldsymbol{\xi}}_A\right)$ and $\hat{\boldsymbol{\xi}}_B \leftarrow Parent\left(\hat{\boldsymbol{\xi}}_B\right)$ and apply the same logic.

We now provide an example uncertainty set and sample tree for which this decision rule over-constrains the variables.

Example 5. Over-conservative application of nonanticipativity constraint. Consider the two-stage uncertainty set $\Xi = \left\{(\xi^1,\xi^2)\,|0 \leq \xi^1,\xi^2 \leq 10\right\}$ with sample tree

$$\mathcal{T} = root \begin{cases} (10,0)\,, & \begin{cases} (0,0) & :A \\ (10,0) & :B \end{cases} \\ \\ (10,10)\,, & \begin{cases} (0,10) & :C \\ (10,10) & :D \end{cases} \end{cases}$$

where we have labelled each leaf sample. The uncertainty sets corresponding to each leaf are

$$\begin{aligned}
\Xi\left(\hat{\boldsymbol{\xi}}_A\right) &= \left\{\left(\xi^1,\xi^2\right) \middle| 0 \le \xi^1 \le 5,\ 0 \le \xi^2 \le 5\right\}\\
\Xi\left(\hat{\boldsymbol{\xi}}_B\right) &= \left\{\left(\xi^1,\xi^2\right) \middle| 5 \le \xi^1 \le 10,\ 0 \le \xi^2 \le 5\right\}\\
\Xi\left(\hat{\boldsymbol{\xi}}_C\right) &= \left\{\left(\xi^1,\xi^2\right) \middle| 0 \le \xi^1 \le 5,\ 5 \le \xi^2 \le 10\right\}\\
\Xi\left(\hat{\boldsymbol{\xi}}_D\right) &= \left\{\left(\xi^1,\xi^2\right) \middle| 5 \le \xi^1 \le 10,\ 5 \le \xi^2 \le 10\right\}
\end{aligned}$$

which we can manually inspect and determine that we must enforce the nonanticipativity constraints $\mathbf{x}_A^2 = \boldsymbol{x}_C^2$ and $\mathbf{x}_B^2 = \boldsymbol{x}_D^2$. However, consider the application of the decision rule above to this problem for $t = 2$ ($\mathbf{x}^2\left(\xi^1\right)$):

- $\hat{\boldsymbol{\xi}}_A$ and $\hat{\boldsymbol{\xi}}_B$ are siblings, and $t_{A,B} = 1 < t$, so they do not require a nonanticipativity constraint.
- $\hat{\boldsymbol{\xi}}_A$ and $\hat{\boldsymbol{\xi}}_C$ are not siblings, so we must compare their parents. The parents are siblings, and differ in ξ^2 so they do require a nonanticipativity constraint.
- $\hat{\boldsymbol{\xi}}_A$ and $\hat{\boldsymbol{\xi}}_D$ are not siblings, so we must compare their parents. The parents are siblings, and differ in ξ^2 so we add a nonanticipativity constraint. However, this constraint is not required.

Thus the decision rule is over-conservative in this case.

The problem of deciding whether or the nonanticipativity constraints need to be enforced for time stage t can be determined by the solution of an optimization problem. Let us express the partitions for the two samples of interest $\hat{\boldsymbol{\xi}}_i$ and $\hat{\boldsymbol{\xi}}_j$ as

$$\begin{aligned}
\Xi\left(\hat{\boldsymbol{\xi}}_i\right) &= \left\{\boldsymbol{\xi} \in \Xi \,\middle|\, F\boldsymbol{\xi} \le f\right\},\\
\Xi\left(\hat{\boldsymbol{\xi}}_j\right) &= \left\{\boldsymbol{\xi} \in \Xi \,\middle|\, G\boldsymbol{\xi} \le g\right\},
\end{aligned}$$

and consider the optimization problem

$$\begin{aligned}
INTERSECT\left(t\right) = \max\ & \alpha && && \\
\text{subject to } & F\boldsymbol{\psi} + \alpha\mathbf{e} && && \le f\\
& G\boldsymbol{\phi} + \alpha\mathbf{e} && && \le g\\
& \boldsymbol{\psi}^s && = \boldsymbol{\phi}^s && \forall s \in \{1, \dots t-1\}\\
& \boldsymbol{\psi} \in \Xi, \boldsymbol{\phi} \in \Xi. && &&
\end{aligned} \tag{18}$$

If problem 18 is infeasible, then we do not need to enforce the nonanticipativity constraints for time stage t as no value of $\boldsymbol{\xi}^{1,\dots,t-1}$ appears in both partitions. If the problem is feasible with objective zero, then there is as a value of $\boldsymbol{\xi}^{1,\dots,t-1}$ that is on the boundary of the partitions, so again we do not need to enforce the nonanticipativity constraints as we could simply perturb the partitions to make the decision unique. If the objective is greater than zero, then we must enforce the non-anticipitavity constraints. We can use this technique to reduce the number of nonanticipativity constraints required by applying it only to those pairs of partitions that the above conservative decision rule says require them. This should be a cheap operation (compared to solve the AMIO) as we will have to check far less than all possible pairs, and the uncertainty set is typically a low-dimensional set. Given these tools, we can now state the multistage nested variant.

ALGORITHM 4. Nested variant of iterative partitioning algorithm (multistage).

1. **Initialization**. Define $\mathcal{T}^1$ to be the initial tree of samples, consisting of one root node (any $\hat{\boldsymbol{\xi}} \in \Xi$). Set iteration counter $k \leftarrow 1$.

2. **Partitioned Problem**. Solve the following partitioned version of 2, with one partition for every $\hat{\boldsymbol{\xi}}_j \in Leaves\left(\mathcal{T}^k\right)$:

$$\begin{aligned} z_{tree}\left(\mathcal{T}^k\right) = \min_{\mathbf{x},z} \quad & z && (19)\\ \text{subject to } & \sum_{t=1}^{T} \mathbf{c}^t\left(\boldsymbol{\xi}\right)\cdot \mathbf{x}_i^t \le z && \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in Leaves\left(\mathcal{T}^k\right)\\ & \sum_{t=1}^{T} \mathbf{a}_i^t\left(\boldsymbol{\xi}\right)\cdot \mathbf{x}_j^t \le b_i\left(\boldsymbol{\xi}\right) && \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in Leaves\left(\mathcal{T}^k\right),\ i\in\{1,\dots,m\}\\ & \mathbf{x}_i^t = \mathbf{x}_j^t && \forall \hat{\boldsymbol{\xi}}_i, \hat{\boldsymbol{\xi}}_j \in Leaves\left(\mathcal{T}^k\right)\ iff\ \dots\\ & \mathbf{x} \in \mathcal{X} && (20) \end{aligned}$$

with solution $(\bar{z}, \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \dots)$. The partitions are defined as in Equation 17, and the nonanticipativity constraints are enforced using the logic stated above.

3. **Grow set**. Initialize $\mathcal{T}^{k+1} \leftarrow \mathcal{T}^k$. For every leaf $\hat{\boldsymbol{\xi}}_j \in Leaves\left(\mathcal{T}^{k+1}\right)$ add the samples $\hat{\Xi}\left(\bar{z}, \bar{\mathbf{x}}_j, \Xi\left(\hat{\boldsymbol{\xi}}_j\right)\right)$ as children of that leaf. Terminate if termination criteria are reached otherwise set $k \leftarrow k+1$ and go to 2.

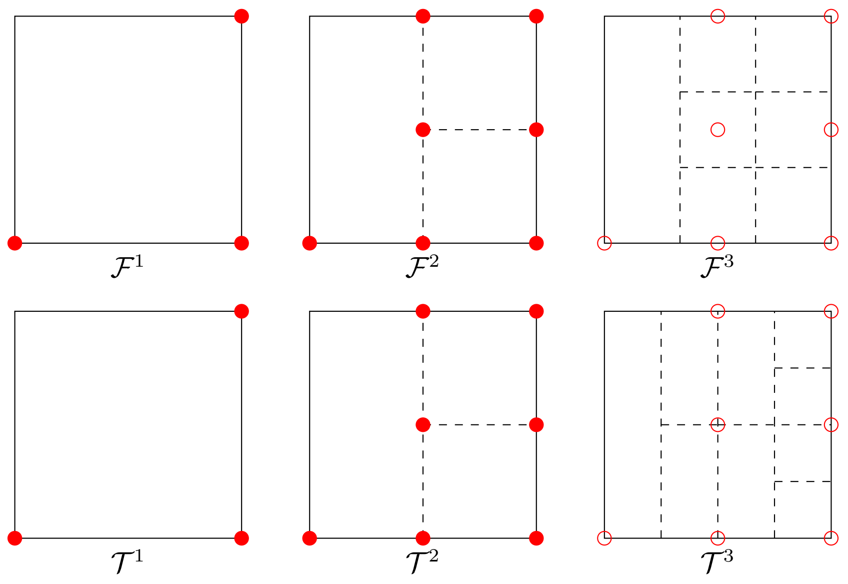

Figure 5 Comparison of the uncertainty set partitions for three iterations of the multistage non-nested (top) and nested (bottom) variants for the inventory control problem of Examples 13, 4 and 6. Both are the same in the first two iterations, but diverge in the third. The solid points represent the active samples after solution, and the hollow points in the third iteration represent the samples that define the partitions (some points overlap).

