STOCHASTIC DUAL DYNAMIC PROGRAMMING AND ITS VARIANTS – A REVIEW

CHRISTIAN FÜLLNER* AND STEFFEN REBENNACK*

Abstract. We provide a tutorial-type review on stochastic dual dynamic programming (SDDP), as one of the state-of-the-art solution methods for large-scale multistage stochastic programs. Since introduced about 30 years ago for solving large-scale multistage stochastic linear programming problems in energy planning, SDDP has been applied to practical problems from several fields and is enriched by various improvements and enhancements to broader problem classes. We begin with a detailed introduction to SDDP, with special focus on its motivation, its complexity and required assumptions. Then, we present and discuss in depth the existing enhancements as well as current research trends, allowing for an alleviation of those assumptions.

Key words. stochastic dual dynamic programming, dynamic programming, multistage stochastic programming, sequential decision problems, large-scale optimization, linear programming, nested Benders decomposition, sampling-based optimization, global optimization

1. Introduction.

In many decision-making situations at least some of the data are uncertain. While this uncertainty is often disregarded, the importance of taking it into account during the decision process was already recognized in 1955 by George Dantzig [48]. In stochastic programming, a common approach to achieve this is to split up this process into two different stages: At the first stage, decisions have to be taken before any uncertain data are revealed and to hedge against the existing uncertainty (so-called *here-and-now* decisions). At the second stage, corrective actions, called *recourse* or *wait-and-see* decisions, can be taken, once the realization of the uncertain data is known [27]. Typically, the aim is to determine an optimal decision rule *in expectation* or with respect to some risk measure.

In many practical applications, not only two, but multiple subsequent decisions have to be taken [7]. If these decisions cannot be taken independently, but are coupled by their effects on a system state, *e.g.*, hydroelectric generation affecting the water level of a reservoir, or orders affecting the size of an inventory stock, this can be modeled as a multistage stochastic problem with several subsequent recourse decisions (this is also referred to as *dynamic programming*, and was recently coined *sequential decision problem* in [177]). In such a problem, trade-offs have to be made between using an existing resource immediately or saving it up for later stages, taking into account the future uncertainty.

Stochastic dual dynamic programming (SDDP) is an algorithm to tackle such multistage stochastic problems in order to compute, or at least approximate, an optimal *policy*, that is, a strategy or decision rule providing the best here-and-now decision as well as the best wait-and-see decisions for any stage and any given realization of the uncertain data. It was first proposed by Pereira and Pinto in 1991 in [160].

Historically, SDDP has its roots in two separate research streams dealing with sequential decision problems. The first one is *stochastic dynamic programming* (SDP), which is closely related to stochastic optimal control and Markov decision processes. Here, a crucial assumption is that the uncertain data on different stages of the decision process are independent of each other (or at least Markovian). In this case, multistage stochastic problems can be expressed by dynamic programming equations (DPE), which decompose the large-scale problem by stages into several smaller subproblems.

*Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Germany (christian.fuellner@kit.edu, steffen.rebennack@kit.edu)

These DPE exploit the famous optimality principle by Bellman [13], which allows one to express the optimal objective value from some stage t onwards, given some state x_{t-1}, recursively by means of some stage-t objective function and a so-called *expected value function* $\mathscr{Q}_t(\cdot)$, modeling the expected optimal objective value from stage $t+1$ onwards, given the new state x_t. We formally introduce these concepts in Section 2.4.

The DPE can be solved exactly by SDP solution methods, such as value iteration [13]. Basically, this method is based on traversing the stages backwards and evaluating the expected value functions $\mathscr{Q}_t(\cdot)$ for all possible states x_{t-1} (concept of a lookup table). Each such evaluation requires solving an optimization problem for all possible realizations of the uncertain data, which, in turn, requires finding an optimal decision over all possible actions. For this evaluation to be possible, it is assumed that the state space, the action space and the scenario space are finite – otherwise they have to be discretized. However, even in the discrete case, enumerating all possible combinations is computationally intractable for all but low dimensions, as the number of evaluations suffers from combinatorial explosion. This phenomenon is known as the *curse of dimensionality* of SDP [176]. In order to circumvent this, *approximate dynamic programming* (ADP) methods have been developed, where expected value functions are approximated instead of being evaluated exactly (or where optimal policies are approximated using different strategies) [176, 177]. SDDP can be regarded as one such method. Due to its close relation with SDP it also heavily relies on the assumption of stagewise independence.

A second perspective on SDDP is one from *stochastic programming*. Traditionally, in this field, multistage uncertain data are often modeled by a scenario tree, which branches at each stage and consists of finitely many possible scenarios. Scenario trees do not require the stochastic data process to be stagewise independent. Using finite scenario trees and assuming linearity, a multistage stochastic program can be reformulated as a large-scale linear programming problem [186]. However, in this extensive form such a problem usually is way too large to be solved by monolithic approaches, since the number of decision variables and constraints grows exponentially in the number of stages. To cope with this challenge, special solution techniques are required which decompose the problem. Based on the L-shaped method for solving two-stage stochastic programs [237] (a special variant of Benders decomposition [17]), one such idea is the extension of Benders-type solution methods to the multistage setting. The *nested Benders decomposition* (NBD) method by Birge [25] is such an extension. It can be interpreted as a nested sequence of solving two-stage stochastic programs while traversing the scenario tree. In contrast to SDP, in NBD the functions $\mathscr{Q}_t(\cdot)$ are not evaluated at all possible states, but iteratively approximated by linear functions called cutting-planes or *cuts*, starting from a rough initial relaxation. Such approximation is possible, since $\mathscr{Q}_t(\cdot)$ can be proven to be convex in x_{t-1} for LPs. It also allows to consider a continuous state space without discretization.

While NBD is a reasonable method to solve multistage stochastic linear programs of moderate time horizons (maximum 4 or 5 time steps), for larger problems, it is still computationally prohibitive, as the scenario tree grows exponentially in the number of stages. As a relief, several methods have been proposed to combine the cutting-plane approximations in NBD with sampling techniques from simulation [41, 57, 109]. The most prominent among these methods is SDDP. From this perspective, SDDP can be considered a sampling-based variant of NBD. In order to use the sampling step in a beneficial way, compared to NBD, SDDP comes with the additional prerequisite that the data process is stagewise independent.

Application-wise, the development of SDDP is closely related to hydrothermal

operational planning, which attempts to determine cost-optimal generation decisions for thermal and hydroelectric power plants over several stages, while ensuring system balance and satisfaction of technical constraints. Since future water availability is affected by uncertain inflows into hydro reservoirs, this optimization problem can be considered multistage, stochastic, and thus very complex.

Prior to SDDP, various solution techniques had been proposed to tackle this type of problem. Among those are simulation models, linear programming techniques (either based on assuming inflows as deterministic or based on reformulating stochastic LPs into a deterministic equivalent), special variants of dynamic programming and SDP [239]. However, all of these techniques either do not consider the uncertain nature of inflows, suffer from the aforementioned curses of dimensionality or do not guarantee convergence. For operating a large-scale power system dominated by hydro power these shortcomings are severe, as they prohibit a cost-minimal and reliable, but at the same time computationally efficient operational planning. The development of SDDP by Pereira and Pinto was directly driven by the endeavor to replace SDP with a more efficient optimization technique in operating the Brazilian power system. While it avoids *some* of the computational drawbacks of SDP or NBD (sometimes advertized as "breaking the curse of dimensionality"), SDDP comes with its own shortcomings, as we thoroughly discuss in this paper.

Since its invention in 1991 SDDP has gained enormous interest, both from a theoretical and an application perspective. To this date, it can be considered one of the state-of-the-art solution methods for large-scale multistage stochastic problems. For this reason, it is used in various practical applications to optimize decision processes, for instance hydrothermal operational planning, portfolio optimization or inventory management, see Section 9.

Several extensions and improvements of SDDP have been proposed by now, many of them attempting to relax the originally required theoretical assumptions, making SDDP applicable to broader problem classes. Others strive for improving the performance of SDDP because, despite its merits, the algorithm may take too long to converge for large problem instances.

Due to both, the sheer amount and the variety of proposed enhancements, SDDP has developed into a wide-ranging research area with several sub-branches, becoming increasingly difficult to keep track of. In this article, we give a comprehensive tutorial-type review on SDDP-related research, covering its basic principle and assumptions, strengths and weaknesses, existing extensions and current research trends.

1.1. Structure. The structure of this review is summarized in Table 1. The review can be divided into four major parts. In the first part (Sections 2 to 8), we discuss the basic mechanism of SDDP. This includes formal preliminaries to formulate multistage stochastic decision problems, but also the main algorithmic steps of SDDP and a complexity analysis. In particular, we point out crucial assumptions for standard SDDP to work. In the second part (Sections 9 and 10), we discuss applications, which underline the practical relevance of SDDP, but also the requirement to relax some of the standard assumptions. In the third part (Sections 11 to 20), we discuss various extensions of SDDP to cases where the standard assumptions are relaxed. These extensions comprise modifications of SDDP itself as well as modifications or reformulations of the considered decision problems. Finally, in the fourth part (Section 21), we discuss approaches to improve the computational performance of SDDP.

Table 1: Table of contents.

1.2. Terminology and Notation. As already mentioned, SDDP is linked to several different research fields and communities, such as stochastic programming, dynamic programming, Markov decision processes, optimal control or reinforcement learning, each using different terminology and notation. This aggravates a presentation of SDDP in a form that is familiar and accessible to all those interested.

To our knowledge, the majority of active research on SDDP is conducted by researchers from the stochastic programming community. For this reason, in many sections we resort to stochastic programming language and notation. On the other hand, this review is also dedicated to offer an access to SDDP for practitioners and researchers from fields in which different perspectives and notation are standard. Therefore, we address these differences if required for the understanding of SDDP, and attempt to avoid heavy mathematical programming notation whenever possible, especially in early sections introducing SDDP.

For a general, not SDDP-specific, attempt at unifying different disciplines related to optimization under uncertainty and sequential decision processes into a common framework, we refer to the excellent book [177].

In the following, we denote random variables by bold letters, *e.g.*, $\boldsymbol{\xi}$, and their realizations by letters in normal font, *e.g.*, ξ. To enhance readability, we summarize

Table 2: Abbreviations that are used throughout the text.

(P)AR	(Periodic) Autoregressive process
DPE	Dynamic programming equations
LP	Linear program
MI(N)LP	Mixed-integer (non-)linear program
MSLP	Multistage stochastic linear programming problem
NBD	Nested Benders Decomposition
RHS	Right-hand side
SDP	Stochastic Dynamic Programming
SDDP	Stochastic Dual Dynamic Programming

some recurring abbreviations in Table 2.

2. Preliminaries for SDDP. In order to present SDDP in its standard form, we start by formally introducing the considered decision problem. In particular, we point out assumptions which are crucial for the presented SDDP method to work.

We consider a multistage decision process where decisions x_t have to be taken over some horizon $[T] := \{1, \dots, T\}$ consisting of T stages, with the aim to minimize some objective function subject to constraints. For now, the horizon T is assumed to satisfy the following condition:

ASSUMPTION 1 (Finite and deterministic horizon). *The number $T \in \mathbb{N}$ of stages is finite and deterministic.*

We discuss later how SDDP may be applied to cases where this is not satisfied, see Sections 19 and 20.

2.1. Modeling the Uncertainty. The data in the considered decision process can be subject to uncertainty, which is revealed over time. To this end, we consider a filtered probability space $(\Omega, \mathscr{F}, \mathbb{P})$ with sample space Ω, σ-algebra $\mathscr{F}$ and probability measure $\mathbb{P}$, which models the uncertainty over the horizon $[T]$. Further let $\mathscr{F}_1, \dots, \mathscr{F}_T$ with $\mathscr{F}_T := \mathscr{F}$ be a sequence of σ-algebras containing the events observable up to time t, thus defining a filtration with $\mathscr{F}_1 \subseteq \mathscr{F}_2 \cdots \subseteq \mathscr{F}_T$, and let Ω_t be the sample space restricted to stage $t \in [T]$. We then define a stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ with random vectors $\boldsymbol{\xi}_t : \Omega_t \to \mathbb{R}^{\kappa_t}, \kappa_t \in \mathbb{N}$, over the probability space. These random vectors are assumed to be $\mathscr{F}_t$-measurable functions. We denote their support by $\Xi_t \subseteq \mathbb{R}^{\kappa_t}$ for all $t \in [T]$. For the first stage, the data are assumed deterministic, *i.e.*, Ξ_1 is a singleton. For each random vector $\boldsymbol{\xi}_t$, we denote a specific realization by ξ_t.

As a crucial ingredient for SDDP to work, we assume that the uncertainty on different stages does not depend on each other.

ASSUMPTION 2 (Stagewise independence). *For all $t \in [T]$, the random vector $\boldsymbol{\xi}_t$ is independent of the history $\xi_{[t-1]} := (\xi_1, \dots, \xi_{t-1})$ of the data process.*

Under Assumption 2, the random vectors $\boldsymbol{\xi}_t$ are often referred to as *noises*. This assumption is common in dynamic programming, but not standard in stochastic programming. In practical applications it may not be satisfied. We address how to apply SDDP to problems with stagewise dependent uncertainty in Section 14.

Additionally, we take the following assumptions for the stochastic process.

ASSUMPTION 3 (Known distribution). *The probability distribution $F_{\boldsymbol{\xi}}$ of the data*

process $(\boldsymbol{\xi}_t)_{t\in[T]}$ *is known.*

ASSUMPTION 4 (Exogeneity). *The random variables* $\boldsymbol{\xi}_t$ *are exogeneous, i.e., the distribution* $F_{\boldsymbol{\xi}}$ *of the data process* $(\boldsymbol{\xi}_t)_{t\in[T]}$ *is independent of decisions* $(x_t)_{t\in[T]}$.

ASSUMPTION 5 (Finite randomness). *The support* Ξ_t *of* $\boldsymbol{\xi}_t$ *is finite for all* $t \in [T]$. *The number of noise realizations at stage* $t \in [T]$ *is given by* $q_t \in \mathbb{N}$ *with* $q_1 = 1$.

We discuss how to apply SDDP if Assumption 3 is not satisfied in Section 13. If Assumption 4 is not satisfied, the problem is said to have *decision-dependent* uncertainty [122]. As this case is not covered in the literature on SDDP so far, we do not discuss the relaxation of this assumption.

Assumption 5 is a key assumption for SDDP and standard in dynamic programming and stochastic programming in order to obtain computationally tractable problems. We discuss possible ways to treat such problems in Section 11. As $\boldsymbol{\xi}_t$ is a discrete and finite random variable for all $t \in [T]$, its distribution $F_{\boldsymbol{\xi}}$ is defined by finitely many realizations $\xi_{tj}, j = 1, \dots, q_t$, and assigned probabilities p_{tj}.

The stagewise independent and finite data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ can be illustrated by a *recombining scenario tree* [186], also called *scenario lattice* [136]. On each stage $t \in [T]$, its nodes represent the possible noise realizations $\xi_{tj}, j = 1, \dots, q_t$. Due to stagewise independence (Assumption 2) all nodes at the same stage have an identical set of child nodes with the same noise realizations and associated probabilities. We call paths $\xi = (\xi_t)_{t\in[T]}$ through the complete tree (stage-T) *scenarios* and index them by $s \in \mathcal{S}$. Note that for each scenario ξ^s, there exists some $j_s \in \{1, \dots, q_t\}$ such that $\xi_t^s = \xi_{tj_s}$. The total number of different scenarios modeled by the tree is $|\mathcal{S}| = \prod_{t\in[T]} q_t$. An example of a recombining scenario tree is presented in Figure 1.

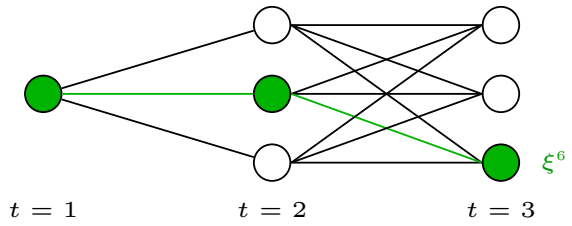


Fig. 1: Recombining tree with 3 realizations per stage and highlighted scenario ξ^6.

2.2. The Decision Process. With the stochastic process in mind, we can now turn to the decision process. At stage 1, the *here-and-now* decision x_1 is taken to hedge against the uncertainty in the following stages. At those stages, recourse decisions $\boldsymbol{x}_t \in \mathbb{R}^{n_t}, n_t \in \mathbb{N}$, can be taken under knowledge of the realization of the data process at stage t. This decision process is illustrated in Figure 2.

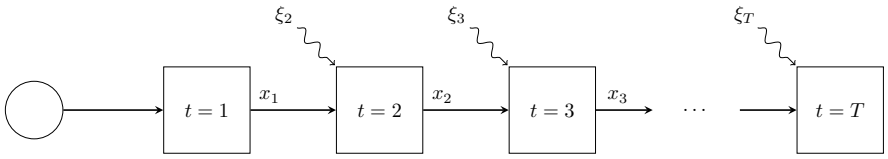


Fig. 2: Multistage decision process with uncertainty.

In other words, the paradigm is that decisions can be taken *after* the uncertainty corresponding to stage t has unfolded (so-called *wait-and-see* decisions), making $\boldsymbol{x}_t(\xi_t)$ a function of $\boldsymbol{\xi}_t$, and by that a random variable. We account for that using a bold symbol. Importantly, $\boldsymbol{x}_t(\cdot)$ does only depend on realizations up to stage t, but does not anticipate future events or decisions. Future events are only considered using distributional information. Therefore, $\boldsymbol{x}_t(\cdot)$ is $\mathscr{F}_t$-measurable [209]. As we will see, $\boldsymbol{x}_t(\cdot)$ may also depend on the choice for $\boldsymbol{x}_{t-1}(\cdot)$ and so on, so that despite stagewise independence (Assumption 2), $\boldsymbol{x}_t(\cdot)$ is actually a function of the whole history $\xi_{[t]}$ of the data process.

A sequence of decision functions $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t\in[T]}$ is called a *policy* and provides a decision rule for all stages $t \in [T]$ and any realization of the data process. By the previous arguments, such a policy is *non-anticipative*, modeling a sequence of nested conditional decisions. The aim of the decision process is to determine an *optimal* policy with respect to a given objective function and a given set of constraints.

In this context, the following assumptions are standard for SDDP.

ASSUMPTION 6 (Linearity). *All functions occurring in the objective and the constraints are linear.*

ASSUMPTION 7 (Consecutive coupling). *Only decisions on consecutive stages can be linked by constraints.*

ASSUMPTION 8 (Risk-neutral policy). *The aim is to determine an optimal risk-neutral policy.*

As not all of these assumptions are guaranteed to be satisfied for an arbitrary problem in practice, we discuss possible ways to relax them in Sections 15 and 16 (for Assumption 6), Section 18 (for Assumption 7) and Section 12 (for Assumption 8).

Under Assumptions 6 and 8, the optimization objective can be expressed as

$$\min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \mathbb{E}\left[\sum_{t\in[T]} \left(\boldsymbol{c}_t(\xi_t)\right)^\top \boldsymbol{x}_t(\xi_{[t]})\right], \tag{2.1}$$

with data vectors $c_t \in \mathbb{R}^{n_t}$ for all $t \in [T]$ and $\mathbb{E}[\cdot]$ denoting the expected value.

Under Assumptions 6 and 7, for all $t \in [T]$, the constraints on the decisions can be expressed using the $\mathscr{F}_t$-measurable set-valued mappings $\mathcal{X}_t(\cdot)$, which for any x_{t-1} and any $\xi_t \in \Xi_t$ are defined by

$$\mathcal{X}_t(x_{t-1}, \xi_t) := \Big\{x_t \in X_t \subset \mathbb{R}^{n_t} \;:\; T_{t-1}(\xi_t)x_{t-1} + W_t(\xi_t)x_t = h_t(\xi_t)\Big\}. \tag{2.2}$$

Here, $h_t \in \mathbb{R}^{m_t}$ are real data vectors (for $m_t \in \mathbb{N}$), T_t and W_t are real-valued ($m_{t+1} \times n_t$) and ($m_t \times n_t$) data matrices and X_t is a non-empty polyhedron, *e.g.*, modeling non-negativity constraints.

As stated before, some (or all) of the problem data can be subject to uncertainty. Hence, for all $t \in [T]$, we consider random variables $\boldsymbol{c}_t(\xi_t), \boldsymbol{T}_{t-1}(\xi_t), \boldsymbol{W}_t(\xi_t)$ and $\boldsymbol{h}_t(\xi_t)$ depending on realizations of $\boldsymbol{\xi}_t$. X_t is considered deterministic. Note again that the first stage is assumed to be deterministic, and that $T_0 \equiv 0$ and $x_0 \equiv 0$. Hence, we define $\mathcal{X}_1 :\equiv \mathcal{X}_1(x_0, \xi_1)$.

Remark 2.1. For notational simplicity, when we deal with finite random variables $\boldsymbol{\xi}_t$, we often index the vectors and matrices c_t, T_{t-1}, W_t and h_t with $j = 1, \ldots, q_t$ if we address specific realizations, *e.g.*, $c_{tj} := c_t(\xi_{tj})$. □

Remark 2.2 (Dynamic programming perspective). In dynamic programming, Markov decision processes or optimal control, usually a slightly different perspective on sequential decision processes is chosen (see [177] for a comprehensive overview). The main difference is that the occurring variables are split into *state variables* and actual *decisions*. State variables $s_t \in S_t$ model the system state at some stage t. S_t is called the *state space*. Importantly, state variables may not only comprise the resource state, but also the information or belief state of a system [177]. Decision variables model local decisions on a stage t given a state s_t. In dynamic programming they are usually discrete and called *actions* $a_t \in A_t(s_t)$, in optimal control they are usually continuous and called *controls* $u_t \in U_t(s_t)$. $A_t(s_t)$ and $U_t(s_t)$ are the *action space* or *control space*, respectively. The actions or controls are what an agent actually decides on given the current state s_t, whereas the new state s_{t+1} is uniquely determined as $s_{t+1} = \mathcal{T}_t(s_t, u_t, \xi_{t+1})$ using a given transition function $\mathcal{T}_t(\cdot)$ which captures the system dynamic. Therefore, from this perspective, a policy is a sequence of mappings $\boldsymbol{\pi}_t : S_t \to U_t$ from the state space to the control (or action) space. By proper modeling of the state variable, Assumption 7 is naturally satisfied.

In our above setting, states and actions are intertwined. We can set $s_t = (x_{t-1}, \xi_t)$ and $u_t = x_t$ to switch perspectives [6]. The state space, control space and transition function are then implicitly given by (2.2) and the definition of $\boldsymbol{\xi}_t$.

Whereas our above definitions are prevalent in the literature on SDDP, sometimes also an optimal control perspective is adopted, *e.g.*, in the French community working on SDDP (see for example [85]). However, in this case usually only the resource state r_t is explicitly considered as a state variable (while not including information on $\boldsymbol{\xi}_t$). Translating our above setting, this implies that $r_t = x_{t-1}$ with state space $R_t = X_t$, $u_t = x_t$ and due to $r_{t+1} = u_t$, both the control space $U_t(r_t, \xi_t)$ and the transition function $\mathcal{T}_t(r_t, u_t, \xi_t)$ are given by the equations in (2.2).

It is worth mentioning that the distinction between state variables and controls (actions) is not only a matter of notation, though, but also relevant computationally because the complexity of SDDP differs in the state and control dimension (see also Remark 2.6 and Subsection 4.2). □

Given the constraint sets (2.2) for all $t \in [T]$, let $\mathcal{X}_0 := \{x_0\}$ and recursively define

$$\mathcal{X}_t := \bigcup_{x_{t-1} \in \mathcal{X}_{t-1}} \bigcup_{\xi_t \in \Xi_t} \mathcal{X}_t(x_{t-1}, \xi_t)$$

for all $t \in [T]$ [75]. Using these definitions, we are able to state assumptions which we require for the feasibility of our decision problem:

ASSUMPTION 9. *(Feasibility and Compactness)*

(a) For all $t \in [T]$, all $x_{t-1} \in \mathcal{X}_{t-1}$ and almost all $\xi_t \in \Xi_t$, the set $\mathcal{X}_t(x_{t-1}, \xi_t)$ is a non-empty compact subset of $\mathbb{R}^{n_t}$ (relatively complete recourse).

(b) The set $\mathcal{X}_t$ is bounded for all $t \in [T]$.

Remark 2.3. Note that the linearity assumption (see Assumption 6), immediately implies that Assumption 9 (a) is not only satisfied for all $x_{t-1} \in \mathcal{X}_{t-1}$, but for all $x_{t-1} \in \operatorname{conv}(\mathcal{X}_{t-1})$, where $\operatorname{conv}(S)$ denotes the convex hull of a set S. □

The set $\mathcal{X}_t \in \mathbb{R}^{n_t}$ is called *reachable set* in [75] and *effective feasible region* in [124]. It may as well be referred to as the state space sometimes, because in our setting x_t also takes the role of a state variable. However, in other cases the larger polyhedral set X_t may be called state space.

The boundedness of $\mathcal{X}_t$ in (b) is required for some of the convergence results on SDDP presented in Section 4. It follows naturally if X_t is bounded, since $\mathcal{X}_t \subseteq X_t$. Property (a) is convenient, but not necessarily required. We discuss possible ways to relax it in Section 17.

With all the ingredients defined, we can now model the decision problem in a form that can be tackled by SDDP. Based on its properties, in the following we refer to this problem as a *multistage stochastic linear programming problem* (MSLP). If not specified otherwise, throughout this paper, we assume that (MSLP) satisfies Assumptions 1 to 9. We first discuss two different modeling approaches which are common in the literature.

2.3. Single-problem Formulation. One way to model the decision problem (MSLP) is to formulate it as a single optimization problem. This modeling approach is common in the stochastic programming community. The optimization problem can be obtained by combining (2.1) with the constraints in (2.2) for all $t \in [T]$.

Then, under Assumptions 1 to 9, (MSLP) can be written as

$$v^* := \begin{cases} \min\limits_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} & \mathbb{E}\left[\sum\limits_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right] \\ \text{s.t.} & x_1 \in \mathcal{X}_1 \\ & \boldsymbol{x}_t(\xi_{[t]}) \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_{[t]}\ \forall t = 2, \ldots, T \\ & \boldsymbol{x}_t(\cdot)\ \mathscr{F}_t\text{-measurable} \quad \forall t = 2, \ldots, T. \end{cases} \tag{2.3}$$

Importantly, the decision variables $x_t \in \mathbb{R}^{n_t}$ depend on $\boldsymbol{\xi_t}$ (and on x_{t-1}), so in this representation we optimize over policies. A policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ is called *feasible* (or *admissible*) if it satisfies the constraints in (MSLP) for almost every realization of the random data [209].

Assumption 9 (a) implies that the feasible set of (MSLP) is compact and non-empty, and by linearity of the objective (Assumption 6) it follows that v^* is finite.

Due to optimizing over policies, without Assumption 5, (MSLP) is an infinite-dimensional optimization problem. With Assumption 5, however, it can be reformulated to a more accessible form. More precisely, it can be reformulated to a large-scale deterministic problem, the so-called *deterministic equivalent* of (MSLP) in *extensive form* (see [209]). To this end, let $\mathcal{S}$ denote the set of all (stage-T) scenarios. Then, for each scenario $s \in \mathcal{S}$ a separate copy x_t^s of variables x_t can be introduced, so that the optimization over implementable policies translates to an optimization over a finite number of decision variables. However, the problem size grows exponentially in the number of stages T. Therefore, even for a finite number of scenarios, this large-scale LP is too large to be solved by off-the-shelf solvers for all but very small instances.

A preferable solution approach is therefore to use tailored solution techniques which decompose (MSLP) into smaller subproblems. Note that from Assumption 7 and the definition of $\mathcal{X}_t(\cdot)$ in (2.2), it is evident that the constraints of (MSLP) are block-diagonal, as only consecutive stages are coupled in the constraints. This is visualized in Figure 3.

This sequential and block-diagonal structure can be exploited to achieve the required decomposition. This is crucial for the derivation of SDDP. Interestingly, this decomposition idea directly leads to the second common modeling approach for our decision problem.

2.4. Dynamic Programming Equations. An alternative, but equivalent way to model (MSLP) is to exploit the well-known optimality principle by Bellman [13] and

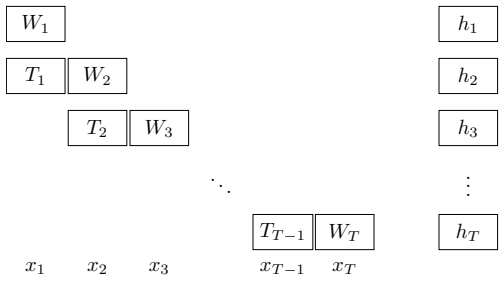


Fig. 3: Block-diagonal structure of constraints in (MSLP).

to formulate a recursion of so-called *dynamic programming equations* (DPE), where a multistage decision process with stagewise independent (or Markovian) uncertainty is modeled as a coupled sequence of optimization problems.

Whereas this modeling approach is often applied in stochastic programming as a way to reformulate and decompose the single problem (2.3) into a computationally tractable form, in dynamic programming it often serves as the starting point of modeling decision problems. However, in contrast to many approaches in dynamic programming we do not discretize x_t, see also Section 5.1.

Under Assumptions 1 to 9, for $t = T, \dots, 2$, the DPE are given by

$$Q_t(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \mathscr{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t), \end{cases} \tag{2.4}$$

where

$$\mathscr{Q}_{t+1}(x_t) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1})\right] \tag{2.5}$$

and $\mathscr{Q}_{T+1}(x_T) \equiv 0$. $Q_t(\cdot, \cdot)$ is called *value function* and $\mathscr{Q}_t(\cdot)$ is called *expected value function*, *(expected) cost-to-go function*, *future cost function* or *recourse function*. For the first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathscr{Q}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{2.6}$$

For a formal proof of the equivalence of (2.3) and its DPE, we refer to [209] and Section 12. Importantly, in subproblem (2.4) x_t is a deterministic variable and not a function because a fixed realization of ξ_t is considered.

We should emphasize that the equivalence of (2.3) and its DPE does not require Assumption 5. This implies that also the DPE (2.4)-(2.6) are computationally intractable in case of general continuous random variables. While the subproblems are deterministic and finite-dimensional, there exist infinitely many value functions $Q_t(\cdot, \cdot)$ and the evaluation of $\mathscr{Q}_t(\cdot)$ requires the evaluation of (multidimensional) integrals. Therefore, also from this perspective Assumption 5 is crucial.

Remark 2.4 (Dynamic programming control perspective). Recall Remark 2.2. Using a distinction between state variables r_t and controls u_t, the DPE to (MSLP) can be formulated as

$$Q_t(r_t, \xi_t) = \min_{u_t \in U_t(r_t, \xi_t)} f_t(u_t, \xi_t) + \mathscr{Q}_{t+1}(\mathcal{T}_t(r_t, u_t, \xi_t)). \tag{2.7}$$

Bellman Operator. In the French literature on SDDP, in addition to taking the optimal control perspective discussed in Remarks 2.2 and 2.4, a more formal way to define the DPE is prevalent, see [75, 126] for instance. To this end, a linear Bellman operator $\widehat{\mathfrak{B}}_t$ is introduced, which applied to some lower semicontinuous function $V : \mathbb{R}^{n_t} \to \mathbb{R} \cup \{+\infty\}$ is defined as [75]

$$\widehat{\mathfrak{B}}_t(V)(x_{t-1}, \xi_t) := \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} \big(c_t(\xi_t)\big)^\top x_t + V(x_t), \tag{2.8}$$

i.e., it maps (x_{t-1}, ξ_t) to the optimal value of an optimization problem containing function $V(\cdot)$. We can then further define the operator

$$\mathfrak{B}_t(V)(x_{t-1}) := \mathbb{E}\Big[\widehat{\mathfrak{B}}_t(V)(x_{t-1}, \boldsymbol{\xi}_t)\Big]. \tag{2.9}$$

Setting V to $\mathscr{Q}_t(\cdot)$ for $t = 2, \dots, T$, the (expected) value functions can then be recursively defined in a very compact form. We summarize the different notations for a better overview:

$$\boxed{\begin{array}{c} \widehat{\mathfrak{B}}_t(\mathscr{Q}_{t+1})(x_{t-1}, \xi_t) = Q_t(x_{t-1}, \xi_t) \\ \mathfrak{B}_t(\mathscr{Q}_{t+1})(x_{t-1}) = \mathscr{Q}_t(x_{t-1}) \end{array}}$$

In the remainder of this work, we stick to notation (2.4), as it is most common in the literature on SDDP which we reference in this paper.

We obtain the following properties for the DPE which are standard for SDDP:

LEMMA 2.5. *Under Assumptions 1 to 9, for the DPE defined by (2.4)-(2.6) the following properties hold:*

(a) We have relatively complete recourse, i.e., *for any $x_{t-1} \in \mathcal{X}_{t-1}$, the stage-$t$ subproblem (2.4) is feasible for all $\xi_t \in \Xi_t$.*

(b) The value functions $Q_t(\cdot, \cdot)$ and expected value functions $\mathscr{Q}_t(\cdot)$ are finite-valued on $\mathrm{conv}(\mathcal{X}_{t-1})$ for all $t = 2, \dots, T$ and all $\xi_t \in \Xi_t$.

(c) Problem (2.6) is feasible and bounded.

Remark 2.6. In addition to Remark 2.2, we should highlight that (MSLP) (both, in single-problem formulation (2.3) and DPE (2.4)-(2.6)) can be straightforwardly enhanced with local decision variables $y_t \in Y_t$ and local constraints, not appearing in different stages. In principle, they can even be incorporated without changes to our models by extending the dimension of the (state) variables x_t and adapting the matrices T_t and W_t accordingly. However, as we explain in Section 4, the complexity of SDDP grows exponentially in the dimension of the state space, so this is computationally detrimental and should be avoided. Instead, purely local variables and constraints should be handled separately from the ones we introduced above. This approach is referred to as *generalized dual dynamic programming* (GDDP) in [18].

While almost every practical application will require the introduction of these additional elements, in this work, for the most part we restrict to coupling variables and constraints which are required to illustrate the mechanics of SDDP. ◻

Remark 2.7. In general, the local objective functions may also include the states x_{t-1} instead of only depending on x_t and ξ_t. For notational simplicity, we consider a less general form of the objective function. ◻

2.5. Approximations of the Value Functions. The main challenge in exploiting the DPE to solve (MSLP) is that the (expected) value functions are not

known in analytical form in advance. The key idea in SDDP is to iteratively approximate them from below using linear functions, which are called *cutting-planes*, or short *cuts*. Together, these linear functions build polyhedral outer approximations $\underline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$ for all $t = 2, \dots, T$, which we refer to as *cut approximations*. In that regard, SDDP can be considered as a special variant of Kelley's cutting-plane method [118] and closely related to Benders decomposition [17], see also Section 5.2. Note that in contrast to SDP this avoids a state discretization, as $Q_t(\cdot, \cdot)$ and $\mathscr{Q}_t(\cdot)$ do not have to be evaluated at all possible states, but only at well-chosen trial points where new cuts are constructed, cf. Section 5.1.

For this approximation by cuts, the following properties are crucial.

THEOREM 2.8 ([27]). *Let $x_{t-1} \in conv(\mathcal{X}_{t-1})$. Then, under Assumptions 1 to 9, for all $t = 2, \dots, T$ and a given noise realization ξ_t, the value function $Q_t(\cdot, \xi_t)$*

(a) is piecewise linear and convex in (h_t, T_{t-1}),
(b) is piecewise linear and concave in c_t,
(c) is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.

The main idea here is that given the definition of $\mathcal{X}_{t-1}(\cdot)$ in (2.2), h_t, T_{t-1} and x_{t-1} do only appear in the right-hand side (RHS) of problem (2.4). Therefore, the dual feasible set is independent of those elements. It possesses finitely many extreme points. This assures piecewise linearity of $Q_t(\cdot, \cdot)$, as known from parametric optimization. The convexity follows with the linearity (Assumption 6) and all vectors and matrices being part of convex sets.

Theorem 2.8 directly implies the piecewise linearity and convexity of $\mathscr{Q}_t(\cdot)$.

COROLLARY 2.9 ([27]). *Under Assumption 5 and the premises of Theorem 2.8, for all $t = 2, \dots, T$, $\mathscr{Q}_t(\cdot)$ is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.*

Theorem 2.8 and Corollary 2.9 also directly imply the Lipschitz continuity of the (expected) value functions.

COROLLARY 2.10. *Under Assumptions 1 to 9, for all $t = 2, \dots, T$ and all $\xi_t \in \Xi_t$, $Q_t(\cdot, \xi_t)$ and $\mathscr{Q}_t(\cdot)$ are Lipschitz continuous on $conv(\mathcal{X}_{t-1})$.*

Replacing the true expected value functions with cut approximations in (2.4), we can define *approximate value functions*

$$\underline{Q}_t(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \underline{\mathcal{V}}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t). \end{cases} \tag{2.10}$$

Trivially, for $\mathscr{Q}_{T+1}(\cdot) \equiv 0$, we have $\underline{\mathcal{V}}_{T+1}(\cdot) \equiv 0$.

Note that apart from x_{t-1} and ξ_t, $\underline{Q}_t(\cdot, \cdot)$ is also a function of the cut approximation $\underline{\mathcal{V}}_{t+1}(\cdot)$. This is especially relevant when these approximations are iteratively updated in SDDP, leading to different approximate value functions. Using the Bellman operators defined in (2.8)-(2.9) this can be expressed in a very concise way:

$$\underline{Q}_t(\cdot, \cdot) = \mathfrak{B}_t(\underline{\mathcal{V}}_{t+1})(\cdot, \cdot).$$

Similarly, we could express this by adding an argument to $\underline{Q}_t(\cdot, \cdot)$, *i.e.*, by writing $\underline{Q}_t(x_{t-1}, \xi_t \,|\, \underline{\mathcal{V}}_{t+1})$ or $\underline{Q}_t(\underline{\mathcal{V}}_{t+1})(x_{t-1}, \xi_t)$. However, for notational simplicity, we do not state this explicitly, but when dealing with SDDP use the iteration index i for distinction. This means that $\underline{Q}_t^i(\cdot, \cdot)$ indicates that $\underline{Q}_t(\cdot, \cdot)$ is considered with cut approximation $\underline{\mathcal{V}}_{t+1}^i(\cdot)$.

We summarize the different notations for a better overview:

$$\boxed{\begin{aligned}\widehat{\mathfrak{B}}_t(\underline{\mathcal{V}}_{t+1})(x_{t-1},\xi_t) &= \underline{Q}_t(x_{t-1},\xi_t)\\ \mathfrak{B}_t(\underline{\mathcal{V}}_{t+1})(x_{t-1}) &= \underline{\mathscr{Q}}_t(x_{t-1}) := \mathbb{E}_{\boldsymbol{\xi_t}}\left[\underline{Q}_t(x_{t-1},\boldsymbol{\xi_t})\right]\end{aligned}} \tag{2.11}$$

Finally, we can observe that given that the cut approximations $\underline{\mathcal{V}}_{t+1}(\cdot)$ are polyhedral, the approximate value functions $\underline{Q}_t(\cdot,\cdot)$ inherit the previously stated properties from $Q_t(\cdot,\cdot)$. In particular:

LEMMA 2.11. *Let $\underline{\mathcal{V}}_{t+1}(\cdot)$ be a polyhedral function. Then, under Assumptions 1 to 9, for all $t = 2,\dots,T$ and a given noise realization ξ_t, $\underline{Q}_t(\cdot,\xi_t)$ is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.*

On the other hand, as they are polyhedral, the cut approximations $\underline{\mathcal{V}}_t(\cdot)$ for $t = 2,\dots,T$ are *nonlinear* functions. Importantly for computations, subproblems (2.10) can still be formulated as LPs by using a partial epigraph reformulation and the fact that $\underline{\mathcal{V}}_t(\cdot)$ is defined as the maximum of finitely many affine functions (modeled by some set $\mathcal{K}$ with $|\mathcal{K}| \in \mathbb{N}$):

$$\underline{Q}_t(x_{t-1},\xi_t) = \begin{cases} \min\limits_{x_t,\theta_{t+1}} & \left(c_t(\xi_t)\right)^\top x_t + \theta_{t+1}\\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1},\xi_t)\\ & -(\beta^i_{t+1,k})^\top x_t + \theta_{t+1} \geq \alpha^i_{t+1,k}, \quad \forall i\ \forall k \in \mathcal{K}. \end{cases} \tag{2.12}$$

This LP contains an additional decision variable θ_{t+1} and finitely many additional linear constraints indexed by i and k. The structure and indexing of these constraints become clear in the next section when we present the cut generation process for SDDP.

3. Standard SDDP.

We are now able to introduce SDDP in its standard form.