To demonstrate the differences from the multistage non-nested variant we extend Example 4 to a third iteration.

Example 6. In Example 4 we solved the three-stage inventory problem using the non-nested variant for two iterations. The first and second iteration for the nested and non-nested variants produce the same partitions, so we will start from the solution of the non-nested variant. In particular, the active samples are as follows: we have active samples $(5,5)$, $(50,5)$, $(50,95)$ for $\hat{\Xi}\left(\hat{\boldsymbol{\xi}}_1 = (5,5)\right)$, $(50,5)$, $(95,5)$, and $(95,50)$ for $\hat{\Xi}\left(\hat{\boldsymbol{\xi}}_2 = (95,5)\right)$, and finally $(50,50)$, $(95,50)$, $(95,95)$ for $\hat{\Xi}\left(\hat{\boldsymbol{\xi}}_3 = (95,95)\right)$. We display the induced partitions for both the non-nested and the nested variants in Figure 5. The third stage objective value for the non-nested variant is 17100 (with 7 partitions, 63% of non-adaptive) and 16200 for the nested variant (with 9 partitions, 60% of non-adaptive).

In this section we demonstrated that both variants can be adapted to the multistage case. Interestingly the number of partitions we have at any iteration is not directly connected to the number of time stages. However, we expect that in practical problems the number of partitions will scale with the number of time stages as the uncertainty set dimension grows. In the example used in this section the non-nested variant had 3 unique samples at the end of iteration 1, 7 unique samples at the end of iteration 2, and 13 at the end of iteration 3. The nested variant had 3 leaves at the end of iteration 1, 9 leaves at the end of iteration 2, and 27 at the end of iteration 3. However, the nested variant had a better objective value than the non-nested variant. For this reason we suggest practitioners evaluate both variants to evaluate what provides the best balance of solution quality and computational efficiency for the problem at hand.

4. Bounds on Performance

In this section, we present results for three types of bounds for different aspects of our methods and AMIO. These bounds are useful for their ability to improve computational tractability. The first bound is perhaps the most critical as it provides an estimate of the gap between our finitely adaptable solution and the fully adaptive solution. The second bound is an upper bound on the objective value of a subsequent iteration of the algorithm. This bound is only available for the nested variant, which we show has monotonically decreasing objectives, whereas the non-nested variant may become worse from iteration-to-iteration. Finally we provide a lower bound on the objective value of a subsequent iteration using duality theory, which may be useful for the purposes of early termination.

4.1. Lower Bound on Fully Adaptive Solution

The finite adaptability methods of this paper are an approximation of the fully adaptive AMIO problem (2). In particular, the difficult-to-calculate solution to (2) is a lower bound on the objective value of our finitely adaptable problem at any iteration with either variant. Without knowledge of the objective value of the fully adaptive solution, we must use an easier to calculate lower bound with the information we have. Given this lower bower bound on the fully adaptive solution, we can estimate the gap between our finitely adaptive solution and the fully adaptive solution. We propose using the set of samples available at the end of iteration k to obtain this lower bound. That is, after the solution of the partitioned problem at iteration k, we can augment the set of samples (i.e. $\mathcal{F}^{k+1}, \mathcal{T}^{k+1}$) and use those samples to construct and solve a deterministic problem whose objective provides the lower bound.

PROPOSITION 3. Consider the solution of the optimization problem

$$\begin{aligned} z_{lower}(\mathcal{A}) = \min_{\mathbf{x},z} \quad & z \\ \text{subject to} \quad & \sum_{t=1}^{T} \mathbf{c}^t\left(\hat{\boldsymbol{\xi}}_i\right) \cdot \mathbf{x}_i^t \le z && \forall \hat{\boldsymbol{\xi}}_i \in \mathcal{A} \\ & \sum_{t=1}^{T} \mathbf{A}^t\left(\hat{\boldsymbol{\xi}}_i\right) \cdot \mathbf{x}_i^t \le \mathbf{b}\left(\hat{\boldsymbol{\xi}}_i\right) && \forall \hat{\boldsymbol{\xi}}_i \in \mathcal{A} \\ & \mathbf{x}_i^t = \mathbf{x}_j^t && \forall \hat{\boldsymbol{\xi}}_i, \hat{\boldsymbol{\xi}}_j \in \mathcal{A} \text{ such that } \hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1} = \hat{\boldsymbol{\xi}}_j^{1,\ldots,t-1} \\ & \mathbf{x} \in \mathcal{X}, \end{aligned} \tag{21}$$

where $\mathcal{A}$ is a set of samples $\hat{\boldsymbol{\xi}} \in \Xi$. Then $z_{lower}(\mathcal{A}) \le z_{full}$, where is z_{full} is the solution to (2).

Proof Let $\bar{\mathbf{x}}^t(\boldsymbol{\xi}^{1,\ldots,t-1})$ be the optimal solution to (2). We can thus construct a feasible solution $\tilde{\mathbf{x}}_i^t$ to (21) using the mapping

$$\tilde{\mathbf{x}}_i^t = \bar{\mathbf{x}}^t\left(\hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1}\right)$$

which satisfies the linear constraints because $\bar{\mathbf{x}}^t(\boldsymbol{\xi}^{1,\ldots,t-1})$ is feasible for all $\boldsymbol{\xi} \in \Xi$, and satisfies the nonanticipativity constraints because if $\hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1} = \hat{\boldsymbol{\xi}}_j^{1,\ldots,t-1}$ then $\bar{\mathbf{x}}^t\left(\hat{\boldsymbol{\xi}}_i^{1,\ldots,t-1}\right) = \bar{\mathbf{x}}^t\left(\hat{\boldsymbol{\xi}}_j^{1,\ldots,t-1}\right)$ and thus $\tilde{\mathbf{x}}_i^t = \tilde{\mathbf{x}}_j^t$. This constructed solution for (21) will have the same objective value as the solution for (2), so $z_{lower}(\mathcal{A}) \le z_{full}$.

We construct the set $\mathcal{A}$ in (21) differently for each variant: in the non-nested variant the set $\mathcal{A}$ is simply $\mathcal{F}^k$, and in the nested variant it is all samples in the tree $\mathcal{T}^k$ - not just the leaf samples. We thus have the bound

$$z_{lower}(\mathcal{A}) \le z_{full} \le \begin{cases} z_{nonnested}\left(\mathcal{F}^k\right) \\ z_{nested}\left(\mathcal{T}^k\right) \end{cases}$$

and we can use the relative gap between $z_{lower}(\mathcal{A})$ and $z_{variant}$ as a termination criterion, e.g. terminate if

$$\frac{z_{variant} - z_{lower}}{z_{lower}} \leq \alpha.$$

We note that there is a close relationship between this bound and the *a posteriori* "scenario based bound" of Hadjiyiannis et al. (2011) in a two-stage setting. Indeed they also discuss the use of "binding" uncertainty realizations, much like our active samples. The primary difference in the bound is addition of the multistage nonanticipativity constraints, which they do not need to consider for their formulation.

A useful property of this bound is that as we obtain more samples the lower bound improves monotonically, so at each iteration we shrink the bound gap from above and below. However, calculating a solution to (21) may be computationally intensive in its own right. A solution to the partitioned problems (16) or (19) can be mapped to a feasible solution to (21), which enables us to provide an initial solution for the branch-and-bound solver. This can reduce solve times for the bounding problem, which we do not necessarily need to solve to provable optimality.

4.2. Upper Bound on Subsequent Iterations

We now present an upper bound on the objective value of the partitioned problem at iteration $k+1$ relative to the solution at iteration k for the nested variant, and demonstrate by example that the non-nested variant does not have a useful upper bound of this nature.

Proposition 4. The objective values of $z_{nonnested}(\mathcal{F}^k)$ do not necessarily decrease monotonically in k, and a solution to (6) at iteration k does necessarily provide a feasible solution for (6) at iteration $k+1$.

Proof We provide an example of this behavior as proof. Consider the two-stage AMIO problem

$$\min_{w,x,y,z} \; z$$

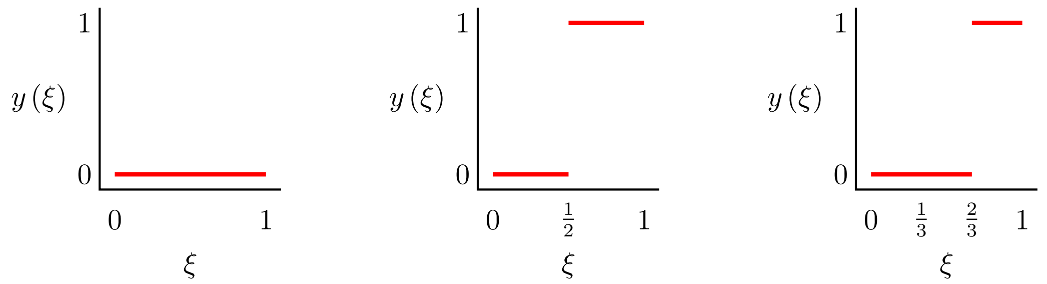

Figure 6 Optimal policies for y as a function of ξ for one (left), two (center), and three (right) partitions for the problem described in the proof for Proposition 4 using the non-nested algorithm. The two partition policy is the same as optimal policy for y in this problem, but it is impossible to express the optimal policy with these particular three partitions.

$$\begin{aligned}
\text{subject to } & z \geq x + 20w(\xi) - 10y(\xi) && \forall \xi \in [0,1] \\
& w(\xi) \geq \xi && \forall \xi \in [0,1] \\
& y(\xi) \leq \min\left\{x,\ \xi + \frac{1}{2}\right\} && \forall \xi \in [0,1] \\
& y(\xi) \in \{0,1\} \\
& x \in \{0,1\},
\end{aligned}$$

which has one first-stage variable x and two second-stage variables $w(\xi)$ and $y(\xi)$. If we solve the non-adaptable version of the problem at iteration 1 we obtain the solution $w = 1$, $x = 0$, $y = 0$, with objective value $z = 20$. The active samples are $\hat{\xi}_1 = 0$ and $\hat{\xi}_2 = 1$, so at iteration 2 we have partitions $\Xi\left(\hat{\xi}_1 = 0\right) = \left[0, \frac{1}{2}\right]$ and $\Xi\left(\hat{\xi}_2 = 1\right) = \left[\frac{1}{2}, 1\right]$. The solution at iteration 2 is $w_1 = \frac{1}{2}$, $w_2 = 1$, $x = 1$, $y_1 = 0$, $y_2 = 1$, with objective value $z = 11$. It can be shown that this is equivalent to the fully adaptive solution for this problem.

If we now construct a new set of partitions using the active samples $\hat{\xi}_1 = 0$, $\hat{\xi}_2 = \frac{1}{2}$, and $\hat{\xi}_3 = 1$, we obtain a solution with a worse objective $14\frac{1}{3}$. This is due to the impossibility of expressing the optimal policy for y with these three partitions, which we demonstrate visually in Figure 6.

Proposition 5. The objective values of $z_{nested}\left(\mathcal{T}^k\right)$ do decrease monotonically in k, and a solution to (9) at iteration k provides a feasible solution for (9) at iteration $k+1$.

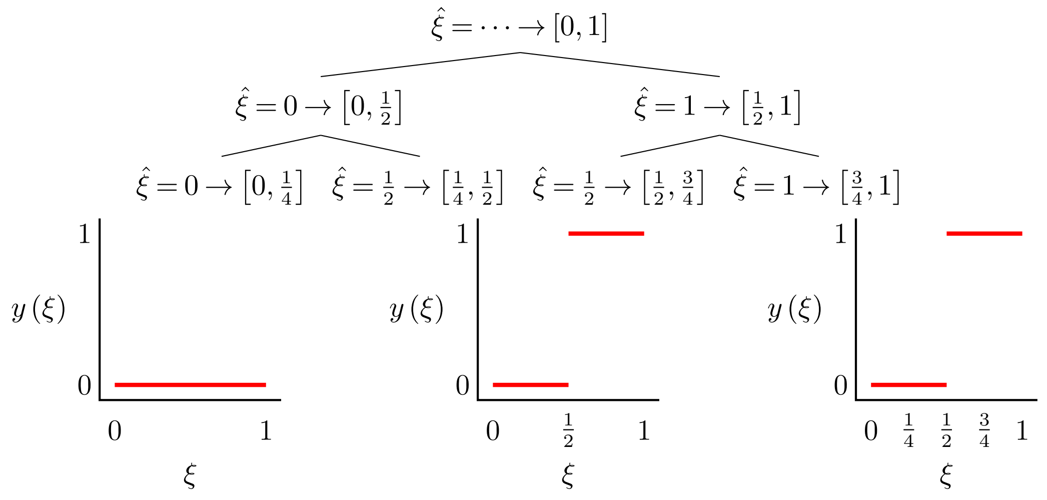

Figure 7 Optimal policies for y as a function of ξ for one (left), two (center), and four (right) partitions for the problem described in Theorem 4 using the nested algorithm. The two and four partition policies are the same as fully adaptive optimal policy for y in this problem.

Proof Consider the sample tree $\mathcal{T}^k$ at iteration k and the solution to the corresponding optimization problem (9). Each leaf $\hat{\boldsymbol{\xi}}_i$ of the tree corresponds to a partition of the uncertainty set $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$, and a second-stage solution $\mathbf{x}_i^2$. For each leaf we will obtain new samples, which we will use in iteration $k+1$ to create sub-partitions of $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$. The solution at iteration at k is by definition feasible for all $\boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_i\right)$, and as we only sub-partition existing partitions, $\mathbf{x}_i^2$ at iteration k is a feasible value for all second-stage variables corresponding to the sub-partitions of $\Xi\left(\hat{\boldsymbol{\xi}}_i\right)$ at iteration $k+1$. Because the solution at iteration k is a feasible solution for the problem at iteration $k+1$, we will obtain a solution no worse than at iteration k.

We can demonstrate Proposition 5 in practice for the example problem in Proposition 4. The nested variant produces identical partitions and solutions for the first two iterations but in the third iteration it produces partitions $\left[0, \frac{1}{4}\right], \left[\frac{1}{4}, \frac{1}{2}\right], \left[\frac{1}{2}, \frac{3}{4}\right], \left[\frac{3}{4}, 1\right]$, which allows the optimal policy that we found at iteration 2 to be also be found at iteration 3. This is demonstrated in Figure 7.

4.3. Lower Bound on Subsequent Iterations

We now present a third bound that bounds the maximum improvement for subsequent iterations. Consider the two-stage adaptive *linear* optimization (ALO) problem

$$\begin{aligned} z^*_{nopart} = \min_{\mathbf{x},z} \quad & z \\ \text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2(\boldsymbol{\xi}) \le z \\ & \mathbf{a}^1_i \cdot \mathbf{x}^1 + \mathbf{a}^2_i \cdot \mathbf{x}^2(\boldsymbol{\xi}) \le \mathbf{b}_i \cdot \boldsymbol{\xi} \qquad \forall \boldsymbol{\xi} \in \Xi,\ i \in \{1,\dots,m\} \\ & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{22}$$

where the uncertainty set Ξ is a polyhedron. We note that if $\mathbf{b}_i = \mathbf{0}$ the constraint is not uncertain and should be included in set $\mathcal{X}$, so we will assume all $\mathbf{b}_i \neq \mathbf{0}$. Consider solving the non-adaptive version of (22) using a cutting plane method: as the uncertainty is only in the right-hand-side of the problem it is sufficient to solve once, for each constraint i, the cutting plane problem

$$\hat{\boldsymbol{\xi}}_i = \arg\min_{\boldsymbol{\xi}\in\Xi} \ \mathbf{b}_i \cdot \boldsymbol{\xi}$$

The deterministic problem created through the cutting plane method is thus

$$\begin{aligned} z^*_{nopart} = \min_{\mathbf{x},z} \quad & z \\ \text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2 \le z \\ & \mathbf{a}^1_i \cdot \mathbf{x}^1 + \mathbf{a}^2_i \cdot \mathbf{x}^2 \le \mathbf{b}_i \cdot \hat{\boldsymbol{\xi}}_i \qquad \forall i \in \{1,\dots,m\} \\ & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{23}$$

with dual variable π_i associated to each constraint i. Consider now the set $\mathcal{A}$ of active samples $\hat{\boldsymbol{\xi}}_i$ for each constraint in (23), and use them to create the partitioned problem

$$\begin{aligned} z^*_{part} = \min_{\mathbf{x},z} \quad & z \\ \text{subject to } & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2_j \le z \qquad \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{A} \\ & \mathbf{a}^1_i \cdot \mathbf{x}^1 + \mathbf{a}^2_i \cdot \mathbf{x}^2_j \le \mathbf{b}_i \cdot \boldsymbol{\xi} \qquad \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}_j\right),\ \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{A},\ i \in \{1,\dots,m\} \\ & \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{24}$$

We now seek to bound the maximum improvement in objective value between unpartitioned problem and partitioned problem, i.e., a bound on $z^*_{nopart} - z^*_{part}$. To do so, we first prove a result about the objective for a single partition.