3.1. Main Principle.

SDDP consists of two main steps in each iteration i, a *forward pass* and a *backward pass* through the stages $t \in [T]$.

In each forward pass, using the approximate value functions $\underline{Q}^i_t(\cdot,\cdot)$ (recall that this implies using cut approximation $\underline{\mathcal{V}}^i_{t+1}(\cdot)$ in (2.10)), a sequence of *trial points* $(x_t)_{t\in[T]}$ is generated, at which then new cuts are constructed in the following backward pass to improve the approximation. These trial points are also called *incumbents* or *candidate* solutions, and their sequence is called a *state trajectory* (especially in optimal control). The idea behind this approach is that the approximate value functions implicitly define a feasible (suboptimal) policy for problem (MSLP). The trial points are generated by evaluating this policy for one or several scenarios which are sampled from $\mathcal{S}$, *i.e.*, by solving the respective subproblems. This has the advantage that cuts are constructed at points which (at least for some scenario) are optimal given the current cut approximation. This step can also be interpreted as a Monte Carlo *simulation* of the current policy.

In the backward pass, subgradients of $\mathscr{Q}_t(\cdot)$ at the trial points are used to construct cuts, passing them back to the previous stage and updating $\underline{\mathcal{V}}^i_t(\cdot)$ to $\underline{\mathcal{V}}^{i+1}_t(\cdot)$ for all $t = 2,\dots,T$. This way, if not optimal, the current policy is amended (at least if the *right* scenario is sampled). In this step, also a *true* lower bound $\underline{v}$ for v^* is determined.

Remark 3.1 (Statistical learning perspective). The basic principle of SDDP can also be interpreted from a perspective of supervized learning as *learning* a policy (or expected value functions $\mathscr{Q}_t(\cdot)$ for all $t = 2,\dots,T$) or *training* a model of this

policy (or cut approximations $\underline{\mathcal{V}}_t(\cdot)$ for all $t = 2, \ldots, T$) using backpropagation. In the forward pass the inputs are propagated through the stages using the current model, and in the backward pass cuts (representing the error of the current approximation) are propagated back through the stages to update the model. □

Algorithm 3.1 provides a pseudo-code for SDDP. We now provide a more detailed and technical look at the algorithmic steps.

Algorithm 3.1 SDDP

Input: Problem (MSLP) satisfying Assumptions 1 to 9. Bounds $\underline{\theta}_t, t = 2, \ldots, T$. Stopping criterion.

`Initialization`

Initialize cut approximations with $\theta_t \geq \underline{\theta}_t$ for all $t = 2, \ldots, T$.
Initialize lower bound with $\underline{v}^0 = -\infty$.
Set iteration counter to $i \leftarrow 0$.

`SDDP Loop`

while Stopping criterion not satisfied **do**
  Set $i \leftarrow i + 1$.

  `Forward Pass`

  Sample a subset $\mathcal{K} \subseteq \mathcal{S}$ of scenarios.
  Solve the approximate first-stage problem (3.1) to obtain trial point $x_1^i = x_1^{ik}$ for all $k \in \mathcal{K}$.
  for stages $t = 2, \ldots, T$ **do**
    for samples $k \in \mathcal{K}$ **do**
      Solve the approximate stage-t subproblem (2.10) associated with $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k)$ to obtain trial point x_t^{ik}.
    end for
  end for

  `Backward Pass`

  for stages $t = T, \ldots, 2$ **do**
    for samples $k \in \mathcal{K}$ **do**
      for noise terms $j = 1, \ldots, q_t$ **do**
        Solve the updated approximate stage-t subproblem (2.10) associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj})$. Store the optimal value and dual vector π_t^{ikj}.
      end for
      Use relations (3.4)-(3.5) and (3.7) to create an optimality cut for $\mathscr{Q}_t(\cdot)$.
      Update the cut approximation $\underline{\mathcal{V}}_t^i(\cdot)$ to $\underline{\mathcal{V}}_t^{i+1}(\cdot)$ using relation (3.6).
    end for
  end for
  Solve the approximate first-stage problem (3.8) to obtain a lower bound $\underline{v}^i$.
end while

Output: (Approximately) optimal feasible policy for (MSLP) defined by x_1^i and cut approximations $\underline{\mathcal{V}}_t^i(\cdot), t = 2, \ldots, T$. x_1^i defines an (approximately) optimal solution to problem (2.6) with $\overline{v}_{\mathcal{K}}^i \approx v^*$.

3.2. Forward Pass. At the start of each iteration i, at first a subset $\mathcal{K} \subseteq \mathcal{S}$ of scenarios is sampled with $|\mathcal{K}| \ll |\mathcal{S}|$ (note that we may equivalently sample stage by stage during the forward pass). The number of samples $|\mathcal{K}|$ may vary by iteration, but we do not state this possible dependence explicitly. Traditionally, and most commonly, in SDDP some random sampling is used, but also a deterministic sampling is possible. We further discuss sampling techniques in Section 6.

Then, at the first-stage, the approximate subproblem

$$\min_{x_1 \in \mathcal{X}_1(x_0)} c_1^\top x_1 + \underline{\mathcal{V}}_2^i(x_1). \tag{3.1}$$

is solved, which yields the trial point $x_1^i = x_1^{ik}$ for all $k \in \mathcal{K}$. Afterwards, for each stage $t = 2, \ldots, T$ and each sample $k \in \mathcal{K}$, recursively the approximate value functions $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k)$ are evaluated (this means that the subproblems (2.10) are solved for x_{t-1}^{ik}, ξ_t^k and the current cut approximation $\underline{\mathcal{V}}_{t+1}^i(\cdot)$). This way, for each sample $k \in \mathcal{K}$, a sequence of trial points $(x_t^{ik})_{t \in [T]}$ is obtained.

The forward pass of SDDP is illustrated in Figure 4 for the recombining scenario tree from Figure 1 and $\mathcal{K} = \{1, 3, 9\}$, *i.e.*, $|\mathcal{K}| = 3$. The three sampled scenario paths are highlighted in green. The figure shows that for sample paths ξ^3 and ξ^9 the same node is reached at stage 3.

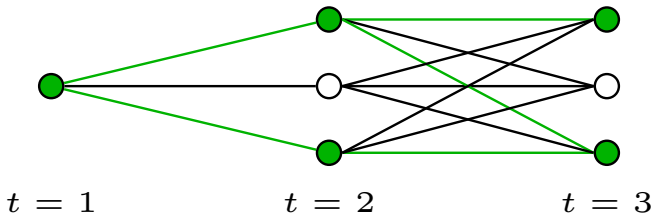


Fig. 4: Illustration of SDDP forward pass for $|\mathcal{K}| = 3$.

Remark 3.2 (Initialization). Before the first backward pass, the cut approximations $\underline{\mathcal{V}}_t(\cdot)$ are not initialized yet, so the forward pass subproblems (3.1) and (2.10) are not well-defined. This can be addressed using different strategies. First, in iteration $i = 1$ the forward pass can be skipped, and a feasible state trajectory $(x_t^1)_{t \in [T]}$ for the backward pass can be user-defined or taken randomly instead. Second, such a state trajectory can be computed heuristically via a greedy approach where $\underline{\mathcal{V}}_t^1(\cdot) \equiv 0$ is assumed in the forward pass subproblems for all $t \in [T]$, so no coupling exists in the objective. Third, the cut approximations $\underline{\mathcal{V}}_t(\cdot)$ may be initialized with a valid user-defined lower bound $\underline{\theta}_t$ for all $t \in [T]$. This approach is taken in the description in Algorithm 3.1. ∎

3.3. Backward Pass. Main Principle. The backward pass starts at stage T. Here, for all samples $k \in \mathcal{K}$, we consider subproblems (2.10) for the trial point x_{T-1}^{ik} computed in the forward pass, all noise realizations $\xi_{Tj}, j = 1, \ldots, q_T$, and $\underline{\mathcal{V}}_{T+1}^{i+1}(\cdot) \equiv 0$. That is, we consider functions $\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{tj})$ for $j = 1, \ldots, q_T$.

As $\underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ is convex in x_{T-1} by Lemma 2.11, it can be underestimated by a linear function using some subgradient $\beta_{Tkj}^i \in \partial \underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ for any $j = 1, \ldots, q_T$

and any $k \in \mathcal{K}$:

$$\underline{Q}_T^{i+1}(x_{T-1}, \xi_{Tj}) \geq \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) + (\beta_{Tkj}^i)^\top (x_{T-1} - x_{T-1}^{ik}).$$

Since $\underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ is a lower approximation of the true value function $Q_T(\cdot, \xi_{Tj})$, this directly implies

$$Q_T(x_{T-1}, \xi_{Tj}) \geq \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) + (\beta_{Tkj}^i)^\top (x_{T-1} - x_{T-1}^{ik}).$$

Taking expectations with respect to $\boldsymbol{\xi}_T$ on both sides, we obtain

$$\begin{aligned}
&\mathscr{Q}_T(x_{T-1}) \\
&\geq \mathbb{E}_{\boldsymbol{\xi}_T}\left[\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \boldsymbol{\xi}_T)\right] + \mathbb{E}_{\boldsymbol{\xi}_T}\left[(\boldsymbol{\beta}_{Tj}^i)^\top (x_{T-1} - x_{T-1}^{ik})\right] \\
&= \mathbb{E}_{\boldsymbol{\xi}_T}\left[\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \boldsymbol{\xi}_T) - (\boldsymbol{\beta}_{Tk}^i)^\top x_{T-1}^{ik}\right] + \left(\mathbb{E}_{\boldsymbol{\xi}_T}\left[\boldsymbol{\beta}_{Tk}^i\right]\right)^\top x_{T-1} \\
&= \underbrace{\sum_{j=1}^{q_T} p_{Tj} \left(\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) - (\beta_{Tkj}^i)^\top x_{T-1}^{ik}\right)}_{=:\alpha_{Tk}^i} + \underbrace{\left(\sum_{j=1}^{q_T} p_{Tj} \beta_{Tkj}^i\right)}_{=:\beta_{Tk}^i}^\top x_{T-1},
\end{aligned} \tag{3.2}$$

where we exploit the finiteness of $\boldsymbol{\xi}_T$ (Assumption 5). α_{Tk}^i is called *cut intercept* and β_{Tk}^i is called *cut gradient*. Defining

$$\phi_{Tk}^i(x_{T-1}) := \alpha_{Tk}^i + (\beta_{Tk}^i)^\top x_{T-1},$$

we can express (3.2) as

$$\mathscr{Q}_T(x_{T-1}) \geq \phi_{Tk}^i(x_{T-1}). \tag{3.3}$$

Inequality (3.3) defines a cut for $\mathscr{Q}_T(\cdot)$. Such a cut is constructed for each $k \in \mathcal{K}$. With these new cuts, the cut approximation $\underline{\mathcal{V}}_T^i(\cdot)$ is updated to

$$\underline{\mathcal{V}}_T^{i+1}(x_{T-1}) := \max\left\{\underline{\mathcal{V}}_T^i(x_{T-1}),\ \phi_{T1}^i(x_{T-1}), \ldots, \phi_{T|\mathcal{K}|}^i(x_{T-1})\right\}.$$

Thus, assuming that $|\mathcal{K}|$ does not change over the iterations, $\underline{\mathcal{V}}_T^{i+1}(\cdot)$ consists of $i|\mathcal{K}|$ affine functions $\phi_{Tk}^i(\cdot)$, cf. formulation (2.12).

In the same way, for stages $t = T-1, \ldots, 2$, cuts for $\mathscr{Q}_t(\cdot)$ can be constructed by solving subproblems (2.10) for the trial points x_{t-1}^{ik} and all noise realizations $\xi_{tj}, j = 1, \ldots, q_t$. Importantly, by going backwards through the stages, at stage t we can already factor in the cuts that have been constructed at the following stage $t+1$, thus using a better approximation as the basis to construct a new cut. This means that we consider $\underline{\mathcal{V}}_{t+1}^{i+1}(\cdot)$ and by that $\underline{Q}_t^{i+1}(\cdot, \cdot)$ with index $i+1$ in the backward pass of iteration i.

As for stage T, we obtain

$$\mathscr{Q}_t(x_{t-1}) \geq \underbrace{\sum_{j=1}^{q_t} p_{tj} \left(\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_t) - (\beta_{tkj}^i)^\top x_{t-1}^{ik}\right)}_{=:\alpha_{tk}^i} + \underbrace{\left(\sum_{j=1}^{q_t} p_{tj} \beta_{tkj}^i\right)}_{=:\beta_{tk}^i}^\top x_{t-1}, \tag{3.4}$$

where β^i_{tkj} denotes a subgradient of $\underline{Q}^{i+1}_t(\cdot, \xi_{tj})$ at x^{ik}_{t-1} for $k \in \mathcal{K}, j = 1, \dots, q_t$. Again, by defining

$$\phi^i_{tk}(x_{t-1}) := \alpha^i_{tk} + (\beta^i_{tk})^\top x_{t-1},$$

we can obtain a cut

$$\mathscr{Q}_t(x_{t-1}) \geq \phi^i_{tk}(x_{t-1}) \tag{3.5}$$

for each $k \in \mathcal{K}$ and can update the cut approximation to

$$\underline{\mathcal{V}}^{i+1}_t(x_{t-1}) := \max\left\{\underline{\mathcal{V}}^i_t(x_{t-1}),\ \phi^i_{t1}(x_{t-1}), \dots, \phi^i_{t|\mathcal{K}|}(x_{t-1})\right\}. \tag{3.6}$$

Computing Subgradients. So far, we have discussed the main idea of the cut generation process in the backward pass of SDDP, which is based on evaluating approximate value functions $\underline{Q}^{i+1}_t(\cdot, \cdot)$ and using subgradients for them at trial points x^{ik}_{t-1}. For the interested reader, we now address in more detail how to compute those subgradients. This step uses *dual information*, *i.e.*, it is based on the duality theory of convex programs. For simplicity, we assume $X_t = \{x_t \in \mathbb{R}^{n_t} \ : \ x_t \geq 0\}$ for all $t \in [T]$.

Consider stage T, some $k \in \mathcal{K}$ and some $j \in \{1, \dots, q_T\}$. Then, the dual problem to the linear stage-T subproblem (2.10) is

$$\begin{cases} \max\limits_{\pi_T} & \left(h_{Tj} - T_{T-1,j} x^{ik}_{T-1}\right)^\top \pi_T \\ \text{s.t.} & W^\top_{Tj} \pi_T \leq c_{Tj}. \end{cases}$$

Let π^{ikj}_T be an optimal dual basic solution. Such solution does always exist by relatively complete recourse and boundedness (see Assumption 9 and Lemma 2.5). By strong duality of linear programs, it follows

$$\begin{aligned} \underline{Q}^{i+1}_T(x^{ik}_{T-1}, \xi_{Tj}) &= \left(h_{Tj} - T_{T-1,j} x^{ik}_{T-1}\right)^\top \pi^{ikj}_T \\ &= -(\pi^{ikj}_T)^\top T_{T-1,j} x^{ik}_{T-1} + (\pi^{ikj}_T)^\top h_{Tj}. \end{aligned}$$

Importantly, the dual feasible set does not depend on x_{T-1}, but remains unchanged for all trial points. In particular, π^{ikj}_T is always dual feasible, but not necessarily dual optimal for all x_{T-1}. Therefore, and because of minimization, it follows

$$\begin{aligned} \underline{Q}^{i+1}_T(x_{T-1}, \xi_{Tj}) &\geq -(\pi^{ikj}_T)^\top T_{T-1,j} x_{T-1} + (\pi^{ikj}_T)^\top h_{Tj} \\ &= -(\pi^{ikj}_T)^\top T_{T-1,j}(x_{T-1} + x^{ik}_{T-1} - x^{ik}_{T-1}) + (\pi^{ikj}_T)^\top h_{Tj} \\ &= \underline{Q}^{i+1}_T(x^{ik}_{T-1}, \xi_{Tj}) - (\pi^{ikj}_T)^\top T_{T-1,j}(x_{T-1} - x^{ik}_{T-1}). \end{aligned}$$

Hence,

$$\beta^i_{Tkj} = -(\pi^{ikj}_T)^\top T_{T-1,j}$$

is a subgradient of $\underline{Q}^{i+1}_T(\cdot, \xi_{Tj})$ at x^{ik}_{T-1}.

The previous derivation provides some additional insight. Since the dual feasible set is polyhedral and does not depend on x_{T-1}, for each noise term $\xi_{Tj}, j = 1, \dots, q_T$, there exist only finitely many dual extreme points (dual basic solutions) that can be attained. Therefore, only finitely many different cut coefficients can be generated. This is crucial for some convergence proofs of SDDP, as we discuss later.

For earlier stages $t = T-1, \dots, 2$, the dual problem to subproblem (2.10) looks a bit more sophisticated, as the cut approximations $\underline{\mathcal{V}}_{t+1}^{i+1}(\cdot)$ have to be taken into account, which requires additional dual multipliers ρ_t^r for all cuts $r \in \Gamma_{t+1}$, where Γ_{t+1} denotes the index set of cuts generated for the following stage. However, the derivation is completely analogous and, again, we arrive at

$$\underline{Q}_t^{i+1}(x_{t-1}, \xi_{tj}) \geq \underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj}) - (\pi_t^{ikj})^\top T_{t-1,j}(x_{t-1} - x_{t-1}^{ik}),$$

so that

$$\beta_{tkj}^i = -(\pi_t^{ikj})^\top T_{t-1,j} \tag{3.7}$$

is a subgradient of $\underline{Q}_t^{i+1}(\cdot, \xi_{tj})$ at x_{t-1}^{ik}. Interestingly, the optimal dual multipliers ρ_t^{rikj} are not explicitly required in this formula.

3.4. Bounds and Stopping. At the first stage, the subproblem

$$\underline{v}^i := \min_{x_1 \in \mathcal{X}_1(x_0)} c_1^\top x_1 + \underline{\mathcal{V}}_2^{i+1}(x_1). \tag{3.8}$$

is solved. As $\underline{\mathcal{V}}_2^{i+1}(\cdot)$ is a lower approximation of $\mathcal{Q}_2(\cdot)$, $\underline{v}^i$ is a valid lower bound to the optimal value v^* of (MSLP). This bound can be initialized with $\underline{v}^0 = -\infty$ or any a priori known lower bound for v^*.

In contrast, we are not guaranteed to obtain a valid upper bound for v^* during iterations of standard SDDP, as we only consider a small subset $\mathcal{K} \subseteq \mathcal{S}$ of all scenarios. This means that in the forward pass, the feasible policy for (MSLP), which is implicitly defined by the current cut approximations $\underline{\mathcal{V}}_t^i(\cdot), t = 2, \dots, T$, is only evaluated for a subset of all scenarios. By evaluating these scenarios in the objective of (MSLP) and taking the sample average

$$\overline{v}_{\mathcal{K}}^i := \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \underbrace{\sum_{t=1}^{T} \big(c_t(\xi_t^k)\big)^\top x_t^{ik}}_{=: v^i(\xi^k)} \tag{3.9}$$

we only obtain an unbiased estimator of the true upper bound $\overline{v}^i$ (a *statistical upper bound*) associated with the current policy, see Section 7 for more details.

After each iteration of SDDP, one or several stopping criteria are checked, which may or may not be based on $\overline{v}_{\mathcal{K}}^i$. We discuss different stopping criteria in detail in Section 7. If SDDP does not stop, a new iteration $i+1$ is started with a forward pass.

3.5. Cut Properties. We discuss convergence of SDDP in Section 4. It relies on three key properties of the derived cuts:

LEMMA 3.3. *For any stage $t = 2, \dots, T$ and any $k \in \mathcal{K}$, the functions $\phi_{tk}^i(\cdot)$ are*
(a) valid *lower approximations of $\mathcal{Q}_t(\cdot)$,*
(b) tight *for $\underline{\mathcal{Q}}_t^{i+1}(\cdot)$ (as defined in (2.11)) at x_{t-1}^{ik},*
(c) finite*, i.e., only finitely many different cuts can be generated, if we restrict to dual basic solutions to generate cuts.*

Proof. Property (a) follows immediately from (3.3) and (3.5). (b) holds because of strong duality for LPs and taking expected values over the obtained optimal values. Alternatively, we can rearrange the RHS of inequality (3.4) to obtain

$$\phi_{tk}^i(x_{t-1}) = \underline{\mathcal{Q}}_t^{i+1}(x_{t-1}^{ik}) + \sum_{j=1}^{q_t} p_{tj} (\beta_{tkj}^i)^\top (x_{t-1} - x_{t-1}^{ik}). \tag{3.10}$$

Inserting x_{t-1}^{ik} yields $\phi_{tk}^{i}(x_{t-1}^{ik}) = \underline{\mathscr{Q}}_t^{i+1}(x_{t-1}^{ik})$. Property (c) follows by induction using the arguments on the dual feasible region previously discussed for stage T. ∎

Note that $\phi_{tk}^{i}(\cdot)$ is not necessarily tight for the true expected value function $\mathscr{Q}_t(\cdot)$ in early iterations for $t \neq T$, but might provide a loose cut only. However, by the finiteness and tightness properties it can be shown recursively, that eventually the derived cuts become tight for $\mathscr{Q}_t(\cdot)$ as well. In fact, after finitely many steps, the polyhedral function $\mathscr{Q}_t(\cdot)$ is represented exactly for all $t = 2, \ldots, T$. This is a key property for the convergence of SDDP.

3.6. Illustrative Example. To illustrate the key steps of SDDP, we present a simple example.

EXAMPLE 3.4. *Consider the 3-stage (MSLP)*

$$\begin{aligned} \min \quad & x_1 + x_2 + x_{31} + x_{32} \\ \text{s.t.} \quad & x_1 \leq 6 \\ & x_2 \geq \xi_2 - x_1 \\ & x_{31} - x_{32} = \xi_3 - x_2 \\ & x_1, x_2, x_{31}, x_{32} \geq 0, \end{aligned} \tag{3.11}$$

which is inspired by Example 2 in Chapter 5 of [27]. The uncertain data in the RHS is stagewise independent and uniformly distributed with $\xi_2 \in \{4, 5, 6\}$ *and* $\xi_3 \in \{1, 2, 4\}$.

Problem (3.11) *has not entirely the same structure as problem (MSLP), but can be easily converted to it by introducing slack variables. However, for illustrative purposes, we abstain from this. The problem can be expressed by means of the value functions*

$$Q_3(x_2, \xi_3) = \begin{cases} \min\limits_{x_3} & x_{31} + x_{32} \\ \text{s.t.} & x_{31} - x_{32} = \xi_3 - x_2 \\ & x_{31}, x_{32} \geq 0 \end{cases} \tag{3.12}$$

and

$$Q_2(x_1, \xi_2) = \begin{cases} \min\limits_{x_2} & x_2 + \mathscr{Q}_3(x_2) \\ \text{s.t.} & x_2 \geq \xi_2 - x_1 \\ & x_2 \geq 0. \end{cases}$$

The first-stage problem then is

$$v^* = \begin{cases} \min\limits_{x_1} & x_1 + \mathscr{Q}_2(x_1) \\ s.t. & x_1 \in [0, 6]. \end{cases}$$

The optimal solution is given by $x_1^* = 3$ *with* $v^* = \frac{53}{9}$.

As shown in [27], the stage-3 value functions can be written in closed-form as $Q_3(x_2, \xi_3) = |\xi_3 - x_2|$ *for all scenarios. Taking expectations, a closed-form expression*

for $\mathscr{Q}_3(\cdot)$ can be derived, and by recursion we obtain

$$\mathscr{Q}_2(x_1) = \begin{cases} \frac{23}{3} - \frac{16}{9}x_1, & x_1 \in [0,1] \\ \frac{67}{9} - \frac{10}{9}x_1, & x_1 \in [1,2] \\ \frac{59}{9} - \frac{10}{9}x_1, & x_1 \in [2,3] \\ \frac{47}{9} - \frac{2}{3}x_1, & x_1 \in [3,4] \\ \frac{31}{9} - \frac{2}{9}x_1, & x_1 \in [4,5] \\ \frac{7}{3}, & x_1 \in [5,6]. \end{cases}$$

The optimal value is $v^* = \frac{56}{9}$.

We apply SDDP for illustration. We assume loose initial bounds $\theta_2, \theta_3 \geq -10$ for simplicity. In the forward pass, we sample one scenario path per iteration, i.e., $|\mathcal{K}| = 1$. In iteration 1, let $(\xi_2, \xi_3) = (5, 4)$ define this path. Solving the approximate subproblems (2.10) for all stages $t = 1, 2, 3$ and $(\xi_2, \xi_3) = (5, 4)$, we obtain $\overline{v}^i_{\mathcal{K}} = 6$. In fact, this is no valid upper bound for v^*.

In the backward pass, cuts for $\mathscr{Q}_t(\cdot), t = 2, 3$, are derived at the trial points. For stage 3, the cut gradient is $\beta_3(5) = 1$. Moreover, $\underline{\mathscr{Q}}^2_3(5) = \frac{8}{3}$. With formulas (3.5) and (3.10) this yields the cut $\mathscr{Q}_3(x_2) \geq -\frac{7}{3} + x_2$, which is incorporated into the stage-2 subproblems. Solving these problems yields the cut $\mathscr{Q}_t(x_1) \geq \frac{23}{3} - 2x_1$. At the first stage, the lower bound computes to $\underline{v}^1 = \frac{5}{3}$.

The expected value functions and the obtained cuts for three iterations are depicted in Figure 5. In the second and the third iteration, the same scenario path $(\xi_2, \xi_3) = (6, 1)$ is sampled in the forward pass.

Figure 6 displays the bounds $\underline{v}^i$ and $\overline{v}^i_{\mathcal{K}}$ for ten iterations of SDDP. It shows that the lower bounds stabilize quickly at v^*, whereas the values of $\overline{v}^i_{\mathcal{K}}$ oscillate around v^*.

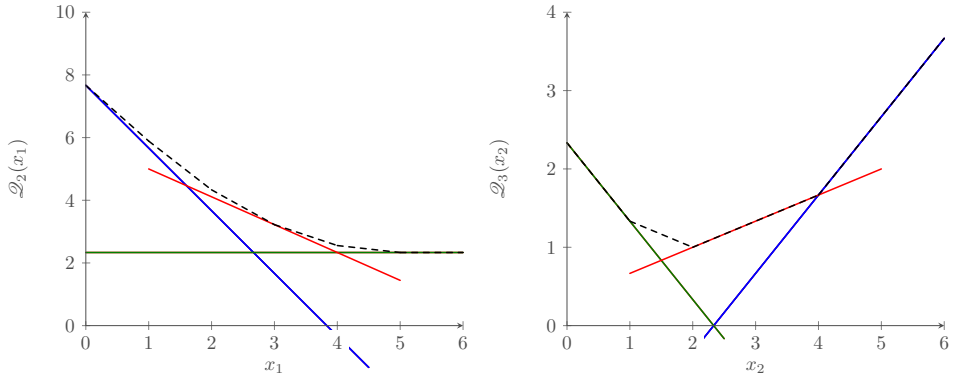


Fig. 5: Expected value functions for Example 3.4 with cuts obtained in first three iterations depicted in blue, green and red.

3.7. Policy Assessment. As mentioned before, in standard SDDP no valid upper bound $\overline{v}$ for v^* is determined. While in each iteration a statistical upper

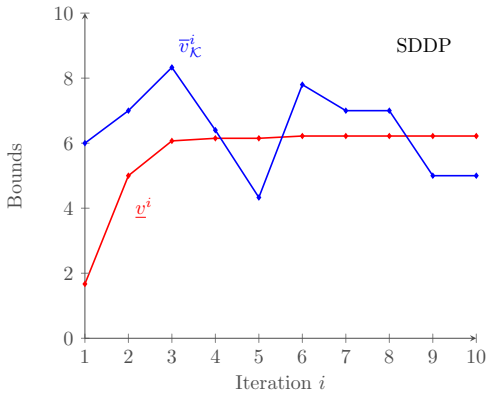


Fig. 6: Bounds for 10 iterations of SDDP applied to Example 3.4.

bound (3.9) can be computed, the number of samples $|\mathcal{K}|$ may often be too small to appropriately assess the quality of the current policy. In particular, $|\mathcal{K}|$ is often chosen to be 1 in practice, and thus $\overline{v}^i_{\mathcal{K}}$ is not a meaningful estimate for $\overline{v}$.

Therefore, to assess the obtained policy, usually an additional forward simulation is conducted once SDDP has terminated. For this simulation a much higher number of sample paths through the scenario tree is used, *e.g.* $|\mathcal{K}| \in \{1000, 10000\}$, leading to a reasonable estimator $\overline{v}_{\mathcal{K}}$. In this step, the simulation can be either performed *in-sample* (using sample paths through the recombining scenario tree) or *out-of-sample* (using the true underlying distribution, *e.g.*, if $\boldsymbol{\xi}_t$ is a continuous random variable that is discretized to satisfy Assumption 5, see Section 11).

Remark 3.5. In the light of Remark 3.1 this policy assessment step can also be interpreted from a statistical learning perspective. After the model has been trained, a model validation (using in-sample data) or a model test (using out-of-sample data) are performed. □

4. Convergence and Complexity. The convergence behavior of SDDP has been thoroughly analyzed over the years. We discuss the main convergence results in this section. We first focus on *finite* convergence of SDDP, and then afterwards discuss the actual convergence *rate*, *i.e.*, the computational complexity of SDDP. Our overview is loosely based on the literature review in [75].

4.1. Finite Convergence. The first convergence analyses related to SDDP have been conducted in [41] and [132], however implicitly assuming independence of sampled random variables and convergent subsequences of algorithm iterates. A first complete convergence proof is given by Philpott and Guan in [171] for the case where uncertainty only enters the RHS of (MSLP) (in fact, they consider a more general algorithm than SDDP, including sampling in the backward pass). The same reasoning is used by Shapiro [206] for the case where also W_t, c_t and T_{t-1} are uncertain.

The convergence behavior of SDDP can be explained using two main arguments: First, as stated in Lemma 3.3, only finitely many different cuts, and by that only finitely many different cut approximations $\underline{\mathcal{V}}_t(\cdot)$ can be constructed for all $t = 2, \dots, T$. This result requires linearity (Assumption 6) and finite random variables (Assumption 5). Moreover, these finitely many cuts also satisfy some tightness property, which implies that they are sufficient to exactly represent the polyhedral (expected) value

functions (see Theorem 2.8). For a deterministic algorithm, this would result in finite convergence to the true optimal point and value (see the convergence properties of Benders decomposition [17] and Kelley's cutting-plane method [118]).

For SDDP, it has to be taken into account that scenarios are sampled in the forward pass. This means that the cut approximations might not further improve for some iterations if the *wrong* scenarios are sampled. Therefore, the second key argument for many proofs of finite convergence of SDDP is that each scenario is visited infinitely many times with probability 1 given that the algorithm does not terminate. Intuitively, this means that after finitely many iterations the *right* scenarios will be sampled with probability 1, leading to the construction of a new cut. This requirement is satisfied under *independent sampling*, that is, if the sampling in the forward pass of Algorithm 3.1 is random and independent of previous iterations. It is also satisfied for an exhaustive enumeration of all scenarios in the sampling process. We should emphasize that this argument is purely theoretical in order to establish convergence results for SDDP. When applying SDDP in practice, it is usually not even possible to sample each scenario *once* in reasonable time.

Using these two arguments, the following main convergence result can be obtained

THEOREM 4.1 (Almost sure finite convergence of SDDP). *Under Assumptions 1 to 9 and using an independent random sampling procedure in the forward pass, SDDP converges with probability 1 to an optimal policy of (MSLP) in a finite number of iterations.*

Importantly, almost sure finite convergence to an optimal policy of (MSLP) does not imply that the trajectories $(x_t^{ik})_{t\in[T]}, k \in \mathcal{K}$, and the corresponding sample averages $\overline{v}^i_{\mathcal{K}}$ obtained in SDDP converge, as both are random and depend on the current sample $\mathcal{K}$. However, the lower bounds $\underline{v}^i$ obtained in SDDP converge to v^*.

Deterministic Sampling. Recently, convergence analyses of SDDP and related algorithms have often made use of deterministic sampling techniques instead of random sampling in line 6 of Algorithm 3.1 [10, 11]. Here, the idea is that the approximation error in SDDP can be controlled and guided to zero in a deterministic way if in each iteration scenarios are sampled for which the current approximation gap is maximized. This requires, however, that the approximation gap itself can be bounded rigorously. Therefore, in addition to the lower cut approximation $\underline{\mathcal{V}}_t(\cdot)$ also an upper approximation $\overline{\mathcal{V}}_t(\cdot)$ is constructed and iteratively refined [10, 241], so that deterministic lower bounds $\underline{v}^i$ and upper bounds $\overline{v}^i$ are computed in each iteration. For more details on deterministic sampling and deterministic upper bounds we refer to Sections 6 and 8.

Generalizations. It has been shown that some of the basic assumptions (Assumptions 1 to 9) can be relaxed without compromising convergence of SDDP. Girardeau et al. [85] analyze the case where SDDP is applied to multistage problems with nonlinear convex subproblems, *i.e.*, Assumption 6 is relaxed. In this case, the value functions $Q_t(\cdot)$ are no longer polyhedral, but still convex. The authors show that almost sure convergence is still satisfied as long as some convexity and compactness assumptions and some tightened recourse assumption are satisfied. We discuss this result in detail in Section 15 when we formally introduce convex multistage stochastic nonlinear problems. The main idea is that even without polyhedrality, $Q_t(\cdot)$ can be guaranteed to be Lipschitz continuous, so that the approximations of $\mathscr{Q}_t(\cdot)$ get better in a whole neighborhood of the trajectories $(x_t^{ik})_{t\in[T]}, k \in \mathcal{K}$.

Guigues generalizes this convergence result to the risk-averse case where Assumption 8 is relaxed [92]. Forcier and Leclère prove convergence for (MSLP) without finite

randomness, *i.e.*, dropping Assumption 5. Further convergence proofs are provided for multi-cut SDDP [8], SDDP with cut selection [8, 94], adaptive partition-based SDDP [215] (see also Section 21), using SDDP with saddle cuts [59] (see also Section 14) and variants of distributionally robust SDDP [69, 169] (see also Section 13), Another proof of almost sure finite convergence for extensions to non-convex problems is provided in [241].

4.2. Complexity. Theorem 4.1 guarantees almost sure finite convergence of SDDP. While this result is of theoretical interest, it may not be very relevant in practical applications, as it provides no result on the rate of convergence. As pointed out in [75] and mentioned before, especially the argument of scenarios being sampled repeatedly (infinitely many times) is almost never applicable to SDDP in practice due to the sheer amount of scenarios in $\mathcal{S}$. Important for the rate of convergence are the computational cost per iteration and the required number of iterations.

Cost per Iteration. For the computational cost per iteration, the number of LPs to be solved in the backward pass is crucial. Per sample $k \in \mathcal{K}$ in the forward pass, q_t subproblems are solved for each stage except for $t = 1$ in the backward pass. Therefore, the total number of LPs solved is $1 + |\mathcal{K}| \sum_{t=2}^{T} q_t$. Hence, the number of problems to be solved grows linearly in the number of stages T, in the number of samples $|\mathcal{K}|$ and in the number of noise terms q_t [186].

Expected Number of Iterations. The computational bottleneck for SDDP is the expected required number of iterations to achieve convergence. Recently, there has been active research on computing theoretical bounds on this number, with Lan [124] as well as Zhang and Sun [241] publishing similar results using slightly different approaches. In both cases, the authors start by considering some case of deterministic sampling before enhancing their results to the random sampling variant of SDDP (in [124] the associated algorithm is referred to as *explorative dual dynamic programming* (EDDP)). We discuss deterministic sampling in more detail in Section 6. The main idea to derive iteration bounds is the following: By exploiting Lipschitz continuity of $\underline{\mathcal{V}}_t(\cdot)$ and $\mathscr{Q}_t(\cdot)$, it is possible to control the approximation error also at points where no cuts are constructed, as long as they lie in a neighborhood of some trial point x_t^{ik}. As long as the state space is bounded for all $t \in [T]$ (cf. Assumption 9), it can be completely covered by finitely many such neighborhoods [241]. A similar reasoning is applied in [75].

More formally, Lan [124] introduces the notion of *saturated points* $\bar{x}_{t-1}$, in which the approximation of $\mathscr{Q}_t(\cdot)$ is already ε-close for some predefined tolerance $\varepsilon > 0$, *i.e.*,

$$\mathscr{Q}_t(\bar{x}_{t-1}) - \underline{\mathcal{V}}_t^i(\bar{x}_{t-1}) \leq \varepsilon,$$

and *distinguishable points* $\bar{x}_{t-1}$, which have at least a δ-distance to the set X_{t-1}^{sat} of already saturated points for some $\delta > 0$, that is

$$\|\bar{x}_{t-1} - x_{t-1}\| > \delta, \quad \forall x_{t-1} \in X_{t-1}^{sat}.$$

If some trial point x_t^{ik} is saturated and distinguishable, the iteration i can be called *effective* [75]. Using deterministic sampling, all iterations in SDDP can be shown to be effective, and thus the number of iterations can be bounded in the aforementioned way. For random sampling, this is not true, but the probability for an effective iteration is at least $\frac{1}{N}$ with $N := \Pi_{t=2}^{T-1} n_t$.

In the light of Assumption 9 (b), for any $t \in [T]$, we call the bound D_t satisfying

$$\|x_t - x_t'\| \leq D_t, \ \forall x_t, x_t' \in \mathcal{X}_t$$

the *diameter* of the state space. Additionally, let L denote a Lipschitz constant for the objective function of (MSLP), which exists due to Corollary 2.10.

Then, the following complexity results are satisfied by SDDP.

THEOREM 4.2 (Complexity of SDDP [124, 241]). *Let $D_t \leq D$ for all $t \in [T]$. For some arbitrary $\varepsilon > 0$, the (expected) number of required iterations of SDDP (Algorithm 3.1) to obtain*

- *an ε-optimal solution using deterministic sampling is*
  - *polynomial in $T, (\frac{1}{\varepsilon}), L$ and D,*
  - *exponential in n_t,*
- *an ε-optimal solution using deterministic sampling, given that $\mathcal{X}_t$ is finite with cardinality $|\mathcal{X}_t| \leq \overline{X}$, is*
  - *linear in T and $\overline{X}$*
- *a $(T\varepsilon)$-optimal solution using deterministic sampling is*
  - *linear in T,*
  - *polynomial in $T, (\frac{1}{\varepsilon}), L$ and D,*
  - *exponential in n_t,*
- *an ε-optimal solution using random sampling is*
  - *polynomial in $q_t, (\frac{1}{\varepsilon}), L$ and D,*
  - *exponential in T and n_t.*

This means that for standard SDDP (using random sampling) the expected number of iterations grows exponentially in the horizon T and the dimension n_t of the state space. This is computationally important. The exponential complexity with respect to the state dimension is not that surprising, as it is well-known for cutting-plane methods [153] and inherited by SDDP. Similarly, the exponential complexity with respect to the number of stages directly follows from the exponential number of scenarios that may have to be sampled in the worst-case. Interestingly, under deterministic sampling, the complexity is independent of the number q_t of noise terms per stage, as this number only affects the computational cost per iteration.

We see that using some deterministic sampling scheme a polynomial or even linear iteration complexity in T can be achieved, whereas the iteration complexity in the state space cannot be alleviated [241].

The complexity results in [124, 241] have been further generalized by Forcier and Leclère [75]. They provide results for a generalized framework of SDDP-related algorithms, including SDDP with inexact cuts or regularization (see also Section 21), risk-averse SDDP (see also Section 12) and extensions to convex nonlinear or non-convex mixed-integer (nonlinear) problems (see also Sections 15 and 16).

5. Comparison with Related Methods.

We briefly compare SDDP to solution methods that it is (historically) related to, as discussed in Section 1.

5.1. Relation to SDP.

SDDP is closely related to stochastic dynamic programming (SDP). SDP usually is applied in a setting where not only state variables, but additional local variables are considered, see Remarks 2.2 and 2.4. Therefore, the DPE and value functions are considered in the form of (2.7), which we repeat here for convenience:

$$Q_t(x_{t-1}, \xi_t) = \min_{u_t \in U_t(x_{t-1}, \xi_t)} f_t(u_t, \xi_t) + \mathscr{Q}_{t+1}(\mathcal{T}_t(x_{t-1}, u_t, \xi_t)).$$

The main idea of SDP is to explicitly evaluate the (expected) value functions for all possible cases during a forward or backward iteration through the stages $t \in [T]$.

This is only possible if the support Ξ_t of $\boldsymbol{\xi}_t$ and the state space $\mathcal{X}_t \subset X_t$ are finite for all $t \in [T]$. Otherwise, infinitely many evaluations would be required. Additionally, it is required that also the action space $U_t(x_{t-1}, \xi_t)$ is finite for all $x_{t-1} \in X_{t-1}, \xi_t \in \Xi_t$, so that the minimum in (2.7) can be computed by finitely many evaluations. For this reason, all these sets may have to be discretized first [176].

The computational effort of SDP scales linearly in T and in the cardinalities $|X_t|, |U_t(x_{t-1}, \xi_t)|$ and $|\Xi_t|$. The three sets might be multidimensional, and thus require to be discretized in each dimension n_t, $\tilde{n}_t$, and κ_t. Hence, their cardinality grows exponentially in these dimensions, which is computationally prohibitive for high-dimensional problems. This is known as the *curse of dimensionality* of SDP, see also Section 1.

SDDP avoids the requirements of state space and action space discretization by not evaluating $\mathscr{Q}_t(\cdot), t \in [T]$, exactly for all (finitely many) possible actions and states, but approximating them by an iteratively refined polyhedral outer approximation $\underline{\mathcal{V}}_t(\cdot)$, constructed by linear cuts. It can thus be considered an *approximate dynamic programming* (ADP) method.

5.2. Relation to NBD. In stochastic programming, it is common practice to consider problems (MSLP) with finite randomness (Assumption 5), but without the requirement of stagewise independence of $\boldsymbol{\xi}_t$ (Assumption 2). In that case the uncertainty can be modeled by a finite scenario tree, which compared to the recombining tree from Section 2 exhibits some path dependence and satisfies the usual *tree* property that each node n has a finite set of child nodes $\mathcal{C}(n)$, but a unique parent node $a(n)$. An example of a scenario tree with $T = 3$ and $|\mathcal{S}| = 9$ is illustrated in Figure 7. This scenario tree represents the same number of scenarios $|\mathcal{S}|$ as the recombining one in Figure 1, but requires $\sum_{t=2}^{T} q_t^{t-1} + 1$ instead of $\sum_{t=2}^{T} q_t + 1$ nodes.

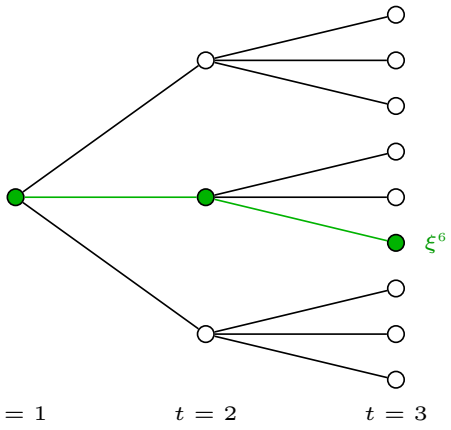


Fig. 7: Scenario tree with 9 scenarios and ξ^6 highlighted.

To solve (MSLP) associated with a general scenario tree, in principle the same approximation approach as in SDDP can be used. However, due to the path dependence, the value functions $Q_t(\cdot, \cdot)$ and expected value function $\mathscr{Q}_t(\cdot, \cdot)$ depend on the history $\xi_{[t-1]}$ of the data process $(\boldsymbol{\xi}_t)_{t \in [T]}$. In other words, each node n has its own value function $Q_n(\cdot)$, and with each node (except for leaf nodes) is associated an expected value function $\mathscr{Q}_{\mathcal{C}(n)}(\cdot)$. Therefore, to update the approximations $\underline{\mathcal{V}}^i_{\mathcal{C}(n)}(\cdot)$ of

all $\mathscr{Q}_{\mathcal{C}(n)}(\cdot)$ in each iteration, all nodal subproblems have to be solved in the backward pass, which in turn requires to compute trial points $x^i_{a(n)}$ for all nodes, *i.e.*, solving all nodal subproblems in the forward pass as well.

Because of its close relation to the L-shaped method for solving two-stage stochastic linear programs [237] and to Benders decomposition [17] this solution method is often called *nested Benders decomposition* (NBD). It was first proposed by Birge in 1985 [26] and can be interpreted as a decomposition method for the extensive form of the deterministic equivalent of (MSLP). Contrary to SDDP, NBD guarantees that valid lower bounds $\underline{v}$ and upper bounds $\overline{v}$ of v^* are determined in each iteration and by that allows for a deterministic stopping criterion in a straightforward way. The upper bounds can be computed as

$$\overline{v}^i := \mathbb{E}\left[\sum_{n\in\mathcal{T}} c_n^\top x_n^i\right],$$

where $\mathcal{T}$ is the set of all nodes in the scenario tree.

On the other hand, due to the sheer amount of subproblems to be solved in each iteration, which grows exponentially in T, it is only computationally tractable for problems of moderate size. By moderate we mean instances with some hundreds or a few thousand scenarios, and 4 or 5 stages at maximum [235].

Furthermore, for general scenario trees also sampling scenarios from $\mathcal{S}$ in the forward pass does not necessarily help to reduce the computational burden and to speed-up the solution process, as it reduces the computational effort per iteration, but at the same time implies that the cut approximations $\underline{\mathcal{V}}^i_{\mathcal{C}(n)}(\cdot)$ are only improved for some $\mathscr{Q}_{\mathcal{C}(n)}(\cdot)$ in each iteration. Under stagewise independence (Assumption 2) this is different. The scenario tree collapses to a recombining tree. This means that for any stage t, many differing scenarios share the same nodes, and thus value functions. In particular, there exists only one expected value function $\mathscr{Q}_t(\cdot)$ for each $t = 2, \dots, T$. Therefore, even if only a sample $\mathcal{K} \subset \mathcal{S}$ of scenarios is considered in each iteration i, still the cut approximations $\underline{\mathcal{V}}^i_t(\cdot)$ for all $\mathscr{Q}_t(\cdot)$ are updated with new cuts.

From this perspective, SDDP can be interpreted as a sampling variant of NBD which reduces the computational effort per iteration significantly [186], but heavily relies on stagewise independence of $(\boldsymbol{\xi}_t)_{t\in[T]}$ in order to leverage the sampling with respect to value function approximations.

Remark 5.1 (Cut-sharing). In the literature, the aforementioned property of SDDP is often referred to as *cut-sharing*. This is best understood by representing the stagewise independent data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ using a standard scenario tree. In this case, at any stage t, all nodes have the same set of successor nodes. If now only a sample $\mathcal{K}$ of scenarios is considered in iterations of SDDP, only a subset of nodes is visited. Nonetheless, the cuts constructed for a specific node are valid for all equivalent nodes in the tree as well, so they are *shared* with other nodes/scenarios.

As mentioned above, a recombining scenario tree provides a more precise picture. For each stage, scenarios *share nodes* in the recombining tree and there exists only *one* function $\mathscr{Q}_t(\cdot)$ to be approximated. Therefore, the phrase *cut-sharing* is sometimes considered misleading. □

5.3. Complexity Comparison. We summarize the main complexity results for SDDP and the related methods in Table 3.

In contrast to SDP, SDDP does not require a state space and action space discretization. Especially, the latter is computationally important in practice, whereas

Table 3: Complexity of SDDP and related solution methods.

	Det. Equiv.	NBD	SDDP	SDP
Requirements				
stagewise independence	no	no	yes	yes (*)
state discretization	no	no	no	yes
action discretization	no	no	no	yes
noise discretization	yes	yes	yes	yes
Complexity (#)				
in no. of stages T	expon.	expon.	expon.	linear
in state dimension n_t	polyn.	expon.	expon.	expon.
in action dimension $\tilde{n}_t$	polyn.	polyn.	polyn.	expon.
in no. of realizations q_t (§)	polyn.	polyn.	polyn.	linear
Progressivity				
of bounds	yes	yes	yes	no

(*) Markovian uncertainty is possible as well.

(#) Comprises no. of iterations, effort per iteration, subproblem sizes, subproblem solution etc.

(§) Note that q_t in itself is exponential in the noise dimension κ_t and polynomial in the discretization precision per noise component.

the former may yield computational improvements, but at least does not translate into an improvement of the worst-case complexity class. On the other hand, SDDP does not have linear complexity in T.

Another difference is that SDDP (as NBD and most solvers for the deterministic equivalent) approximates v^* with improving lower (and upper) bounds. This means that if the computation time is increased also the quality of the approximation improves. On the contrary, standard solution methods for SDP, such as backward induction, either manage to solve a problem in a given time limit or not, but do not use improving approximations. In particular, stopping SDP prematurely does not provide valid bounds for v^*.

Compared to NBD, SDDP mainly reduces the computational effort per iteration significantly, but does not get rid of the exponential growth of the computational cost with respect to T. In return, it heavily relies on stagewise independence (Assumption 2) and has worse complexity with respect to the state dimension n_t.

We can conclude that SDDP, while mitigating some of the weaknesses of SDP and NBD (sometimes advertized as "breaking the curse of dimensionality"), does not manage to leave the respective worst-case complexity classes. On the contrary, it inherits some of the complexity drawbacks of both methods. Still, in many applications (where not worst-case complexity is decisive) it shows considerable performance improvements compared to SDP and NBD, especially for problems with continuous action space, a medium number of stages T and a moderate state dimension n_t. While Theorem 4.2 indicates that convergence may take extremely long in large-scale applications, and too long to be computationally tractable, SDDP has shown good performance for large-scale instances of (MSLP) in many applications, as we discuss in Section 9. This is also due to various improvements, which we address in the following sections.

6. Sampling. Sampling is a central element of SDDP, see Section 3, and in particular line 6 of Algorithm 3.1. In the forward pass, a finite number $|\mathcal{K}|$ of scenarios is sampled to simulate the current policy and compute a trajectory of trial points $(x_t^{ik})_{t\in[T]}$ for all $k \in \mathcal{K}$. Often, this sampling is done from a finite set of scenarios $\mathcal{S}$ (see Assumption 5), with $|\mathcal{K}| \ll |\mathcal{S}|$. Alternatively, it is possible to directly sample from a given (continuous) distribution.

In this section, we discuss different sampling techniques which can be used in SDDP. As indicated in Sections 3 and 4, we can distinguish between *random sampling* and *deterministic sampling* methods. In standard SDDP, as originally proposed in [159], random sampling is used. Here, the main requirement is that the samples should be independent and identically distributed (i.i.d.). This is important for two reasons:

(1) This way, almost sure finite convergence of SDDP can be ensured, as any scenario is sampled infinitely many times with probability 1, assuming that the algorithm does not terminate, see Section 4.
(2) In the originally proposed stopping criterion of SDDP a confidence interval is used, which is built using the sample mean $\overline{v}^i_{\mathcal{K}}$ (3.9), see Section 7. However, by the Central Limit Theorem, even an approximate confidence interval can only be obtained for a sequence of i.i.d. random variables.

6.1. Monte Carlo Sampling. The simplest sampling method satisfying the above requirement is Monte Carlo (MC) sampling. Here, samples are drawn randomly from the probability distribution of $\boldsymbol{\xi}_t$ in each iteration, by first sampling from a uniform distribution and then using appropriate transforms. Under stagewise independence (Assumption 2), this is done independently for each stage $t \in [T]$.

As the quantities $v^i(\xi^k)$ are i.i.d., the value $\overline{v}^i_{\mathcal{K}}$ (3.9) that can be computed in the SDDP forward pass is an unbiased estimator of $\overline{v}^i$ and according to the Strong Law of Large Numbers converges to $\overline{v}^i$ for $|\mathcal{K}|$ approaching infinity. Still, the sampling error can be significant. The variance of $\overline{v}^i_{\mathcal{K}}$ can be estimated by $\frac{1}{|\mathcal{K}|}\left(\sigma^i_{\overline{v},\mathcal{K}}\right)^2$. This means that the variance can be reduced either by increasing the number of samples $|\mathcal{K}|$ or by reducing the sample variance $\left(\sigma^i_{\overline{v},\mathcal{K}}\right)^2$. Increasing the sample size may look promising at first glance, but may become computationally intractable in practice [157]. Recall that for every sample $k \in \mathcal{K}$ a number of $1 + \sum_{t=2}^{T} q_t$ subproblems has to be solved in the backward pass of each iteration. Therefore, the more promising approach is combining MC sampling with variance reduction techniques [157].

6.2. Variance Reduction Techniques. Incorporating variance reduction techniques into sampling in SDDP is studied extensively in [112, 157]. For a review on sampling techniques in stochastic programming in general, we refer to [111].

Randomized QMC Sampling. In [112], it is proposed to use Quasi-Monte Carlo (QMC) sampling within SDDP. In this case, instead of randomly sampling from the uniform distribution, a deterministic sequence of points $u^1, \ldots, u^N$ from $(0,1)^{\kappa_t}$ is chosen. This is done in such a way that the sampled points fill $(0,1)^{\kappa_t}$ as homogeneously as possible (so the empirical distribution is as close to a uniform distribution as possible). Then, after an appropriate transformation, they provide a better representation of $\boldsymbol{\xi}_t$ than randomly sampled points.

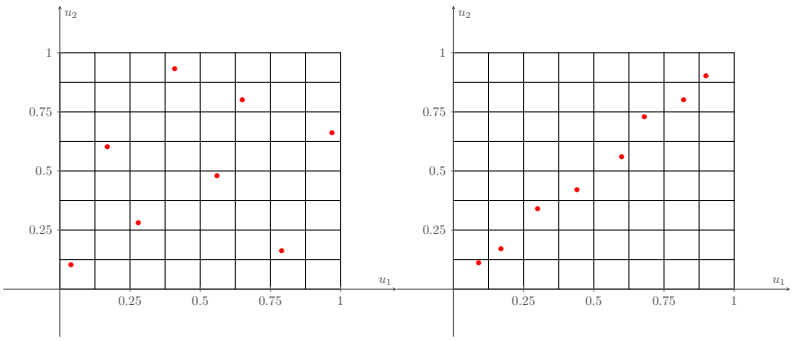
A drawback of QMC methods is that the sample points are not random, the obtained estimator is biased and no confidence interval can be established. *Randomized* QMC (RQMC) methods, where the choice of QMC points is combined with some kind of randomness, avoid this drawback and allow for standard error estimation [112].

Compared to MC sampling, RQMC methods achieve better convergence rates of

$\mathcal{O}\big(|\mathcal{K}|^{-1}(\log|\mathcal{K}|)^{\kappa_t}\big)$, and thus are considered more efficient. However, the convergence rate depends on the dimension κ_t of $\boldsymbol{\xi}_t$ [112].

Latin Hypercube Sampling. In Latin Hypercube Sampling (LHS) [148], the space $(0,1)^{\kappa_t}$ is divided into equidistant subintervals and then scenarios are sampled from each subinterval in such a way that in each row and column of the grid only one point is sampled. This is illustrated in Figure 8 (a).

In this way, again, a more homogeneous distribution of the sample points can be obtained, and compared to MC sampling, the variance can be reduced. On the flipside, poor space-filling or correlation between the sample points has to be ruled out, see Figure 8 (b), which requires significant additional effort.



(a) Good space-filling. (b) Correlated sample points.

Fig. 8: Latin Hypercube Sampling for two dimensions.

Incorporation into SDDP. It is important to notice that while reducing the variance compared to the classical MC estimators, scenarios sampled by RQMC and LHS are no longer i.i.d. Therefore, both sampling techniques cannot be incorporated into SDDP without modification, if convergence properties should not be compromised. Homem-de-Mello et al. [112] therefore suggest to build sampling blocks. This means that the total number of samples $|\mathcal{K}|$ is divided into M blocks $\ell = 1, \dots, M$ with $M \geq 5$ a divisor of $|\mathcal{K}|$. Then, for each block ℓ, $|\mathcal{K}'| := |\mathcal{K}|/M$ scenarios are obtained using conditional sampling with RQMC or LHS, which are not independent. For each $k' \in \mathcal{K}'$, values $v^i(\xi^{k'})$ are determined and averaged to $\overline{v}^{i,\ell}$.

This is repeated for each block ℓ. Then, the mean $\overline{v}^i_{\mathcal{K}}$ of all values $\overline{v}^{i,\ell}, \ell = 1, \dots, M$, and the sample variance are determined. As the scenarios of different blocks are independent, this still yields a useful confidence interval to stop the algorithm.

Another challenge reported in [112] is that it is computationally expensive to generate samples using RQMC for high dimensions. To reduce the computational effort, it may be reasonable to apply RQMC only to important components, *e.g.*, to early stages in $[T]$, and standard MC or LHS to the other ones. This strategy is called *padding* and applied after 6 or 12 stages for numerical tests in [112].

Experiments in [112] imply that RQMC and LHS both lead to upper bounds $\overline{v}_{\mathcal{K}}$ oscillating around the lower bound $\underline{v}$ more quickly compared to MC sampling.

6.3. Importance Sampling. In [157], Parpas et al. propose incorporating importance sampling into SDDP. In contrast to the previously described techniques, it can be used to obtain i.i.d. samples in the forward pass.

The main idea of importance sampling in general is to attach different importance to subregions of the sample space and to sample more often from subregions of higher importance. In the context of SDDP, this means that it is sampled with priority from scenarios that contribute more to the value of the expected value functions $\mathscr{Q}_t(\cdot)$.

This is achieved by sampling from a different distribution than the original one, the so-called *importance sampling distribution*, but correcting the bias introduced by this difference. Then, an importance sampling estimate of $\overline{v}$ can be calculated as

$$\overline{v}_{\mathcal{K}}^{IS,i} := \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} v^i(\xi^k) \Lambda(\xi^k)$$

with $\Lambda(\boldsymbol{\xi}) := \frac{f(\boldsymbol{\xi})}{g(\boldsymbol{\xi})}$, where f denotes the original distribution and g the importance sampling distribution. The likelihood function $\Lambda(\cdot)$ is used to correct for sampling from the wrong distribution. It can be shown that importance sampling can reduce the variance of sampling estimators significantly. In the SDDP case, as shown in [157], the variance is minimized for the choice

$$g_t^*(\xi_t) := \frac{|Q_t(x_{t-1}^{ik}, \xi_t)|}{\mathbb{E}_f |Q_t(x_{t-1}^{ik}, \boldsymbol{\xi}_t)|} f_t(\xi_t).$$

However, clearly, this *zero-variance distribution* is a theoretical construct and not known, which is referred to as the *curse of circularity*. Therefore, it is proposed to first approximate g^* using a framework including Kernel density estimation [157].

In numerical experiments, SDDP with importance sampling is shown to outperform MC and QMC sampling based methods, given that it is difficult to sample from the original probability distribution and that the original problem has moderate or high variance [157].

6.4. Deterministic Sampling. As already discussed in Section 4, in line 6 of SDDP (Algorithm 3.1) also some deterministic sampling can be used. In this case, $|\mathcal{K}| = 1$. In the literature, two different approaches are considered.

Worst Approximation Sampling. The first one requires that in addition to the (lower) cut approximation $\underline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$ also an upper approximation $\overline{\mathcal{V}}_t(\cdot)$ is constructed and iteratively refined in SDDP. Assume that in the forward pass on stage $t-1$ the trial point x_{t-1}^i has been computed. Then, for stage t the approximate subproblem (2.10) is solved for x_{t-1}^i and for all noise terms $\xi_{tj}, j = 1, \ldots, q_t$, yielding optimal states x_{tj}. For the next stage, the trial point $x_t^i = x_{tj'}$ is chosen such that

$$j' \in \operatorname*{arg\,max}_{j=1,\ldots,q_t} \left\{ \overline{\mathcal{V}}_t^i(x_{tj}) - \underline{\mathcal{V}}_t^i(x_{tj}) \right\},$$

i.e., that the gap between the current upper and lower approximations is maximized. This corresponds to sampling noise term $\xi_{tj'}$ on stage t.

This form of deterministic sampling is used for SDDP in [241]. Its computational drawback is that at each stage q_t subproblems have to be solved instead of only $|\mathcal{K}| \ll q_t$. A similar approach was first proposed by Baucke et al. in [10, 11] and called *problem-child node selection*. However, their setting differs a bit from original SDDP, as each subproblem contains specific variables $x_{tj}, j = 1, \ldots, q_t$, for all random outcomes, and therefore in their case only one subproblem has to be solved in the sampling step. Another related sampling scheme is used in *robust dual dynamic programming* (RDDP) [83]. In that case, $\xi_{tj'}$ is determined by solving a special upper bounding problem containing $\overline{\mathcal{V}}_t^i(\cdot)$

Explorative Sampling. Explorative deterministic sampling is proposed in [124] as part of EDDP. It is based on the concepts of saturated and distinguishable points, which we introduced in Section 4.2. As for the previous sampling scheme, the idea is to solve the forward pass subproblems for all $\xi_{tj}, j = 1, \dots, q_t$. Instead of maximizing an approximation gap, however, the trial point $x_t^i = x_{tj'}$ is chosen such that

$$j' \in \operatorname*{arg\,max}_{j=1,\dots,q_t} \min_{x_t \in X_t^{sat}} \|x_{tj} - x_t\|,$$

i.e., the minimum distance to already saturated points is maximized. In other words, a maximum distinguishable point is chosen.

As shown in [75], worst approximation sampling and explorative sampling are equivalent in the sense that both approaches are guaranteed to lead to effective iterations, see Section 4.2.

7. Stopping Criteria. In each iteration i of SDDP, a valid lower bound $\underline{v}^i$ for the optimal value v^* is determined. Additionally, a statistical upper bound $\overline{v}^i_{\mathcal{K}}$ can be computed. Since the latter is not necessarily valid, an important question is when to consider an obtained policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ as (approximately) *optimal* and to stop the SDDP method, see line 4 of Algorithm 3.1. If the stopping criterion is too conservative, the algorithm may iterate much longer than required, if it is too optimistic, then SDDP may stop prematurely.

Confidence Stopping Criteria. In their seminal work on SDDP, Pereira and Pinto propose to use a confidence interval based stopping criterion [160]. An approximate confidence interval for a true valid upper bound $\overline{v}^i$ is determined as follows using the estimates $v^i(\xi^k)$ from (3.9).

Under random independent sampling, the values $v^i(\xi^k)$ are i.i.d. random variables with expected value $\overline{v}^i$ and variance $(\sigma^i)^2$. Moreover, knowing the sample mean $\overline{v}^i_{\mathcal{K}}$ (3.9), we can define a standardized random variable

$$Z^i_{\mathcal{K}} := \frac{\overline{v}^i_{\mathcal{K}} - \overline{v}^i}{\frac{\sigma^i}{\sqrt{K}}}. \tag{7.1}$$

According to the Central Limit Theorem, this random variable asymptotically, that is, for $|\mathcal{K}| \to \infty$, follows a standard normal distribution $\mathcal{N}(0,1)$. This implies that for sufficiently large $|\mathcal{K}|$, $Z^i_{\mathcal{K}}$ is approximately standard normal distributed.

Due to symmetry of the standard normal distribution, it follows

$$\mathbb{P}(-z_{1-\alpha/2} \le Z^i_{\mathcal{K}} \le z_{1-\alpha/2}) \approx 1 - \alpha,$$

where $z_{1-\alpha/2}$ denotes $(1 - \frac{\alpha}{2})$-quantiles of $\mathcal{N}(0,1)$ for some level $\alpha \in (0,1)$.

Inserting (7.1) and rearranging yields an approximate $(1-\alpha)$-confidence interval for the true upper bound $\overline{v}^i$:

$$\left[\overline{v}^i_{\mathcal{K}} - z_{1-\frac{\alpha}{2}} \frac{\sigma^i}{\sqrt{|\mathcal{K}|}}, \overline{v}^i_{\mathcal{K}} + z_{1-\frac{\alpha}{2}} \frac{\sigma^i}{\sqrt{|\mathcal{K}|}}\right].$$

As σ^i is unknown, it can be replaced by the sample standard distribution $\sigma^i_{\overline{v},K}$ which is defined by the sample variance

$$(\sigma^i_{\overline{v},\mathcal{K}})^2 := \frac{1}{|\mathcal{K}| - 1} \sum_{k\in\mathcal{K}} (v^i(\xi^k) - \overline{v}^i_{\mathcal{K}})^2.$$

In that case, the standardized variable approximately follows a Student's t-distribution with degree of freedom $|\mathcal{K}| - 1$. In the literature on SDDP, even in this case, the $(1-\alpha)$-confidence interval for the true upper bound $\overline{v}^i$ is usually approximated using a standard Normal distribution [209], though, which yields:

$$\left[\overline{v}^i_{\mathcal{K}} - z_{1-\frac{\alpha}{2}} \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\sqrt{|\mathcal{K}|}}, \overline{v}^i_{\mathcal{K}} + z_{1-\frac{\alpha}{2}} \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\sqrt{|\mathcal{K}|}}\right]. \tag{7.2}$$

Pereira and Pinto propose choosing $\alpha = 0.05$, which implies $z_{1-\alpha/2} = 1.96$, and stopping SDDP if the lower bound $\underline{v}^i$ is included in this confidence interval [160].

As pointed out by Shapiro [206], this stopping criterion has several flaws. The higher the sample variance $(\sigma^i_{\overline{v},\mathcal{K}})^2$, the earlier $\underline{v}^i$ exceeds the lower end of the confidence interval, which provides a misguided incentive to increase $(\sigma^i_{\overline{v},\mathcal{K}})^2$. The same is true for increasing the confidence $1-\alpha$, which contradicts the intuition behind α. Additionally, faster stopping can be achieved by reducing the sample size $|\mathcal{K}|$. Finally, the above stopping criterion may favor premature stopping, as it is rather unlikely that $\overline{v}^i$ is located exactly at the lower bound of the confidence interval. For these reasons, Shapiro proposes a more conservative stopping criterion where SDDP terminates if the difference between the upper bound of the confidence interval (7.2) and $\underline{v}^i$ is sufficiently small.

Sometimes it is also suggested to include values $v^j(\xi^k)$ from previous iterations $j < i$ in (3.9), for instance if $|\mathcal{K}|$ is too small to obtain a reasonable bound. However, this destroys the independence between the different samples. Thus, the Central Limit Theorem can no longer be applied and the confidence-based stopping criteria are not applicable. [53].

A Hypothesis Test Perspective. Considering that hypothesis tests and confidence intervals are closely related, the above stopping criterion can also be interpreted in terms of a hypothesis test with hypotheses [112]:

$$H_0 : \overline{v}^i = \underline{v}^i, \quad \text{against} \quad H_1 : \overline{v}^i \neq \underline{v}^i.$$

The null hypothesis H_0 is tested using the test statistic $\overline{v}^i_{\mathcal{K}}$, which is assumed to be approximately normal distributed. This can again be reasoned using the Central Limit Theorem for sufficiently large $|\mathcal{K}|$. Then, the region of acceptance for H_0 in iteration i is given by the interval (7.2): If the lower bound $\underline{v}^i$ does not exceed the lower bound of this region, then optimality is rejected. Otherwise, there is no compelling reason to reject it, so it is retained. By choosing α, the type I error (rejecting optimality although SDDP has converged) can be controlled. However, this comes at the cost of a possibly high type II error (stopping the algorithm prematurely) [112].

Different Hypothesis Tests. To avoid stopping prematurely, Homem-de-Mello et al. propose a modified hypothesis test controlling type I and type II errors simultaneously [112]. The basic principle is very similar to above, even if it is presented for a one-sided hypothesis test with $H_0 : \overline{v}^i \leq \underline{v}^i$. The key difference is that if $\underline{v}^i$ lies inside of the region of acceptance, the hypothesis of optimality is not necessarily retained, but still may be rejected. In particular, stopping SDDP should be prevented if the true upper bound $\overline{v}^i$ exceeds the lower bound $\underline{v}^i$ considerably. As $\overline{v}^i$ is not known, we cannot observe when this event occurs, but we can predefine a bound $\gamma > 0$ on the probability of stopping given that it happens. For fixed γ and α, and given sample estimates, we can then compute a percentage difference δ^i between $\overline{v}^i$ and $\underline{v}^i$, for

which the probability of a type II error (premature stopping) is bounded by γ:

$$\delta^i = (z_{1-\alpha} + z_{1-\gamma}) \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\underline{v}^i \sqrt{|\mathcal{K}|}}. \tag{7.3}$$

If δ^i is below some predefined threshold $\overline{\delta}$, the sample estimates guarantee that for deviations larger than $\overline{\delta}$, the type II error is under control. Therefore, SDDP stops. Otherwise, the control of the type II error is not considered sufficient, and the algorithm proceeds. In other words, SDDP only terminates when $\underline{v}^i$ lies inside of the region of acceptance *and* when the type II error is bounded by γ for a sufficiently small percentage difference δ^i.

Summarized, the following procedure is used in each iteration i [112]:

1. Compute ρ^i as the ratio of the left interval boundary in (7.2) and $\underline{v}^i$.
2. If $\rho^i \leq 1$, then compute δ^i according to (7.3).
   a) If $\delta^i < \overline{\delta}$, then stop SDDP.
   b) Otherwise, start a new iteration $i+1$ of SDDP (or adapt α, γ or $\mathcal{K}$).

   Otherwise, start a new iteration $i+1$ of SDDP.

Computational experiments with $\overline{\delta} = 0.1$ and $\gamma = 0.05$ indicate that this stopping criterion is effective in preventing SDDP from premature stopping [112]. Still, it is a heuristic, and so far, no proposed statistical testing procedure guarantees that the probability of stopping prematurely is bounded by some $\gamma > 0$ in general.

Predefined Criteria. The previous statistical stopping criteria are computationally demanding and require $|\mathcal{K}|$ to be sufficiently large to yield reasonable approximate confidence intervals. Furthermore, in practical applications (MSLP) is often too large to achieve convergence in reasonable time. Finally, the statistical stopping criteria do not necessarily generalize to extensions of SDDP, such as risk-averse variants, see Section 12. Therefore, in practice often more convenient stopping criteria are used for SDDP. For instance, it is common to stop SDDP after a fixed number of iterations $I \in \mathbb{N}$, after a fixed number of cuts $|\mathcal{K}|I$, after a predefined time or if the lower bounds $\underline{v}^i$ have stalled. Neither guarantees that an optimal policy is determined, though.

Deterministic Stopping. Finally, SDDP can be stopped deterministically as long as valid upper bounds $\overline{v}^i$ for v^* are computed in addition to lower bounds $\underline{v}^i$. In that case, for some predefined optimality tolerance $\varepsilon > 0$, SDDP stops with an (approximately) optimal policy if $\overline{v}^i - \underline{v}^i \leq \varepsilon$. This stopping criterion requires significant additional computational effort to determine true upper bounds $\overline{v}^i$. Hence, there is a trade-off between achieving a more reasonable stopping criterion and spending computational resources on computations offside of the core elements of SDDP. We address how such exact upper bounds can be computed in the next section.

Summarizing, despite various attempts at developing reasonable termination criteria for SDDP, optimally stopping SDDP remains an open challenge.

8. Exact Upper Bounds and Upper Approximations. The idea of computing deterministic upper bounds $\overline{v}$ for v^* and deterministic upper approximations $\overline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$ has drawn a lot of interest in the research community recently, both in analyzing the convergence behavior of SDDP, see Section 4, and in developing deterministic stopping criteria, see Section 7.

An intuitive way to determine upper approximations $\overline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$ is based on the observation that due to convexity of $\mathscr{Q}_t(\cdot)$ all its secants lie above or on its graph. Therefore, an upper approximation is possible by a convex combination of points $(x_{t-1}, \mathscr{Q}_t(x_{t-1}))$. To obtain an approximation on the whole state space, it can be

extended using a regularization with a Lipschitz constant L_t of $\mathscr{Q}_t(\cdot)$. Such constant exists according to Corollary 2.10. In this light, $\overline{\mathcal{V}}_t(\cdot)$ can be constructed as [241]

$$\overline{\mathcal{V}}_t(x_{t-1}) = \operatorname{co}\left(\min_{m=1,\dots,M_t} \left\{ \mathscr{Q}_t(x_{t-1}^m) + L_t \|x_{t-1} - x_{t-1}^m\| \right\} \right), \tag{8.1}$$

where $\operatorname{co}(f)$ denotes the convex envelope of function f. This is illustrated in Figure 9.

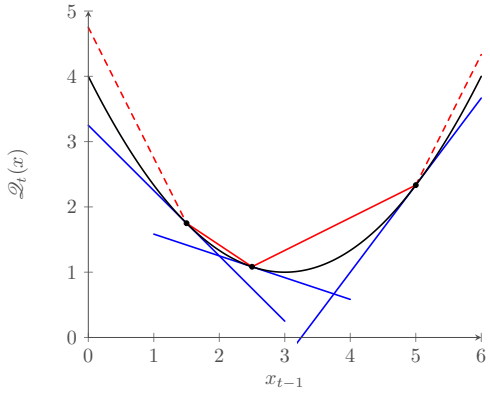


Fig. 9: Inner and outer approximation of $\mathscr{Q}_t(\cdot)$.

Alternatively, by interpreting this idea from a set perspective, the convex epigraph $\operatorname{epi}(\mathscr{Q}_t)$ of $\mathscr{Q}_t(\cdot)$ can be approximated by the convex hull $\operatorname{conv}(w_{t-1}^1, \dots, w_{t-1}^{M_t})$ of finitely many points $w_{t-1} := (x_{t-1}, \mathscr{Q}_t(x_{t-1}))$ in $\operatorname{epi}(\mathscr{Q}_t)$.

In principle, there are two different approaches to realize this idea. One uses the above perspectives, which we refer to as *primal*, and one is related to some *dual* perspective on SDDP and its value functions [104, 126].

8.1. Primal Inner Approximation. Similar to subproblems (2.10), based on upper approximations $\overline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$, approximating subproblems can be defined by replacing $\mathscr{Q}_t(\cdot)$ with $\overline{\mathcal{V}}_t(\cdot)$ in the DPE for all $t \in [T]$. This idea is first introduced by Philpott et al. [168]. As they consider only the RHS of (MSLP) to be uncertain, we adopt this assumption, although it is not required.

For stages $t = T-1, \dots, 2$, each element m in a given set of points $x_t^1, \dots, x_t^{M_{t-1}}$ and each $\xi_{tj}, j = 1, \dots, q_t$, the following subproblem can be solved by backward recursion:

$$\overline{Q}_t(x_{t-1}^m, \xi_{tj}) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \overline{\mathcal{V}}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^m, \xi_{tj}). \end{cases} \tag{8.2}$$

Here, as indicated in (8.1), the upper approximation $\overline{\mathcal{V}}_{t+1}(\cdot)$ is defined as a convex combination of points $\left(x_t^m, \overline{\mathscr{Q}}_{t+1}(x_t^m)\right), m = 1, \dots, M_t$. The key difference is that instead of $\mathscr{Q}_{t+1}(x_t^m)$ here $\overline{\mathscr{Q}}_{t+1}(x_t^m) := \mathbb{E}\left[\, \overline{Q}_{t+1}(x_t^m, \boldsymbol{\xi}_{t+1})\right]$ is used, as $\mathscr{Q}_{t+1}(\cdot)$ is not

known:

$$\overline{\mathcal{V}}_{t+1}(x_t) := \begin{cases} \min\limits_{w} & \sum\limits_{m=1}^{M_t} w_m \overline{\mathscr{Q}}_{t+1}(x_t^m) \\ \text{s.t.} & \sum\limits_{m=1}^{M_t} w_m x_t^m = x_t \\ & \sum\limits_{m=1}^{M_t} w_m = 1 \\ & w_m \geq 0, \quad m = 1, \dots, M_t. \end{cases} \tag{8.3}$$

Furthermore, compared to (8.1) no regularization is used.

By recursion, it can be shown that

$$\overline{Q}_t(x_{t-1}^m, \xi_{tj}) \geq Q_t(x_{t-1}^m, \xi_{tj})$$

for all $m = 1, \dots, M_{t-1}$ and $j = 1, \dots, q_t$. This implies

$$\overline{\mathscr{Q}}_t(x_{t-1}^m) \geq \mathscr{Q}_t(x_{t-1}^m).$$

The first-stage problem then yields

$$\overline{v}^{IA} := \begin{cases} \min\limits_{x_t} & c_1^\top x_1 + \overline{\mathcal{V}}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1, \end{cases} \tag{8.4}$$

with $\overline{v}^{IA}$ an exact valid upper bound to v^*.

Based on these definitions, in principle, the backward pass of SDDP can be enhanced at each stage $t = T-1, \dots, 2$, by solving subproblems (8.2) for all $m = 1, \dots, M_t$, and each $\xi_{tj}, j = 1, \dots, q_t$, and updating $\overline{\mathcal{V}}_t(\cdot)$ according to (8.3). At the first stage, by solving subproblem (8.4), then an upper bound can be computed.

A key challenge with this approach is to appropriately choose the set of points $x_{t-1}^m, m = 1, \dots, M_{t-1}$ for each stage t and iteration i. On the one hand, they should be chosen such that as much of $\mathcal{X}_{t-1}$ is spanned as possible. On the other hand, choosing (at least some of) those points as extreme points leads to $M_t \geq 2^{n_t}$ points, *i.e.*, the number of points grows exponentially in the dimension of the state space.

An alternative is to use the trial points from the SDDP forward pass [168]. Even using these points, the computational effort may become excessive, though. Similarly to the SDDP backward pass, subproblems (8.2) have to be solved for each stage $t \in [T]$, each point $x_{t-1}^m, m = 1, \dots, M_{t-1}$, and each noise term $\xi_{tj}, j = 1, \dots, q_t$. However, the number M_{t-1} of points to be considered grows with each iteration, as it contains all previous trial solutions. It is therefore suggested to only use the upper bound computation every few hundred iterations, and not to permanently incorporate it into the backward pass [168]. This hinders using the upper bounds $\overline{v}^{IA}$ in the stopping criterion of SDDP in each iteration, though.

Moreover, the obtained bounds $\overline{v}^{IA}$ may be very loose, especially in problems (MSLP) with a high number of stages. Computational tests are required to assess whether the information gain justifies the additional computational effort and, possibly, higher number of iterations.

Baucke et al. provide a different perspective on the previous inner approximation idea [10]. Instead of (8.3), they use its dual representation

$$\overline{\mathcal{V}}_{t+1}(x_t) = \begin{cases} \max\limits_{\mu,\lambda} & x_t^\top \lambda + \mu \\ \text{s.t.} & (x_t^m)^\top \lambda + \mu \leq \overline{\mathcal{Q}}_{t+1}(x_t^m), \quad m = 1, ..., M_t. \end{cases} \tag{8.5}$$

This shows that $\overline{\mathcal{V}}_{t+1}(\cdot)$ can be equivalently described by maximizing over the coefficients of all supporting hyperplanes for points $\left(x_t^m, \overline{\mathcal{Q}}_{t+1}(x_t^m)\right), m = 1, \ldots, M_t$.

In [10], the dual problem is additionally regularized, *i.e.*, enhanced by constraint

$$\|\lambda\| \leq L_t,$$

with L_t denoting a Lipschitz constant of $\overline{Q}_t(\cdot, \cdot)$. This is equivalent to regularizing the primal problem (8.3) with the dual norm to $\|\cdot\|$, cf. (8.1). This way, a reasonable approximation is also achieved for points outside of the convex hull of the set defined by the points $x_t^m, m = 1, \ldots, M_t$.

Using this expression for the inner approximation functions, Baucke et al. propose a deterministic algorithm for multistage stochastic convex programs. In their case, subproblems (8.2) are solved in each backward pass iteration, and $\overline{\mathcal{V}}_{t+1}^i(\cdot)$ is updated by adding constraint $(x_t^{\widetilde{m}})^\top \lambda + \mu \leq \overline{\mathcal{Q}}_{t+1}^i(x_t^{\widetilde{m}})$ for the current iterate $x_t^{\widetilde{m}}$ (to incorporate it into the subproblems (8.2), however, again its dual (8.3) is used). The proposed algorithm differs in further regards from standard SDDP, for instance it requires a multi-cut approach, see Section 21, and uses a worst approximation sampling, see Section 6.4. Moreover, choosing a reasonable and valid value for L_t can be very challenging, but is crucial for the proposed method to work as intended.

8.2. Dual SDDP. To compute deterministic upper bounds $\overline{v}$ for v^* recently a dual perspective on SDDP and the DPE (2.4) has gained attention.

Using Convex Conjugates of Value Functions. The first proposal in this context, by Leclère et al. [126], exploits convex conjugates and the related duality concepts to derive *dual value functions* for (MSLP) where uncertainty only appears in the RHS $h_t(\xi_t)$.

Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{-\infty, \infty\}$. Then its *convex conjugate* $f^\star(\cdot)$ is defined as [193]

$$f^\star(\lambda) := \sup_{x \in \mathbb{R}^n} \lambda^\top x - f(x).$$

For (MSLP), the convex conjugates $D_t(\cdot) := Q_t^\star(\cdot)$ of the value functions $Q_t(\cdot)$ can be considered as dual value functions for $t = 2, \ldots, T$. It can be shown that these functions also satisfy some DPE with linear subproblems on each stage. Whereas Leclère et al. consider a more general setting including control variables u_t (see Remark 2.2), for (MSLP) as defined in Section 2 (and especially under Assumption 5), for $t = 2, \ldots, T$, these subproblems can be expressed by

$$D_t(\lambda_{t-1}) := \begin{cases} \min\limits_{\lambda_t, \mu_t, \gamma_t} & \sum\limits_{j=1}^{q_t} p_{tj} \Big(- h_{tj}^\top \mu_{tj} + D_{t+1}(\lambda_{tj}) \Big) \\ \text{s.t.} & T_{t-1}^\top \left(\sum\limits_{j=1}^{q_t} p_{tj} \mu_{tj} \right) - \sum\limits_{j=1}^{q_t} p_{tj} \gamma_{tj} + \lambda_{t-1} = 0 \\ & W_t^\top \mu_{tj} = \lambda_{tj} + c_t, \qquad j = 1, \ldots, q_t \\ & \gamma_{tj} \leq 0, \qquad j = 1, \ldots, q_t. \end{cases} \tag{8.6}$$

For the first stage, we obtain a deterministic problem, which by $T_0 \equiv 0$ simplifies to

$$D_1(\lambda_0) = \min_{\mu_1} \; h_1^\top \mu_1 + D_t\big(W_1^\top \mu_1 - c_1\big)$$

for some arbitrary initial $\lambda_0 \leq 0$ (note that more general formulations of (MSLP) may lead to a dependence on λ_0).

Using this dynamic recursion, it is possible to apply an SDDP-type algorithm, called *dual SDDP*, to $D_t(\cdot)$, using iteratively improving outer approximations $\mathfrak{D}_t^i(\cdot)$ for $D_t(\cdot)$. Analogously to SDDP, this iterative method yields a converging deterministic lower bound for the first-stage optimal value, *i.e.*, $\mathfrak{D}_1^i(\lambda_0) \leq D_1(\lambda_0)$. Applying conjugacy theory again, we obtain

$$\overline{v}^i = \big(\mathfrak{D}_1^i\big)^\star(x_0) \geq D_1^\star(x_0) = Q_1^{\star\star}(x_0) = Q_1(x_0) = v^*.$$

Hence, deterministic upper bounds for v^* can be obtained as conjugates of the first-stage approximations $\mathfrak{D}_t^i(\cdot)$ evaluated at $x_0 = 0$, and $(\overline{v}^i)_i$ defines a sequence converging to v^* [126].

Using the Dual of (MSLP). Guigues et al. propose an alternative way to define dual value functions and DPE that can be exploited in a dual SDDP algorithm [104]. Instead of working with conjugates of the primal value functions $Q_t(\cdot)$, they first derive the dual to (MSLP) formulated as a single problem (2.3), and then show that this dual problem can be decomposed using DPE and dual value functions

$$\widetilde{D}_t(\pi_{t-1}) := \begin{cases} \max\limits_{\pi_t} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(- h_{tj}^\top \pi_{tj} + \widetilde{D}_{t+1}(\pi_{tj})\Big) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(T_{t-1,j}^\top \pi_{tj}\Big) + W_{t-1}^\top \pi_{t-1} \leq c_{t-1}. \end{cases} \tag{8.7}$$

It can be argued that these dual DPE are simpler and more intuitive, as they do not require conjugacy theory. Moreover, we immediately obtain that the first-stage optimal value $\widetilde{D}_1(\pi_0)$ equals v^* by strong duality for linear programs. Therefore, using outer approximations $\widetilde{\mathfrak{D}}_t^i(\cdot)$ of these value functions in dual SDDP, again a sequence $(\overline{v}^i)_i$ of deterministic and valid upper bounds can be computed which converges to v^* [104]. On the other hand, the dual value functions $\widetilde{D}_t(\cdot)$ cannot be directly related to the original value functions $Q_t(\cdot)$.

Remark 8.1. Even if the dual DPE (8.6) and (8.7) are derived using different tools and perspectives, they are still closely related. Note that subproblem (8.6) can be reformulated as

$$D_t(\lambda_{t-1}) = \begin{cases} \min\limits_{\lambda_t, \mu_t} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(- h_{tj}^\top \mu_{tj} + D_{t+1}(\lambda_{tj})\Big) \\ \text{s.t.} & T_{t-1}^\top \left(\sum\limits_{j=1}^{q_t} p_{tj} \mu_{tj} \right) + \lambda_{t-1} \leq 0 \\ & W_t^\top \mu_{tj} = \lambda_{tj} + c_t, \qquad j = 1, \ldots, q_t. \end{cases}$$

Using the last constraint, the state λ_{t-1} can be expressed through the dual variables μ_{t-1} from the previous stage: $\lambda_{t-1} = W_{t-1}^\top \mu_{t-1} - c_{t-1}$. Exploiting this, the

subproblems only contain dual variables μ_t, which have to be considered as state variables. By adapting the optimization sense in the objective, we get exactly the structure of (8.7). □

We can make the following additional observations with respect to the dual DPE (8.6) and (8.7). First, in both cases, the subproblems are not necessarily bounded. Therefore, in both cases, artificial bounds are introduced. In [104] they are chosen as $\pi_t \in [\underline{\pi}_t, \overline{\pi}_t]$, whereas in [126] Lipschitz continuity of $\mathscr{Q}_t(\cdot)$ is exploited to impose the bounds $\|\lambda_t\|_\infty \leq L_t$ for Lipschitz constants $L_t, t = 2, \dots, T$. It is assumed that these bounds are chosen sufficiently large to not affect the optimal solutions.

Second, even if the primal DPE (2.4) are assumed to have relatively complete recourse (see Assumption 9 and Lemma 2.5), this does not necessarily translate to the dual subproblems. To ensure feasibility, Guigues et al. propose to either use feasibility cuts (also see Section 17) or a penalization approach [104].

Third, in contrast to the primal perspective, the subproblems do not decompose by realizations of $\boldsymbol{\xi}_t$, but contain separate dual variables π_{tj} (or $\lambda_{tj}, \mu_{tj}, \gamma_{tj}$, respectively) for all $j = 1, \dots, q_t$. In the forward pass of dual SDDP the trial point π_t^i (or λ_t^i) that is used as a parameter in the following stage is sampled from these variables. A decoupling can be achieved using a Lagrangian relaxation [44].

Finally, if W_t and c_t become uncertain as well, then the value functions and subproblems additional depend on ξ_t. In fact, in formulation (8.7) the state space has to be extended to include the history ξ_{t-1} of the stochastic process, as the problem contains W_{t-1} and c_{t-1} [104].

Again, an SDDP-type algorithm, also referred to as *dual SDDP* in [104], can be applied to the DPE (8.7). This algorithm is presented in Algorithm 8.1. The two variants of dual SDDP have been extended to the risk-averse case [43] (see also Section 12) and to problems with infinite horizon (see also Section 19) [208].

Dual Inner Approximation. First and foremost, dual SDDP is an alternative to (primal) SDDP to approximate v^* by converging deterministic upper bounds $\overline{v}^i$. However, as shown in [126], if the dual DPE (8.6) are used, then the obtained approximations $\mathfrak{D}_t^i(\cdot)$ may be translated to inner approximations $\overline{\mathcal{V}}_t^i(\cdot)$ of the *primal* value functions $Q_t(\cdot)$. This way, policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ for (MSLP) can be computed. The inner approximations can be computed as Lipschitz regularizations (see Sect. 17) of the convex conjugate of the outer approximations $\mathfrak{D}_t^i(\cdot)$, which is shown to be equivalent to solving problem (8.5) with regularization $\|\lambda\|_\infty \leq L_t$. The key difference to the approach in [10] is the way the primal supporting points x_t^m are determined, that is, by the slopes of the dual outer approximation [126].

Incorporation into SDDP. While dual SDDP can be applied on its own to approximate v^*, and even compute policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$, it seems reasonable to incorporate it into (primal) SDDP in order to compute deterministic upper *and* lower bounds for v^*. Guigues et al. suggest to use both variants of SDDP in parallel [104]. In contrast, Leclère et al. propose a framework where primal and dual SDDP are intertwined [126]:

1. Run a forward pass of (primal) SDDP, yielding trial solutions $(x_t^i)_{t\in[T]}$ for the sampled scenario path (the authors choose $|\mathcal{K}| = 1$).
2. Run a backward pass of (primal) SDDP using the trial solutions x_{t-1}^i, obtaining new slopes π_t^i from the cuts.
3. Run a backward pass of dual SDDP using the slopes $\lambda_t^i = \pi_t^i$, obtaining new cuts for the dual problem.

Algorithm 8.1 Dual SDDP from [104]

Input: Dual to problem (MSLP) satisfying Assumptions 1 to 9. Appropriate multiplier bounds. Stopping criterion.

`Initialization`

Initialize cut approximations with bounded $\widetilde{\mathfrak{D}}_t^0(\cdot)$ for all $t = 2, \dots, T$.
Initialize upper bound with $\overline{v}^0 = +\infty$.
Set iteration counter to $i \leftarrow 0$.

`Dual SDDP Loop`

while Stopping criterion not satisfied **do**
  Set $i \leftarrow i + 1$.

  `Forward Pass`

  Solve the first-stage problem (defined by replacing $\widetilde{D}_2(\cdot)$ with $\widetilde{\mathfrak{D}}_2^i(\cdot)$ and adding multiplier bounds in (8.7)). Store the trial point π_1^i.
  for stages $t = 2, \dots, T$ **do**
    Solve the stage-t subproblem (defined by replacing $\widetilde{D}_{t+1}(\cdot)$ with $\widetilde{\mathfrak{D}}_{t+1}^i(\cdot)$ and adding multiplier bounds in (8.7)) for π_{t-1}^i to obtain $\pi_{tj}^i, j = 1, \dots, q_t$.
    Sample $\widetilde{j}$ from $j = 1, \dots, q_t$ and set $\pi_t^i = \pi_{t\widetilde{j}}$.
  end for

  `Backward Pass`

  for stages $t = T, \dots, 2$ **do**
    Solve the updated stage-t subproblem (2.10) (defined by replacing $\widetilde{D}_{t+1}(\cdot)$ with $\widetilde{\mathfrak{D}}_{t+1}^{i+1}(\cdot)$ and adding multiplier bounds in (8.7)) for π_{t-1}^i. Store the optimal value $\overline{D}_t(\pi_{t-1}^i)$ and the optimal dual vector x_{t-1}^i.
    Compute

$$\alpha_t^{D,i} := \overline{D}_t(\pi_{t-1}^i) - \left(\beta_t^{D,i}\right)^\top \pi_{t-1}^i$$

    and

$$\beta_t^{D,i} := -W_{t-1} x_{t-1}^i.$$

    Update the cut approximation of $\widetilde{D}_t(\cdot)$ to

$$\widetilde{\mathfrak{D}}_t^{i+1}(x_{t-1}) := \min\left\{\widetilde{\mathfrak{D}}_t^i(x_{t-1}),\ \alpha_t^{D,i} + \left(\beta_t^{D,i}\right)^\top \pi_{t-1}\right\}.$$

  end for
  Solve the first-stage problem (defined by replacing $\widetilde{D}_2(\cdot)$ with $\widetilde{\mathfrak{D}}_2^{i+1}(\cdot)$ and adding multiplier bounds in (3.8)) to obtain an upper bound $\overline{v}^i$.
end while

Output: Upper bound $\overline{v}^i$ for v^*.

4. Run a forward pass of dual SDDP, to obtain a new dual trajectory $(\tilde{\lambda}_t^i)_{t\in[T]}$ and update the cuts along this trajectory.

One computational drawback of this framework, and of dual SDDP in general, is that each iteration of dual SDDP is much more computational expensive than for standard (primal) SDDP. This hampers the application of a solely deterministic

stopping criterion for very large problems [104, 126].

9. Applications. In this section, we present different application areas of SDDP. We also point out applications in which some of the Assumptions 1 to 9 are not satisfied, and therefore either modifications of (MSLP) or algorithmic extensions are required in order to apply SDDP. These use cases can be regarded as a motivation for the enhancements of SDDP that we cover in the following sections.

9.1. Power System Optimization. By far the dominating application field of SDDP is power system optimization, in particular, the operational planning of energy systems including hydro storages by a central planner. This is due to its adequacy for such problems, but also due to its origins in optimizing the operational planning of the Brazilian hydrothermal system [159, 160].

In general, solving power system optimization problems is a very complex task, as it allows for incorporation of various technical and economical details and uncertainties [117, 156, 189, 190, 191, 217, 218, 242]. Including all these details in one single problem is computationally intractable. Therefore, usually a hierarchy of problems is considered, dealing with different time-scales and perspectives [51], such as short-term dispatch (a few days or weeks), mid-term operational planning (1-2 years) and long-term operational planning (3-5 years) [76, 88]. Results from a long-term model can then be incorporated into one with a shorter horizon, but more detail in other modeling aspects.

9.1.1. Long-term Operational Planning. SDDP is most prominently used for long-term operational planning (LTOP) of hydrothermal power systems, also called long-term hydrothermal scheduling (LTHS). In the research literature, SDDP has been applied to LTOP of various hydrothermal systems, with the most prominent ones being the hydro power dominated systems in Brazil [15, 32, 33, 34, 46, 51, 52, 53, 56, 91, 104, 112, 133, 134, 136, 140, 143, 168, 173, 211, 212, 213, 216, 224, 234], other Central or South American countries [6, 74, 186, 221], Norway [87, 195] and New Zealand [167, 169, 238]. Additionally, to this day, SDDP is applied by the Brazilian system operator ONS in practice [141, 142].

The aim in LTOP is to determine an optimal policy for the amount of power to be generated by thermal and hydroelectrical utilities over some planning horizon of several years (usually with monthly resolution) such that demand is satisfied, technical constraints are fulfilled and the expected cost is minimized [167]. The main focus is on managing hydro reservoirs, and thus the water resource efficiently. This is not trivial. While there is an incentive to use all the water in a reservoir immediately, as no fuel costs occur, also the potential value of storing water for later stages has to be considered, with taking into account the uncertainty of future inflows. For this reason, it can be beneficial to retain water in wet periods for following dryer periods. The ability to store water in reservoirs leads to a temporal coupling of the stages. The number of inflow realizations q_t per stage is typically chosen in a range between 20 and 100. For $T = 60$, this yields a scenario tree with about $1.15 \cdot 10^{78}$ or 10^{120} scenarios. Per forward pass, either a single scenario [52] or 100 to 200 scenarios are sampled.

LTOP can be used to illustrate some of the challenges and limits of standard SDDP, and thus motivate the necessity of extensions.

Autoregressive Uncertainty. In LTOP, the main source of uncertainty are future (usually monthly) inflows into the reservoirs. These inflows often show seasonality and a temporal or spatial coupling which has to be considered in modeling.

Therefore, usually *autoregressive* (AR) processes are used to model and forecast them, in particular *periodic autoregressive* (PAR) [140, 141] and related models [146]. This means that for each reservoir and each month a different AR model is fitted, or in other words, that the parameters in the AR model are allowed to differ between months.

Additionally, often hydro reservoirs are organized in cascade systems. Then, the generation of one turbine may affect the inflow of downstream reservoirs, such that they cannot be managed separately. For this reason, inflows often do not only show temporal correlation and seasonality, but also spatial correlation. To address this, instead of PAR, *spatial periodic autoregressive* (SPAR) models can be used [134]. These models are still linear, but instead of only autoregressive components, *i.e.*, lags of ξ_{it} for some reservoir i, also lags of the inflows of neighboring reservoirs i' are used to explain ξ_{it}. Apart from inflow lags, also different exogenous variables, such as climate indices, precipitation or sea temperature can be used to explain inflows [131, 172].

Whenever an AR process is used for the uncertain data, the assumption of stagewise independence (Assumption 2) is not satisfied. This motivates an extension of SDDP able to handle stagewise dependent uncertainty. We discuss this in Section 14.

Nonlinear Uncertainty. When modeling hydro inflows, the error terms in the AR process are usually assumed to be i.i.d. with normal or log-normal distribution [51, 134]. In the latter case, the model is also referred to as a geometric PAR (GPAR) model [136]:

$$\ln(\xi_t) = \gamma_t + \Phi_t \ln(\xi_{t-1}) + \eta_t. \tag{9.1}$$

GPAR models are usually more accurate in modeling inflows, as these often tend to positive skewness and are thus not normally distributed. Moreover, they have the advantage that the requirement of non-negative inflows is naturally satisfied.

On the other hand, solving (9.1) for ξ_t yields an AR process with multiplicative instead of additive error terms [212], which is a nonlinear model. Incorporating this into the DPE destroys the convexity of $\mathscr{Q}_t(\cdot)$, making a direct application of SDDP impossible. Instead, the nonlinear model has to be approximated linearly [212]. Another idea is to normalize the inflows first using a Box-cox transformation. As such a transformation is nonlinear, still a linear approximation is required afterwards, though [175]. Further strategies to avoid non-negative inflows and nonlinearities are discussed in [51, 184]. In [49] it is suggested to apply bootstrapping to resample directly from the historical residuals instead of applying a nonlinear transformation.

Continuous Uncertainty and Distributional Uncertainty. As stated before, usually a normal or log-normal distribution is assumed for the error terms in the inflow models, both being continuous distributions (an exception is [178] where inflows are modeled as a continuous process with discrete random errors). For this reason, the assumption of finite discrete random variables (Assumption 5) is not satisfied. Additionally, the chosen distribution for the model may not coincide with the *true distribution* of the uncertain data. This raises the questions of how to handle continuous uncertainty and distributional uncertainty in SDDP. We address these questions in Section 11 and Section 13.

Computational Performance. Despite the amenities of SDDP, its performance may suffer for problems with a large number of state variables, due to its exponential complexity in the state dimension n_t, see Section 4.2. For instance, SDDP is computationally prohibitive for a complete model of the Brazilian energy system consisting of about 150 thermal plants and more than 150 hydro storages [52]. This is aggravated if the state dimension is artificially increased, *e.g.*, in order to deal with

stagewise dependent uncertainty, see Section 14. As a relief, it is common practice to aggregate reservoirs based on their region and hydrological properties in so-called *energy equivalent reservoirs* (EER) [4], thus reducing the state dimension [142]. However, this comes with an increased abstraction, and may lead to suboptimal policies. Moreover, as outlined in [51], the EER modeling may introduce some nonlinearities into the system, which have to be mitigated by linearization.

The computational complexity with respect to the state space also makes general performance improvements for SDDP indispensable, which we discuss in Section 21.

End-of-horizon Effect. Another challenge when applying SDDP to LTOP in practice is the so-called *end-of-horizon effect*. It relates to the effect that obtained policies do not guarantee a continuous and reliable energy supply *after* the planning period, because in an optimal policy, all energy remaining in the reservoirs will be used at the end of the planning period. A typical planning horizon for LTOP are 5 years with a monthly resolution, leading to 60 stages. A common practice to mitigate the end-of-horizon effect is to add 60 more stages to the problem, *i.e.*, to consider a problem with 120 stages [212], even if only decisions of the first half are about to be implemented. Alternatively, it seems natural to analyze how SDDP can be applied for problems with an infinite horizon or with a random horizon, where Assumption 1 is not satisfied. We address this in Sections 19 and 20.

Risk-aversion. Due to the high importance of system reliability and stability to prevent outages and electricity shortages, system planners may favor more risk-averse policies compared to the risk-neutral ones obtained by standard SDDP. Therefore, there has been an increased interest recently to take risk aversion into account when applying SDDP to LTOP [115, 216]. However, as Assumption 8 is no longer satisfied, this requires to extend standard SDDP to a risk-averse variant. We discuss different approaches to achieve this in Section 12.

9.1.2. Medium-term Operational Planning. Structurally, medium-term operational planning problems (MTOP) do not differ much from LTOP. The main difference is that a shorter, one- or two-year time horizon is considered [51, 167, 168, 186].

Price Uncertainty in the Objective. Especially on a medium-term time horizon, SDDP has also been adopted from the traditional setting with a single system operator to more market-driven systems, in which several electricity suppliers are active. In such systems, besides inflows also spot prices can be considered uncertain. This imposes an additional challenge to SDDP, as it leads to stagewise dependent uncertainty in the objective. We discuss this in detail in Section 14. To deal with this challenge, for instance, for the operational planning of the Norwegian hydro-storage system, usually a combined SDP/SDDP approach is used [86, 87, 88, 107, 108].

Water Head Effect. In LTOP the so-called *water head effect* of hydro storages is often disregarded, but it may become decision-relevant in (MTOP). This effect describes that the production of a hydro plant increases with the net head of the reservoir. As this production function is multiplied with the water discharge, it introduces non-convexities to the problem. Therefore, if this nonlinear effect is explicitly considered, suitable extensions of SDDP to non-convex problem are required [38, 110, 170]. We cover such extensions in Section 16.

9.1.3. More Energy Applications. We briefly summarize further applications of SDDP in power system optimization.

Short-term Dispatch. SDDP is particularly suited for long-term planning, but it can also be applied to short-term economic dispatch problems [39, 55, 125, 155]. For shorter time horizons, it may be reasonable to include additional system aspects,

for instance power flow and security constraints, reserve energy or different ancillary services [139, 224]. If security constraints are considered, usually linear DC power flow models are used, but recently also AC power flow has gained interest [119, 194].

Another research stream considers CO_2 emissions, which can be covered by imposing an emission quota system [14, 187, 185] or by introducing emission trading [188]. The first approach leads to an (MSLP) which has no block-diagonal structure (Assumption 7). We discuss how SDDP can be applied in this case in Section 18.

Using a reasonable extension to mixed-integer programs, see Section 16, also unit commitment problems are accessible by the SDDP idea [244].

Different Storage Systems. As different types of storage systems can be modeled similar to hydro storages, SDDP is also applicable to such systems, for instance, to optimize gas storage facilities [235] or energy storages in microgrids [22].

Optimal Bidding. Instead of minimizing expected system cost from the perspective of a central system operator, in strategic bidding problems power plant operators attempt to determine an optimal bidding policy in order to maximize their expected revenue, while taking into account information uncertainty, for example with respect to inflows or the market-clearing price; see [220, 222] for an overview.

Since the future revenue functions of the price-maker have a sawtooth shape, the resulting problem is non-convex [221]. Therefore, to apply the SDDP idea, tailor-made extensions are required, *e.g.*, convexifications, approximations by saddle cuts [59] or by step functions [170, 238]. For methodological details, we refer to Section 16.

Recently, also applying SDDP to optimize trading in continuous intraday markets has gained attention [214].

Investment Planning. An important long-term optimization problem in power systems is to make optimal (risk-averse) investment decisions, either with respect to the expansion of renewables [35, 130, 223] or to conventional projects.

For conventional power systems, common investment problems address the questions of generation expansion or transmission expansion. The main challenge with such problems is that they naturally impose the introduction of integer decision variables. Therefore, in such a case relaxations [154] or appropriate extensions of SDDP, *e.g.*, SDDiP [244], have to be used (see Section 16). Alternatively, SDDP can be incorporated into a larger Benders decomposition framework, where at the first stage binary investment decisions are taken and at the second stage a multistage stochastic linear program is solved by SDDP [185]. Similar applications are considered in [56] and [45] with a special focus on risk and reliability constraints.

Coping with Renewable Uncertainty. An increasing share of renewable energy sources introduces more variability to an energy system, which has to be taken into account and balanced by appropriate mechanisms. The usage of distributed grid-level storage, such as batteries or electric vehicles, for smoothing out the variable generation of renewables is examined using SDDP in [70, 245].

9.2. Water Resource Management. In many energy applications of SDDP, managing water resources plays a key role, as it couples subsequent stages. Apart from energy optimization, SDDP is also applied to more general water resource management problems, where not only energy production, but also water usage for irrigation in agriculture [162, 229], flow requirements for navigation [229], groundwater [144] or ecological constraints [228] are taken into account in the operational planning of reservoirs. Also related is the problem of river basin management [196].

Additionally, SDDP is used for assessing various quantities in hydrological systems, *e.g.*, the value of water [232], risk for dam projects [2, 231], resource vulnera-

bilities [197] or benefits and costs of cooperation or non-cooperation [145, 230].

9.3. Portfolio Management. The optimal management of a portfolio of investments, also referred to as *asset allocation*, can be modeled as an (MSLP) [47]. The aim is to distribute a fixed investment sum among a finite number of assets with uncertain returns, in such a way that the expected return at the end of the considered horizon is maximized. By selling or buying certain amounts of assets, the investor can restructure his portfolio in each time period. Usually, both operations are associated with transactions costs, which leads to a very complex problem [233].

In the literature on SDDP, asset allocation problems are quite popular to test proposed improvements and enhancements of SDDP, such as regularization [97], cut-sharing [91] or inexact cuts [8]. Since most investors are risk-averse, asset allocation problems are a popular application [64, 67, 68, 113, 120, 121], but also one of the main drivers for the development of risk-averse SDDP, which we introduce in Section 12.

For applications of practical interest, asset allocation becomes very challenging, as pointed out in [233]. First, risk aversion parameters such as λ_t or α_t, see Section 12, are not intuitive to choose in such a way that the true preferences of an investor are appropriately represented. For this reason, the authors propose to solve a risk-constrained model with one-period conditional AVaR constraints instead of a usual risk-averse SDDP approach. Second, assuming stagewise independence of asset returns may prove unrealistic, requiring a more sophisticated approach such as incorporating a Markov chain, see Section 14. Moreover, the large supply of potential assets leads to a high-dimensional state space.

9.4. Further Applications. Although the focus is on the previous applications, occasionally also other types of applications are investigated using SDDP. Among those applications are dairy farming [65, 90], newsvendor problems [5, 157], inventory management [8, 63, 94, 104], lot-sizing [226] and routing problems [64]. In [54] and [244] airline revenue management is explored, which is an established problem in dynamic programming, but requires integer variables.

10. Software. Until recently, SDDP implementations have been solely restricted to closed research projects or commercial products. For commercial products, most established is the SDDP implementation by PSR, a Brazilian energy consultancy [179]. A newer stochastic programming software, which also includes SDDP ideas, is provided by Quantego and can be accessed using MATLAB, Python and Java [180]. For research projects, various different implementations exist, covering programming languages like AMPL, C++, Fortran, GAMS, Java or MATLAB, see [61].

In the last few years, open-source implementations have gained more and more interest, with the aim to increase research transparency, enhance research exchange and benchmarking, and facilitate access to SDDP in industry and science [61]. The most prominent programming language in this regard is Julia [21], which provides its own algebraic modeling language JUMP [66] and is increasingly used in operations research and especially stochastic progamming. By now, with `StochDynamicProgram.jl` [127], `StructDualDynProg.jl` [128] and `SDDP.jl` [61] there exist three SDDP implementations in Julia. Similarly, SDDP packages are available in MATLAB (`FAST` [36]), C++ (`StOpt` [84]) and Python (`msppy` [54]).

Currently, `SDDP.jl`, which is based on the concept of policy graphs [60], can be considered the most comprehensive package. It provides many of the features described in this paper, such as cut selection, parallelism, Markov chain SDDP, objective states, belief states, SDDiP, as well as different stopping criteria and sampling

approaches. Moreover, it includes some of the approaches discussed for distributionally robust and risk-averse SDDP. However, as most other packages, it requires the underlying stochastic process to be finite. Thus, if Assumption 5 is not satisfied, some discretization has to be applied a priori. Then, the results obtained by SDDP are valid for the discretized problem, but not put into perspective with respect to the *true* problem. `msppy`, on the other hand, integrates both, the discretization by SAA and the solution by SDDP in one package and, thus, can naturally be applied to problems with continuous uncertainty [54].

A more detailed comparison of currently available libraries is presented in [61].

11. SDDP for Continuous Uncertainty [relaxing Assumption 5]. So far, we assumed the uncertainty in (MSLP) to be modeled by some discrete and finite random process, see Assumption 5, in order for SDDP to be applicable. Until the recent work by Forcier and Leclère [75], also all convergence proofs for SDDP leveraged Assumption 5. However, in many practical applications, this assumption is not justified. For example, if the stochastic process governing the uncertain data is modeled by a time series model, the random error terms are usually assumed to follow a continuous distribution [206], see Section 9. In the remainder of this section, we denote a problem with such a continuous data process by $(\widetilde{P})$.

As pointed out in Section 2.3, for sizes of practical interest, problem $(\widetilde{P})$ is considered computationally intractable. Therefore, if the true distribution $F_{\boldsymbol{\xi}}$ of the stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is continuous, usually an approximation with finitely many scenarios is used. In the literature on multistage stochastic programming, a variety of techniques are proposed to generate (and reduce) scenario tree approximations of continuous stochastic processes. For an overview we refer to [135].

Before focusing on the most prominent technique in the next section, we should note that recently Forcier and Leclère [75] presented a (theoretical) extension of SDDP that directly works with continuous uncertainty. They show that under linearity (Assumption 6), given that $\boldsymbol{W}_t(\boldsymbol{\xi}_t)$ is finitely supported and given that $\boldsymbol{c}_t(\boldsymbol{\xi}_t)$ and $(\boldsymbol{T}_{t-1}(\boldsymbol{\xi}_t), \boldsymbol{h}_t(\boldsymbol{\xi}_t))$ are independent, the ideas of SDDP may be extended to the case of continuous random variables $\boldsymbol{\xi}_t$. Instead of generating cuts for $\mathscr{Q}_t(\cdot)$ by deriving them for each realization $j = 1, \dots, q_t$ separately and then aggregating, which is only possible for finite random variables, the main idea is that cuts are derived for $\mathscr{Q}_t(\cdot)$ directly. To achieve this, the support Ξ_t of $\boldsymbol{\xi}_t$ is partitioned. For the partition $\mathcal{P}$ then special *partitioned expected value functions* $V_{\mathcal{P},t}(\cdot)$ are defined, in which the corresponding original value functions $Q_t(\cdot)$ and the expectation operator $\mathbb{E}[\cdot]$ are exchanged. This allows to evaluate and approximate them even for continuous $\boldsymbol{\xi}_t$. If $V_{\mathcal{P},t}(\cdot)$ is guaranteed to underestimate $\mathscr{Q}_t(\cdot)$ for all x_{t-1} (validity) and to coincide with $\mathscr{Q}_t(\cdot)$ at some point $\bar{x}_{t-1}$ (tightness), it is called *adapted* to $\bar{x}_{t-1}$. Given an adapted partition, by deriving cuts for $V_{\mathcal{P},t}(\cdot)$ as in standard SDDP, valid and tight cuts for $\mathscr{Q}_t(\cdot)$ can be obtained even for problems $(\widetilde{P})$. Forcier and Leclère also show how adapted partitions can be constructed for (MSLP) explicitly under certain assumptions. However, we are not aware of any applications and computational tests of the resulting extension of SDDP.

11.1. Sample Average Approximation (SAA). The most common approximation approach for continuous uncertainty is to use random sampling. That means that the distribution $F_{\boldsymbol{\xi}}$ is *approximated* using an empirical distribution $\widetilde{F}_N$ with a finite number N of scenarios, which is obtained by sampling from $F_{\boldsymbol{\xi}}$ [206]. This yields an approximating problem $(\widetilde{P}_N)$, which then can be handled by SDDP. Often, this

technique is referred to as *sample average approximation* (SAA), especially, if classical Monte Carlo sampling is used. We discuss SAA and the application of SDDP to solve an SAA problem in more detail now. For a general analysis of SAA, we refer the interested reader to [209].

SAA and SDDP. Under stagewise independence of $(\boldsymbol{\xi}_t)_{t\in[T]}$ (Assumption 2), it is desirable to preserve this property in the SAA problem, especially if the latter should be solved by SDDP. To achieve this, random sampling can be applied to each stage $t = 2, \dots, T$ independently with sample size $\widetilde{q}_t$ [206]. The obtained SAA has a total number of $N = \prod_{t=2}^{T} \widetilde{q}_t$ scenarios, *i.e.*, the number of scenarios is exponentially growing in the number of stages [206].

For the SAA problem $(\widetilde{P}_N)$, for each stage $t = 2, \dots, T$ and each sample $j = 1, ..., \widetilde{q}_t$, the DPE can be written as

$$\widetilde{Q}_t\big(x_{t-1}, \widetilde{\xi}_{tj}\big) := \begin{cases} \min\limits_{x_t} & \big(c_t\big(\widetilde{\xi}_{tj}\big)\big)^\top x_t + \widetilde{\mathscr{Q}}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \widetilde{\xi}_{tj}) \end{cases} \tag{11.1}$$

where

$$\widetilde{\mathscr{Q}}_{t+1}(x_t) := \frac{1}{N_{t+1}} \sum_{j=1}^{\widetilde{q}_{t+1}} \widetilde{Q}_{t+1}\big(x_t, \widetilde{\xi}_{t+1,j}\big) \tag{11.2}$$

and $\widetilde{\mathscr{Q}}_{T+1}(\cdot) \equiv 0$. For the first stage, we obtain

$$\widetilde{v}_N := \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \widetilde{\mathscr{Q}}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{11.3}$$

The DPE (11.1)-(11.3) can be approached by SDDP as described in Section 3. However, in contrast to the problems considered there, the SAA problems are random, as they depend on a sample from the true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$.

SAA Properties. Since the aim is to solve the original problem $(\widetilde{P})$, the central question is how the solution and the bounds obtained by applying SDDP to the SAA problem $(\widetilde{P}_N)$ relate to the solution of $(\widetilde{P})$. We denote the optimal value of $(\widetilde{P})$ by $\widetilde{v}^*$ and the bounds obtained by SDDP in iteration i with $\underline{\widetilde{v}}^i$ and $\overline{\widetilde{v}}^i_{\mathcal{K}}$. We summarize important properties of SAA.

(P.11.1) *Consistency.* It can be shown that the optimal value $\widetilde{\boldsymbol{v}}_N$ provides a consistent estimator of the true optimal value $\widetilde{v}^*$, *i.e.*, $\lim_{\widetilde{q}_2,\dots,\widetilde{q}_T\to\infty} \widetilde{\boldsymbol{v}}_N = \widetilde{v}^*$ with probability 1 [206, 209]. The intuition behind this is that asymptotically, the structure of the true process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is recovered. In practical applications, increasing $\widetilde{q}_t$ to infinity is computationally intractable, though.

(P.11.2) *Bias.* $\widetilde{\boldsymbol{v}}_N$ is a biased estimator of $\widetilde{v}^*$, more precisely, $\mathbb{E}[\widetilde{\boldsymbol{v}}_N] \leq \widetilde{v}^*$ for all N [209], since only a subset of all scenarios is considered and the decisions are optimized with respect to these scenarios [52]. This means that solving the SAA problem provides a (converging) estimator of a lower bound for $\widetilde{v}^*$ [203].

(P.11.3) *Lower Bounds.* In each iteration i of SDDP, we have $\underline{\widetilde{v}}^i \leq \widetilde{v}_N$. Therefore, $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}^i] \leq \widetilde{v}^*$ [206], and the SDDP lower bound is a statistical lower bound for $\widetilde{v}^*$. Note, however, that both, $\widetilde{\boldsymbol{v}}_N$ and $\underline{\widetilde{\boldsymbol{v}}}^i$, are lower bounds in expectation only, whereas this is not clear for one specific SAA problem $(\widetilde{P}_N)$.

(P.11.4) *Upper Bounds.* Applying SDDP to the DPE (11.1)-(11.3) yields a policy. Under relatively complete recourse (see Assumption 9) with respect to the

true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$, this policy also yields feasible decisions if applied to any realization $(\xi_t)_{t\in[T]}$ of this true process. By computing

$$\mathbb{E}\left[\sum_{t=1}^{T}\big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t^i\big(\xi_{[t]}\big)\right] \tag{11.4}$$

with the expectation taken with respect to the true process, a valid upper bound for $\widetilde{v}^*$ can be obtained [206].

(P.11.5) The sample mean $\overline{\widetilde{v}}^i_{\mathcal{K}}$ determined in iteration i in SDDP is an unbiased and consistent estimator of (11.4). Hence, $\mathbb{E}\left[\overline{\widetilde{\boldsymbol{v}}}^i_{\mathcal{K}}\right] \geq \widetilde{v}^*$.

Even with these theoretical properties, solving $(\widetilde{P})$ using SAA may be computationally intractable. Shapiro shows that even under relatively complete recourse (see Assumption 9) and stagewise independence (Assumption 2) of the true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$, the total number of scenarios required in SAA problem $(\widetilde{P}_N)$ to solve $(\widetilde{P})$ with a reasonable accuracy $\varepsilon > 0$ grows exponentially in the number of stages [204]. Therefore, he proposes to use smaller sample sizes $\widetilde{q}_t$ for later stages, although then the accuracy of the solution cannot be guaranteed anymore [205].

Clearly, there exists a trade-off between the quality of the obtained bounds for $\widetilde{v}^*$ and the computational tractability of the SAA problem. Approximating $F_{\boldsymbol{\xi}}$ with F_N using very large sample sizes $\widetilde{q}_t$ for all $t = 2, \dots, T$, a much better representation of the original process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is obtained, leading to a better approximation of $\widetilde{v}^*$. However, in this case, it may be even impossible to solve the SAA problem to optimality in reasonable time, as it may take too long until all scenarios are *eventually* sampled [206]. On the other hand, a very rough approximation yields a problem $(\widetilde{P}_N)$, which can be solved efficiently by SDDP, but does not provide reasonable information about the solution to the true problem $(\widetilde{P})$ [121].

11.2. Assessing Policy Quality. As it is computationally intractable to solve an SAA problem of $(\widetilde{P})$ with a sample size that guarantees a predetermined accuracy, in practice, usually moderate sample sizes are used. For example, in [52], sample sizes with branching numbers $\tilde{q}_t$ between 5 and 200 are tested.

The bounds $\underline{\widetilde{v}}^i$ and $\overline{\widetilde{v}}^i_{\mathcal{K}}$ in SDDP are determined using one specific sample of $(\boldsymbol{\xi}_t)_{t\in[T]}$. Therefore, they only measure the *in-sample* performance of the determined feasible policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$. To assess its quality for the original problem $(\widetilde{P})$, *i.e.*, its *out-of-sample* performance, it is required to evaluate it with respect to the original process $(\boldsymbol{\xi}_t)_{t\in[T]}$. Such an evaluation also allows one to compare policies obtained for different SAA problems, which can be helpful in designing appropriate sampling techniques and sample sizes [52].

Various techniques have been proposed in stochastic programming to measure the performance of feasible policies, such as analyzing optimality conditions, assessing solution stability or estimating the optimality gap [52]. Specifically for SDDP, Morton et al. have made substantial contributions [42, 52, 121], which are based on estimating the optimality gap ([121] analyzes a risk-averse variant of SDDP, see Section 12). We discuss their ideas for the risk-neutral case thoroughly in the remainder of this subsection. In accordance with [52], we only consider uncertainty in the RHS of $(\widetilde{P})$.

Estimating the Optimality Gap. For some feasible policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$, let $\widetilde{v}(\xi) = \sum_{t=1}^{T} c_t x_t\big(\xi_{[t]}\big)$ denote the random cost for some arbitrary scenario path $\xi = (\xi_1, \dots, \xi_T)$. From (P.11.4) we have $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] \geq \widetilde{v}^*$. Therefore, the optimality gap

induced by policy $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t\in[T]}$ can be expressed as

$$\Delta := \mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] - \widetilde{v}^* \geq 0.$$

This gap cannot be directly evaluated because the optimal value $\widetilde{v}^*$ is not known. Using some lower bound for $\widetilde{v}^*$, Δ can be overestimated though. Such lower bound is given by $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$, see (P.11.3). This yields

$$\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] - \mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}] \geq \Delta \geq 0. \tag{11.5}$$

Still, the left-hand side of (11.5) is computationally infeasible to evaluate. It requires excessive computational effort to evaluate policy $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t\in[T]}$ for all possible scenarios to obtain $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Furthermore, from SDDP only one specific realization of $\underline{\widetilde{v}}$ is known. Therefore, in [52] it is proposed to use estimators for both terms to derive an approximate one-sided confidence interval bounding Δ from above.

Upper Bound Estimation. The SDDP policy $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t\in[T]}$ is feasible for the original problem $(\widetilde{P})$, see (P.11.4). Hence, it can be evaluated for any realization of $(\boldsymbol{\xi}_t)_{t\in[T]}$ to assess its out-of-sample performance. Let us sample M_u i.i.d. scenario paths from $(\boldsymbol{\xi}_t)_{t\in[T]}$. For each of those sampled scenarios $\xi^\ell, \ell = 1,\dots,M_u$, the SDDP subproblems (2.10) are solved in forward direction, yielding $x_t(\xi^\ell_{[t]})$ and $\widetilde{v}(\xi^\ell)$ [52]. An upper bound estimator is then defined by the sample mean

$$U_{M_u} := \frac{1}{M_u}\sum_{\ell=1}^{M_u} \widetilde{v}(\xi^\ell). \tag{11.6}$$

Similarly to the in-sample estimator, this estimator is an unbiased and consistent estimator of $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Its sample variance is given by [52]

$$\sigma_U^2 := \frac{1}{M_u - 1}\sum_{\ell=1}^{M_u} (\widetilde{v}(\xi^\ell) - U_{M_u})^2. \tag{11.7}$$

Alternatively, an upper bound estimator can be obtained by sampling a finite number of different SAA problems, and applying the SDDP policy $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t\in[T]}$ to each of them [42]. This comes at the cost of increased computational effort.