LEMMA 1. Given a $\hat{\boldsymbol{\xi}} \in \mathcal{A}$ define the partitioned optimization problem

$$\begin{aligned} z^*_{onepart} = \min_{\mathbf{x},z} \quad & z & (25) \\ \text{subject to } \; & \mathbf{c}^1 \cdot \mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2 \leq z \\ & \mathbf{a}^1_i \cdot \mathbf{x}^1 + \mathbf{a}^2_i \cdot \mathbf{x}^2_j \leq \mathbf{b}_i \cdot \boldsymbol{\xi} \qquad \forall \boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}\right), i \in \{1,\ldots,m\} \\ & \mathbf{x} \in \mathcal{X}. \end{aligned}$$

For each constraint i solve the cutting plane problem

$$\arg \min_{\boldsymbol{\xi} \in \Xi\left(\hat{\boldsymbol{\xi}}\right)} \mathbf{b}_i \cdot \boldsymbol{\xi}$$

to obtain a $\tilde{\boldsymbol{\xi}}_i$. The objective $z^*_{onepart}$ is bounded by the relationship

$$z^*_{onepart} \geq z^*_{nopart} + \sum_{i=1}^{m} \mathbf{b}_i \cdot \left(\tilde{\boldsymbol{\xi}}_i - \hat{\boldsymbol{\xi}}_i\right) \pi_i$$

where $\hat{\boldsymbol{\xi}}_i$ is the active sample for constraint i in Problem (23).

Proof $\boldsymbol{\pi}$, the dual solution to (23), is a sub-gradient of the objective function of (23) with respect to the right-hand-side of the constraints. That is, if the right-hand-side changes by $\boldsymbol{\Delta}$, the objective changes by at most $\boldsymbol{\pi} \cdot \boldsymbol{\Delta}$. The change in the right-hand-side between (23) and (25) is determined by the change in the active uncertain parameters, i.e. $\Delta_i = \mathbf{b}_i \cdot \left(\tilde{\boldsymbol{\xi}}_i - \hat{\boldsymbol{\xi}}_i\right)$. Thus the change in objective is at least $\sum_{i=1}^{m} \mathbf{b}_i \cdot \left(\tilde{\boldsymbol{\xi}}_i - \hat{\boldsymbol{\xi}}_i\right) \pi_i$.

We now use this lemma to provide our bound on the improvement between two iterations.

THEOREM 2. For each partition $\Xi\left(\hat{\boldsymbol{\xi}}_j\right)$ of (24) calculate a $\tilde{\boldsymbol{\xi}}_{j,i}$ for each constraint i as in Lemma 1. The objective z^*_{part} is bounded by the relationship

$$z^*_{part} \geq z^*_{nopart} + \sum_{i=1}^{m} \mathbf{b}_i \cdot \left(\tilde{\boldsymbol{\xi}}_{j,i} - \hat{\boldsymbol{\xi}}_i\right) \pi_i \qquad \forall \hat{\boldsymbol{\xi}}_j \in \mathcal{A}$$

Proof This follows from Lemma 1. The objective of z^*_{part} is the largest of the respective objective of each partition $(\mathbf{c}^1\mathbf{x}^1 + \mathbf{c}^2 \cdot \mathbf{x}^2_j)$ so the lower bound is the largest (worst) of the bounds on each individual partition. That is, the bound is the most pessimistic of the partition bounds.

We can illustrate this bound using the following two-stage ALO problem.

Example 7. Two-stage stock allocation problem. Consider the following two-stage ALO problem

$$\begin{aligned} z = \max\ & 3(S_1+S_2) - (B_1+B_2) \\ \text{s.t. }\ & 0 \le S_1 \le B_1 \\ & 0 \le S_2 \le B_2 \\ & S_1 \le D_1 \qquad \forall \mathbf{D} \in U \\ & S_2 \le D_2 \qquad \forall \mathbf{D} \in U \\ & U = \left\{ \mathbf{D} \mid \frac{|D_1 - 20|}{10} + \frac{|D_2 - 10|}{5} \le 1 \right\} \end{aligned}$$

where $\mathbf{S}$ is the second-stage decision (amount sold at a store) which is dependent on how much stock is bought in the first stage (B) and the uncertain demand ($\mathbf{D}$). The optimal objective value z for the non-adaptive problem is 30. For the uncertain constraint on S_1 we find that the active sample is $\hat{\mathbf{D}} = (10,10)$ with $\pi_1 = 2$ and correspondingly for S_2 we find that the sample is $\hat{\mathbf{D}} = (20,5)$ with $\pi_2 = 2$ as well. If now create the partitioned problem, but only obtain the cuts/active samples, we find that for the partition induced by $\hat{\mathbf{D}} = (10,10)$ we have active samples corresponding to $\tilde{\mathbf{D}} = (10,10)$ and $\tilde{\mathbf{D}} = (15,7.5)$, giving a partial upper bound (as we are maximizing) for that partition of

$$30 + 2\,[1\ \ 0] \begin{bmatrix} 10-10 \\ 10-10 \end{bmatrix} + 2\,[0\ \ 1] \begin{bmatrix} 10-15 \\ 10-7.5 \end{bmatrix} = 35.$$

For the partition induced by $\hat{\mathbf{D}} = (20,5)$ we have active cuts corresponding to $\tilde{\mathbf{D}} = (15,7.5)$ and $\tilde{\mathbf{D}} = (20,5)$, giving a partial upper bound of

$$30 + 2\,[1\ \ 0] \begin{bmatrix} 20-15 \\ 5-7.5 \end{bmatrix} + 2\,[0\ \ 1] \begin{bmatrix} 20-20 \\ 5-5 \end{bmatrix} = 40.$$

The bound on the objective value for the partitioned problem is thus 35, the more pessimistic of the two partial bounds. The actual objective value of the partitioned problem is 32.5, but we note the best possible objective value (with full adaptability) is 35.

This bound generalizes naturally to multiple iterations and time stages, which we omit for brevity. However, the bound as stated is dependent on both the uncertainty being only in the right-hand-side, and on the problem being continuous. The first restriction cannot be relaxed, but the second may be overcome in some cases. For mixed-integer problems it may be sufficient to fix integer variables to get a continuous problem, solve and use the dual solution of that. The bound no longer necessarily holds, but may be able to provide insights regardless.

5. Computational Experiments

The goal of this section is to demonstrate that, for problems of interest from the literature, the methods described in this paper are practical. In particular, we provide comparisons with alternative methods, guidance on the question of whether to use the nested or non-nested variant, and how the methods scale. To do so we consider a two-stage binary capital budgeting problem (Section 5.1), a mulitstage lot sizing problem (Section 5.2), and finally an extension of the lot sizing problem to the network setting (Section 5.3). We will summarize the conclusions drawn from these benchmarks in Section 6.

5.1. Capital Budgeting

For our first example we revisit the capital budgeting problem from Hanasusanto et al. (2014) where we must decide which of N projects to pursue to maximize profits. We are constrained by a known budget B, and the cost c_i and profit r_i for each facility i are functions of uncertain parameters $\boldsymbol{\xi}$. We have the option of pursuing a project either before or after these uncertain parameters are observed. If we pursue a project after they are observed then we generate only a fraction $\theta \in [0, 1)$ of the profit - a penalty for delaying the decision. This can be expressed as the adaptive binary optimization problem

$$\max_{z,\mathbf{x}} \quad z$$

$$\begin{aligned}
\text{subject to} \quad & \mathbf{r}(\boldsymbol{\xi}) \cdot \left(\mathbf{x}^1 + \theta \mathbf{x}^2(\boldsymbol{\xi})\right) \geq z && \forall \boldsymbol{\xi} \in \Xi \\
& \mathbf{c}(\boldsymbol{\xi}) \cdot \left(\mathbf{x}^1 + \mathbf{x}^2(\boldsymbol{\xi})\right) \leq B && \forall \boldsymbol{\xi} \in \Xi \\
& \mathbf{x}^1 \in \{0,1\}^N \\
& \mathbf{x}^2(\boldsymbol{\xi}) \in \{0,1\}^N && \forall \boldsymbol{\xi} \in \Xi,
\end{aligned}$$

where $\mathbf{x}^1$ is the first-stage decision to pursue now, and $\mathbf{x}^2$ is the second-stage decision to pursue later.

We will use our finite partitioning to approximate a solution to this fully adaptable problem. We generate instances of size size $N \in \{5, 10, \ldots, 30\}$ using the same parameters as in Hanasusanto et al. (2014), which we reproduce here for completeness. We use four uncertain parameters for all problem sizes N and define the uncertainty set to be $\Xi = \left\{\boldsymbol{\xi} \,\middle|\, \boldsymbol{\xi} \in [-1,1]^4\right\}$. The project costs and profits are, for all i,

$$c_i(\boldsymbol{\xi}) = \left(1 + \frac{1}{2}\boldsymbol{\Phi}_i \cdot \boldsymbol{\xi}\right) c_i^0 \qquad \text{and} \qquad r_i(\boldsymbol{\xi}) = \left(1 + \frac{1}{2}\boldsymbol{\Psi}_i \cdot \boldsymbol{\xi}\right) r_i^0$$

respectively where c_i^0 and r_i^0 are the nominal costs and profits. We sample c_i^0 from the interval $[0, 10]$, and set $r_i^0 = c_i^0/5$. The rows of the matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ are sampled from the unit simplex in $\mathbb{R}^4$, with the budget B set to $\frac{1}{2}\mathbf{e} \cdot \mathbf{c}^0$ and θ set to 0.8.

We used both variants described in Section 2. It is illustrative to consider the size of the adaptable formulation at each iteration for the methods proposed in this paper versus the approach in Hanasusanto et al. (2014). Their approach uses $\mathcal{O}(KN)$ binary variables and $\mathcal{O}(2^K KN)$ continuous variables where K is the number of partitions (selected *a priori*), with fractionality-inducing "big-M" constraints. For both our variants our partitioned problem at iteration k has at most $(1 + 2^{k-1})N$ binary variables (with one auxiliary continuous variable to represent the objective). We do not make any substantial changes in problem structure, just the duplication of the second-stage variables for each partition.

To compare our results with Hanasusanto et al. (2014) we measured performance our method by the same metric: the improvement in objective relative to the solution with no adaptability,

$$\frac{z_{variant}^{k=k_{max}} - z_{variant}^{k=1}}{z_{variant}^{k=1}}. \tag{26}$$

They demonstrate an improvement by this metric of approximately 65% with their two-partition solution, 90% with their three-partition solution, and 110% with their four-partition solution. However, even with only $N = 20$ almost all instances failed to solve to optimality using their method within 2 hours.

We add a new metric, gap, which estimates how far we are from the fully adaptive solution using the bound presented in Section 4.1,

$$\frac{z^k_{upper} - z^k_{variant}}{z^k_{upper}}, \tag{27}$$

where z^k_{upper} is the objective value of the bound problem. Table 1 shows that for problems of size $N = 10$ we can obtain an average improvement of 77% in 3 seconds with the nested variant. Table 2 shows similar results for problems of size $N = 40$ with an 80% improvement in 3 seconds with the nested variant. We attained an average improvement of 110% with the non-nested variant in an average of 388 seconds for $N = 40$, similar to the best improvement demonstrated by Hanasusanto et al. (2014). While our method requires multiple solves and has more binary variables, we suggest the reason our approach is far more tractable is that it preserves the relatively simple problem structure, allowing the MIO solver to branch smarter and generate good solutions using heuristics. We note that the results obtained in Postek and Den Hertog (2014) achieve similar results, although a more precise comparison is difficult due not only to the use of different solver settings and hardware, but that both methods can trade off time/iterations for solution quality.

When comparing the two variants we note that the nested variant delivers improvements similar to the non-nested variant in a much smaller amount of time, but that the biggest improvement over eight iterations was with the non-nested variant. Figures 8 and 9 demonstrate the improvement and gap versus time for $N = 10$ and $N = 40$, and seem to suggest there is little difference between the two in terms of progress made per unit time.

5.2. Multistage Lot Sizing

The multistage lot sizing problem described in Bertsimas and Georghiou (2013) is a challenging AMIO problem that features a mix of continuous and binary decisions across multiple time stages.

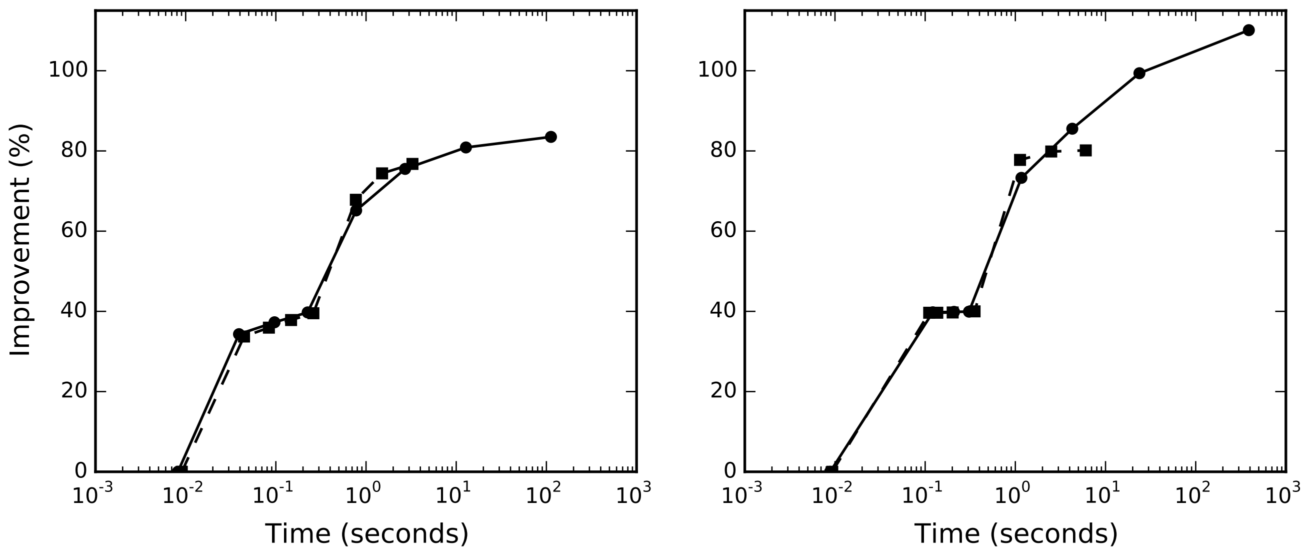

Figure 8 Comparison of rate of improvement in objective for the capital budgeting example of the partitioned problem (Equation (26)) for $N = 10$ (left) and $N = 40$ (right). The nested variant is the dashed line with a square iteration marker, and non-nested variant is a solid line with circular iteration markers.