Lower Bound Estimation with Several SAA Problems. From SDDP, only one single realization of $\underline{\widetilde{v}}$ is known. Hence, it is not directly possible to determine a sampling error for this point estimate and to derive a confidence interval for $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$.

One approach to derive a lower bound estimator is to solve a finite number of different SAA problems with SDDP and to determine the mean of the lower bounds $\underline{\widetilde{v}}$. To be precise, M_l different SAA problems are constructed, each by sampling $\widehat{q}_t$ realizations per stage from $(\boldsymbol{\xi}_t)_{t\in[T]}$. Then SDDP is run, yielding the lower bounds $\underline{\widetilde{v}}^\ell, \ell = 1,\dots,M_l$ [52]. The sample mean

$$L_{M_l} := \frac{1}{M_l}\sum_{\ell=1}^{M_l} \underline{\widetilde{v}}^\ell \tag{11.8}$$

then defines an estimator for $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$ with sample variance

$$\sigma_l^2 := \frac{1}{M_l - 1}\sum_{\ell=1}^{M_l} (\underline{\widetilde{v}}^\ell - L_{M_l})^2.$$

Note that instead of lower bounds $\widetilde{\underline{v}}^{\ell}$, also the optimal values $\widetilde{v}_N^{\ell}$ could be used in estimator (11.8) [52]. We already discussed in Section 11.1 that it may be computationally intractable to solve one single SAA problem to optimality, though. Thus, using $\widetilde{\underline{v}}^{\ell}$ may be computationally preferable.

In principle, applying SDDP to not only one, but several SAA problems and building the mean of the obtained bounds seems very reasonable from a statistical perspective, as the outcome of one SAA problem is random. This also has another possible benefit: If SDDP is run for M_l different SAA problems ($\widetilde{P}_N^l$), each of these problems yields a different feasible policy. By calculating the upper bound estimator U_{M_u} (11.6) for each of them, directly M_l different policies could be compared.

However, for problems with multiple stages and for sufficiently high $\widehat{N}_t$, this becomes computationally intractable, even without solving ($\widetilde{P}_N^l$) exactly. Therefore, de Matos et al. [52] follow the strategy to run SDDP once for some SAA problem with larger branch size $\widetilde{q}_t$ to determine a high quality policy and then, afterwards, to run SDDP for M_l SAA problems with smaller branch size $\widehat{q}_t$ only to produce the lower bound estimate L_{M_l} and assess the quality of that policy. In their numerical tests, they choose values between 5 and 200 for $\widetilde{q}_t$ and 5 for $\widehat{q}_t$. In general, it is not clear though, how to choose $\widehat{q}_t$ to reach a reasonable trade-off between computational tractability and an appropriate quality of the lower bound estimator.

Lower Bound Estimation with One SAA Problem. An alternative and less costly lower bound estimator is derived by only using the existing SAA problem, which has been applied to determine the policy that is to be assessed [52].

The idea is then to use the SDDP outcome $\widetilde{\underline{v}}$ as the point estimate L_{M_l} for the lower bound. To estimate the unknown sampling error of $\widetilde{\underline{v}}$, the sampling error of the in-sample upper bound estimator is used. This means that M_l scenarios are sampled from F_N (the SAA problem distribution), and formulas (11.6) and (11.7) with M_l in the role of M_u are used to compute an upper bound estimate $\widetilde{\overline{v}}_{M_l}$ and sample error σ_l^2. The idea behind applying this sampling error is that $\widetilde{\underline{v}}$ and $\mathbb{E}[\widetilde{\overline{\boldsymbol{v}}}_{M_l}]$ are equal if SDDP has been run to optimality. However, this also implies that if SDDP has not converged (or if $\widetilde{q}_t$ is not sufficiently large) the sampling error may be underestimated, and thus the confidence intervals drawn from this become overly optimistic [52].

Confidence Intervals. Using the bound estimators and their sample variances, asymptotically valid confidence intervals can be derived [52].

$$\left(-\infty, U_{M_u} + t_{M_u-1,\alpha}\frac{\sigma_U}{\sqrt{M_u}}\right]$$

is an asymptotically valid, and for finite M_u approximate, $(1-\alpha)\%$ confidence interval for $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Here, $t_{M_u-1,\alpha}$ denotes the $(1-\alpha)$-level quantile of a student's t distribution with $M_u - 1$ degrees of freedom. Similarly,

$$\left[L_{M_l} - t_{M_l-1,\alpha}\frac{\sigma_l}{\sqrt{M_l}}, \infty\right)$$

is an asymptotically valid, and for finite M_l approximate, $(1-\alpha)\%$ confidence interval for $\widetilde{v}^*$. Using only one SAA problem, this confidence interval is only valid if SDDP has converged and if $\widetilde{q}_t$ is sufficiently large. Combining both intervals yields

$$\left[0, [U_{M_u} - L_{M_l}]_+ + t_{M_l-1,\alpha}\frac{\sigma_l}{\sqrt{M_l}} + t_{M_u-1,\alpha}\frac{\sigma_U}{\sqrt{M_u}}\right]$$

as a one-sided approximate confidence interval for the optimality gap Δ [52]. Here, $[x]_+ := \max\{x, 0\}$.

11.3. Variance Reduction Techniques. Instead of MC sampling, also importance sampling [157] and variance reduction techniques (see Section 6.2) can be applied to obtain SAA estimators with reduced bias and variance.

In [112], numerical tests comparing MC, LHS and RQMC indicate that RQMC yields the most promising results when it comes to determining representative SAA problems. In [52] also MC, LHS and RMC are compared for different branch sizes and policy evaluation strategies. The results indicate that with both LHS and RQMC, a reduction of bias and sampling error, a higher policy quality and tighter confidence intervals can be achieved in comparison with MC sampling, especially for smaller branch sizes $\widetilde{q}_t$. For smaller branch sizes LHS appears to be superior, while RQMC yields better results for larger branch sizes. While showing higher variability for MC sampling, if combined with RQMC and LHS sampling, the computationally preferable lower bound estimator using only in-sample scenarios from the existing SAA yields comparable results to the approach solving several SAA problems [52].

12. Risk-averse SDDP [relaxing Assumption 8]. In SDDP, as described in Section 3, a risk-neutral optimal policy is determined for (MSLP) (see Assumption 8). More precisely, (MSLP) minimizes the expectation of the total objective value over all stages $t \in [T]$ over feasible policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$, which satisfy non-anticipativity and all constraints. Hence, it can be formulated as the single problem (2.3) with objective

$$\min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \mathbb{E}\left[\sum_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right]. \tag{12.1}$$

As discussed in Section 2.4, this problem can be expressed equivalently using the DPE (2.4)-(2.6). This equivalence is based on two important properties of expected values, first the so-called *tower property*

$$\mathbb{E}_{\boldsymbol{\xi}_t}[\boldsymbol{Z}_t(\xi_t)] = \mathbb{E}_{\xi_{[t-1]}}\big[\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{[t-1]}}[\boldsymbol{Z}_t(\xi_t)]\big] \tag{12.2}$$

for some random variable $\boldsymbol{Z}_t$, and second its strict monotonicity (see property (R2') below for a formal definition) [207].

Recall that the objective value $\sum_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})$ is random, and its realizations depend on realizations of $(\boldsymbol{\xi}_t)_{t\in[T]}$. For some specific realization, the SDDP policy may produce an objective value which widely deviates from the expectation in (12.1). In practice, decision makers are often anxious not only to find a policy causing low costs *on average*, but also to avoid the risk of extremely high cost situations. This motivates to consider *risk-averse* approaches in stochastic programming.

For multistage stochastic programming, incorporating risk-aversion has been a popular research topic in the last decade. This includes theoretical fundamentals on dynamic risk measures [200] as well as algorithmic developments, such as rolling horizon approaches with chance constraints or AVaR constraints, which take risk aversion into account in the constraints of (MSLP) [102, 103]. For SDDP, most focus has been on replacing expectations in the objective (12.1) with some multi-period risk measure $\mathcal{R}[\cdot]$ (see below for a formal definition). This yields the multistage risk-averse

problem $(P_{\mathcal{R}})$:

$$
\begin{aligned}
\min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \quad & \mathcal{R}\left[\left(\boldsymbol{c}_1(\xi_1)\right)^\top \boldsymbol{x}_1(\xi_{[1]}), \ldots, \left(\boldsymbol{c}_T(\xi_T)\right)^\top \boldsymbol{x}_T(\xi_{[T]})\right] \\
\text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\
& \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2, \ldots, T \\
& \boldsymbol{x}_t(\cdot) \ \mathscr{F}_t\text{-measurable} \quad \forall t = 2, \ldots, T.
\end{aligned} \tag{12.3}
$$

We cover risk-averse SDDP in detail in the remainder of this section, but start with some theoretical concepts.

12.1. Risk Measures. In this section, we introduce some required foundations of risk measures, especially for multistage problems. As our focus is on algorithmic aspects of SDDP, we refer to the comprehensive coverage of this topic in [207, 209] for technical definitions and derivations.

12.1.1. Static Risk Measures. A *static* (or *one-period*) risk measure is a function $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ from the space $\mathcal{Z}$ of random variables $\boldsymbol{Z}$ to $\bar{R} := \mathbb{R} \cup \{-\infty, +\infty\}$. Often, $\mathcal{Z}$ is assumed to be $\mathcal{L}_1(\Omega, \mathscr{F}, \mathbb{P})$, *i.e.*, the space of all $\mathscr{F}$-measurable functions with finite first moments, as this ensures well-definedness and finiteness of many common risk measures. Importantly, since random variables are functions themselves, risk measures are actually *functionals*. This is sometimes emphasized by calling them *risk functionals* or *risk mappings*.

We summarize some well-known static risk measures:

- The expected value $\mathbb{E}[\cdot]$ is the most common risk measure. It is completely risk-neutral.
- The *value-at-risk* $\mathrm{VaR}_\alpha[\cdot]$ to level $\alpha \in (0, 1)$ is defined as the left-side $(1-\alpha)$-quantile of the cumulative distribution of some random variable $\boldsymbol{Z}$:

$$
\mathrm{VaR}_\alpha[\boldsymbol{Z}] := \inf\left\{u \in \mathbb{R} \ : \ \mathbb{P}(Z \leq u) \geq 1 - \alpha\right\}. \tag{12.4}
$$

  Note that this definition is not used consistently in the literature, and that the RHS of (12.4) may also be defined as $\mathrm{VaR}_{1-\alpha}[\boldsymbol{Z}]$.
- The *average value-at-risk* $\mathrm{AVaR}_\alpha[\cdot]$ to level $\alpha \in (0, 1)$ for some random variable $\boldsymbol{Z}$ is defined by [192]

$$
\mathrm{AVaR}_\alpha[\boldsymbol{Z}] := \inf\left\{u \in \mathbb{R} \ : \ u + \frac{1}{\alpha}\mathbb{E}\left[[Z - u]_+\right]\right\}, \tag{12.5}
$$

  where $[x]_+$ is defined as $\max\{x, 0\}$. Note that the infimum is always attained in our SDDP setting of finite randomness (Assumption 5) and finite value functions $Q_t(\cdot)$ (see Lemma 2.5).

  Remark 12.1. $\mathrm{AVaR}_\alpha[\cdot]$ is also called *conditional value-at-risk*, *expected shortfall*, *expected tail loss* or *superquantile*. In the literature on risk-averse stochastic programming, the first alternative is most frequently used with notation $\mathrm{CVaR}_\alpha[\cdot]$, but to avoid confusion when we introduce conditional risk measures later, we stick to average value-at-risk. □

  It can be shown that an equivalent formulation of $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ is given by [206]

$$
\mathrm{AVaR}_\alpha[\boldsymbol{Z}] = \mathrm{VaR}_\alpha[\boldsymbol{Z}] + \frac{1}{\alpha}\mathbb{E}\Big[\big[\boldsymbol{Z} - \mathrm{VaR}_\alpha[\boldsymbol{Z}]\big]_+\Big], \tag{12.6}
$$

i.e., $u^* = \mathrm{VaR}_\alpha[\boldsymbol{Z}]$ minimizes the RHS in (12.5).
$\mathrm{AVaR}_\alpha[\cdot]$ has some beneficial properties compared to $\mathrm{VaR}_\alpha[\cdot]$. It does not only consider the probability mass beyond $\mathrm{VaR}_\alpha[\cdot]$, but also its distribution, *e.g.*, if it has fat or long tails. Moreover, it allows to retain convexity of optimization problems, as we discuss later on. $\mathrm{VaR}_\alpha[\cdot]$ and $\mathrm{AVaR}_\alpha[\cdot]$ are illustrated in Figure 10.

Remark 12.2. For continuous random variables $\boldsymbol{Z}$, $\mathrm{AVaR}_\alpha[\cdot]$ may as well be defined as

$$\mathrm{AVaR}_\alpha[\boldsymbol{Z}] = \mathbb{E}\big[\boldsymbol{Z} | \boldsymbol{Z} \geq \mathrm{VaR}_\alpha[\boldsymbol{Z}]\big].$$

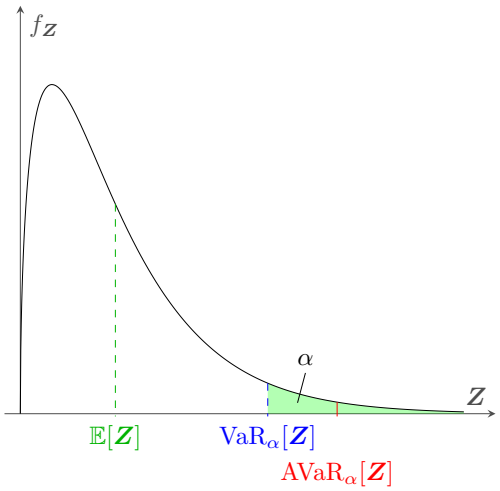


Fig. 10: $\mathrm{VaR}_\alpha[\boldsymbol{Z}]$ and $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ for a gamma distributed random variable $\boldsymbol{Z}$.

- In stochastic programming, often a convex combination of $\mathbb{E}[\cdot]$ and $\mathrm{AVaR}[\cdot]$ is considered, that is

$$\widehat{\rho}_{\alpha,\lambda}[\boldsymbol{Z}] := (1-\lambda)\mathbb{E}[\boldsymbol{Z}] + \lambda \mathrm{AVaR}_\alpha[\boldsymbol{Z}] \tag{12.7}$$

  for some $\lambda \in [0,1]$. The parameters λ and α control the risk-aversion. Choosing $\lambda = 0$ yields the standard risk-neutral model.
- For some $\gamma > 0$, the *entropic risk measure* is defined by

$$\mathbb{ENT}_\gamma[\boldsymbol{Z}] := \frac{1}{\gamma} \log\big(\mathbb{E}[e^{\gamma \boldsymbol{Z}}]\big). \tag{12.8}$$

  It generalizes $\mathbb{E}[\cdot]$ (for $\gamma \to 0$) and $\operatorname{ess\,sup}[\cdot]$ (for $\gamma \to \infty$), where $\operatorname{ess\,sup}[\boldsymbol{Z}]$ denotes the essential supremum of a random variable $\boldsymbol{Z}$.

It is often required that risk measures satisfy some special properties, especially in an optimization context. First, we assume that all considered risk measures are proper. Another desired property is *coherence*, a concept introduced by Artzner et al. [3]. We employ a slightly different definition from [209] and state it for the general case of continuous random variables:

DEFINITION 12.3. *A risk measure $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ is called* coherent*, if it satisfies*

Table 4: Properties of common risk measures.

	(R1)	(R2)	(R3)	(R4)	(R2')	(R5)
$\mathbb{E}[\cdot]$	✓	✓	✓	✓	✓	✓
$\mathrm{VaR}_\alpha[\cdot]$	-	✓	✓	✓	-	✓
$\mathrm{AVaR}_\alpha[\cdot]$	✓	✓	✓	✓	-	✓
$\widehat{\rho}_{\alpha,\lambda}[\cdot]$	✓	✓	✓	✓	✓*	✓
$\mathbb{ENT}_\gamma[\cdot]$	✓	✓	✓	-	✓	✓

* only for $\lambda \in [0, 1)$.

(R1) Convexity: for any $\boldsymbol{Z}_1, \boldsymbol{Z}_2 \in \mathcal{Z}$ and all $\lambda \in [0, 1]$ it holds

$$\rho(\lambda \boldsymbol{Z}_1 + (1-\lambda)\boldsymbol{Z}_2) \leq \lambda\rho(\boldsymbol{Z}_1) + (1-\lambda)\rho(\boldsymbol{Z}_2),$$

(R2) Monotonicity: If $\boldsymbol{Z}_1 \leq \boldsymbol{Z}_2$ almost surely, then $\rho(\boldsymbol{Z}_1) \leq \rho(\boldsymbol{Z}_2)$,
(R3) Translation Equivariance: If $a \in \mathbb{R}$ and $\boldsymbol{Z} \in \mathcal{Z}$, then $\rho(\boldsymbol{Z} + a) = \rho(\boldsymbol{Z}) + a$,
(R4) Positive Homogeneity: If $\lambda > 0$ and $\boldsymbol{Z} \in \mathcal{Z}$, then $\rho(\lambda \boldsymbol{Z}) = \lambda\rho(\boldsymbol{Z})$.

A risk measure satisfying only properties (R1), (R2) and (R3) is called *convex*. In fact, a key feature of coherent risk measures is that they are convex, and thus convex objective functions as they appear in $(P_{\mathcal{R}})$ and its DPE remain convex if $\rho[\cdot]$ is applied to them. $\mathrm{VaR}_\alpha[\cdot]$ is not a coherent risk measure, but $\mathrm{AVaR}_\alpha[\cdot]$ is [164]. Therefore, in optimization $\mathrm{AVaR}_\alpha[\cdot]$ is usually preferred over $\mathrm{VaR}_\alpha[\cdot]$.

As we exploit later, for every coherent risk measure there exists a dual representation as the *worst-case expectation* over some class of probability distributions over $(\Omega, \mathscr{F})$ [3]. More precisely, let $\mathcal{P}$ be a convex set of probability measures, then a coherent risk measure $\rho[\cdot]$ can be expressed as

$$\rho[Z] = \sup_{\mathbb{P} \in \mathcal{P}} \mathbb{E}_{\mathbb{P}}[Z]. \tag{12.9}$$

We introduce some additional relevant properties.

DEFINITION 12.4. *Let $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ be some risk measure. Then, the following properties can be defined.*

(R2') If the inequalities in (R2) in Definition 12.3 are strict, we call this property strict monotonicity.
(R5) Law Invariance: $\rho[\cdot]$ is called law invariant *with respect to $\mathbb{P}$, if for all $\boldsymbol{Z}, \boldsymbol{Z}' \in \mathcal{Z}$ with the same distribution also $\rho(\boldsymbol{Z}) = \rho(\boldsymbol{Z}')$ holds.*

Property (R5) implies that the risk measure ρ only depends on the distribution of the considered random variable $\boldsymbol{Z}$.

We summarize properties of the previously introduced risk measures in Table 4.

Remark 12.5. A classical approach in economics is to take risk aversion into account by means of non-decreasing and convex disutility (or concave utility) functions $g : \mathbb{R} \to \bar{\mathbb{R}}$ that are applied to some random variable $\boldsymbol{Z}$ before taking expectations. However, the obtained risk measure $\rho[\boldsymbol{Z}] = \mathbb{E}[g(\boldsymbol{Z})]$ does not satisfy property (R3) which is required to equivalently express $(P_{\mathcal{R}})$ using DPE. □

12.1.2. Multi-period Risk Measures. In a multistage setting, static, *i.e.*, one-period, risk measures have to be extended to several periods, more precisely, to a

sequence of random variables $\boldsymbol{Z} := \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_T$, which in our case model the stagewise objectives of (MSLP). We define such multi-period risk measures as functionals $\mathcal{R} : \mathcal{Z} \to \bar{\mathbb{R}}$ with $\mathcal{Z} = \mathcal{Z}_1 \times \mathcal{Z}_2 \times \ldots \times \mathcal{Z}_T$, where $\mathcal{Z}_t$ denotes the space of random variables $\boldsymbol{Z}_t$ for each stage t.

Choosing multi-period risk measures in a reasonable way is a challenging task. First, it is not clear how risk should be measured in a multistage setting [113]. Several different options exist [64, 113, 209], such as

$$\mathcal{R}[\boldsymbol{Z}] = \rho[\boldsymbol{Z}_1 + \cdots + \boldsymbol{Z}_T] \qquad \text{(end-of-horizon risk)} \tag{12.10}$$

$$\mathcal{R}[\boldsymbol{Z}] = \rho_1\Big[\boldsymbol{Z}_1 + \rho_{2|\mathcal{F}_1}\big[\boldsymbol{Z}_2 + \ldots + \rho_{T|\mathcal{F}_{T-1}}[\boldsymbol{Z}_T]\cdots\big]\Big] \qquad \text{(nested risk)} \tag{12.11}$$

$$\mathcal{R}[\boldsymbol{Z}] = \rho[\boldsymbol{Z}_1] + \ldots + \rho[\boldsymbol{Z}_T] \qquad \text{(stage-wise risk).} \tag{12.12}$$

Here, $\rho[\cdot]$ is some static risk measure, and $\rho_{t|\mathcal{F}_{t-1}}[\cdot], t = 2, \ldots, T$, (later also denoted by $\rho_{t|\xi_{[t-1]}}[\cdot]$) is a family of *conditional* risk measures, each mapping from $\mathcal{Z}_t$ to $\mathcal{Z}_{t-1}$ and defined as the static risk measure $\rho_t[\cdot]$ conditioned on $\mathscr{F}_{t-1}$ (or $\xi_{[t-1]}$, respectively). If $\rho_t[\cdot]$ is law-invariant (property (R5) in Definition 12.4), then $\rho_{t|\mathcal{F}_{t-1}}[\cdot]$ can be obtained by replacing the given distribution with the corresponding conditional distribution [209]. Usually the same static risk measure $\rho[\cdot]$ is chosen for all $\rho_t[\cdot], t = 2, \ldots, T$. Note that coherence of conditional risk measures can be defined completely analogously to unconditional ones. The idea of nested conditional risk measures goes back to Ruszczyński and Shapiro [201].

Remark 12.6. Under stagewise independence (Assumption 2), as we assume it for SDDP, the conditional risk measures $\rho_{t|\mathcal{F}_{t-1}}[\cdot]$ in (12.11) no longer depend on $\mathcal{F}_{t-1}$, and thus coincide with $\rho_t[\cdot]$ [209]. □

Second, in an optimization context, multi-period risk measures have to be carefully chosen, in such a way that the resulting problem $(P_\mathcal{R})$ possesses desirable properties. In addition to convexity, especially time-consistency is a crucial property.

12.1.3. Time Consistency. In the literature, various different definitions of time consistency exist, see among others [37, 113, 50, 165, 207] and references within. The term is ambiguous in the sense that it is used for risk measures, policies and optimization problems. We only state some of these concepts that are relevant for SDDP, and for technical definitions and detailed discussions refer to [68, 113, 207, 209].

A common definition is that an optimal policy $\big(\bar{\boldsymbol{x}}_t(\xi_{[t]})\big)_{t\in[T]}$ for $(P_\mathcal{R})$ (see (12.3)) is called *time consistent* if for any $\tau \in [T]$, the policy $\big(\bar{\boldsymbol{x}}_t(\xi_{[t]})\big)_{t=\tau,\ldots,T}$ is optimal for $(P_\mathcal{R})$ restricted to horizon $t = \tau, \ldots, T$ conditional on $\mathscr{F}_{\tau-1}$ and $\bar{x}_{\tau-1}$ [209]. This means that the optimal policy remains optimal after some of the uncertain data has been revealed. The problem $(P_\mathcal{R})$ is then called *weakly time consistent*, if at least one of its optimal policies is time consistent, or *time consistent*, if every optimal policy is time consistent [209] (note that there exist deviating definitions in the literature).

Policies obtained using DPE (such as (2.4)-(2.6)) naturally satisfy time consistency. Therefore, the concept of time consistency is closely related to equivalently reformulating $(P_\mathcal{R})$ (see (12.3)) into DPE [209]. For nested risk measures $\mathcal{R}[\cdot]$, see (12.11), this equivalence holds under strict monotonicity (property (R2') in Definition 12.4) of ρ_t (or $\rho_{t|\xi_{[t-1]}}$, respectively) for all $t = 2, \ldots, T$. More precisely, under (R2'), by interchanging risk measures and minimization operators, $(P_\mathcal{R})$ with nested

risk can be expressed in the nested fashion [209]

$$\min_{x_1 \in \mathcal{X}_1} c_1^\top x_1 + \rho_2 \Bigg[\min_{\boldsymbol{x}_2 \in \mathcal{X}_2(x_1)} (\boldsymbol{c}_2(\xi_2))^\top \boldsymbol{x}_2 + \rho_{3|\xi_{[2]}} \Big[\dots \\ \dots + \rho_{T|\xi_{[T-1]}} \Big[\min_{\boldsymbol{x}_T \in \mathcal{X}_T(x_{T-1})} (\boldsymbol{c}_T(\xi_T))^\top \boldsymbol{x}_T \Big] \dots \Big] \Bigg], \tag{12.13}$$

which naturally allows for a reformulation to DPE. Note that for stage 2 no conditional expectation is used as the first-stage data is deterministic. If ρ_t (or $\rho_{t|\xi_{[t]}}$) only satisfy (R2) instead of (R2'), then only weak consistency of $(P_\mathcal{R})$ is guaranteed, as any optimal policy for the DPE is also optimal for problem $(P_\mathcal{R})$ with nested risk, but not necessarily vice versa.

As indicated by Table 4, $\text{AVaR}_\alpha[\cdot]$ is not strictly monotone. Therefore, even if applied in a nested conditional way, time consistency is not assured. In contrast, it can be ensured using risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ defined in (12.7), given that $\lambda \in [0, 1)$. A drawback of nested risk is that it is less amenable to suitable interpretations, although some economic interpretations are possible [198].

For one-period risk measures $\rho[\cdot]$ that are applied as an end-of-horizon risk measure (12.10), it is well known that time consistency is often not satisfied. For instance, some simple examples in [68, 113] show that using a one-period risk measure $\rho[\cdot]$, such as $\text{VaR}_\alpha[\cdot]$ or $\text{AVaR}_\alpha[\cdot]$, in this setting leads to time-inconsistent decisions. Moreover, in [198], an illustrative example is presented in which even under stagewise independence (Assumption 2), the risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ does not yield time-consistent policies from an end-of-horizon perspective. To achieve time consistency, it is required that problem $(P_\mathcal{R})$ (see (12.3)) with end-of-horizon risk measure $\rho[\cdot]$ can be converted to an equivalent problem with nested risk using the corresponding conditional risk measures $\rho_{|\xi[t]}$. For this reason, Dowson et al. [64] define time consistency (in their case referred to as *conditional consistency*) of a one-period risk measure $\rho[\cdot]$ as an equivalence between the associated end-of-horizon risk and nested risk.

In fact, the only law-invariant coherent one-period risk measures $\rho[\cdot]$ allowing for such an equivalent reformulation between an end-of-horizon risk and a nested risk perspective are $\mathbb{E}[\cdot]$ and $\operatorname{ess\,sup}[\cdot]$ [209]. Therefore, the coherent and law-invariant risk measure $\text{AVaR}_\alpha[\cdot]$ does not even guarantee weak time consistency for $(P_\mathcal{R})$ if it is applied as an end-of-horizon risk measure. It can be shown, though, that the non-coherent, but convex risk measure $\mathbb{ENT}_\gamma[\cdot]$ from (12.8) is conditionally consistent, and thus is sufficient to ensure time consistency of $(P_\mathcal{R})$. The equivalence of different formulations for problem $(P_\mathcal{R})$ is illustrated in Figure 11.

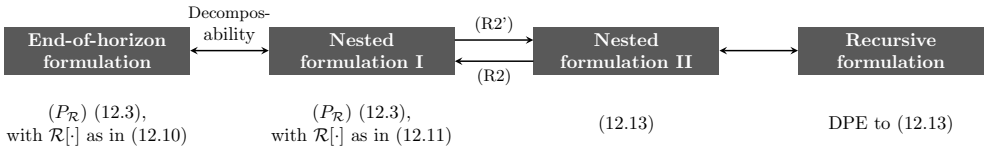


Fig. 11: Different forms of $(P_\mathcal{R})$ and conditions for their equivalence.

Remark 12.7. In view of conditional consistency, note that nested risk measures $\mathcal{R}[\cdot]$ from (12.11) can always be expressed equivalently using an associated end-of-horizon risk measure (12.10), called *composite risk measure*. However, as the previous

discussion shows, this composite risk measure only equals $\rho[\cdot]$ if the latter allows for a decomposition using its conditional analogues; similar to (12.2) [207, 209]. □

Additionally, some notion of time consistency can be satisfied using *expected conditional risk measures* $\mathcal{R}[\cdot]$, which measure the risk stage by stage (see (12.12)), as long as the included (conditional) risk measures are coherent [113]. Applying such a risk measure in $(P_{\mathcal{R}})$ (problem (12.3)), we obtain the problem

$$\begin{aligned} \min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \quad & c_1^\top x_1 + \rho_2\left[\left(\boldsymbol{c}_2(\xi_2)\right)^\top \boldsymbol{x}_2(\xi_{[2]})\right] + \mathbb{E}_{\xi_{[2]}}\left[\rho_{3|\xi_{[2]}}\left[\left(\boldsymbol{c}_3(\xi_3)\right)^\top \boldsymbol{x}_3(\xi_{[3]})\right]\right] \\ & \qquad + \cdots + \mathbb{E}_{\xi_{[T-1]}}\left[\rho_{T|\xi_{[T-1]}}\left[\left(\boldsymbol{c}_T(\xi_T)\right)^\top \boldsymbol{x}_T(\xi_{[T]})\right]\right] \\ \text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\ & \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2, \ldots, T \\ & \boldsymbol{x}_t(\cdot) \ \mathscr{F}_t\text{-measurable} \quad \forall t = 2, \ldots, T. \end{aligned} \tag{12.14}$$

12.1.4. Polyhedral Risk Measures. Multiperiod polyhedral risk measures $\mathcal{R}[\cdot]$ are a special type of risk measure, which for a time horizon of $T \in \mathbb{N}$ can be formulated as the optimal value of certain T-stage stochastic linear programs [71]. The arguments of the risk measure, *e.g.*, in our case the objective function of (MSLP), enter these linear programs on the RHS.

In [100], multiperiod extended polyhedral risk measures are introduced, for which the corresponding linear program has a slightly more general form. This class comprises polyhedral risk measures, spectral risk measures and also $\mathrm{AVaR}_\alpha[\cdot]$. These risk measures can be shown to be convex and coherent under certain assumptions [100].

The main strength of (extended) polyhedral risk measures is that they can naturally be used in a multistage stochastic programming setting. The LP representation of $\mathcal{R}[\cdot]$ and the original LP formulation of (MSLP) can be conflated to a single large-scale risk-neutral linear programming problem $(P_{\mathcal{R}})$, which allows for a reformulation by means of DPE [100].

12.2. Towards Considering Risk in SDDP. In the remainder of this section, we discuss the incorporation of risk-aversion into SDDP from an algorithmic perspective. As a first step, this requires to derive tractable DPE for $(P_{\mathcal{R}})$. Then, SDDP has to be adapted to those DPE, which potentially affects the cut generation, the upper bound computation and the sampling.

The first two methodological studies of risk-averse SDDP are [100] for problems with end-of-horizon risk (12.10), in particular using polyhedral risk measures, and [206] for problems with nested conditional risk mappings (12.11). Since then several extensions of SDDP have been proposed based on various risk measures. While some articles on this topic also cover SAA [121, 206, 212], see Section 11, we restrict to finite random variables here.

Remark 12.8 (SDDP with Polyhedral Risk Measures). As stated in Section 12.1.4, polyhedral risk measures have the advantage that DPE can be derived in a straightforward way. These DPE can then be approached by standard risk-neutral SDDP. Guigues and Römisch derive the associated cut formulas and give a convergence proof for some special cases of extended polyhedral risk measures [100] and the special case of spectral risk measures [101]. This approach to SDDP has been successfully applied for $\mathrm{AVaR}_\alpha[\cdot]$ in [91].

Despite this straightforward approach, polyhedral risk measures also pose a significant challenge to SDDP. The stage-t subproblems have to be enhanced with ad-

ditional state variables z_{t-1} and $y_1, \dots, y_{t-1}$, which are required to store the history of previous decisions. In general, such a state space expansion is unfavorable, as it may lead to prohibitive computational cost [168], see the complexity results in Section 4.2. The specific computational cost depends on the chosen extended polyhedral risk measure. □

12.3. SDDP with Nested Risk Measures. As mentioned in Section 12.1.3, to obtain a risk-averse problem $(P_{\mathcal{R}})$ with time-consistent solutions, it is often proposed to use (conditional) coherent one-period risk measures $\rho[\cdot]$ (or $\rho_{t|\xi_{[t]}}[\cdot]$) for all $t \in [T]$ in a nested fashion. This yields the nested problem (12.13). We denote its optimal value by $v^*_{\mathcal{R}}$. As indicated before, we can derive an equivalent formulation using DPE [209]. Using Remark 12.6 they become

$$Q_{\mathcal{R},t}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \mathscr{Q}_{\mathcal{R},t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \tag{12.15}$$

with some *risk-adjusted value function*

$$\mathscr{Q}_{\mathcal{R},t+1}(x_t) := \rho_{t+1}\left[Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1})\right] \tag{12.16}$$

and $\mathscr{Q}_{\mathcal{R},T+1}(\cdot) \equiv 0$. The corresponding first-stage problem is

$$v^*_{\mathcal{R}} = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathscr{Q}_{\mathcal{R},2}(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{12.17}$$

Fortunately, for coherent risk measures $\rho_t[\cdot], t \in [T]$, also the nested risk measure $\mathcal{R}[\cdot]$ preserves convexity of $\mathscr{Q}_{\mathcal{R},t+1}(\cdot)$. Therefore, a cutting-plane approximation as in SDDP can be applied.

Nested conditional risk measures are by far the most frequently chosen approach for risk-averse extensions of SDDP [68, 105, 113, 121, 167, 168, 206, 212]. Most typically, the risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ (see (12.7)) is used, which is coherent according to Table 4. For the remainder of Section 12.3, we therefore set $p_t[\cdot] = \widehat{\rho}_{\alpha_t,\lambda_t}[\cdot]$ for all $t \in [T]$, if not specified otherwise.

12.3.1. Reformulating the DPE. The general DPE for $(P_{\mathcal{R}})$ with nested risk measures are formulated in (12.15)-(12.17). To determine $\mathscr{Q}_t(\cdot), t \in [T]$, for $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ specifically, the AVaR of $Q_t(\cdot,\cdot)$ has to be evaluated. Using its definition as the optimal value of an optimization problem with decision variable $u \in \mathbb{R}$ [192], see (12.5), we are able to further reformulate the DPE, so that only expectation operators occur.

Additional State Variable Approach. Using (12.5), the risk-adjusted value function (12.16) can be expressed as

$$\begin{aligned} \mathscr{Q}_{\mathcal{R},t+1}(x_t) = \min_{u_t \in \mathbb{R}} \; \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[&(1-\lambda_{t+1}) Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) \\ &+ \lambda_{t+1}\Big(u_t + \frac{1}{\alpha_{t+1}}\big[Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) - u_t\big]_+\Big)\Big]. \end{aligned} \tag{12.18}$$

Recall that λ_t and α_t, $t = 2, \dots, T$, are user-controlled parameters.

The minimization over u_t can be incorporated into the stage-t subproblems [206], which yields

$$\widetilde{Q}_{\mathcal{R},t}(x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t, u_t} & \big(c_t(\xi_t)\big)^\top x_t + \lambda_{t+1} u_t + \widetilde{\mathscr{Q}}_{\mathcal{R},t+1}(x_t, u_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \tag{12.19}$$

with some modified risk-adjusted value function

$$\widetilde{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[(1-\lambda_{t+1})\widetilde{Q}_{\mathcal{R},t+1}(x_t,\boldsymbol{\xi}_{t+1}) + \frac{\lambda_{t+1}}{\alpha_{t+1}}\big[\widetilde{Q}_{\mathcal{R},t+1}(x_t,\boldsymbol{\xi}_{t+1})-u_t\big]_+\Big], \tag{12.20}$$

$\widetilde{\mathscr{Q}}_{\mathcal{R},T+1}(\cdot,\cdot)\equiv 0$ and $\lambda_{T+1}\equiv 0$ [206]. The first stage changes to

$$v^*_{\mathcal{R}} = \begin{cases} \min\limits_{x_1,u_1} & c_1^\top x_1 + \lambda_2 u_1 + \widetilde{\mathscr{Q}}_{\mathcal{R},2}(x_1,u_1) \\ \text{s.t.} & x_1\in\mathcal{X}_1. \end{cases} \tag{12.21}$$

The risk-adjusted value functions $\widetilde{\mathscr{Q}}_{\mathcal{R},t+1}(\cdot,\cdot)$ differ from the ones defined in (12.18), but can be proven to be convex as well.

With equations (12.19)-(12.21), the risk measures $\rho_{\alpha_t,\lambda_t}[\cdot]$ are incorporated into the subproblems, such that only expectations have to be evaluated in the DPE. However, as pointed out in [121], in comparison with the DPE (2.4)-(2.6) of the risk-neutral case, we still observe some fundamental differences: First, an additional, albeit one-dimensional, state variable $u_t\in\mathbb{R}$ is introduced at each stage to estimate the VaR-level, augmenting the state space by one. Second, the risk-adjusted value functions $\mathscr{Q}_{\mathcal{R},t+1}(\cdot,\cdot)$ do not only depend on x_t, but also on u_t and parameters λ_t,α_t. Third, they contain the nonlinear, *i.e.*, piecewise linear, function $[\cdot]_+$.

Philpott and de Matos provide an alternative reformulation of the DPE, eliminating the nonlinear expression via an epigraph reformulation [167]. To this end, the random term in the brackets in (12.20) is fully incorporated into the value functions. For $t=2,\dots,T-1$, this yields

$$\begin{aligned} &\widehat{Q}_{\mathcal{R},t}(x_{t-1},u_{t-1},\xi_t) \\ &= \begin{cases} \min\limits_{x_t,u_t,w_t} & (1-\lambda_t)\Big(\big(c_t(\xi_t)\big)^\top x_t + \lambda_{t+1}u_t + \widehat{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_t)\Big) + \dfrac{\lambda_t}{\alpha_t}w_t \\ \text{s.t.} & x_t\in\mathcal{X}_t(x_{t-1},\xi_t) \\ & w_t - \big(c_t(\xi_t)\big)^\top x_t - \lambda_{t+1}u_t - \widehat{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_t) \geq -u_{t-1}. \end{cases} \end{aligned} \tag{12.22}$$

Using this formulation, the risk value function is defined more naturally as

$$\widehat{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[\widehat{Q}_{\mathcal{R},t+1}(x_t,u_t,\boldsymbol{\xi}_{t+1})\Big]. \tag{12.23}$$

Again, $\widehat{\mathscr{Q}}_{\mathcal{R},T+1}(\cdot,\cdot)\equiv 0$ and $\lambda_{T+1}\equiv 0$.

The first-stage problem reads then

$$v^*_{\mathcal{R}} = \begin{cases} \min\limits_{x_1,u_1,w_1} & c_1^\top x_1 + \lambda_2 u_1 + \widehat{\mathscr{Q}}_{\mathcal{R},2}(x_1,u_1) \\ \text{s.t.} & x_1\in\mathcal{X}_1. \end{cases} \tag{12.24}$$

In comparison to the formulation (12.19)-(12.21) by Shapiro [206], additional variables and constraints have to be introduced. Both formulations allow application of SDDP, but share the drawback of augmenting the state space. Since the computational effort of SDDP grows exponentially in the state space dimension, see Theorem 4.2, such increase should be avoided.

Modifying the Probability Measure. An alternative idea is to exploit that $u^* = \mathrm{VaR}_\alpha[\boldsymbol{Z}]$ in the definition of $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ (see (12.5)) and that $\mathrm{VaR}_\alpha[\boldsymbol{Z}]$ is the $(1-$

α)-quantile of a random variable $\boldsymbol{Z}$. As we assume finite randomness (Assumption 5) and solve the subproblems for all realizations $\xi_{tj}, j = 1, \dots, q_t$, in the backward pass of SDDP, this quantile can be manually determined for the value functions [212].