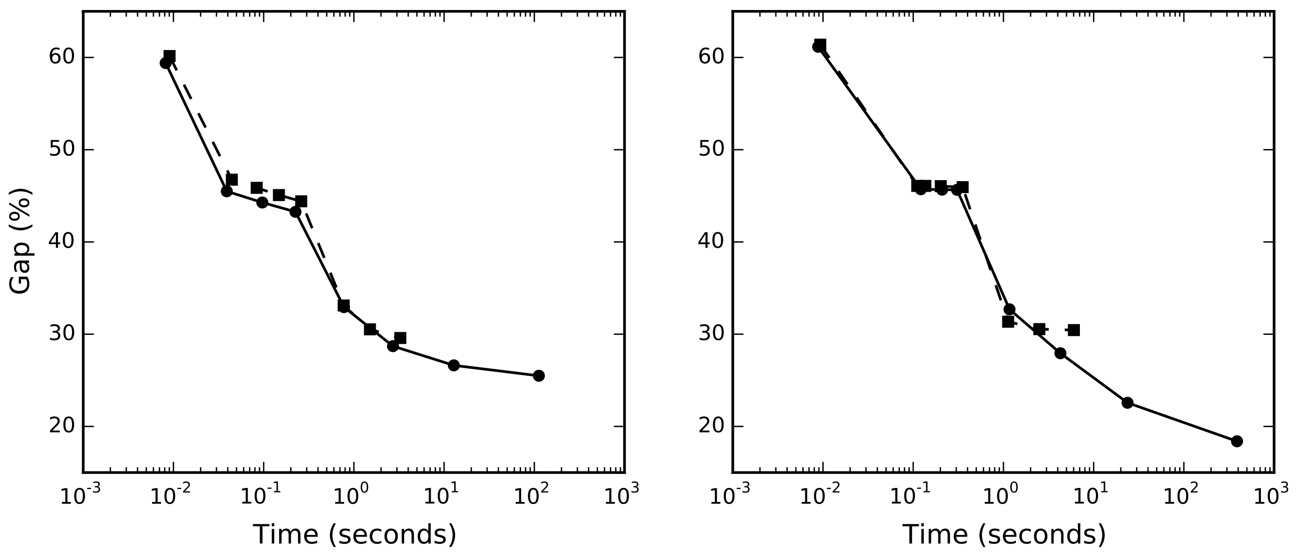

Figure 9 Comparison of rate of improvement in gap (Equation (27)) for the capital budgeting example of the partitioned problem versus the unpartitioned problem ($k = 1$) for $N = 10$ (left) and $N = 40$ (right). The nested variant is the dashed line with a square iteration marker, and non-nested variant is a solid line with circular iteration markers

		Iteration							
$N = 10$	Variant	1	2	3	4	5	6	7	8
Total Time (s)	Non-nested	0.0	0.0	0.1	0.2	0.8	2.7	12.8	112.4
	Nested	0.0	0.0	0.1	0.1	0.2	0.7	1.5	3.2
Improvement (%)	Non-nested	0	34	37	40	65	76	81	83
	Nested	0	34	36	38	40	68	74	77
Gap (%)	Non-nested	62	48	46	45	34	29	27	34
	Nested	62	48	47	46	45	34	31	30

Table 1 Average results over 20 instances of the capital budgeting problem for both variants of the iterative method for the capital budgeting problem with $N = 10$. "Total time" is the total cumulative time to solve the partitioned problems across the iterations. "Improvement" is defined in Equation (26), "gap" in Equation (27).

		Iteration							
$N = 40$	Variant	1	2	3	4	5	6	7	8
Total Time (s)	Non-nested	0.0	0.1	0.2	0.3	1.2	4.3	23.8	388.2
	Nested	0.0	0.1	0.1	0.2	0.4	1.1	2.5	6.0
Improvement (%)	Non-nested	0	40	40	40	73	86	99	110
	Nested	0	40	40	40	40	78	80	80
Gap (%)	Non-nested	62	46	46	46	33	28	23	18
	Nested	62	46	46	46	46	31	31	30

Table 2 Average results over 20 instances of the capital budgeting problem for both variants of the iterative method for the capital budgeting problem with $N = 40$. "Total time" is the total cumulative time to solve the partitioned problems across the iterations. "Improvement" is defined in Equation (26), "gap" in Equation (27).

We have used small versions of this problem throughout this paper as a motivating example. We now state the full problem using the notation of Bertsimas and Georghiou (2013). Our goal is to minimize the cost of stock ordered plus holding costs, subject to the need to satisfy uncertain demand. The deterministic problem, where demand $\boldsymbol{\xi}$ is known, is

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{y},\mathbf{I}} \quad & \sum_{t=2}^{T} \left(c_x x^{t-1} + c_h I^t + \sum_{m=1}^{M} c_m q_m y_m^t \right) && && (28) \\
\text{subject to} \quad & I^{t-1} + x^{t-1} + \sum_{,m=1}^{M} q_m y_m^t - \xi^t = I^t && \forall t \in \{2,\dots,T\} \\
& \sum_{s=1}^{t-1} x^s \le \bar{x}_{tot,t} && \forall t \in \{2,\dots,T\} \\
& I^t \ge 0 && \forall t \in \{2,\dots,T\} \\
& x^{t-1} \ge 0 && \forall t \in \{2,\dots,T\} \\
& \mathbf{y}^t \in \{0,1\}^M, && \forall t \in \{2,\dots,T\}
\end{aligned}$$

where x^t is the (continuous, non-negative) amount of stock ordered at time stage t for delivery at time stage $t+1$ at a unit cost of c_x, y_m^t is a binary decision indicating whether to order a fixed amount q_m of stock for immediate delivery at time t at a unit cost of c_m, and I^t is the level of inventory at time t. We do not allow stock levels to become negative and we impose a unit holding cost c_h for any stock remaining at the end of stage t. The formulation as stated is not suitable for an AMIO formulation as we have demand in an equality constraint, however we can eliminate $\mathbf{I}$ from the problem by observating that

$$I^t = \sum_{s=2}^{t} \left(x^{s-1} + \sum_{m=1}^{M} q_m y_m^s - \xi^s \right).$$

We can now move to the AMIO setting where $\boldsymbol{\xi}$ is uncertain. To compare our methods with Bertsimas and Georghiou (2013) we use the same box uncertainty set

$$\Xi = \left\{ \boldsymbol{\xi} \,\middle|\, \xi^1 = 1,\ l^t \le \xi^t \le u^t \quad \forall t \in \{2,\dots,T\} \right\},$$

with the certain parameters sampled at random from the following ranges or set to fixed values: c_x from $[0,5]$, c_m from $[c_x,10]$, c_h from $[0,10]$, q_m set to $100/M$, $\bar{x}_{tot,t}$ set to $\sum_{s=1}^{t} \bar{x}^s$ where $\bar{x}^s$ is drawn from $[0,100]$, and finally l^t and u^t from $[0,25]$ and $[75,100]$ respectively. As described in Section (2.4) we can mix affine adaptability with our finite partitioning method, which we do here for the continuous decisions x^t. Thus we have x^t restricted to piecewise affine functions of all demand realized by the time of the decision, and the binary decisions y_m^t restricted to piecewise constant functions of the demand. To reduce the number of partitions we use only samples for active constraints and only if they correspond to active partitions (as described in Section 2.5).

We evaluated the performance of both variants over 30 randomly generated instances for $T \in \{2,4,6,8,10\}$, with results sumarized in Table 3 (for $M=2$) and Table 4 (for $M=3$). We used four iterations for both variants, and calculated a gap

$$\frac{z_{upper}^{k_{max}} - z_{lower}^{k_{max}}}{2\left(z_{upper}^{k_{max}} - z_{lower}^{k_{max}}\right)}, \tag{29}$$

using the same formula as in Bertsimas and Georghiou (2013). The initial gap includes affine adaptability for the continuous decisions, but no partitioning. Both variants performed similarly well, substantially tightening the gap and improving over the initial unpartitioned solution. Figure 10 suggests that much of the improvement occurs with only one or two iterations and a relatively small amount of time. The number of partitions (Figure 11) grows slower than the maximum exponential growth that is possible, most likely due to the practical modifications made to the method described above. In constrast to Bertsimas and Georghiou (2013) we find that our method is far more tractable as it solves all problems with four iterations in under 5 minutes. We hypothesize that this is due, like in the the capital budgeting example, to our methods preservation of problem structure. The piecewise linear formulation proposed by Bertsimas and Georghiou (2013) adds fractionality and modifies the problem structure substantially. The similar method used in Postek and Den Hertog (2014) has relatively similar performance for smaller problem sizes but our method appears to have a significantly lower gap for larger values of T, although at the cost of higher computation times.

5.3. Multistage Lot Sizing in a Network

We extend the multistage lot sizing problem of Section 5.2 to the network case, whereby we must now make ordering decisions at N locations rather than 1. We allow stock to be moved instantly between stores at a cost, which significantly grows the size of the problem as we now need $O\left(N^2\right)$

	Initial	Non-nested		Nested	
T	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)
2	24.7	0.0	15.3	0.0	12.0
4	17.1	0.4	11.7	0.4	11.3
6	20.1	9.0	12.6	6.0	14.4
8	19.3	40.4	13.5	42.5	14.4
10	19.5	281.5	12.3	416.9	14.5

Table 3 Results for the multistage lot sizing problem for the non-nested and nested variant for the $M = 2$ case. Gap is calculated using (29): "initial gap" is the gap before any partitioning (at end of iteration 1), "time" is the total time for four iterations, and the variant gaps are the gaps at the end of iteration 4. All numbers are averages.