Without loss of generality, assume that for all $t = 2, \dots, T$ and any fixed trial solution $\bar{x}_{t-1}$ the values of $Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{tj})$ are ordered for all $j = 1, \dots, q_t$. That means, we have $Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{t1}) \leq \cdots \leq Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{t,q_t})$. With this argument, in (12.18) the variable u_t can be replaced by the $(1-\alpha)$-quantile $Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*})$ with j^* chosen such that $\sum_{j=1}^{j^*} p_{t+1,j} \geq 1 - \alpha_{t+1}$:

$$\begin{aligned}\mathscr{Q}_{\mathcal{R},t+1}(x_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[&(1-\lambda_{t+1}) Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) + \lambda_{t+1}\Big(Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*}) \\ &+ \frac{1}{\alpha_t}\big[Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) - Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*})\big]_+\Big)\Big].\end{aligned} \tag{12.25}$$

In SDDP, relation (12.25) cannot directly be applied, since $Q_{\mathcal{R},t+1}(\cdot, \xi_{t+1,j})$ is not known and also not evaluated for all $j = 1, \dots, q_{t+1}$. However, the same principle can also be applied to the approximate value functions $\underline{Q}_{\mathcal{R},t+1}(\cdot, \xi_{t+1,j})$. Because of monotonicity (R2) of ρ_{t+1}, we have $\rho_{t+1}\left[Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1})\right] \geq \rho_{t+1}\left[\underline{Q}_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1})\right]$, so this yields valid lower approximations.

In [168], this idea is considered from a dual perspective and used to reformulate the risk measure (12.7) even before formulating the DPE. The key idea is to use the dual representation of $\mathrm{AVaR}_\alpha[\cdot]$, see (12.9), which is given by

$$\mathrm{AVaR}_\alpha[\boldsymbol{Z}] = \begin{cases} \sup\limits_{\zeta} & \sum\limits_{j=1}^{q} p_j \zeta_j Z(\xi_j) \\ \text{s.t.} & \sum\limits_{j=1}^{q} p_j \zeta_j = 1 \\ & \zeta_j \geq 0, \quad j = 1, \dots, q \\ & \zeta_j \leq \dfrac{1}{\alpha}, \quad j = 1, \dots, q. \end{cases} \tag{12.26}$$

It shows that $\mathrm{AVaR}_\alpha[\cdot]$ can be interpreted as some worst-case probability measure $\widetilde{\mathbb{P}}$ with $\widetilde{p}_j := p_j \zeta_j$ for all $j = 1, \dots, q$.

As shown in [168], using this definition and explicitly computing the supremum, risk measure (12.7) can be written as

$$\widehat{\rho}_{\alpha_t, \lambda_t}[\boldsymbol{Z}_t] = \sum_{j=1}^{q_t} p_{tj} \zeta_{tj} Z_t(\xi_{tj}) \tag{12.27}$$

with

$$\zeta_{tj} = \begin{cases} (1-\lambda_t), & j < j^*, \\ (1-\lambda_t) + \dfrac{1}{p_{tj^*}}\Big(\lambda_t - \dfrac{\lambda_t}{\alpha_t} \displaystyle\sum_{n=j^*+1}^{q_t} p_{tn}\Big), & j = j^*, \\ (1-\lambda_t) + \dfrac{\lambda}{\alpha_t}, & j > j^*. \end{cases} \tag{12.28}$$

Again, note that the true value functions $Q_t(\cdot)$ are not known explicitly in advance, and therefore the worst-case probability measure $\widetilde{\mathbb{P}}$ stemming from (12.26)

is not known either. However, it can be approximated in SDDP. In particular, the DPE (12.15)-(12.17) and their approximations can be used with expectations as in standard SDDP, but with a modified probability measure that is iteratively updated. More precisely, ζ_{tj} changes with $\bar{x}_{t-1}$, so the modified probabilities have to be re-computed for each stage t, iteration i and sample k in SDDP. This principle is also extended to general coherent risk measures in [168].

Recently, this kind of change of the probability measure has also been discussed in [133]. Instead of adapting the ordering and j^* based on $\underline{Q}_{t+1}^{i+1}(\cdot)$ and x_t^i in each iteration i, it is assumed that j^* is stable with respect to different values of x_{t-1}. Its stable value is then approximated by counting the number of iterations in which an index j exceeds $\mathrm{VaR}_\alpha[\underline{Q}_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_t)]$. This is considered as a good proxy for the ordering of the actual value functions. A different approximation strategy is proposed in [80]. As a key ingredient, multi-cut SDDP is used, see Section 21.2.1. An ordering of the most current values $\theta_{t+1,j}^i$ obtained for each cut approximation $\underline{\mathcal{V}}_{tj}^i(\cdot), j = 1, \ldots, q_t$, at stage t is then used as a proxy for the risk-adapted probability measure $\widetilde{\mathbb{P}}$.

The ordering, and thus the probability measure $\widetilde{\mathbb{P}}$, can either be updated dynamically within SDDP or be determined by running risk-averse SDDP once in advance to identify the outcomes contributing to $\mathrm{AVaR}_\alpha[\cdot]$. The latter approach has the advantage that the changed probability measure $\widetilde{\mathbb{P}}$ can be fixed for the following run, which yields a *risk-neutral* problem and allows for application of standard SDDP.

For the third-stage of Example 3.4, the expected risk value function $\mathscr{Q}_{\mathcal{R},3}(\cdot)$ obtained by applying (12.27) and (12.28) to (12.16) is illustrated in Figure 12 for $\alpha = 0.05$ and different values of λ. It can be seen that with choosing larger values for λ, representing a higher risk-aversion, the stage-3 cost increases compared to the risk-neutral case ($\lambda = 0$).

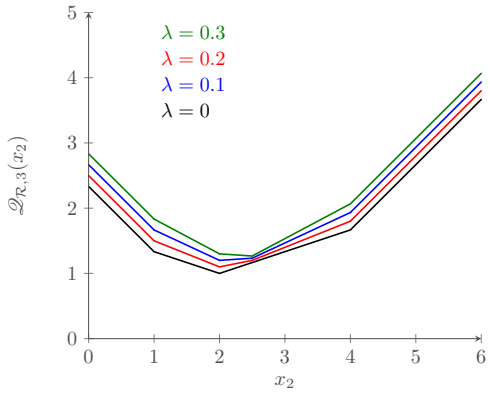


Fig. 12: $\mathscr{Q}_{\mathcal{R},3}(\cdot)$ from Example 3.4 for $\alpha = 0.05$ and different values of λ.

As an overview, the different forms of DPE for $(P_\mathcal{R})$ using a nested (conditional) risk measure based on $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ are summarized in Table 5. All approaches in Table 5 to formulate the DPE allow for a solution of a risk-averse problem $(P_\mathcal{R})$ using SDDP. Some approaches are more efficient, since the state space, the decision space or the number of constraints are not augmented. Others are advantageous in the sense that $\mathscr{Q}_{\mathcal{R},t}(\cdot)$ is expressed by a neat formula, and thus cut formulas can be derived more easily. With some epigraph reformulation, for all the approaches all subproblems can

Description	Source	DPE
- general		(12.15)-(12.17)
- augmented state, sophisticated formula for $\mathscr{Q}_{\mathcal{R},t}(\cdot)$	[206]	(12.19)-(12.21)
- augmented state, additional constraints and variables	[167]	(12.22)-(12.24)
- $\mathrm{VaR}_{\alpha_t}[Q_t(\cdot)]$ explicitly determined, sophisticated formula for $\mathscr{Q}_{\mathcal{R},t}(\cdot)$	[212]	(12.19), (12.21), (12.25)
- modified probability measure	[168]	(12.15)-(12.17), (12.27)-(12.28)
- modified probability measure	[133]	(12.19), (12.21), (12.25)

Table 5: DPE formulations for $(P_{\mathcal{R}})$ using a nested (conditional) risk measure based on $\widehat{\rho}_{\alpha,\lambda}[\cdot]$.

be formulated as LPs.

12.3.2. Forward and Backward Pass. The forward pass of SDDP basically remains the same as for risk-neutral SDDP from Section 3, see Algorithm 3.1. That is, $k \in \mathcal{K}$ scenarios are sampled and considered, with $\mathcal{K} \subset \mathcal{S}$ and $|\mathcal{K}| \ll |\mathcal{S}|$. However, the subproblems and the associated approximate value functions $\underline{Q}^i_{\mathcal{R},t}(x^{ik}_{t-1}, \xi^k_t)$ differ from the risk-neutral case. Instead of subproblems (2.10), one of the DPE from Table 5 are chosen and the occurring risk-adjusted value functions $\mathscr{Q}_{\mathcal{R},t+1}(\cdot)$ are replaced by cut approximations $\underline{\mathcal{V}}^i_{\mathcal{R},t+1}(\cdot)$. The sampling technique in line 6 of Algorithm 3.1 usually remains the same as in risk-neutral SDDP, meaning that random sampling with respect to the distribution of $\boldsymbol{\xi}$ is used. However, it is also possible to sample scenarios with "bad" outcomes with higher probability based on proxies of the probability measure $\widetilde{\mathbb{P}}$ [80, 133]. This *biased sampling* can be considered similar to the importance sampling techniques presented in Section 6.

In the backward pass, as in risk-neutral SDDP, at each stage $t = T, \ldots, 2$, the subproblems are solved for each trial solution $x^{ik}_{t-1}, k \in \mathcal{K}$, and possible stage-$t$ realization $\xi^k_{tj} \equiv \xi_{tj}, j = 1, \ldots, q_t$, using an updated cut approximation $\underline{\mathcal{V}}^{i+1}_{\mathcal{R},t+1}(\cdot)$. On stage t, a new cut for $\mathscr{Q}_{\mathcal{R},t}(\cdot)$ is derived and handed back to stage $t-1$. The main difference to risk-neutral SDDP is again the definition of $\mathscr{Q}_{\mathcal{R},t}(\cdot)$. Therefore, the cut formulas used in line 18 of Algorithm 3.1 have to be adapted to the individual approach chosen. For the technical derivation of subgradients in such cases, we refer to the references in Table 5.

12.3.3. Upper Bound Determination and Stopping. A challenge in applying SDDP to risk-averse problems is to determine upper bounds for $v^*_{\mathcal{R}}$, and allowing for a reasonable stopping criterion. The reason is that most upper bound construction methods from the risk-neutral case, see Sections 7 and 8, cannot be efficiently extended to the risk-averse case.

Recall that in the risk-neutral case, a feasible policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ is determined in the backward pass and evaluated in the forward pass for different scenarios $k \in \mathcal{K}$, yielding a sequence of trial points $(x^{ik}_t)_{t\in[T]}$. Then, a statistical upper bound $\overline{v}_{\mathcal{K}}$ for v^* is determined as the sample average of the objective values of all these sample paths ξ^k, see (3.9). Analogously, a true upper bound $\overline{v}$ can be obtained by taking the expectation of such objective value for all scenarios $\xi^s, s \in \mathcal{S}$.

However, this is possible only due to the tower property (12.2) of expected values,

which is required for the equivalence of the end-of-horizon formulation (12.1) and the nested formulation (12.13), see the discussion in Section 12.1.3. For most coherent risk measures this property does not hold, and thus a *direct analogue* to the (statistical) upper bound (3.9) from risk-neutral SDDP cannot be constructed.

As determining reasonable upper bounds is an important ingredient of SDDP, developing appropriate upper bound estimators has been an active research field in the last decade. In the following, we discuss different approaches that have been proposed. In reviewing them, we mostly follow the presentation of Kozmík and Morton [121], who provide a comprehensive study within their own work on upper bound estimators.

A Sample Average Estimator. In Section 12.3.1, we managed to formulate each $\rho_t[\cdot]$ only by means of expectations in (12.20). Still, this does not assure the tower property, since the risk-adjusted value functions $\mathscr{Q}_{\mathcal{R},t}(\cdot)$ contain a nested nonlinearity due to the $[\cdot]_+$-function. However, we can derive an estimator similar to (3.9) [121]. To this end, we remove the expectation in (12.20) to obtain

$$\begin{aligned}\hat{v}_t(\xi_t^k) := (1-\lambda_t)\Big(\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}(\xi_t^k)\Big) \\ + \lambda_t u_{t-1}^k + \frac{\lambda_t}{\alpha_t}\Big[\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}(\xi_t^k) - u_{t-1}^k\Big]_+,\end{aligned} \tag{12.29}$$

where we replace the value functions $Q_{\mathcal{R},t+1}(\cdot)$ by the estimator of the following stage. For stage T it follows $\hat{v}_{T+1}(\xi_T^k) \equiv 0$ and for the first stage

$$\hat{v}(\xi^k) := c_1^\top x_1 + \hat{v}_2(\xi_1^k). \tag{12.30}$$

Equation (12.30) provides a recursive estimator for the cost associated with sample path ξ^k. This estimator has to be evaluated by backward recursion starting with stage T. Importantly, formula (12.29) is only used for upper bound estimation, whereas the forward and backward problems in SDDP are still based on the original DPE (12.19)-(12.21). Determining estimator (12.30) for all scenarios $\xi^k, k \in \mathcal{K}$, sampled in the forward pass of SDDP, we can form an upper bound estimator

$$U^n := \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \hat{v}(\xi^k), \tag{12.31}$$

which resembles the sample average estimator (3.9) from risk-neutral SDDP.

It can be shown that $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)] \geq v_{\mathcal{R}}^*$ and that U^n is an unbiased and consistent estimator of $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)]$, so it is a statistical upper bound [121]. However, U^n is also observed to have a large variance. Kozmík and Morton [121] identify as the main reason that only a small portion of the sampled scenarios contributes to estimating $\mathrm{AVaR}_\alpha[\cdot]$, while most scenarios solely contribute to the expectation. Therefore, a very large number of scenarios would be required for an appropriate estimate.

More crucially, because expectations are not taken conditionally on each stage as in (12.20), and due to to division by $\alpha_t \in (0,1)$, small or large values are very likely to propagate from late to earlier stages in the recursion to determine $\hat{v}(\xi^k)$ [121]. Therefore, the upper bound $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)]$ can significantly deviate from $\overline{v}_{\mathcal{R}}$, *i.e.*, the upper bound induced by the current policy in SDDP. In computational experiments, an upward bias is observed that makes U^n practically useless for large T [210].

Remark 12.9. We should note that recently a very similar recursive upper bound estimator to $\hat{v}(\xi^k)$ and U^n has been proposed in [105], but for a general class of risk measures instead of only $\widehat{\rho}_{\alpha,\lambda}[\cdot]$. The main difference is that there SDDP is applied to

a risk-averse stochastic optimal control model which deviates from our setting introduced in Section 2. In particular, states and controls are explicitly distinguished, and the decision on the controls is taken *before* $\boldsymbol{\xi}_t$ is realized. In this setting, the negative multiplicative effects observed in [121, 210] can be circumvented, and a computationally efficient statistical upper bound is obtained. □

Conditional Sampling Estimator. For the above reasons, estimator U^n in (12.31) is rarely considered in the literature on risk-averse SDDP. Instead, Shapiro discusses a conditional sampling estimator [206]. Here, the idea is to estimate the expectations (12.20) in the nested structure conditionally by sampling on each stage. Since in principle, the upper bound estimator can be determined independently of the scenarios sampled in the forward pass, we denote the set of samples by $\mathcal{M}$ instead of $\mathcal{K}$. $\mathcal{M}_t$ denotes the corresponding scenario set for stage t.

For each stage, $t = 2, \dots, T$, this yields [121]

$$\hat{v}_t^c(\xi_t^k) := \frac{1}{|\mathcal{M}_t|} \sum_{m_t \in \mathcal{M}_t} \Bigg[(1-\lambda_t)\Big(\big(c_t(\xi_t^{m_t})\big)^\top x_t^{m_t} + \hat{v}_{t+1}^c(\xi_t^{m_t}) \Big)$$

$$+ \lambda_t u_{t-1}^{m_t} + \frac{\lambda_t}{\alpha_t} \Big[\big(c_t(\xi_t^{m_t})\big)^\top x_t^{m_t} + \hat{v}_{t+1}^c(\xi_t^{m_t}) - u_{t-1}^{m_t} \Big]_+ \Bigg],$$

and for the first stage the estimator

$$U^c := c_1^\top x_1 + \hat{v}_2^c(\xi_1).$$

As Shapiro himself points out, this estimator has two significant drawbacks. It requires $\prod_{t=2}^{T} |\mathcal{M}_t| + 1$ subproblems to be solved, which is exponentially growing in the number of stages. Moreover, the obtained upper bounds are typically not very tight. Therefore, estimator U^c should not be useful for large-scale problems [121].

Importance Sampling Estimators. Some of the drawbacks of estimator U^n may also be addressed by importance sampling [120, 121], see Section 6 for an introduction. By sampling scenarios associated with $\mathrm{AVaR}_\alpha[\cdot]$ with higher importance, it is possible to better represent it, and thus reduce the variance of the estimator. Based on this idea, Kozmík and Morton put forward different importance sampling upper bound estimators [121], which are further enhanced in [120].

Using importance sampling with respect to $\mathrm{AVaR}_\alpha[\cdot]$ creates a considerable challenge, though. In order to determine the importance sampling distribution for some stage t, it has to be identified which scenarios are associated with $\mathrm{AVaR}_\alpha[\cdot]$ on that stage, *i.e.*, which of them provide a value $Q_{\mathcal{R},t}(x_{t-1}^k, \xi_{tj}^k)$ beyond the $(1-\alpha)$-quantile. If we estimate this by solving subproblems for several ξ_{tj}^k and determining $Q_{\mathcal{R},t}(x_{t-1}^k, \xi_{tj}^k)$, we face a similar computational burden as for conditional sampling.

Kozmík and Morton propose the following approach: They use an *approximation function* $d_t(x_{t-1}, \xi_t)$, which estimates the recourse value of the decisions x_{t-1} after ξ_t has been observed [121]. Instead of solving the subproblems for several ξ_{tj}^k, they simply evaluate $d_t(x_{t-1}^k, \xi_{tj}^k)$ and sort these values. Based on the obtained order, it can be decided then which scenarios are used to estimate $\mathrm{AVaR}_\alpha[\cdot]$, *i.e.*, $u_t := \mathrm{VaR}_{\alpha_t}[d_t(x_{t-1}, \boldsymbol{\xi}_t)]$ is determined.

This allows defining an importance sampling distribution depending on x_{t-1} [121]. For simplicity, we assume that all scenarios are equally likely in the original stage-t distribution F_t of $\boldsymbol{\xi}_t$, that is, the corresponding probability density function $f_t(\cdot)$

satisfies $f_t(\xi_{tj}) = \frac{1}{q_t}$ for all $j = 1, \ldots, q_t$. Then, it follows:

$$g_t(\boldsymbol{\xi}_t|x_{t-1}) := \begin{cases} \dfrac{1}{2\lfloor \alpha_t q_t \rfloor}, & d_t(x_{t-1}, \boldsymbol{\xi}_t) \geq u_t, \\ \dfrac{1}{2(q_t - \lfloor \alpha_t q_t \rfloor)}, & d_t(x_{t-1}, \boldsymbol{\xi}_t) < u_t. \end{cases}$$

This distribution ensures that it is equally likely to draw sample observations above and below u_t. Note that the formula presented in [121] looks a bit different, since it is presented in the context of SAA.

Defining weights

$$\Lambda_t(\boldsymbol{\xi}_t|x_{t-1}) := \frac{f_t(\boldsymbol{\xi}_t)}{g_t(\boldsymbol{\xi}_t|x_{t-1})}$$

and multiplying them along the sample paths

$$\Lambda(\xi^k) := \prod_{t=2}^{T} \Lambda_t(\xi_t^k|x_{t-1})$$

we can derive the estimator

$$U^i := \frac{1}{\sum_{k\in\mathcal{K}} \Lambda(\xi^k)} \sum_{k\in\mathcal{K}} \Lambda(\xi^k)\hat{v}(\xi^k). \tag{12.32}$$

This estimator is similar to (12.31), as the same recursive term $\hat{v}(\xi^k)$ is used, but combined with importance instead of standard MC sampling.

With the assumptions of relatively complete recourse (based on Assumption 9) and stagewise independence (Assumption 2), estimator (12.32) is asymptotically valid, *i.e.*, for $|\mathcal{K}| \to \infty$, U^i converges to $\mathbb{E}_f[\hat{\boldsymbol{v}}(\xi)]$ with probability 1 (recall that $\mathbb{E}_f[\hat{\boldsymbol{v}}(\xi)] \geq v^*_{\mathcal{R}}$). Moreover, for sufficiently good choice of $d_t(\cdot)$, it can be assumed that the variance is lower than for U^n [121].

Based on this idea, even better estimators are developed in [120, 121], for example by not only sampling scenarios associated with $\text{AVaR}_\alpha[\cdot]$ with higher priority, but also using only scenarios which contribute to the $[\cdot]_+$-term to estimate AVaR [121]:

$$\begin{aligned} \hat{v}_t^d(\xi_t^k) := (1-\lambda_t)\Big(\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}^d(\xi_t^k)\Big) \\ + \lambda_t u_{t-1}^k + \mathcal{I}[d_t(x_{t-1}, \xi_t) \geq u_d]\frac{\lambda_t}{\alpha_{t-1}}\Big[\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}^d(\xi_t^k) - u_{t-1}^k\Big]_+. \end{aligned}$$

Here $\mathcal{I}[\cdot]$ denotes an indicator function. For the first stage it follows

$$\hat{v}^d(\xi^k) := c_1^\top x_1 + \hat{v}_2^d(\xi_1^k).$$

Combining this with (11.6), we obtain

$$U^d := \frac{1}{\sum_{k\in\mathcal{K}} \Lambda(\xi^k)} \sum_{k\in\mathcal{K}} \Lambda(\xi^k)\hat{v}^d(\xi^k).$$

The practical applicability of this estimator relies heavily on satisfaction of the following goodness assumption with respect to $d_t(\cdot)$:

$$Q_{\mathcal{R},t}(x_{t-1}, \xi_t) \geq \text{VaR}_{\alpha_t}[Q_{\mathcal{R},t}(x_{t-1}, \xi_t)] \;\Leftrightarrow\; d_t(x_{t-1}, \xi_t) \geq \text{VaR}_{\alpha_t}[d_t(x_{t-1}, \xi_t)],$$

which means that $d_t(\cdot)$ correctly classifies whether a realization is in the upper α-tail of the recourse value distribution.

It is proven that this estimator is asymptotically valid as well, but also provides tighter upper bounds than U^i in expectation, as long as the above goodness assumption is satisfied. Moreover, a smaller variance should be expected [121]. Numerical results in [121] illustrate that even for a medium number of stages, estimator U^d provides significantly better upper bounds than U^n, U^c and U^i and that also the variance of the estimators is reduced significantly. However, despite reducing the variance, even U^i and U^d may still show a considerable upward bias with respect to the upper bound $\overline{v}_{\mathcal{R}}$ induced by the current policy [210].

Apart from the above sampling estimators, some completely different strategies may be used to obtain upper bounds for $v^*_{\mathcal{R}}$ or to define some stopping criteria for SDDP in the risk-averse case.

Using Deterministic Upper Bounds. As already discussed in Section 8, we may circumvent the determination of sampling-based upper bound estimators completely if we resort to deterministic upper bounding procedures.

To this end, Philpott et al. [168] extend their inner approximation based upper bounding procedure from Section 8 to the risk-averse case with nested (conditional) coherent risk measures. The main downside of this procedure, to require prohibitively large computational effort for a large number of state variables and an increasing number of cuts, also holds in this case, though.

The alternative deterministic upper bounding procedure based on dual SDDP [104, 126] has been extended to a risk-averse setting as well [43, 44].

Using a Change of Probability Measure. As discussed in Section 12.3.1, following the approach of a change of probability measure, see (12.15)-(12.17) and (12.27), it is also possible to run (risk-averse) SDDP once in advance to approximate the probability measure $\widetilde{\mathbb{P}}$, and then a second time, this time fixing the probability measure to the approximation of $\widetilde{\mathbb{P}}$. This is referred to as solving the *change-of-measure risk-neutral problem* in [133]. Whereas this approach has a lot of computational overhead, the advantage is that a risk-neutral problem can be solved by SDDP and therefore, also the standard stopping, upper bounding and policy assessment techniques can be applied. Solving the change-of-measure risk-neutral problem is not guaranteed to yield optimal policies for $(P_{\mathcal{R}})$, however, Liu and Shapiro [133] report that the quality of the policies is similar to those obtained by risk-averse SDDP.

A similar approach is used in [80], but without running SDDP twice. Instead, the biased sampling strategy from Section 12.3.2 is applied based on approximating $\widetilde{\mathbb{P}}$ with information gained from using multi-cut SDDP. Again, the idea is that by changing the probability measure when sampling, statistical upper bounds can be computed in a standard fashion. Whereas this approach comes with almost no computational overhead (except for relying on multi-cut SDDP, see Section 21.2.1), the theoretical properties of the obtained estimators are not explored.

Fixing the Number of Iterations. This approach is proposed by Philpott and de Matos [167]. They run a risk-neutral variant of SDDP first and then fix the number of iterations required until termination. The same number of iterations is then used in the risk-averse case, avoiding the challenge of upper bound evaluation.

In some practical applications, in which it is computationally intractable to determine a sophisticated upper bound estimator, this approach may be useful. Promising results are reported in [167]. However, there is no theoretical guarantee to find a sufficiently good solution for a risk-averse version of $(P_{\mathcal{R}})$ in the same number of

iterations as for a risk-neutral version. Additionally, for large problems it may already take considerably long to run SDDP one time. Running it a second time for risk-averse problem ($P_{\mathcal{R}}$) may partially annihilate the computational advantage of avoiding upper bound estimation.

Lower Bound Stabilization. As for risk-neutral SDDP, instead of using upper bounds at all, the algorithm can be terminated, once the lower bounds $\underline{v}^i_{\mathcal{R}}$ stabilize. This provides no convergence guarantee but may be worthwhile in large-scale practical applications where other approaches become computationally prohibitive.

Using Benefit Factors. Instead of the lower bounds $\underline{v}^i_{\mathcal{R}}$, it is also possible to condition termination of SDDP on the improvements of the cut approximations $\underline{\mathcal{V}}^i_{\mathcal{R},t}(\cdot), t = 2, \ldots, T$. For that purpose, Brandi et al. define a benefit factor

$$\mathcal{B}^i_{t,k} = \min\left\{1, \frac{\delta(x^{ik}_{t-1})}{\delta^i_{t,\max}}\right\},$$

which determines how much a new cut improves the current cut approximation $\underline{\mathcal{V}}^i_{\mathcal{R},t}(\cdot)$ at x^{ik}_{t-1} [32]. $\delta(x^{ik}_{t-1})$ is the absolute increase, while $\delta^i_{t,\max}$ is a proxy for the maximum improvement possible. For each sample path $k \in \mathcal{K}$, a total benefit factor can be determined by

$$\mathcal{B}^i_k = \max\left\{\mathcal{B}^i_{2,k}, \mathcal{B}^i_{3,k}, \ldots, \mathcal{B}^i_{T,k}\right\}.$$

The risk-averse SDDP method is then stopped if the values $\mathcal{B}^i_k$ for all $k \in \mathcal{K}$ are below a predefined tolerance, either for one iteration or, alternatively and more robustly, for a predefined larger number of iterations.

12.4. SDDP with Entropic Risk Measure. As discussed before, nested risk measures come with some drawbacks. Computation-wise, upper bound determination is very challenging. Additionally, applying a standard one-period risk measure $\rho[\cdot]$, *e.g.*, $\text{AVaR}_\alpha[\cdot]$, as an end-of-horizon risk measure (12.10) and (possibly conditionally) in a nested risk measure (12.11) does not yield equivalent policies [64] (this is only the case if we take the composite risk measure associated with the nested risk measure as end-of-horizon risk; however, this risk measure is usually not known explicitly, see Remark 12.7). This makes nested risk measures difficult to interpret from an end-of-horizon perspective.

For this reason, Dowson et al. [64] propose to apply one-period *conditionally consistent* risk measures in the context of SDDP [64], see also [11, 165]. It can be proven that under some technical assumptions, the class of entropic risk measures $\mathbb{ENT}_\gamma[\cdot]$ (see (12.8)) is the only class of risk measures that is conditionally consistent.

As $\mathbb{ENT}_\gamma[\cdot]$ can be applied in a nested fashion, the DPE (12.15)-(12.17) are valid in this case. Moreover, since $\mathbb{ENT}_\gamma[\cdot]$ is a convex risk measure, the (risk-adjusted) value functions are convex. Therefore, SDDP can be applied to derive polyhedral outer approximations.

As for standard SDDP, first, for each scenario $k \in \mathcal{K}$ and all possible stage-t realizations $\xi^k_{tj} \equiv \xi_{tj}, j = 1, \ldots, q_t$, approximate versions of subproblems (12.15) are solved to obtain $\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1}, \xi_{tj})$. Then, based on the dual form of $\mathbb{ENT}_\gamma[\cdot]$, the following auxiliary problem can be solved to evaluate the risk-adjusted value function:

$$\mathbb{ENT}_\gamma\big[\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1},\xi_t)\big]$$

$$= \begin{cases} \max\limits_{\tilde{p}_t} & \sum\limits_{j=1}^{q_t} \tilde{p}_{tj}\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1},\xi_{tj}) - \dfrac{1}{\gamma_t}\sum\limits_{j=1}^{q_t}\tilde{p}_{tj}\cdot\log\left(\dfrac{\tilde{p}_{tj}}{p_{tj}}\right) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t}\tilde{p}_{tj} = 1 \\ & \tilde{p}_{tj}\geq 0, \quad j=1,\ldots,q_t. \end{cases} \tag{12.33}$$

Here, parameter p_{tj} denotes the nominal probabilities of realizations ξ_{tj}, which usually equal $\frac{1}{q_t}$, and the decision variable $\tilde{p}_{tj}$ denotes an alternative probability based on the entropic risk measure. In this way, problem (12.33) can be regarded as building the expectation based on some modified probability measure and with some additional penalty term. Problem (12.33) can be solved algorithmically, but as stated in [64], also a closed form for $\tilde{p}^*_{tj}$ can be derived. Using $\tilde{p}^*_{tj}$ and $\mathbb{ENT}_\gamma\big[\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1},\xi_t)\big]$, cuts can then be constructed and handed back to the previous stage.

The entropic risk measure does not only ensure conditional consistency of the obtained policies, but it also allows for upper bound computation as in standard SDDP, because the tower property can be employed for $\mathbb{ENT}[\cdot]$. However, these advantages come at the cost of an aggravated interpretation of the risk measure compared to AVaR-based ones. In this context, it is particularly difficult to make a reasonable choice for the parameter $\gamma_t > 0$ [64].

12.5. SDDP with Expected Conditional AVaR. Another class of multi-period risk measures that can be used as an alternative to nested risk measures are *expected conditional risk measures*, which we briefly introduced in Section 12.1.3 [68, 113]. Here, conditional expectations are used to avoid the risk measure nesting, which proves beneficial in determining upper bounds in SDDP, as it avoids the aforementioned computational difficulties, while still time consistency is ensured.

Recall the risk-averse problem $(P_\mathcal{R})$ using expected conditional risk measures stated in (12.14). Using $\rho_t[\cdot] = \mathrm{AVaR}_{\alpha_t}[\cdot]$ yields the so called $\mathbb{E}$-AVaR or *multi-period average value-at-risk* [113], which goes back to Pflug and Ruszczyński [166].

As stated in [113], by some lengthy reformulations, the objective function of problem (12.14) can be expressed in a nested way. Therefore, equivalent DPE can be derived and time consistency is assured. Moreover, the $[\cdot]_+$-function can be reformulated by an epigraph approach. Then, for $t = 2,\ldots,T$, the DPE read

$$\check{Q}_{\mathcal{R},t}(x_{t-1},u_t,\xi_t) = \begin{cases} \min\limits_{x_t,u_{t+1},w_t} & \dfrac{1}{\alpha_t}w_t + u_{t+1} + \check{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_{t+1}) \\ \text{s.t.} & x_t\in\mathcal{X}_t(x_{t-1},\xi_t) \\ & w_t - \big(c_t(\xi_t)\big)^\top x_t \geq -u_t \\ & w_t \geq 0 \end{cases} \tag{12.34}$$

with

$$\check{\mathscr{Q}}_{\mathcal{R},t+1}(x_t,u_{t+1}) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[\check{Q}_{\mathcal{R},t}(x_{t-1},u_t,\boldsymbol{\xi}_t)\Big], \tag{12.35}$$

$\check{\mathscr{Q}}_{\mathcal{R},T+1}(\cdot,\cdot)\equiv 0$ and first stage

$$v^*_\mathcal{R} = \begin{cases} \min\limits_{x_1,u_2} & c_1^\top x_1 + u_2 + \check{Q}_{\mathcal{R},2}(x_1,u_2,\xi_t) \\ \text{s.t.} & x_1\in\mathcal{X}_1. \end{cases} \tag{12.36}$$

In contrast to using nested conditional risk measures, the DPE here only depend on nested sums of (conditional) expectations, *i.e.*, have the same structure as in the risk-neutral case. Hence, standard SDDP can be applied. This has the advantage to allow one to use upper bounding techniques developed for risk-neutral SDDP.

12.6. Bi-objective SDDP. An alternative to risk-averse formulations that allows one to achieve a trade-off between obtaining the best policy in expectation (*e.g.*, the policy with the lowest expected costs) and avoiding bad extreme outcomes (*e.g.*, power outages or load shedding in an electricity network) is to formulate a multistage problem (MSLP) with multiple competing objectives that are optimized simultaneously. Recently, a variant of SDDP for bi-objective problems has been put forward by Dowson et al. [62].

Let $\widetilde{c}_t(\xi_t)$ and $\widehat{c}_t(\xi_t)$ denote the objective coefficients for stage $t \in [T]$ and the two competing objectives. For all but trivial cases, there exists no policy which yields the best objective value with respect to both objectives

$$\widetilde{v}^* := \min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \underbrace{\mathbb{E}\left[\sum_{t \in [T]} \big(\widetilde{\boldsymbol{c}}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]}) \right]}_{=:\widetilde{v}(\boldsymbol{x})}$$

and

$$\widehat{v}^* := \min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \underbrace{\mathbb{E}\left[\sum_{t \in [T]} \big(\widehat{\boldsymbol{c}}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]}) \right]}_{=:\widehat{v}(\boldsymbol{x})},$$

meaning that the two objectives are truly conflicting.

For this reason, if there is no clear preference for one of the objectives, usually the aim is to compute *Pareto-optimal* policies. A policy $\big(\bar{\boldsymbol{x}}_t(\xi_{[t]})\big)_{t \in [T]}$ is Pareto-optimal if it cannot be improved in one objective without getting worse in the other one, *i.e.*, if there exists no other policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t \in [T]}$ such that $\widetilde{v}(\boldsymbol{x}) \geq \widetilde{v}(\bar{\boldsymbol{x}})$ and $\widehat{v}(\boldsymbol{x}) > \widehat{v}(\bar{\boldsymbol{x}})$ (or the other way around). Pareto-optimal solutions are also called *non-dominated*, and the set of non-dominated objective vectors is called the *Pareto front* [62].

A standard approach to compute Pareto-optimal solutions in optimization is to use some *scalarization* approach in which both conflicting objectives are combined to a weighted sum, which is then optimized in a deterministic single-objective problem. In our case, the DPE (2.4)-(2.6) can be adapted to

$$Q_t(x_{t-1}, \xi_t, \lambda) := \begin{cases} \min\limits_{x_t} & \big(\lambda \widetilde{c}_t(\xi_t) + (1-\lambda)\widehat{c}_t(\xi_t)\big)^\top x_t + \mathscr{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \tag{12.37}$$

where

$$\mathscr{Q}_{t+1}(x_t, \lambda) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1}, \lambda) \right] \tag{12.38}$$

and $\mathscr{Q}_{T+1}(x_T) \equiv 0$. For the first stage, we obtain

$$v^*(\lambda) = \begin{cases} \min\limits_{x_1} & \big(\lambda \widetilde{c}_1 + (1-\lambda)\widehat{c}_1\big)^\top x_1 + Q_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{12.39}$$

SDDP can then be applied to these DPE. In the proposed variant, λ is adapted dynamically. To this end, in each iteration i, after the backward pass, one stage $t \in [T]$

is randomly and independently sampled and the corresponding subproblem is solved again for x_{t-1}^k, ξ_t^k and λ^i. Then, λ^i is updated to λ^{i+1}, where the latter is determined as the closest λ to λ^i such that the optimal basis of the constraint equation system changes.

It is proven that this variant of SDDP converges almost surely to the Pareto front of bi-objective (MSLP) in finitely many iterations. Note that technically speaking not *all* Pareto-optimal policies are guaranteed to be identified by SDDP because for some λ multiple optimal policies may exist. However, all Pareto-optimal policies for which $(\widehat{v}(\boldsymbol{x}), \widetilde{v}(\boldsymbol{x}))$ cannot be represented as a strict convex combination of other non-dominated objective vectors are identified under weak assumptions [62].

13. SDDP with Unknown Distribution [relaxing Assumption 3]. In Section 3 we introduced SDDP assuming that the probability distribution $F_{\boldsymbol{\xi}}$ of the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ governing the uncertainty in problem (MSLP) is known, see Assumption 3. This allowed us to sample from this specific distribution in the forward pass of SDDP or, in case of continuous random vectors, to obtain a finite sample average approximation, as described in Section 11.

In practical applications, usually, the true distribution $F_{\boldsymbol{\xi}}$ is not known, though. Often, only historical data is available, *i.e.*, some realization of an unknown true distribution. This data is then used to determine a reasonable estimate for the true distribution, from which the required samples are taken. However, using such an estimation imposes the risk of *overfitting* the SDDP policies to this specific distribution, and thus the available data. Philpott et al. [169] identify this problem as particularly noteworthy if the number of possible outcomes q_t per stage is small. For this reason, it may be reasonable to take a more robust approach and factor in the distributional uncertainty. Considering this type of uncertainty in SDDP is a young research area.

13.1. Distributionally Robust SDDP. One way to consider distributional uncertainty in SDDP is by integrating ideas from robust optimization [16, 20] into (multistage) stochastic programming. More precisely, a set of potential distributions is considered, which is called *distributional uncertainty set* or *ambiguity set* and denoted by $\mathcal{P}$. The expected cost is then minimized over the worst-case probability distribution from this set. This is called *Distributionally Robust Optimization* (DRO).

Usually, the outcomes of the random variables $\boldsymbol{\xi}_t$ are fixed to a finite number of realizations observed in the historical data. The ambiguity set $\mathcal{P}_t$ then models a variety of potential probability measures $\mathbb{P}_t \in \mathcal{P}_t$ supported on this finite set Ξ_t.

In the following, we restrict to DRO specifically in the SDDP context. For a general introduction to DRO, we refer to the review [183] and the tutorial [207]. We assume all assumptions from Section 3 to hold, except for Assumption 3. Furthermore, we only consider uncertainty in the RHS.

Then, the distributionally robust version of (MSLP) can be written as

$$
\begin{aligned}
\min_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} \max_{\mathbb{P}\in\mathcal{P}} \quad & \mathbb{E}\left[\sum_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right] \\
\text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\
& \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2, \ldots, T.
\end{aligned} \tag{13.1}
$$

Remark 13.1. Distributionally robust stochastic programming is closely related to risk-averse stochastic programming. In particular, the operator $\max_{\mathbb{P}\in\mathcal{P}} \mathbb{E}[\cdot]$ can be interpreted as a multi-period risk measure $\mathcal{R}[\cdot]$. This risk measure is coherent [207].

□

For SDDP it is required to reformulate problem (13.1) by means of DPE. This requires that each distribution $\mathbb{P}$ in the ambiguity set $\mathcal{P}$ can be expressed as the cross product of the respective marginal distributions of random vectors $\boldsymbol{\xi}_t$ [207]. Formally,

$$\mathcal{P} := \left\{ \mathbb{P} = \mathbb{P}_1 \times \ldots \times \mathbb{P}_T \ : \ \mathbb{P}_t \in \mathcal{P}_t, t \in [T] \right\}.$$

The ambiguity sets $\mathbb{P}_t$ are assumed to be independent of each other. This property is called *rectangularity* of $\mathcal{P}$ and is reminiscent of the stagewise independence assumption for vectors $\boldsymbol{\xi}_t$. Note that $\mathcal{P}_1$ is a singleton containing one distribution with one possible realization.