	Initial	Non-nested		Nested	
T	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)
2	22.0	0.0	13.2	0.0	9.3
4	15.5	0.4	10.3	0.4	9.4
6	15.7	5.8	11.5	5.2	10.8
8	21.0	41.5	15.7	69.0	13.9
10	20.1	313.6	16.2	343.9	12.9

Table 4 Results for the multistage lot sizing problem for the non-nested and nested variant for the $M = 3$ case. Gap is calculated using (29): "initial gap" is the gap before any partitioning (at end of iteration 1), "time" is the total time for four iterations, and the variant gaps are the gaps at the end of iteration 4. All numbers are averages.

variables at each stage. This increase in complexity is also reflected in a richer uncertainty set. The deterministic problem, where demand $\boldsymbol{\xi}$ is known, is

$$\min_{\mathbf{f},\mathbf{x},\mathbf{y},\mathbf{I}} \quad \sum_{t=2}^{T} \left(\sum_{i=1}^{N} \left(c_x x_i^{t-1} + c_h I_i^t + \sum_{m=1}^{M} c_m q_m y_{i,m}^t \right) + c_f \sum_{i,j=1}^{N} f_{ij}^t \right) \tag{30}$$

subject to $$I_i^{t-1} + x_i^{t-1} + \sum_{m=1}^{M} q_m y_{i,m}^t + \sum_{j=1}^{N} \left(f_{ji}^t - f_{ij}^t \right) - \xi^t = I_i^t \qquad \forall i \in \mathcal{N}, t \in \{2,\ldots,T\}$$

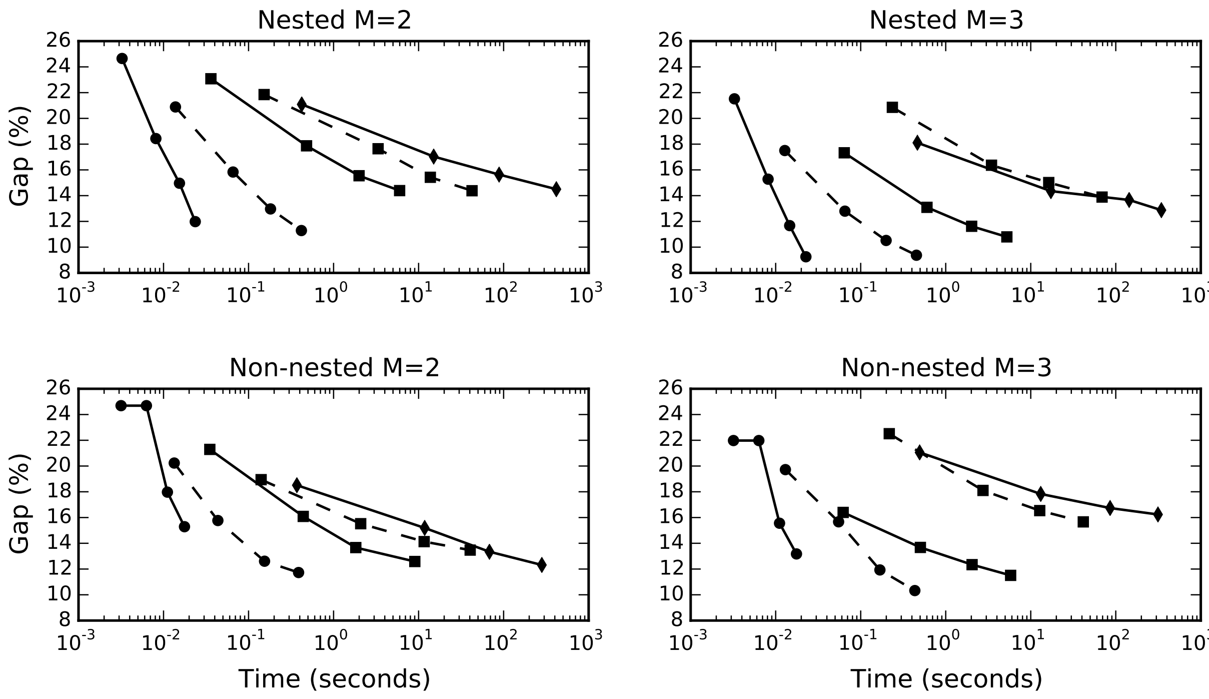

Figure 10 Average gap versus time for the multistage lot sizing problem, with the nested variant (top), non-nested variant (bottom), $M = 2$ (left) and $M = 3$ (right). Lines from left to right represent $T \in \{2, 4, 6, 8, 10\}$.

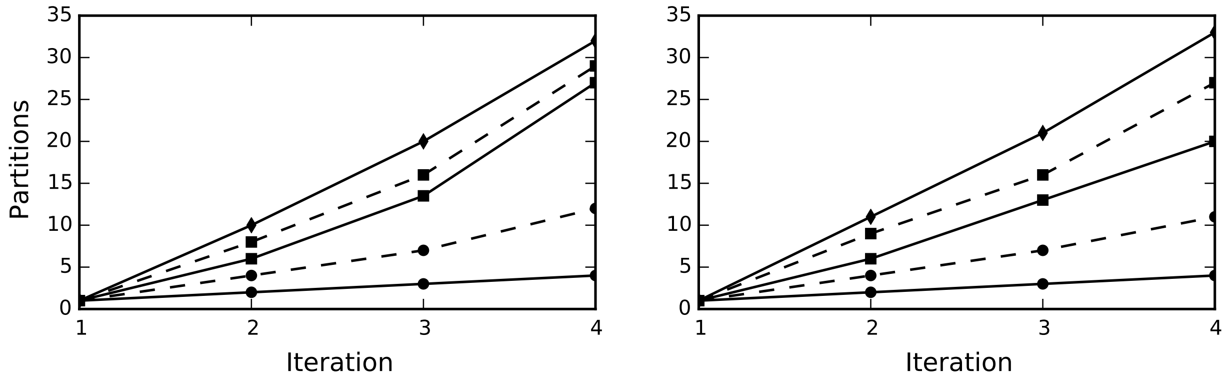

Figure 11 Median number of partitions versus iteration for the multistage lot sizing problem using the nested variant , for $M = 2$ (left) and $M = 3$ (right). Lines from bottom to top represent $T \in \{2, 4, 6, 8, 10\}$.

$$\sum_{s=1}^{t-1} x_i^s \le \bar{x}_{tot,t} \qquad \forall i \in \mathcal{N}, t \in \{2, \dots, T\}$$

$$I_i^t \ge 0 \qquad \forall i \in \mathcal{N}, t \in \{2, \dots, T\}$$

$$x_i^{t-1} \ge 0 \qquad \forall i \in \mathcal{N}, t \in \{2, \dots, T\}$$

$$\mathbf{y}_i^t \in \{0,1\}^M \quad \forall i \in \mathcal{N}, t \in \{2,\dots,T\}$$

$$f_{ij}^t \geq 0 \qquad \forall i,j \in \mathcal{N}, t \in \{2,\dots,T\}$$

where N is the number of locations (indexed by i, $\mathcal{N} = \{1,\dots,N\}$), f_{ij}^t is the amount of stock to move from location i to location j at time t for a per-unit transport cost of c_f, with all other variables and notation as before. For the case of $N=1$ this problem is equivalent to (28). We now extend the uncertainty set of the non-network case to the set

$$\Xi = \left\{ \begin{array}{cc} \xi_i^1 = 1 & \forall i \in \mathcal{N} \\ (1-\alpha)\,\bar{\xi}_i^t \leq \xi_i^t \leq (1+\alpha)\,\bar{\xi}_i^t & \forall i \in \mathcal{N}, \forall t \in \{2,\dots,T\} \\ \sum_i \frac{\left|\xi_i^t - \bar{\xi}_i^t\right|}{\alpha \bar{\xi}_i^t} \leq \Gamma & \forall t \in \{2,\dots,T\} \end{array} \right\},$$

which is similar to the non-networked case but with a limit at every time stage on the deviation from nominal demand across all locations.

Instances are generated in the same manner as the the non-networked case for the parameters that appear in both: c_x, c_m, c_h, q_m, $\bar{x}_{tot,t}$. c_f is sampled from $[0,10]$, $\bar{\xi}_i^t$ is sampled from $[25,75]$, α is set to $\frac{1}{2}$ and Γ to $\sqrt{N}$. As before we restrict $\mathbf{x}$ to be piecewise affine and $\mathbf{y}$ to be piecewise constant, and we additionally restrict $\mathbf{f}$ to be piecewise constant. We again only use samples for active constraints and only if they correspond to active partitions.

Ten instances were generated for each of $T \in \{2,3,4,5,6\}$ and $N \in \{1,2,3,4\}$, and both variants were evaluated on the same set of instances. This problem proved to be much harder than the non-networked lot sizing problem, with problems of size $T=6$, $N \geq 3$ proving to be impossible to solve to integer optimality (with an integrality tolerance of 1%) in the time limit of 1200 seconds. There is some evidence that the nested variant produces lower gaps for cases where all problems solved, and in particular we note that for $N=4$ the nested variant was consistently faster than the non-nested variant.

6. Concluding Remarks

In this paper, we have presented a novel iterative partitioning approach to solving adaptive mixed integer optimization problems. In particular we use finite partitioning to approximate the fully

Non-nested

	N=1		N=2		N=3		N=4	
T	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)
2	0.0	11.9	0.0	15.3	0.1	16.5	0.1	17.4
3	0.0	8.7	0.6	11.7	3.1	12.6	9.1	14.2
4	0.2	8.1	4.9	10.2	32.8	13.0	$182.2^{(9)}$	$15.1^{(9)}$
5	0.7	9.2	28.4	10.8	$300.0^{(8)}$	$13.0^{(8)}$	$1028.3^{(6)}$	$13.8^{(6)}$
6	1.3	9.2	$340.9^{(6)}$	$11.3^{(6)}$	-	-	-	-

Nested

	N=1		N=2		N=3		N=4	
T	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)
2	0.0	9.5	0.0	15.7	0.1	16.3	0.1	16.8
3	0.1	8.3	0.3	9.9	1.1	12.9	7.5	14.2
4	0.1	8.2	2.2	9.9	18.0	12.1	$132.7^{(9)}$	$13.9^{(9)}$
5	0.4	9.0	22.6	10.7	$444.0^{(6)}$	$13.5^{(6)}$	$479.5^{(8)}$	$13.6^{(8)}$
6	1.8	8.4	$312.0^{(8)}$	$11.5^{(8)}$	-	-	-	-

Table 5 Gap as calculated by equation (29) and total time for three iterations of the non-nested and nested variants for the multistage lot sizing problem in a network. Numbers are medians over ten instances for $T \in \{2,3,4,5,6\}$, $N \in \{1,2,3,4\}$. Entries with a superscript in parenthesis represent the number of instances solved within the time limit (1200 seconds). Entries with a dash had no instances solved within the time limit.

adaptable solution, and then use information from the solution of the finitely partitioned problem to further improve the partitioning. The particular partitioning scheme we employ, a Voronoi diagram, is inspired by the observation that a finite partitioning of the uncertainty set can only improve the solution quality if the partitioning separates regions of the uncertainty set that correspond to binding constraints. We find that our method delivers high-quality solutions even as the problem scales in both number of time stages and in the number of decision variables. The time per iteration does increase with the number of time stages, although we observed that the number of partitions created typically grows much slower than the worst-case. The termination criteria we considered, based on an upper and lower bound of the fully adaptive solution, worked well in practice, and can be coupled naturally with a time or iteration limit. The nested variant did not strictly dominate the non-nested variant, but its theoretical properties are attractive and we expect the warm-starting possibility to be very helpful for some AMIO problems where finding a feasible solution is difficult.

We compare our approach with the branch-and-bound technique for integer optimization. In branch-and-bound we use the linear relaxation of an integer optimization problem to provide a lower bound on the true integer solution, and use the information obtained from the solution to branch on a fractional variable (i.e. to partition the problem). We can repeat this process until we are satisfied that the gap between the upper and lower bounds is small enough for our purposes. In our method we approach the AMIO problem from the upper bound by solving a finitely adaptable version of the fully adaptable problem and using the information to further partition the uncertainty set. We show in this paper we can use the information obtained to provide a lower bound, and that we can continue this process until we are satisfied with the gap between bounds. We note that the lower bounds presented to date, including the result in this paper, are good for many problems but they are in need of further improvement. In particular we have observed that the scenario-based lower bound can be weak for some problems, and that a prohibitively large number of scenarios might be required to tighten it. We would draw an analogy again with integer optimization and the ability of cutting planes to significantly tighten the fractionality gap - AMIO has yet to see an equivalent development.

Finally we note that we demonstrated improvements in computational experiments over other methods described in the AMIO literature. These other approaches have either tried to directly optimize the particular partitions used (Hanasusanto et al. 2014) or somewhat equivalently optimize the pieces in a piecewise recourse decision (Bertsimas and Georghiou 2013). Our approach, similar to Postek and Den Hertog (2014), simply picks the partitions based on readily available information, and tries to improve over time with multiple solves. We envision an interesting future direction of work where we pick the partitions using a method like the one described in this paper, and then do some local optimization of the partition structure to extract more better solutions. From the aspect of computational efficiency, it would be interesting to explore ways of further improving the solution time at each iteration by using more information from the previous iteration, much like the relatively small effort required to solve after branching in integer optimization.

References

Aurenhammer, Franz. 1991. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)* **23**(3) 345–405.

Ben-Tal, Aharon, Laurent El Ghaoui, Arkadi Nemirovski. 2009. *Robust optimization*. Princeton University Press.

Ben-Tal, Aharon, Boaz Golany, Arkadi Nemirovski, Jean-Philippe Vial. 2005. Retailer-supplier flexible commitments contracts: a robust optimization approach. *Manufacturing & Service Operations Management* **7**(3) 248–271.

Ben-Tal, Aharon, Alexander Goryashko, Elana Guslitzer, Arkadi Nemirovski. 2004. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming* **99**(2) 351–376.

Bertsimas, Dimitris, David B Brown, Constantine Caramanis. 2011a. Theory and applications of robust optimization. *SIAM review* **53**(3) 464–501.

Bertsimas, Dimitris, Constantine Caramanis. 2010. Finite adaptability in multistage linear optimization. *Automatic Control, IEEE Transactions on* **55**(12) 2751–2766.

Bertsimas, Dimitris, Iain Dunning, Miles Lubin. 2014. Reformulations versus cutting planes for robust optimization. *Optimization Online* .

Bertsimas, Dimitris, Angelos Georghiou. 2013. Design of near optimal decision rules in multistage adaptive mixed-integer optimization. *Optimization Online* .

Bertsimas, Dimitris, Angelos Georghiou. 2014. Binary decision dules for multistage adaptive mixed-integer optimization. *Optimization Online* .

Bertsimas, Dimitris, Vineet Goyal. 2012. On the power and limitations of affine policies in two-stage adaptive optimization. *Mathematical programming* **134**(2) 491–531.

Bertsimas, Dimitris, Dan A Iancu, Pablo A Parrilo. 2010. Optimality of affine policies in multistage robust optimization. *Mathematics of Operations Research* **35**(2) 363–394.

Bertsimas, Dimitris, Dan Andrei Iancu, Pablo A Parrilo. 2011b. A hierarchy of near-optimal policies for multistage adaptive optimization. *Automatic Control, IEEE Transactions on* **56**(12) 2809–2824.

Bertsimas, Dimitris, Eugene Litvinov, Xu Andy Sun, Jinye Zhao, Tongxin Zheng. 2013. Adaptive robust optimization for the security constrained unit commitment problem. *Power Systems, IEEE Transactions on* **28**(1) 52–63.

Chen, Xin, Melvyn Sim, Peng Sun. 2007. A robust optimization perspective on stochastic programming. *Operations Research* **55**(6) 1058–1071.

Chen, Xin, Melvyn Sim, Peng Sun, Jiawei Zhang. 2008. A linear decision-based approximation approach to stochastic programming. *Operations Research* **56**(2) 344–357.

Fischetti, Matteo, Michele Monaci. 2012. Cutting plane versus compact formulations for uncertain (integer) linear programs. *Mathematical Programming Computation* **4**(3) 239–273.

Goulart, Paul J, Eric C Kerrigan, Jan M Maciejowski. 2006. Optimization over state feedback policies for robust control with constraints. *Automatica* **42**(4) 523–533.

Hadjiyiannis, M.J., P.J. Goulart, D. Kuhn. 2011. A scenario approach for estimating the suboptimality of linear decision rules in two-stage robust optimization. *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. 7386–7391.

Hanasusanto, Grani Adiwena, Daniel Kuhn, Wolfram Wiesemann. 2014. Two-stage robust integer programming. *Optimization Online* .

Postek, Krzysztof, Dick Den Hertog. 2014. Multi-stage adjustable robust mixed-integer optimization via iterative splitting of the uncertainty set. *CentER Discussion Paper Series* .

Vayanos, Phebe, Daniel Kuhn, Berç Rustem. 2011. Decision rules for information discovery in multi-stage stochastic programming. *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE, 7368–7373.