With the ambiguity sets $\mathcal{P}_t$, then the DPE can be written as

$$Q_{DR,t}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathscr{Q}_{DR,t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \tag{13.2}$$

with

$$\mathscr{Q}_{DR,t+1}(x_t) := \max_{\mathbb{P}_{t+1} \in \mathcal{P}_{t+1}} \mathbb{E}_{\mathbb{P}_{t+1}} \left[Q_{DR,t+1}(x_t, \boldsymbol{\xi}_{t+1}) \right], \tag{13.3}$$

and $\mathscr{Q}_{DR,T+1}(x_T) \equiv 0$. Compared to Section 3, here, an inner maximization problem is introduced when defining $\mathscr{Q}_{DR,t+1}(\cdot)$ to obtain the expected cost over the worst-case probability measure in $\mathcal{P}_{t+1}$. The first-stage problem reads

$$v_{DR}^* = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathscr{Q}_{DR,2}(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{13.4}$$

How v_{DR}^* and a corresponding optimal policy can be computed algorithmically, heavily depends on the specific choice of the ambiguity sets $\mathcal{P}_t, t = 2, \ldots, T$. Various ambiguity sets are proposed in the literature. Usually, these sets are defined in such a way that they contain all distributions, which are *in some sense* within a given range of some nominal distribution. This nominal distribution, denoted by $\bar{\mathbb{P}}_t$, in turn, is defined by probabilities $\bar{p}_{tj} = \frac{1}{q_t}$ for all $j = 1, \ldots, q_t$, where q_t denotes the number of historical data samples. Based on the measure employed to evaluate the distance between two distributions or probability measures, respectively, different classes of ambiguity sets can be defined.

For SDDP, the following three distance measures have been used so far. In [114], the ℓ_∞ metric with parameter $r > 0$ is used to define the ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q_t} p_{ti} = 1, \ p_{ti} \geq 0, \ \|p_t - \bar{p}_t\|_\infty \leq r \right\}. \tag{13.5}$$

A similar metric, but with the ℓ_2-norm, is used in [169] to define the ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q_t} p_{ti} = 1, \ p_{ti} \geq 0, \ \|p_t - \bar{p}_t\|_2 \leq r \right\}. \tag{13.6}$$

This is a special case of the class of ϕ-divergence distances, see [12]. Both these distance measures are only applicable to discrete distributions supported on the observed

data points, and contrary to the Wasserstein distance discussed below, they do not go out of this initial support.

On the contrary, the Wasserstein distance allows to compare general distributions (see for instance [225]). In our case with finite distributions $\mathbb{P}_t$ and $\bar{\mathbb{P}}_t$, the Wasserstein distance can be defined by the minimization problem

$$\begin{aligned} d_W(\bar{\mathbb{P}}_t, \mathbb{P}_t) := \min_{z} \quad & \sum_{i=1}^{q_t}\sum_{j=1}^{q_t} \|\xi_t^i - \xi_t^j\| z_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{q_t} z_{ij} = \bar{p}_{ti} \quad \forall i = 1, \dots, q_t \\ & \sum_{i=1}^{q_t} z_{ij} = p_{tj} \quad \forall j = 1, \dots, q_t \\ & z_{ij} \geq 0 \quad \forall i, j = 1, \dots, q_t, \end{aligned}$$

where for the norm different choices are possible. It can be interpreted as the amount of probability mass that has to be moved between the distributions. This distance is used in [69] to define the Wasserstein ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q} p_{ti} = 1, \ p_{ti} \geq 0, \ d_W(\bar{\mathbb{P}}_t, \mathbb{P}_t) \leq r \right\}. \tag{13.7}$$

In all three cases, very different strategies are chosen to apply SDDP to the nested min-max structure defined by the DPE (13.2)-(13.4).

13.1.1. Reformulation as a Risk-averse Problem. As shown in [114], using the ℓ^∞-ambiguity set (13.5), the DPE (13.2)-(13.4) can be reformulated to those of a risk-averse multistage problem with nested conditional $\text{AVaR}_\alpha[\cdot]$, that is equations (12.19)-(12.21) with

$$\lambda_{t+1} = 1 - \mu^\ell_{t+1}, \quad \alpha_{t+1} = \frac{\lambda_{t+1}}{\mu^u_{t+1} - \mu^\ell_{t+1}}.$$

Here, $\mu^\ell_{t+1} := \sum_{i=1}^{q_{t+1}} p^\ell_{t+1,i}$ and $\mu^u_{t+1} := \sum_{i=1}^{q_{t+1}} p^u_{t+1,i}$, where $p^\ell_{t+1} := \bar{p}_{t+1} - r$ denotes the probabilities associated with the probability measure at the lower bound of ambiguity set (13.5) (and $p^u_{t+1} := \bar{p}_{t+1} + r$ is defined analogously for the upper bound). Therefore, SDDP can be applied as in this risk-averse setting.

13.1.2. Solving the Inner Maximization Problem Separately. In [169] an ℓ^2-ambiguity set (13.6) is considered. It is then exploited that for convex ambiguity sets, such as the ℓ^2-ambiguity set, $\max_{\mathbb{P}\in\mathcal{P}} \mathbb{E}_{\mathbb{P}}[\cdot]$ can be interpreted as a coherent risk measure, see (12.9), and thus the value functions in the DPE (13.2)-(13.4) remain convex.

To derive linear cuts to approximate these value functions, it is proposed to solve the inner maximization problem identifying the worst-case distribution separately. In the backward pass, for some stage t, first the subproblems are solved for all $j = 1, \dots, q_t$ as usual (see line 16 of Algorithm 3.1). Then, using the obtained values of $\underline{Q}^i_t(x^{ik}_{t-1}, \xi_{tj})$, the inner maximization problem is solved. This can be done algorithmically and in some cases even analytically, as shown in [169]. The obtained worst-case probability measure $\mathbb{P}^*$ can then be used to compute subgradients and

cut coefficients. Even though these coefficients are determined based on cut approximation $\underline{\mathcal{V}}_t^{i+1}(\cdot)$ and on $\mathbb{P}^*$, which does not necessarily coincide with the worst-case probability measure in the true DPE, valid cuts are constructed and convergence is ensured [169].

13.1.3. Using a Dual Representation. If we use the Wasserstein ambiguity set (13.7) in SDDP, we obtain the inner maximization problem

$$\begin{aligned}
\max_{z_t, p_{t+1}} \quad & \sum_{j=1}^{q_t} p_{t+1,j} Q_{t+1}(x_t, \xi_{t+1,j}) \\
\text{s.t.} \quad & \sum_{i=1}^{q_t} \sum_{j=1}^{q_t} d_{t+1,ij} z_{tij} \leq 1 \\
& \sum_{j=1}^{q_t} z_{tij} = \bar{p}_{ti} \quad \forall i = 1, \ldots, q_t \\
& \sum_{i=1}^{q_t} z_{tij} = p_{tj} \quad \forall j = 1, \ldots, q_t \\
& z_{tij} \geq 0 \quad \forall i, j = 1, \ldots, q_t
\end{aligned}$$

with $d_{t+1,ij} = \|\xi_{t+1}^i - \xi_{t+1}^j\|$. Duque and Morton [69] suggest to replace this problem using its dual problem. This way, the value functions can be evaluated by solving the single-level minimization problem

$$Q_{DR,t}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t, \gamma_t, \nu_t} & c_t^\top x_t + r\gamma_t + \sum\limits_{i=1}^{q_{t+1}} q_{t+1}^i \nu_t^i \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \\ & d_{t+1,ij}\gamma_t + \nu_{ti} \geq Q_{DR,t+1}(x_t, \xi_{t+1,j}) \quad \forall i, j = 1, \ldots, q_{t+1} \\ & \gamma_t \geq 0 \end{cases}$$

with dual variables γ_t and ν_t. As proven in [69], these value functions are piecewise linear and convex on $\mathcal{X}_{t-1}$, and therefore can be represented by finitely many linear cuts. However, this approach requires to use multi-cut SDDP, see Section 21.2.1, because otherwise bilinear terms occur.

For all the strategies presented in Sections 13.1.1 to 13.1.3, the forward pass of distributionally robust SDDP remains basically the same as in standard SDDP, with additional flexibility in the sampling step. More precisely, the sampling can be done from the nominal distribution associated with $\bar{\mathbb{P}}_t, t = 2, \ldots, T$, from the current worst-case distribution associated with $\mathbb{P}_t^*$ or more generally from a convex combination of both, defined by some parameter $\beta \in [0, 1]$. If independent sampling is conducted, for $\beta \in [0, 1)$, convergence of distributionally robust SDDP can be established as for standard SDDP [69]. However, challenges to determine valid upper bounds are prevalent similarly to the risk-averse case.

Computational results indicate that taking the dual reformulation approach, better approximations are achieved for multi-cut SDDP than solving the inner maximization in a side computation [69]. Furthermore, out-of-sample tests by Philpott et al. [169] imply that distributionally robust SDDP yields policies which are better suited, *e.g.*, induce lower costs, in periods with a substantial risk of high costs.

13.2. Partially Observable Distributions. A different approach to deal with distributional uncertainty is introduced by Dowson et al. in [63], and is referred to as *partially observable* multistage stochastic programming. The idea is to consider a finite number of potential distributions by combining problem (MSLP) with a hidden Markov model. More precisely, in each stage $t \in [T]$, different nodes can be reached, with each node representing one Markov state. Let $\mathcal{N}$ denote the set of all these nodes except for the root node. Each node reflects a different candidate distribution, possibly with identical realizations $\xi_j, j = 1, \dots, q$, but different associated probabilities.

As a key idea, consider a partition $\mathcal{A}$ of nodes in $\mathcal{N}$ into ambiguity sets $A \in \mathcal{A}$, satisfying $\bigcup_{A \in \mathcal{A}} A = \mathcal{N}$. For example, this partition can be chosen such that there is one ambiguity set A for each stage.

To model the distributional uncertainty, it is now assumed that at any point, only the current ambiguity set is known, while the specific node within it cannot be observed. However, for each node i, a probability b_i is available. In other words, each candidate distribution is considered to be the most accurate representation of the true underlying distribution with a certain probability. These probabilities are stored in a so called *belief state* b. Each time an ambiguity set A is entered and a particular realization $\widetilde{\xi}$ of the random data is observed, the belief state is updated componentwise by applying Bayes' theorem [63].

In contrast to (MSLP) with perfect distribution information (see Assumption 3), the value functions $Q_t(\cdot)$ have to incorporate this belief state. To this end, let $p_{i\ell}$ be the probability of observing $\xi_{i\ell}$ conditional on being in node i with $\ell = 1, \dots, q^i$. Let $\bar{\mathcal{N}}$ describe all nodes including the root node, ω_{jk} the transition probability from node j to k and $B_k(b, \xi)$ the update rule for the belief state being in (unobservable) node k. Furthermore, let x' denote the current trial solution. Then, the expected value function can be written as

$$\mathscr{Q}_B(x', b) := \sum_{j \in \bar{\mathcal{N}}} b_j \sum_{k \in \mathcal{N}} \omega_{jk} \sum_{\ell=1}^{q^k} p_{k\ell} \; Q_k\big(x', B_k(b, \xi_{k\ell}), \xi_{k\ell}\big). \tag{13.8}$$

This means that the value functions $Q_k(\cdot, \cdot)$ depend on a node and an updated belief state, and in (13.8) it is looped over all nodes, weighing the corresponding expected value with the current belief and the transition probabilities between the nodes.

In the forward pass, for each stage $t = 2, \dots, T$, first, a new node is sampled conditionally on the (unobserved) current node. Then, a realization of ξ is sampled conditionally on the obtained node and the associated candidate distribution. Based on these samples, the ambiguity set A and the belief state b are updated. The latter can be considered as an additional state. For a more detailed description, see [63].

As proven in Theorem 1 in [63], the expected value functions $\mathscr{Q}_B(\cdot)$ are saddle functions, as they are convex in x for fixed b, but concave in b for fixed x. Therefore, to apply SDDP, the cut generation has to be adapted to this property, see also Subsection 14.6. This can be achieved by using an outer approximation for x and an inner approximation for b [63]. Apart from the changed cut formula, in line 15 of Algorithm 3.1 it is looped over all nodes in the current ambiguity set A and then conditionally on all possible realizations of $\boldsymbol{\xi}$ on the following stage. In line 18 of Algorithm 3.1, in addition to taking conditional expectations with respect to $\boldsymbol{\xi}$, the cut components are summed over all nodes in A and weighed with the current belief state for these nodes, similar to (13.8). [63].

A different method of combining SDDP with a hidden Markov model is given in [70]. One general drawback of such hidden Markov approaches is that transition

probabilities between the nodes have to be properly defined a priori.

14. Stagewise Dependent Uncertainty [relaxing Assumption 2]. As explained in Sections 2 and 3, stagewise independence (Assumption 2) is a standard assumption in dynamic programming, and thus also for SDDP. It is also crucial for the computational tractability of SDDP compared to NBD because it ensures that there exists only one expected value function $\mathscr{Q}_t(\cdot)$ per stage and that cuts can be shared between scenarios, see Section 5.2. However, in many applications, the uncertain data in (MSLP) (*e.g.*, demand, fuel prices, electricity prices, inflows) shows correlations over time and assuming stagewise independence is not appropriate.

If the uncertainty in problem (MSLP) is stagewise dependent, the expected value functions $\mathscr{Q}_t(\cdot)$ for $t = 2, \dots, T$ do not only depend on x_{t-1}, but implicitly also depend on the history $\xi_{[t-1]}$ of the process $(\boldsymbol{\xi}_t)_{t\in[T]}$. In order to apply SDDP, this dependence has to be taken into account, for instance by reformulating the model or adapting the algorithmic steps in SDDP. In this section, we consider different cases of stagewise dependent uncertainty and ways of how SDDP can be applied in these cases.

14.1. Expanding the State Space. As a first case of stagewise dependent uncertainty, let us assume that the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is a simple linear *autoregressive* (AR) process with lag one, defined by appropriately chosen coefficient vectors γ_t, matrices Φ_t and stagewise independent and i.i.d. error terms η_t:

$$\boldsymbol{\xi}_t = \gamma_t + \Phi_t \boldsymbol{\xi}_{t-1} + \boldsymbol{\eta}_t. \tag{14.1}$$

Remark 14.1. If we still assume finite randomness (Assumption 5), now for $\boldsymbol{\eta}_t$, then $\boldsymbol{\xi}_t$ can be modeled by a classical scenario tree, see Section 5.2. □

The most natural approach to deal with this case, is to reformulate (MSLP) in such a way that it exhibits stagewise independent uncertainty [161]. This can be achieved by including ξ_{t-1} as an additional state variable. Then, as shown in [136],

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \boldsymbol{\xi}_t)\right] &= \mathbb{E}_{\boldsymbol{\eta}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \boldsymbol{\eta}_t)\right] \\ &= \mathbb{E}_{\boldsymbol{\eta}_t}\left[Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \boldsymbol{\eta}_t)\right],\end{aligned}$$

where the second equality holds because η_t and ξ_{t-1} are statistically independent.

By introducing equation (14.1) as a constraint and defining a new value function

$$\widehat{Q}_t(x_{t-1}, \xi_{t-1}, \eta_t) := Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \eta_t), \tag{14.2}$$

and the corresponding expected value function

$$\widehat{\mathscr{Q}}_t(x_{t-1}, \xi_{t-1}) := \mathbb{E}_{\boldsymbol{\eta}_t}\left[\widehat{Q}_t(x_{t-1}, \xi_{t-1}, \boldsymbol{\eta}_t)\right] \tag{14.3}$$

for all $t = 2, \dots, T$, it follows

$$\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \boldsymbol{\xi}_t)\right] = \widehat{\mathscr{Q}}_t(x_{t-1}, \xi_{t-1}).$$

The state variables then consist of the resource state x_{t-1} and the information state ξ_{t-1}, while the stagewise independent uncertainty is modeled by $\boldsymbol{\eta}_t$. Importantly, ξ_t is regarded as a decision variable in the reformulated problem, augmenting the dimension of the decision space.

Remark 14.2. It is worth emphasizing that this approach is presented in different ways in the literature. In some cases, as outlined, equation (14.1) is explicitly incorporated into the DPE as an additional constraint [181, 212]. In some cases, each occurrence of ξ_t in the subproblems is simply replaced by the RHS of (14.1). And in other cases, the dependence on ξ_{t-1} is only expressed by writing $\widehat{Q}_t(\cdot,\cdot,\cdot)$ and $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ as functions of ξ_{t-1}, whereas the explicit relation (14.1) is only considered in the cut generation process [91, 136, 186]. We revisit this observation in the next subsection. □

By the presented procedure, stagewise independence (Assumption 2) is recovered for (MSLP). However, in order to apply SDDP, it also has to be ensured that valid linear cuts for $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ can be derived as functions in both types of state variables. This requires that $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ is convex in both x_{t-1} and ξ_{t-1}. Similarly to Theorem 2.8, it can be shown that under certain assumptions, this property is satisfied.

THEOREM 14.3 ([186]). *Let $\boldsymbol{\xi}_t$ be described by (14.1) and let ξ_{t-1} be contained in some convex set. Then, under Assumptions 1 and 3 to 9, the expected value function $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ is piecewise linear and*

a) convex in x_{t-1} on $\mathcal{X}_{t-1}$ for fixed ξ_{t-1},
b) convex in $\xi_{t-1} = (T_{t-2}, h_{t-1})$ for fixed $x_{t-1}, W_{t-1}, c_{t-1}$,
c) concave in $\xi_{t-1} = c_{t-1}$ for fixed $x_{t-1}, W_{t-1}, T_{t-2}, h_{t-1}$,
d) convex jointly in x_{t-1} and in $\xi_{t-1} = h_{t-1}$ for fixed $W_{t-1}, T_{t-2}, c_{t-1}$.

Theorem 14.3 shows that convexity in both types of state variables is only guaranteed if the stagewise dependent part of the uncertainty only enters the RHS $\boldsymbol{h}_t(\xi_t)$ of problem (MSLP). Note that this still allows for additional stagewise independent uncertainty in c_t, W_t and T_{t-1}. The result also requires linearity of (MSLP) (Assumption 6) and of the AR process (14.1) defining the random variable $\boldsymbol{\xi}_t$.

Under certain assumptions, Theorem 14.3 can be generalized to convex problems (MSLP) and stagewise dependence in the RHS defined by a convex function [91]. Moreover, the result is not limited to lag-one processes, but can be enhanced to AR processes with higher lag order [91]. This is important for practical applications, as often several lags are required to explain a time series appropriately. In contrast, for general nonlinear stochastic processes or for uncertainty in W_t, c_t or T_{t-1}, such a generalization seems not possible. In order to cover such cases, different approaches are required. We discuss those in later parts of this section.

For simplicity, assume that $X_t = \{x_t \in \mathbb{R}^{n_t} \,:\, x_t \geq 0\}$ for all $t \in [T]$ and recall the definition of the approximate subproblem (2.10):

$$\underline{Q}_t(x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t, \theta_{t+1}} & \big(c_t(\xi_t)\big)^\top x_t + \theta_{t+1} \\ \text{s.t.} & W_t(\xi_t)x_t = h_t(\xi_t) - T_{t-1}(\xi_t)x_{t-1} \\ & x_t \geq 0 \\ & -(\beta_{t+1}^r)^\top x_t + \theta_{t+1} \geq \alpha_{t+1}^r, \quad \forall r \in \Gamma_{t+1}, \end{cases} \tag{14.4}$$

where Γ_{t+1} is the index set of previously generated cuts. Then, the result in Theorem 14.3 can be illustrated by means of the feasible region of the LP dual to (14.4),

which can be written as

$$\begin{aligned} \max_{\pi_t, \mu_t} \quad & \big(h_t(\xi_t) - T_{t-1}(\xi_t) x_{t-1}\big)^\top \pi_t + a_{t+1}^\top \mu_t \\ \text{s.t.} \quad & \big(W_t(\xi_t)\big)^\top \pi_t - B_{t+1}^\top \mu_t \leq c_t(\xi_t) \\ & e^\top \mu_t = 1 \\ & \mu_t \geq 0. \end{aligned} \tag{14.5}$$

Here, we collect all cut gradients β_{t+1}^r in a matrix B_{t+1} and all cut intercepts α_{t+1}^r in a vector a_t for compact representation. π_t denotes the dual variable to the original constraints, and μ_t denotes the dual variable to the previously generated cuts.

In the case of linear AR processes in the RHS $\boldsymbol{h}_t(\xi_t)$, the dual feasible region is not affected by the new state variable ξ_{t-1} (and also remains polyhedral). This means that the extreme solutions obtained for one state $\bar{\xi}_{t-1}$ remain valid, although not necessarily optimal, for all other states ξ_{t-1} as well. In contrast, in other cases of stagewise dependence, the dual feasible region and its extreme solutions may change for different states, affecting the properties of $\widehat{\mathscr{Q}}_t(\cdot, \cdot)$ [186].

In sum, for affine and convex AR processes occurring in the RHS, expanding the state recovers stagewise independence (Assumption 2), but at the same time convexity of $\widehat{\mathscr{Q}}_t(\cdot, \cdot)$ in all state variables is preserved. Therefore, SDDP can be used as introduced in Section 3. In this case, the obtained cuts are functions of both state variables and can be formulated with a cut gradient for each of them (compare to (3.5)), *i.e.*,

$$\phi_t(x_{t-1}, \xi_{t-1}) = \alpha_t + \big(\beta_t^x\big)^\top x_{t-1} + \big(\beta_t^\xi\big)^\top \xi_{t-1}.$$

Unfortunately, depending on the dimension κ_{t-1} of ξ_{t-1}, the state space dimension can increase significantly. This effect is amplified for higher lag orders. As the computational complexity of SDDP grows exponentially in this dimension, see Section 4.2, augmenting the state space is detrimental and should be avoided if possible.

14.2. Scenario-Adaptable Cut Formulas. The previously described adverse effect can be alleviated to some degree by a special cut generation approach that was first proposed by Infanger and Morton [116] and later enhanced by de Queiroz and Morton [181] and Guigues [91]. In all these cases, the process model, such as (14.1), is not explicitly incorporated into the subproblems, see Remark 14.2. Instead, it is merely considered within the cut generation process. The main idea is to derive scenario-adaptable closed-form cut formulas, given AR processes with a specific structure, which allow one to adapt the cut generated for one specific history $\bar{\xi}_{[t-1]}$ to different histories $\xi_{[t-1]}$ of the stochastic process, and thus to different scenarios. This way, the cuts can be *shared* between scenarios (see Section 5.2) without the need to incorporate (14.1) into (MSLP) as a constraint. Importantly, these cut formulas lead to the exact same cuts as the previously described approach.

To illustrate this idea, consider a cut derived using dual problem (14.5) without paying any particular attention to the stagewise dependence. For convenience, but without loss of generality, we assume T_{t-1} to be deterministic and the RHS uncertainty to be defined by

$$\boldsymbol{h}_t(\boldsymbol{\xi}_t) = \Phi_t h_{t-1}(\xi_{t-1}) + \boldsymbol{\eta}_t \tag{14.6}$$

with stagewise independent error terms $\boldsymbol{\eta}_t$, similarly to (14.1). We obtain

$$\begin{aligned} \widehat{\mathscr{Q}}_t(x_{t-1}, \xi_{t-1}) &\geq \mathbb{E}_{\boldsymbol{\xi}_t | \xi_{t-1}} \left[-\boldsymbol{\pi}_t^\top T_{t-1} x_{t-1} + \boldsymbol{\pi}_t^\top \boldsymbol{h}_t(\boldsymbol{\xi}_t) + \boldsymbol{\mu}_t^\top \boldsymbol{a}_{t+1} \right] \\ &= \mathbb{E}_{\boldsymbol{\xi}_t | \xi_{t-1}} \left[-\boldsymbol{\pi}_t^\top T_{t-1} \right] x_{t-1} + \mathbb{E}_{\boldsymbol{\xi}_t | \xi_{t-1}} \left[\boldsymbol{\pi}_t^\top \boldsymbol{h}_t(\boldsymbol{\xi}_t) + \boldsymbol{\mu}_t^\top \boldsymbol{a}_{t+1} \right] \end{aligned} \tag{14.7}$$

We can make the following observations:

(i) Since the probabilities in $\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}[\cdot]$ are assumed to not depend on ξ_{t-1} (recall that $\boldsymbol{\eta}_t$ is stagewise independent) and since all scenarios share the same dual feasible region, the cut gradient

$$\beta_t = \mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[-\boldsymbol{\pi}_t^\top T_{t-1}\right] \tag{14.8}$$

derived for one specific scenario $\bar{\xi}_{t-1}$, is valid for all other scenarios as well.

(ii) According to (14.6), the RHS $\boldsymbol{h}_t(\boldsymbol{\xi}_t)$ depends on ξ_{t-1}. Therefore, to evaluate the cut for a specific scenario, this term has to be adapted to this scenario. Otherwise, the cut may become invalid. By (14.6), this term can be split up into a scenario-dependent part depending on ξ_{t-1} and a scenario-independent part depending on η_t only.

(iii) The last term $\boldsymbol{a}_{t+1}$ in (14.7) is the cut intercept of the following stage. As we face stagewise dependence, this intercept is not scenario-independent anymore, but should denote $\boldsymbol{a}_{t+1}(\xi_t)$. Moreover, it is defined recursively: The stage-t intercept includes the stage-$(t+1)$ intercept, which includes the stage-$(t+2)$ intercept and so on. This implies that to evaluate $\boldsymbol{a}_{t+1}(\xi_t)$ for a specific scenario, it is basically required to recursively traverse the whole scenario tree starting form stage t. This is computationally intractable.

To address these observations, the main idea by Infanger and Morton [116] is to express the cut intercept $\alpha_t(\xi_{t-1})$ as the sum of a stagewise independent term α_t^{ind} and a stagewise dependent term $\alpha_t^{\text{dep}}(\xi_{t-1})$:

$$\alpha_t(\xi_{t-1}) = \alpha_t^{\text{ind}} + \alpha_t^{\text{dep}}(\xi_{t-1}). \tag{14.9}$$

Let $\bar{\pi}_t = \mathbb{E}_{\boldsymbol{\eta}_t}[\boldsymbol{\pi}_t]$ and $\bar{\mu}_t = \mathbb{E}_{\boldsymbol{\eta}_t}[\boldsymbol{\mu}_t]$ denote the expected value of the dual variables obtained for realizations of $\boldsymbol{\eta}_t$. As explained, these dual values are valid for any history of the stochastic process due to the structure of the dual feasible set. Let $\bar{\mathscr{P}}_t$ define the $(|\Gamma_t| \times m_t)$-matrix containing the values of $\bar{\pi}_t$ and $\bar{\mathscr{R}}_t$ the $(|\Gamma_t| \times |\Gamma_{t-1}|)$-matrix containing the values of $\bar{\mu}_t$ for the previously determined cuts. Furthermore, let the matrix D_t be defined recursively by

$$D_t = \left[\bar{\mathscr{P}}_{t+1} + \bar{\mathscr{R}}_{t+1} D_{t+1}\right] \Phi_t, \quad D_T = 0. \tag{14.10}$$

Then, as shown in [116], the stagewise dependent cut intercept is given by

$$\alpha_t^{\text{dep}}(\xi_{t-1}) = \left[\bar{\pi}_t + \bar{\mu}_t D_t\right] \Phi_t h_{t-1}(\xi_{t-1}). \tag{14.11}$$

This means that in line 18 of Algorithm 3.1 a cut can be constructed by using formula (14.8) for the gradient and formulas (14.9), (14.10) and (14.11) for the intercept. The stagewise independent term can be either determined by an additional formula or by subtracting (14.11) from $\alpha_t(\xi_{t-1})$ [116]. In order for a cut to be shared with a different scenario at stage $t-1$, it is only required to adapt the stagewise dependent intercept (14.11) to this specific scenario. In other words, a given cut can be *corrected* to be valid for a different history of the stochastic process. In particular, it is not required to add (14.6) as a constraint to the stage-t subproblem or to traverse the whole scenario tree (see Remark 14.1). Instead, only the cut gradient, the stagewise independent part of the intercept and the *cumulative expected dual vector* $\left[\bar{\mathscr{P}}_{t+1} + \bar{\mathscr{R}}_{t+1} D_{t+1}\right] \Phi_t$ have to be stored [116].

Whereas we limited our explanations to a very simple AR process so far, similar cut formulas can be derived for more complex processes [91, 116, 181, 186]. We give

RHS $h_t(\xi_t)$	Autoregressive model for ξ_t Model	Type	Lag	Formula	Source
const.	AR	L	1	$\xi_t = \Phi_t \xi_{t-1} + \eta_t$	[116]
L	AR	L	1	$\xi_t = \Phi_t \xi_{t-1} + \eta_t$	[181]
const.	PAR	L	1	$\xi_t = \varphi_t(\xi_{t-1} - \psi_{t-1}) + \mu_t + \sigma_t \eta_t$	[219]
const.	AR	L	≥ 1	$\xi_t = \sum_{k=1}^{t-1}(\Phi_k^t \xi_k + \Psi_k^t \eta_k) + \eta_t$	[116]
L/C*	AR	L	≥ 1	$\xi_t = \Phi_t \xi_{[t-1]} + \eta_t$	[91]
L/C*	AR	L	≥ 1	$\xi_t = \Phi_t \xi_{[t-1]} + \Psi_t \eta_t + \Theta_t$	[91]
const.	SPAR	L	≥ 1	$\xi_{ti} = \sum_{i'} \sum_{k=1}^{t-1} \Phi_{ii'k}^t \xi_{ti'} + \eta_{ti}$	[134]
const.	AR	NL	1	$\xi_t = \Phi_t(f_t(v_{t-1}) + \xi_{t-1}) + \eta_t$	[116]
const.	AR	NL	≥ 1	$\xi_t = \sum_{k=1}^{t-1}(\Phi_k^t \xi_k + f_k^t(v_k)) + \eta_t$	[116]
C	AR	C	≥ 1	$\xi_t = f_t(\xi_{[t-1]}, \eta_t)$	[91]

L = affine/linear function, C = convex function, NL = general nonlinear function
* only in case of inequality constraints

Table 6: RHS and uncertainty models considered in the literature on SDDP with stagewise dependence to derive scenario-adaptable closed-form cut formulas.

an overview on different cases covered in the literature in Table 6. Some of the process formulas in Table 6 are presented in a simplified form for reasons of clarity, *e.g.*, by omitting standardization and the incorporation of seasonal or periodical effects. For example, this is true for the SPAR processes considered in [134] (also see Section 9), where spatial dependencies between locations i and i' are taken into account.

Importantly, all processes for which scenario-adaptable closed-form cut formulas can be derived require a specific structure, such as linearity, convexity or separability. As shown by Guigues [91], a generalization to convex AR processes and more complex structures in the RHS is possible. For instance, the RHS h_t does not have to be directly described by the stochastic process (constant $\boldsymbol{h}_t \equiv \boldsymbol{\xi}_t$), but may also be defined as some function $\boldsymbol{h}_t(\cdot)$ of $\boldsymbol{\xi}_t$. Moreover, for the affine case, alternative formulas to the ones provided by Infanger and Morton are presented by Guigues [91]. The main difference is that only a minimal subset of coefficients is used, due to defining the process $(\boldsymbol{\xi}_t)_{t\in[T]}$ componentwise and not in vectorial form compared to (14.1) or (14.6). On the other hand, no recursive formula as in (14.10) is provided to compute the cut coefficients. Finally, Guigues shows that also for feasibility cuts (Section 17) scenario-adaptable cut formulas can be derived.

It is important to emphasize that the presented approach only partially mitigates the drawbacks of augmenting the state space. First of all, the history of the stochastic process has to be stored to compute ξ_t, even if such computation is possible outside of the subproblems. Guigues provides a detailed discussion on how state vectors of minimal size can be defined in order to keep the stored information as small as possible [91]. Additionally, due to their dependence on ξ_{t-1}, or $\xi_{[t-1]}$ in general, the expected value functions $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ live in a higher-dimensional space. Therefore, more iterations and cuts may be required to achieve convergence compared to the stagewise independent case, as discussed in Section 4.2.

14.3. Sensitivity of SDDP with AR Processes. Let the uncertainty in (MSLP) be modeled by an AR process. Consider the approach of expanding the state, leading to two types of state variables: x_t and $\xi_{[t]}$. Both contain information

on future resource availability (*e.g.*, hydro storage volume and hydro inflow history affecting future inflows), but they differ in several aspects [216]. First, whereas the information provided by the state x_{t-1} is certain, the information provided by $\xi_{[t-1]}$ enters an AR model predicting future realizations, which still involves uncertainty. Second, the parameters of this AR model are estimated from data, and thus can be subject to estimation errors. Third, in practice it can often be observed that the values in $(\xi_t)_{t\in[T]}$ show higher variability over short time than the values of $(x_t)_{t\in[T]}$. This uncertainty and variability raises the question on how much the solutions obtained in SDDP react to changes in $\xi_{[t-1]}$. This can be examined in a *sensitivity analysis*.

A general approach for sensitivity analysis in SDDP is presented in [104] and applied to an inventory problem with AR demand. Also the sensitivity with respect to AR model parameters Φ_t or γ_t is discussed.

For a hydrothermal problem, in [216], it is shown that the solutions obtained in SDDP are more sensitive to changes in the initial information state ξ_1 than to changes in the initial resource state x_0. Based on the previous observations this leads to the unfavorable side effect of expanding the state space that solutions of SDDP exhibit larger variability. This may have severe consequences in economic applications, such as increasing risk, unpredictability of prices or distorted investment signals.

To address this issue, Soares et al. [216] present different mitigation heuristics, such as regularizing changes in x_t over time, or using the accurate AR model in the forward pass of SDDP, but predefined unconditional samples in the backward pass in order to avoid the dependence of cuts on $\xi_{[t-1]}$. While they report positive computational results, the authors provide no theoretical results on reasonable parameter choice, cut validity and convergence for their heuristics.

14.4. SDDP with Markov Chain. According to Theorem 14.3, a natural extension of SDDP to stagewise dependent uncertainty by expanding the state space is only possible for linear (or at least convex) AR processes appearing in the RHS of problem (MSLP). In all other cases, the convexity of the expected value functions $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ is destroyed. Therefore, in such cases, different approaches are required. One such approach is to incorporate a discrete Markov chain into the uncertainty modeling. This approach is quite established in the literature on SDDP and used in practical applications in various forms.

14.4.1. Modeling. Consider a Markov chain with finitely many possible states $\zeta_\ell, \ell = 1,\dots,L$, with $L \in \mathbb{N}$. At each stage $t \in [T]$, we denote the current state of the Markov chain as ψ_t (again, we assume that ψ_1 is deterministic). The transition probabilities between state $\psi_{t-1} = \zeta_\ell$ at stage $t-1$ and $\psi_t = \zeta_{\ell'}$ at stage t are then denoted by $\omega_{\ell\ell'}$ for $\ell, \ell' \in \{1,\dots,L\}$. For simplicity, we assume the Markov chain to be time-homogeneous, such that $\omega_{\ell\ell'}$ does not depend on t, even though this is not required.

We now assume that the distribution of random variable $\boldsymbol{\xi}_t$ at stage $t \in [T]$ may depend on the state ψ_t of the Markov chain. In other words, for each possible state $\zeta_\ell, \ell = 1,\dots,L$, the distribution of $\boldsymbol{\xi}_t$ may differ. We emphasize this by writing $\boldsymbol{\xi}_t^\ell$.

The value functions $Q_t(\cdot,\cdot)$ for (MSLP) then do not only depend on x_{t-1} and the realization ξ_t of $\boldsymbol{\xi}_t$, but also on the current Markov state ψ_t. As this state can only take finitely many values, we denote this by $Q_{t\ell}(x_{t-1},\xi_t)$, where index ℓ indicates conditioning on $\psi_t = \zeta_\ell$. Based on this definition, the expected value functions can

be expressed as

$$\mathscr{Q}_{t\ell}(x_{t-1}) := \sum_{\ell'=1}^{L} \omega_{\ell\ell'} \mathbb{E}_{\boldsymbol{\xi}_t|\ell'} \left[Q_{t\ell'}(x_{t-1}, \boldsymbol{\xi}_t^{\ell'}) \right]. \tag{14.12}$$

The index ℓ of the expected value function refers to the previous Markov state $\psi_{t-1} = \zeta_\ell$. Compared to standard SDDP, the expectation is not only taken over the realizations of $\boldsymbol{\xi}_t^{\ell'}$, but also the state transitions from ψ_{t-1} to ψ_t are taken into account. Using this definition, the DPE for stages $t = 2, \dots, T$ can be written as

$$Q_{t\ell}(x_{t-1}, \xi_t^\ell) := \begin{cases} \min\limits_{x_t} & \zeta_\ell^\top x_t + \mathcal{Q}_{t+1,\ell}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(\xi_t^\ell). \end{cases} \tag{14.13}$$

Note that the dependence on ζ_ℓ in (14.12) resembles the expanding-the-state approach from Section 14.1. However, there are important differences. ψ_{t-1} does not necessarily enter the subproblems, but can be an underlying (hidden) Markov state. Moreover, it can only take a finite number of different values, whereas $\xi_{[t-1]}$, even if discrete, is treated like a continuous state variable when expanding the state space. Furthermore, as the transition probabilities $\omega_{\ell\ell'}$ may differ for each ζ_ℓ, the cut components are weighted differently and cuts cannot be shared between different Markov states. Consequently, it is required to store separate expected value functions $\mathscr{Q}_{t\ell}(\cdot)$ for each $\ell = 1, \dots, L$. In return, the non-convexity of these functions is circumvented, since each $\mathscr{Q}_{t\ell}(\cdot)$ remains convex and is approximated on its own, see also the discussion in Section 14.4.

As an example, consider a problem with $L = 2$ Markov states and $q^\ell = 2$ realizations for $\boldsymbol{\xi}_t^\ell$ for each of them, which is borrowed from [167]. The corresponding scenario tree with underlying Markov chain is illustrated in Figure 13. For the transition probabilities let $\omega_{11} = q, \omega_{12} = 1 - q, \omega_{21} = 1 - p$ and $\omega_{22} = p$. For all t and $\ell \in \{1, 2\}$, the distribution of ξ_t^ℓ is given by $p_{tj} = \frac{1}{2}$ for $j \in \{1, 2\}$.

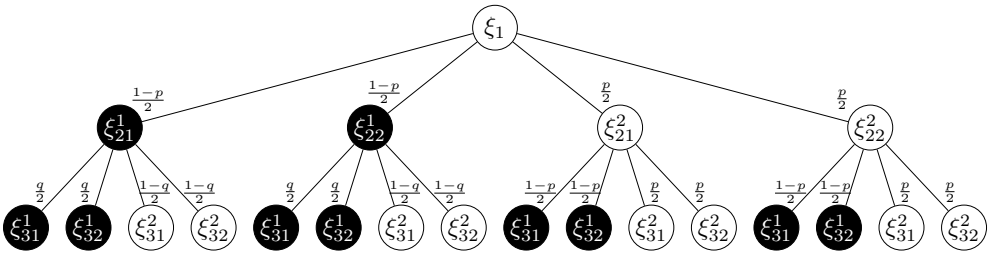


Fig. 13: Scenario tree with underlying Markov chain (state 1 printed in black, state 2 printed in white). Replication from [167].

As an alternative to the scenario tree in Figure 13, the stochastic process with underlying Markov chain can be represented by a Markovian policy graph with finitely many nodes per stage [60]. This approach is also included in the software package `SDDP.jl`, see Section 10.

14.4.2. Adaptation of SDDP. Let us now address how SDDP works in this case. In the forward pass, different approaches are used in the literature. The most natural one is for each stage t and each sample path $k \in \mathcal{K}$, to sample first from

the Markov states and then conditionally from $\boldsymbol{\xi}_t^\ell$ in line 6 of Algorithm 3.1 [168]. Sometimes it is also proposed to use historical values here, *e.g.*, true inflow spot-price combinations [86]. In such a case, it is possible that a spot price is drawn which is not a valid state of the Markov chain. Then, a strategy is to use the *in some sense* closest state from the Markov chain [86]. Another one is to use a linear interpolation between the hyperplanes of neighbouring states [88, 238].

For stages $t = 2, \dots, T$, states $\ell = 1, \dots, L$ and samples $k \in \mathcal{K}$, based on (14.13), the approximate subproblems solved in the forward pass of SDDP have the form

$$\underline{Q}_{t\ell}^i(x_{t-1}^{ik}, \xi_t^{\ell k}) := \begin{cases} \min\limits_{x_t} & \left(c_t(\xi_t^\ell)\right)^\top x_t + \underline{\mathcal{V}}_{t+1\ell}^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_t^\ell). \end{cases} \tag{14.14}$$

Importantly, each function $\mathcal{Q}_{t\ell}(\cdot), \ell = 1, \dots, L$, is approximated by an individual cut approximation $\underline{\mathcal{V}}_{t\ell}(\cdot)$.

In the backward pass of some iteration i, the stages are traversed in backward direction as usual to improve the cut approximations. At each stage t, the subproblems (14.14) including $\underline{\mathcal{V}}_{t\ell}^{i+1}(\cdot)$ are solved for each trial state x_{t-1}^{ik}, $k \in \mathcal{K}$, each stage-t Markov state $\psi_t = \zeta_\ell, \ell = 1, \dots, L$, and all realizations $\xi_{tj}^\ell, j = 1, \dots, q_t^\ell$. This means that in lines 13-21 of Algorithm 3.1 it is not only iterated over t, k and j, but over t, k, ℓ and $j(\ell)$.

Then, for each x_{t-1}^{ik} and $\psi_{t-1} = \zeta_\ell, \ell = 1, \dots, L$, a valid cut can be derived for $\mathcal{Q}_{t\ell}(\cdot)$. Let $\beta_{t\ell kj}^i$ denote a subgradient for $\underline{Q}_{t\ell}^i(\cdot, \cdot)$ at x_{t-1}^{ik}. In accordance with (3.4), but also taking into account the Markov chain transition probabilities, we can then define cut coefficients

$$\begin{aligned} \beta_{t\ell k}^i &:= \sum_{\ell'=1}^{L} \omega_{\ell\ell'} \left(\sum_{j=1}^{q_{t\ell}} p_{t\ell j} \left(\underline{Q}_{t\ell}^{i+1}(x_{t-1}^{ik}, \xi_t^{\ell k}) - (\beta_{t\ell kj}^i)^\top x_{t-1}^{ik} \right) \right), \\ \alpha_{t\ell} &:= \sum_{\ell'=1}^{L} \omega_{\ell\ell'} \left(\sum_{j=1}^{q_{t\ell}} p_{t\ell j} \beta_{t\ell kj}^i \right), \end{aligned} \tag{14.15}$$

where $q_{t\ell}$ and $p_{t\ell j}$ denote the number of realizations and probabilities of $\boldsymbol{\xi}_t^\ell$.

A cut (3.5) for $\mathcal{Q}_{t\ell}(\cdot)$ is then given by function

$$\phi_{t\ell k}^i(x_{t-1}) := \alpha_{t\ell k}^i + (\beta_{t\ell k}^i)^\top x_{t-1}$$

and can be used to update $\underline{\mathcal{V}}_{t\ell}^i(\cdot)$. Philpott et al. derive similar formulas for the multi-cut and risk-averse case [168].

The described approach allows for the incorporation of even nonlinear stagewise dependent uncertainty into SDDP, but also gives rise to some challenges. Among those is the assumption of the Markov property, which may not always be appropriate. Moreover, it is required to define useful values $\zeta_\ell, \ell = 1, \dots, L$, and transition probabilities $\omega_{\ell\ell'}$ for the Markov states [88, 150]. Most importantly, cuts cannot be shared between, but only within Markov states, so that separate expected value functions have to be considered for each $\ell = 1, \dots, L$. Therefore, the number of Markov states should be rather small to preserve computational tractability.

14.4.3. Specific Use Cases. There exist different use cases for modeling the uncertainty in (MSLP) with an integrated Markov chain.

- **Nonlinear Processes.** The data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ can be modeled as a nonlinear AR process or a nonlinear transformation of a linear AR process (see Section 9), which, if handled by expanding the state space, destroys the convexity of $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$. Sometimes such a nonlinear process can be approximated by assuming that the realizations ξ_t depend on an *underlying* system state which follows a Markov process [168], thus not capturing the nonlinearity explicitly in a formula. As the value functions are also not convex in this, possibly continuous, Markov state, the Markov process is approximated using a discrete Markov chain.
- **Regime-switching Models.** Instead of a single AR process, sometimes the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ may be best modeled by a finite set of different AR processes, which are valid representations, and thus active, under different circumstances (*e.g.*, macroeconomic, political or ecological situations). A discrete Markov chain can then be used to model these overall system states, and AR models can be used to describe realizations of the uncertain data conditioned on these states. Such *regime-switching* models are very common in wind forecasting [243].
- **Hybrid SDP/SDDP.** Different parts of the data in (MSLP) exhibit stagewise dependent uncertainty. While some of them, namely uncertainty in the RHS h_t, can be treated by expanding the state space, for others, *e.g.*, stagewise dependent uncertainty in the objective coefficients c_t, it would destroy the convexity of $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$. Therefore, this part of the uncertainty may be modeled by a discrete Markov chain instead. Since one part of the uncertainty is treated as in standard SDDP (allows for cut-sharing between scenarios), while another one is treated by enumerating separate expected value functions for each $\ell = 1,\dots,L$ (cuts cannot be shared between Markov states), this is often referred to as a *hybrid SDP/SDDP* method [86].
  For instance, this setting often occurs in medium-term hydrothermal scheduling problems (see Section 9) when inflow uncertainty in the RHS as well as spot-price uncertainty in the objective function are taken into account. The idea to address this by using a Markov chain goes back to Gjelsvik et al. who modeled this kind of scheduling problem for the Norwegian power system [86, 88, 89]. Since then, this approach has been employed in several applications, for example, hydrothermal scheduling including balancing market bids [107, 108], risk management [115, 123, 149] and fuel contracts [39]. It is also applied to model fuel price uncertainty [158].
  In contrast to the presented general approach, in this case it is usually assumed that the uncertainty in the RHS and in the objective are independent of each other. Therefore, for each state $\zeta_\ell, \ell = 1,\dots,L$, the distribution of $\boldsymbol{\xi}_t$ is the same, and marginal distributions can be used in the expectation in (14.12). Moreover, note that in this specific case the Markov chain states are not underlying the distribution of $\boldsymbol{\xi}_t$, but instead entering the subproblems explicitly, *e.g.*, as objective coefficients. Still SDDP can be applied using the same ideas as above.
- **Markov Chain SDDP.** Assume that the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ *itself* is Markovian, *i.e.*, as in (14.1), $\boldsymbol{\xi}_t$ only depends on ξ_{t-1} for all $t = 2,\dots,T$ instead of the whole history $\xi_{[t-1]}$. In this case, the data process can be represented, or at least approximated (if the random variables $\boldsymbol{\xi}_t$ are continuous), by a discrete Markov chain. This approximation can be obtained by lattice quan-

tization techniques [31, 136]. As it contains only finitely many states per stage $t = 2, \dots, T$, this Markov chain can be illustrated as a recombining scenario tree or *scenario lattice* [136], just as in the case of stagewise independence Assumption 2, see Section 2. The difference is that in the Markov chain case the probabilities of transitions to stage-t nodes may differ between different stage-$(t-1)$ nodes. This also includes the possibility that some stage-t nodes may not be reached from certain stage-$(t-1)$ nodes.

Due to this difference, the (expected) value functions $\mathscr{Q}_{t\ell}(\cdot)$ depend on the states $\zeta_\ell, \ell = 1, \dots, L$, of the Markov chain, and in SDDP cuts are derived for each of these functions separately. This idea is called Markov chain SDDP (MC-SDDP) [136] or *approximate dual dynamic programming* (ADDP) [137, 138], whereas for distinction the approach of expanding the state space is referred to as time series SDDP (TS-SDDP).

MC-SDDP can be considered a special case of the above framework, where the Markov states explicitly enter the subproblems and completely describe the uncertainty, instead of affecting another random variable. More precisely, in (14.12) the conditional expectations and $\boldsymbol{\xi}_t^{\ell'}$ can be omitted, and only probabilities $\omega_{\ell\ell'}$ are required. Analogously, in the cut intercept and gradient formulas (14.15), the summation over j and the probabilities $p_{t\ell j}$ can be omitted.

For problems with moderate state space dimension to MC-SDDP, expanding the state may be computationally favorable as only one expected value function has to be approximated per stage. On the other hand, a computational advantage of MC-SDDP is that the computational effort grows linearly with the number of Markov states only [209]. In contrast, expanding the state leads to a state space dimension increase in which the complexity of SDDP grows exponentially. Moreover, MC-SDDP requires no linearity and is not limited to stagewise dependent uncertainty only appearing in the RHS of (MSLP). As long as the Markov property is satisfied, it allows for stagewise dependent uncertainty in all data c_t, T_{t-1}, W_t and h_t of (MSLP).

The main drawback of MC-SDDP lies in the relation to the true problem $(\widetilde{P})$ in case of a continuous data process $(\boldsymbol{\xi}_t)_{t\in[T]}$, see also Section 11. For SDDP with AR processes and expanding the state space, many results exist that allow for inference of the SAA solution with respect to the true problem, see Section 11. One key property in this regard is that ξ_{t-1} is treated as a possibly continuous state variable in SDDP, such that the derived cuts are also valid at states which are not reached by the scenarios $\xi^s \in \mathcal{S}$ that are considered in SDDP. Similar results are not available for MC-SDDP. In particular, the obtained policy and lower bounds are not necessarily valid for the true problem [136].

In spite of this theoretical downside, Löhndorf and Shapiro [136] report tighter lower bounds and better policies even for the true process based on computational experiments. They conjecture that this is due to a differing exploration of the state space. Expanding the state space introduces additional state variables, which are not under control of the optimal policy (their trajectory is not chosen based on solving the approximate subproblems in the forward pass, but selected randomly in the forward pass). This may lead to selection of states, which do not provide the highest information gain. With MC-SDDP this is partially mitigated by choosing sufficiently different states in advance when constructing the Markov chain.

14.5. Hybrid NBD/SDDP. In the previous section, we presented a hybrid SDP/SDDP method as a tool to model different stagewise dependent uncertain data in (MSLP) by different approaches. Instead of modeling the "complicating" part of the uncertainty by a discrete Markov chain, also a scenario tree can be used. Instead of a hybrid SDP/SDDP method, this yields a hybrid NBD/SDDP method [186], see also Section 5.2.

Assume that the random vector $\boldsymbol{\xi}_t$ modeling the uncertainty in c_t, W_t, T_{t-1} and h_t can be separated into two separate and independent parts, $\boldsymbol{\xi}_t^S$ and $\boldsymbol{\xi}_t^T$. The first vector $\boldsymbol{\xi}_t^S$ can either be stagewise independent or exhibit some linear dependency if it occurs in the RHS. In the latter case, it can be handled by expanding the state space. Within SDDP, in each iteration samples of $\boldsymbol{\xi}_t^S$ are considered. The second vector $\boldsymbol{\xi}_t^T$, on the other hand, may lead to non-convexities in the value functions if it is approached by expanding the state space. Therefore, it is modeled by a scenario tree, which is treated exactly in SDDP. This means that for this particular part of the uncertainty, no samples are drawn, but all scenarios are considered in each iteration of SDDP, as in NBD, see Section 5.2. This approach is similar to hybrid SDP/SDDP in the sense that the expected value functions $\mathscr{Q}_t(\cdot)$ depend on the scenarios from $\boldsymbol{\xi}_t^S$ and that cuts can only be shared within, but not between such scenarios.

By only treating the crucial part $\boldsymbol{\xi}^T$ of $\boldsymbol{\xi}$ as a scenario tree and the remainder $\boldsymbol{\xi}^S$ still by sampling, complex uncertainty processes can be considered, while at the same time the increase of computational complexity is kept as small as possible [186]. To take advantage of this, the scenario tree associated with $\boldsymbol{\xi}^S$ should not be too large.

Compared to hybrid SDP/SDDP, in specific applications the one or the other approach may be favorable. For instance, the Markov chain approaches allow for dependencies between different uncertainty processes. Moreover, if each realization of $\boldsymbol{\xi}_t$ is assigned to one specific Markov state $\zeta_\ell, \ell = 1, \dots, L$, the number of LPs to be solved per iteration can be kept equal to standard SDDP. The scenario tree approach, by contrast, requires independence of $\boldsymbol{\xi}^S$ and $\boldsymbol{\xi}^T$. By design, it considers all combinations of scenarios of $\boldsymbol{\xi}^T$ and $\boldsymbol{\xi}^S$, so no assignment of realizations of $\boldsymbol{\xi}^S$ to scenarios of $\boldsymbol{\xi}^T$ is required. However, the number of LPs to be solved grow exponentially in the number of stages [186]. On the other hand, a scenario tree may be more appropriate to model very complex processes, *e.g.*, referring to macroeconomical, political or structural decisions [186], for which the Markov property is not appropriate.

14.6. Saddle Cuts. We consider the special case of stagewise dependent objective coefficients $\boldsymbol{c}_t(\xi_t)$ in (MSLP), as they appear for uncertain prices models by AR processes. So far, we introduced SDDP with integrated Markov chain as a suitable solution approach in this case. Now, we discuss as second one.

As discussed in Section 14.1, by expanding the state space, stagewise independence (Assumption 2) can be recovered, but in return the expected value functions $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ are no longer convex. In Theorem 14.3 it is shown that $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ is in fact convex in x_{t-1}, but concave in c_{t-1}, which yields a saddle shape. Therefore, linear cuts are not sufficient to approximate them. As a resort, exploiting the saddle shape, special *saddle cuts* can be used.

To derive this formally, in the vein of [59], we assume the objective coefficients to be described by $\big(y_t(\xi_t)\big)^\top C_t$ instead of $c_t(\xi_t)$. While the matrix C_t is considered deterministic, $y_t(\xi_t)$ is defined by the following AR process

$$y_t(\xi_t) = B_t(\xi_t) y_{t-1}(\xi_{t-1}) + b_t(\xi_t) \tag{14.16}$$

for all stages $t = 2, \dots, T$. Here, the matrix B_t and the vector b_t are uncertain and

depend on the realization of $\boldsymbol{\xi}_t$. Thus, the sequence $\big(y_t(\xi_t)\big)_{t=1}^T$ is scenario-dependent.

Inserting relation (14.16) into the objective function and considering y_{t-1} as an additional state variable, for $t = 2, \dots, T$, we obtain the subproblems

$$\widehat{Q}_t(x_{t-1}, y_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t} & \big(B_t(\xi_t)y_{t-1} + b_t(\xi_t)\big)^\top C_t x_t + \widehat{\mathscr{Q}}_{t+1}\big(x_t, B_t(\xi_t)y_{t-1} + b_t(\xi_t)\big) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases}$$

where

$$\widehat{\mathscr{Q}}_{t+1}(x_t, y_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[\widehat{Q}_{t+1}(x_t, y_t, \boldsymbol{\xi}_{t+1}))\right]$$

and $\widehat{\mathscr{Q}}_{T+1}(x_T, y_T) \equiv 0$. For the first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & b_1 C_1 x_1 + \widehat{\mathscr{Q}}_2(x_1, y_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases}$$

The additional state y_{t-1} is referred to as an *objective state*. This state is not allowed to appear in the constraints [59]. As stated before, $\widehat{\mathscr{Q}}_t(\cdot, \cdot)$ is piecewise linear and convex in x_{t-1}, but piecewise linear concave in y_{t-1} and as such, a piecewise bilinear saddle function.

The concept of approximating saddle functions with saddle cuts goes back to Baucke et al., who propose a deterministic algorithm to solve stochastic minimax dynamic programs [11]. A related approach is used in *robust dual dynamic programming* (RDDP), which uses an SDDP-like framework to solve multistage robust programs [83]. The main idea is to compute lower and upper bounding saddle functions, which combine the ideas of an outer approximation by cutting-planes and an inner approximation by convex combinations of function values, the latter of which we discuss thoroughly in Section 8. For stagewise dependent objective coefficients, it is sufficient to only use the lower bounding saddle functions, so-called *saddle cuts*, from [11] to approximate the expected value functions in SDDP.

Let (3.4) define β_t and α_t as in standard SDDP. Then, in line 18 of Algorithm 3.1, the r-th saddle cut for $\widehat{\mathscr{Q}}_{t+1}(\cdot, \cdot)$ is defined as the solution to the optimization problem

$$\begin{aligned} \min_{\gamma_t,\, \theta_{t+1}} \quad & y_t^\top \gamma_t + \theta_{t+1} \\ \text{s.t.} \quad & (y_t^r)^\top \gamma_t + \theta_{t+1} \geq \alpha_{t+1}^r + (\beta_{t+1}^r)^\top x_t \\ & \|\gamma_t\|_\infty \leq \nu \end{aligned} \tag{14.17}$$

where $y_t^r = y_t^{ik}$ denotes the current objective state in iteration i and for scenario $k \in \mathcal{K}$. Importantly, this problem has x_t and y_t as parameters. Hence, a saddle cut gives a valid lower approximation for $\widehat{\mathscr{Q}}_{t+1}(\cdot, \cdot)$ for all x_t and y_t and can be shared between scenarios. Moreover, the saddle cuts are tight at the trial state given by x_t^{ik} and y_t^{ik}, at which they are created.

A crucial part of applying this approach is to bound the decision variable γ_t in (14.17) by an appropriate constant ν. To this end, the expected value functions $\widehat{\mathscr{Q}}_t(\cdot, \cdot)$ are required to be Lipschitz continuous with respect to y_{t-1}. As shown in [11], to ensure validity of the saddle cuts, the parameter ν has to be chosen at least

as large as the Lipschitz constant of $\widehat{\mathscr{Q}}_t(\cdot,\cdot)$ with respect to y_{t-1} under the dual norm $\|\cdot\|_1$ of $\|\cdot\|_\infty$. If it is chosen smaller, this may result in invalid cuts and suboptimal solutions. If it is chosen too large, the cuts may become very weak [59].

Incorporating the saddle cuts, for each stage $t = 2,\dots,T$, iteration i and scenario $k \in \mathcal{K}$, the SDDP subproblems can be formulated as

$$\begin{aligned}&\underline{\widehat{Q}}_t^i(x_{t-1}^{ik}, y_{t-1}^{ik}, \xi_{tj})\\ &= \begin{cases} \min\limits_{x_t,\gamma_t,\theta_{t+1}} & (y_t^{ik})^\top C_t x_t + (y_t^{ik})^\top \gamma_t + \theta_{t+1} \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_{tj}) \\ & (y_t^r)^\top \gamma_t + \theta_{t+1} - (\beta_{t+1}^r)^\top x_t \geq \alpha_{t+1}^r, \quad r \in \Gamma_{t+1} \\ & \|\gamma_t\|_\infty \leq \nu, \end{cases}\end{aligned}$$

where $y_t^{ik} = B_t(\xi_t^k) y_{t-1}^{ik} + b_t(\xi_t^k)$. Apart from considering the additional objective state y_t^{ik}, the forward pass remains of SDDP remains unchanged.

It can be shown that only finitely many different saddle cuts can be constructed. As a consequence, the convergence results are the same as for standard SDDP [59].

14.7. Applying Dual SDDP. A third alternative that is tailored to stagewise dependent objective coefficients $\boldsymbol{c}_t(\xi_t)$ in (MSLP) is to apply dual SDDP [104], as presented in Section 8. Recall the value functions derived from the dual problem of (MSLP):

$$\widetilde{D}_t(\pi_{t-1}) := \begin{cases} \max\limits_{\pi_t} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(- h_{tj}^\top \pi_{tj} + \widetilde{D}_{t+1}(\pi_{tj})\Big) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(T_{t-1,j}^\top \pi_{tj}\Big) + W_{t-1}^\top \pi_{t-1} \leq c_{t-1}. \end{cases} \tag{14.18}$$

These value functions are concave in π_{t-1}. Crucially, here the objective coefficients c_{t-1} appear in the RHS. If $(\boldsymbol{c}_t)_{t\in[T]}$ is described as a linear AR process, we can expand the state space as for the primal subproblems in Section 14.1, and the new state variable $c_{[t-2]}$ appears in the RHS. Therefore, the obtained value functions are also concave in $c_{[t-2]}$ and can be approximated from above by linear cuts. This can be done by applying dual SDDP [104], see Section 8.

14.8. Conditional Cuts. The previously discussed approaches all have in common that they require to expand the state space or to set up a scenario tree or a discrete Markov chain from the true (continuous) data process (or from existing historical data). van Ackooij and Warin propose an alternative approach that works without these requirements [235]. The approach is based on established methods in mathematical finance and optimal stopping theory. A crucial assumption is that the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is Markovian.

Assume that a finite set $\mathcal{S}$ of scenarios $\xi^s, s \in \mathcal{S}$, is given, *e.g.*, historical observations of the data. This set is chosen in advance and not changed within SDDP. The first key ingredient of the proposed variant of SDDP is to partition the set of possible values of $\boldsymbol{\xi}_t$ for each stage $t \in [T]$ into a finite number $|L_t|$ of hypercubes $D_{t\ell}$, $\ell = 1,\dots,|L_t|$, also called *meshes*. This partitioning is done in such a way that approximately a uniform distribution of the samples is achieved [235].

As we explain below, the main idea now is to compute cuts conditioned on specific meshes, *i.e.*, for each mesh a different set of cuts is considered.

In the forward pass of SDDP, a subset $L_t \subseteq \mathcal{S}_t$ of scenarios are sampled for each stage (see line 6 of Algorithm 3.1). This is done with the aim to obtain a trial solution x_t^ℓ for each mesh in expectation for all $t = 2, \dots, T$. Each of these trial solutions is then used in the backward pass to derive cuts.

In the backward pass, for any sequence $(x_t^{i\ell})_{t\in[T]}$ of trial solutions, let $\left(d(t)^{i\ell}\right)_{t\in[T]}$ denote the sequence of corresponding meshes, *i.e.*, $x_t^{i\ell}$ has been determined in the forward pass for $\xi_t^\ell \in D_{t,d(t)^{i\ell}}$. At each stage $t = T, \dots, 2$, the SDDP subproblems are now solved for all scenarios ξ_t^s for which $\xi_{t-1}^s \in D_{t-1,d(t-1)^{i\ell}}$. This means that for each trial solution, all scenarios are considered which share the same mesh with the scenario used to obtain the trial solution. Line 15 in Algorithm 3.1 is adapted accordingly.

After solving these subproblems, the obtained solutions are used to construct cuts. In contrast to standard SDDP, however, the cut coefficients are determined as estimates of the corresponding conditional expectations [235]:

$$\alpha_{t\ell}^i(\xi_{t-1}) = \hat{\mathbb{E}}^S_{|\xi_{t-1}}\left[(\boldsymbol{\pi}_t^{i\ell s})^\top \boldsymbol{h}_t(\xi_t) + \sum_{r\in\Gamma_{t+1}} \boldsymbol{\mu}_t^{i\ell sr}\alpha_{t+1}^r\right]$$

and

$$\beta_{t\ell}^i(\xi_{t-1}) = -\hat{\mathbb{E}}^S_{|\xi_{t-1}}\left[(\boldsymbol{\pi}_t^{i\ell s})^\top T_{t-1}\right].$$

These estimates are computed by linearly regressing the terms for each considered scenario ξ_t^s on a finite number of local base functions, *e.g.*, monomials in $\mathbb{R}^{p_t}$, with support on the considered mesh. Importantly, they are zero outside of this mesh. The idea is that this way a cut of form

$$\mathcal{Q}_t(x_{t-1}, \xi_{t-1}) \geq \phi_{t\ell}^i(x_{t-1}, \xi_{t-1}) = \left(\beta_{t\ell}^i(\xi_{t-1})\right)^\top x_{t-1} + \alpha_{t\ell}^i(\xi_{t-1}), \tag{14.19}$$

can be constructed, which provides a local update of the cut approximation in the current mesh $D_{t-1,d(t-1)}$ and is zero otherwise. Hence, the cut is associated with this specific mesh and stored in a corresponding index set. In following iterations of SDDP, for each subproblem then only the set of cuts is taken into account which is associated with the currently explored mesh [235]. Therefore, these cuts are called *conditional cuts*.

The main drawback of this approach is that the described cuts are not guaranteed to be valid underestimators, so the inequality in (14.19) is not guaranteed to be satisfied, because their formula relies on *estimators* that may deviate from the true conditional expectations.

Still, for problems with a low-dimensional vector $\boldsymbol{\xi}_t$ and Markovian dependency, the policies obtained using conditional cuts are reported to be competitive with those obtained by expanding the state space, but without an increase of the state dimension and without the need to set up a scenario tree [235].

15. Extension to Convex Programs [relaxing Assumption 6]. A natural extension of SDDP can be achieved by relaxing the assumption of linearity, *i.e.*, Assumption 6, but assuming a multistage stochastic convex problem (MSCP). In the

same vein as problem (2.3), this problem can be formulated in the general form

$$v_C^* := \begin{cases} \min\limits_{x_1, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}} & \mathbb{E}\left[\sum\limits_{t\in[T]} \boldsymbol{f}_t(\boldsymbol{x}_t(\xi_{[t]}), \xi_t)\right] \\ \text{s.t.} & g_1(x_1) \leq 0 \\ & g_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \boldsymbol{x}_t(\xi_{[t]}), \xi_t) \leq 0 \quad \forall \xi_{[t]}\ \forall t = 2, \ldots, T \\ & x_t \in X_t \quad \forall t \in [T] \\ & \boldsymbol{x}_t(\cdot)\ \mathscr{F}_t\text{-measurable} \quad \forall t \in [T], \end{cases} \tag{15.1}$$

with $\boldsymbol{f}_t(\cdot)$ and $\boldsymbol{g}_t(\cdot,\cdot)$ some $\mathscr{F}_t$-measurable functions with respect to $\boldsymbol{\xi}$.

We take the following assumptions [85, 92].

ASSUMPTION 10. *For fixed $\xi_t \in \Xi_t$, let $f_t(\cdot, \xi_t)$ and $g_t(\cdot,\cdot,\xi_t)$ (componentwise) be proper, convex, lower semicontinuous and differentiable functions and X_t nonempty convex compact sets for all $t \in [T]$.*

Under stagewise independence (Assumption 2), finite randomness (Assumption 5) and Assumption 10, (MSCP) in (15.1) can be expressed using its DPE in the following form. For $t = 2, \ldots, T$ they read

$$Q_{t,C}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & f_t(x_t, \xi_t) + \mathscr{Q}_{t+1,C}(x_t) \\ \text{s.t.} & g_t(x_{t-1}, x_t, \xi_t) \leq 0 \\ & x_t \in X_t, \end{cases} \tag{15.2}$$

with expected value functions defined as usual by

$$\mathscr{Q}_{t+1,C}(x_t) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1,C}(x_t, \boldsymbol{\xi}_{t+1})\right] \tag{15.3}$$

and $\mathscr{Q}_{T+1,C}(x_T) \equiv 0$. For the first stage, this yields

$$v_C^* = \begin{cases} \min\limits_{x_1} & f_1(x_1) + \mathscr{Q}_{2,C}(x_1) \\ \text{s.t.} & g_1(x_1) = 0 \\ & x_1 \in X_1. \end{cases} \tag{15.4}$$

Applying SDDP to (MSCP) with convergence guarantees requires a more strict recourse assumption compared to Assumption 9.

ASSUMPTION 11. *(Extended relatively complete recourse [85]) Let $\mathrm{aff}(\mathcal{X}_t)$ be the affine hull of the reachable set $\mathcal{X}_t$ and $B_t(\delta_t) = \{y \in \mathrm{aff}(\mathcal{X}_t) : \|y\| < \delta_t\}$ for some $\delta_t > 0$ and some norm $\|\cdot\|$.*

For all $t \in t = 2, \ldots, T$, all $x_{t-1} \in \mathcal{X}_{t-1} + B_t(\delta_t)$ and all $\xi_{tj}, j = 1, \ldots, q_t$, the feasible set of subproblems (15.2) is non-empty.

Intuitively, Assumption 11 demands that feasibility of the subproblems is also ensured for x_{t-1} slightly outside of $\mathcal{X}_t$. This is required in order to guarantee Lipschitz continuity of all value functions $Q_{t,C}(\cdot,\cdot)$ and expected value functions $\mathscr{Q}_{t,C}(\cdot)$ [85]. Additionally, all value functions are convex, and thus can be approximated by linear cuts. Such cuts can be generated using Lagrangian duality. More precisely, for all $t = 2, \ldots, T$, $x_{t-1} \in \mathcal{X}_{t-1}$ and some multipliers $\pi_t \in \mathbb{R}^{m_t}$ (with m_t the dimension of $g_t(\cdot,\cdot)$), we introduce the Lagrangian function

$$L_{t,C}(\pi_t; x_{t-1}, x_t, \xi_t) = f_t(x_t, \xi_t) + \pi_t^\top g_t(x_{t-1}, x_t, \xi_t), \tag{15.5}$$

the corresponding dual function

$$\mathcal{L}_{t,C}(\pi_t; x_{t-1}, \xi_t) = \min_{x_t \in X_t} L_t(\pi_t; x_{t-1}, x_t, \xi_t) \tag{15.6}$$

and the corresponding Lagrangian dual problem

$$\max_{\pi_t \geq 0} \mathcal{L}_t(\pi_t; x_{t-1}, \xi_t). \tag{15.7}$$

Further, we make the following assumption which ensures no duality gap between the primal subproblems (15.2) and their dual problems (15.7) [92]. Here, $\mathrm{ri}(S)$ denotes the relative interior of some set S.

ASSUMPTION 12. *(Slater condition [92]) For all $x_{t-1} \in \mathcal{X}_{t-1}$ and all $\xi_{tj}, j = 1, \ldots, q_t$, there exists $x_t \in ri(X_t)$ such that $g_t(x_{t-1}, x_t, \xi_{tj}) < 0$.*

Then, exploiting differentiability, a subgradient of $\mathscr{Q}_{t,C}(\cdot)$ at $\bar{x}_{t-1}$ is given by

$$\bar{\beta}_t = \partial \mathscr{Q}_t(\bar{x}_{t-1}) = \sum_{j=1}^{q_t} p_{tj} \nabla_{x_{t-1}} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}),$$

where $\bar{x}_{tj}$ is an optimal solution to the primal problem (15.2) and $\bar{\pi}_{tj}$ is an optimal solution to the dual problem (15.7) given ξ_{tj}. Moreover, $\nabla_x h(\cdot)$ denotes the gradient of some function $h(\cdot)$ with respect to x. Using this subgradient, a cut for $\mathscr{Q}_t(\cdot)$ is given by [92]

$$\mathscr{Q}_t(x_{t-1}) \geq \mathscr{Q}_t(\bar{x}_{t-1}) + \bar{\beta}_t^\top (x_{t-1} - \bar{x}_{t-1}). \tag{15.8}$$

Under Assumption 11, the norm of the obtained subgradients can be shown to be bounded [92].

Summarized, to adapt SDDP to the convex setting, in line 10 the convex subproblems (15.2) are solved, in line 16 of Algorithm 3.1 the dual problems (15.7) are solved and in line 18 cuts (15.8) are constructed.

The cut derivation can be generalized to DPE including $\underline{\mathcal{V}}_t(\cdot)$ instead of $\mathscr{Q}_t(\cdot)$. The results can also be generalized to cost functions $f_t(x_{t-1}, x_t, \xi_t)$ depending on the state x_{t-1}, see [92] for details.

Contrary to the linear case, however, the expected value functions $\mathscr{Q}_{t,C}(\cdot)$ are no longer polyhedral. As a consequence, they cannot be represented exactly by a finite number of cuts. However, it can be shown that given the above assumptions and Assumptions 1 to 8 almost sure asymptotic convergence of SDDP is ensured. In [85] this is proven for the case that x_{t-1} only enters the subproblems (15.2) in linear constraints, that is, $g_t(\cdot)$ being a linear function. In [92] the convergence proof is extended to the more general setting presented above. For both convergence proofs also the differentiability requirement can be dropped. As shown in [75], almost sure *finite* convergence can be achieved for ε-optimal policies, for some predefined $\varepsilon > 0$.

In [99], Guigues and Monteiro propose a slightly different algorithmic approach, called StoDCuP (*Stochastic Dynamic Cutting Plane*), in which not only $\mathscr{Q}_t(\cdot), t = 2, \ldots, T$, but also some or all nonlinear functions $f_t(\cdot)$ and $g_t(\cdot)$ are iteratively approximated by affine functions at the trial points visited in the forward pass.

Another variant of SDDP is DASC (*decomposition algorithm for multistage stochastic programs with strongly convex cost functions*), which is introduced in [93]. It can be applied when the (expected) value functions in (MSCP) are *strongly convex*. For this type of problems, it is proposed to approximate them using functions $\underline{\mathcal{V}}_t(\cdot)$

which are defined as the pointwise maximum of quadratic cuts instead of affine cuts. In contrast to standard SDDP, this means that the subproblems to be solved in SDDP become nonlinear, but in return good approximations of the expected value functions are obtained much quicker, and thus less iterations are expected [93]

While most research on SDDP deals with problems (MSLP), some of the extensions presented previously and in the following sections have also been enhanced to the convex case, *e.g.*, risk-aversion [92], inexact cuts [95], regularization [97] or exact upper bounding procedures [10, 126]. [92] contains an extension of the convergence proof from [85] to the risk-averse case. Furthermore, the idea to use inexact cuts is generalized to convex non-differentiable problems [98], see Section 21.

16. Extensions to Mixed-integer and Non-convex Problems [relaxing Assumption 6]. In many practical applications, multistage stochastic problems do involve integer decision variables or nonlinear, but non-convex terms in the objective function or constraints, see Section 9. In general, such programs can be formulated in the same way as in the convex case, but with the functions $f_t(\cdot)$ and $g_t(\cdot)$ possibly being non-convex. Moreover, in this case, X_t is the intersection of a convex compact set, *e.g.*, representing box constraints, with possible integer constraints, *i.e.*, $X_t \subset \mathbb{R}_t^{n_{t1}} \times \mathbb{Z}_+^{n_{t2}}$ with $n_t = n_{t1} + n_{t2}$. We denote the optimal value by v_{NC}^*.

Under stagewise independence (Assumption 2), the DPE can be written as (15.2)-(15.4), but for distinction we denote the value functions by $Q_{t,NC}(x_{t-1}, \xi_t)$ and the expected value functions by $\mathscr{Q}_{t,NC}(x_{t-1})$ for all $t = 2, \dots, T$. Both, integer variables and non-convex functions make this a non-convex multistage stochastic programming problem (MSNCP). Importantly, $Q_{t,NC}(\cdot, \cdot)$ and $\mathscr{Q}_{t,NC}(\cdot)$ are no longer ensured to be convex, but become non-convex functions in x_{t-1}. They are also not guaranteed to be (Lipschitz) continuous. This poses significant challenges on approximation algorithms such as SDDP, as linear cuts are not sufficient to approximate $\mathscr{Q}_{t,NC}(\cdot)$.

To approach (MSNCP) by SDDP, different strategies can be used. As nonlinear or mixed-integer stochastic programming are large research areas on their own, we give a brief overview here and for methodological details refer to the cited literature.

16.1. Convexification. A standard approach in practice is to solve a static convex relaxation ($\widehat{P}_{NC}$) of (MSNCP), which is associated with convex expected value functions $\widehat{\mathscr{Q}}_t(\cdot)$ for all $t \in [T]$. Such relaxation can be achieved by relaxing the integrality constraints and replacing non-convex functions with convex relaxations, such as McCormick envelopes [147]. In this case, the Benders cuts determined by SDDP can be very loose, though. Therefore, only some rough under-approximation $\widehat{v}_{NC}^*$ of the optimal value v_{NC}^* may be determined. However, sometimes this is considered sufficient to obtain reasonable policies for practical implementation. Also note that even if convex relaxations are considered when running SDDP to compute a policy, the simulation of this policy afterwards can be executed including integrality constraints and non-convex functions, see for instance [194].

A second strategy is to keep the subproblems in SDDP non-convex, but to convexify the expected value functions $\mathscr{Q}_{t,NC}(\cdot)$ *in some sense*. Often, in this case, the nonlinearities in (MSNCP) are first relaxed by piecewise linear approximations, such that all subproblems are MILPs [38, 227]. In the backward pass, more specifically, in line 16 of Algorithm 3.1, for all $t = T, \dots, 2$ and all $\xi_{tj}, j = 1, \dots, q_t$, instead of solving an LP relaxation of the subproblems (2.10) (or its LP dual), a Lagrangian relaxation is solved where the coupling constraints $g_t(x_{t-1}, x_t, \xi_{tj}) \leq 0$ are relaxed. For any trial

point x_{t-1}^{ik} and any multiplier $\pi_t \in \mathbb{R}^{m_t}$, this relaxation can be written as

$$\begin{aligned} \mathfrak{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}) := \min_{x_t} \quad & f_t(x_t, \xi_{tj}) + \mathfrak{Q}_{t+1}(x_t) + \pi_t^\top g_t(x_{t-1}^{ik}, x_t, \xi_{tj}) \\ \text{s.t.} \quad & x_t \in \mathcal{X}_t. \end{aligned}$$

In the Lagrangian dual, this dual function is maximized over all multipliers π_t:

$$v_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj}) := \max_{\pi_t \geq 0} \mathfrak{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}). \tag{16.1}$$

It is known from the theory on Lagrangian relaxation that the optimal value $v_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj})$ coincides with the lower convex envelope of $\underline{Q}_{t,NC}^{i+1}(\cdot, \xi_{tj})$ at x_{t-1}^{ik} [82]. Therefore, cuts obtained based on (16.1) are associated with a convexification of the value function. In order to derive utilizable cut formulas from (16.1) some specific conditions have to be satisfied by the constraints. Suppose the constraints $g_t(x_{t-1}, x_t, \xi_t) \leq 0$ can be rewritten as

$$\widehat{g}_t(x_{t-1}) - \bar{g}_t(x_t, \xi_t) \leq 0, \quad \tilde{g}_t(x_t, \xi_t) \leq 0,$$

i.e., the nonlinear function being separable with respect to x_{t-1}, and let π_t^{ikj} denote optimal multipliers in (16.1). Then, in line with Sect. 3.3, *Lagrangian cuts* can be derived as [221]

$$\mathscr{Q}_{t,NC}(x_{t-1}) \geq \alpha_{tk}^i + (\beta_{tk}^i)^\top \widehat{g}_t(x_{t-1}),$$

with

$$\begin{aligned} \alpha_{tk}^i &= \sum_{j=1}^{q_t} p_{tj} \Big(\mathfrak{L}_t(\pi_t^{ikj}; x_{t-1}^{ik}, \xi_{tj}) - (\pi_t^{ikj})^\top \widehat{g}_t(x_{t-1}^{ik}) \Big), \\ \beta_{tk}^i &= \sum_{j=1}^{q_t} p_{tj} \pi_t^{ikj}. \end{aligned}$$

For linear functions $\widehat{g}_t(\cdot)$ and $\bar{g}_t(\cdot, \cdot)$, a similar result is derived in [38].

The obtained Lagrangian cuts provably dominate standard Benders cuts, which can be obtained by solving LP relaxations [221]. However, convergence of SDDP is not guaranteed, since there may still exist a duality gap between $v_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj})$ and $\underline{Q}_{t,NC}^{i+1}(x_{t-1}^{ik}, \xi_{tj})$. Moreover, generating Lagrangian cuts can be computationally costly. Various methods have been proposed to solve the Lagrangian dual (16.1), such as cutting-plane methods [118], subgradient methods [73, 174] or bundle methods [129], but all of them may take considerable time compared to solving an LP relaxation. Advantageously, even suboptimal Lagrangian multipliers π_t yield valid cuts for $\mathscr{Q}_{t,NC}(\cdot)$.

Instead of a static convexification approach [38], Steeger and Rebennack [219, 221], also apply the above principle in a dynamic fashion by considering DPE for the Lagrangian relaxations in the backward pass.

16.2. Exact Methods. Recently, there has been more research on directly applying the SDDP idea to problems (MSNCP) to avoid the requirement of convexification and to close the optimality gap.

Step Functions. Given that the value functions $Q_{t,NC}(\cdot)$ are monotonically increasing or decreasing, they can be approximated by special step functions instead

of affine functions. This idea is incorporated into the SDDP framework in the so-called *mixed-integer dynamic approximation scheme* (MIDAS) [170]. To determine the step functions, mixed-integer linear subproblems have to be solved exactly at each stage and in each iteration. In contrast to the previous approaches, convergence of MIDAS to an approximately optimal policy for (MSNCP) is guaranteed.

SDDiP. For the mixed-integer *linear* case, the *stochastic dual dynamic integer programming* (SDDiP) approach by Zou, Ahmed and Sun [244] allows for the computation of optimal policies for (MSNCP) as long as all state variables x_t are binary (or bounded integer).

Consider the subproblems (2.10), but with binary state variables $x_t \in \{0,1\}^{n_t}$. Similarly to the approaches in [38, 221, 227], Lagrangian dual problems are solved in the backward pass (line 16 of Algorithm 3.1) to derive valid cuts. However, in SDDiP a new class of Lagrangian cuts is proposed. The crucial idea is to introduce local copies z_t of the state variables x_{t-1} and to relax the corresponding copy constraints in the Lagrangian relaxation:

$$\begin{aligned} \mathcal{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}) := \min_{x_t, z_t} \quad & \big(c_t(\xi_{tj})\big)^\top x_t + \mathfrak{Q}_{t+1}(x_t) + \pi_t^\top (x_{t-1}^{ik} - z_t) \\ \text{s.t.} \quad & x_t \in \mathcal{X}_t(z_t, \xi_t) \\ & z_t \in [0,1]^{d_{a(n)}}. \end{aligned}$$

In the Lagrangian dual, this dual function is maximized over all multipliers π_t:

$$\tilde{v}_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj}) := \max_{\pi_t} \ \mathcal{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}).$$

Then, in line 18 of Algorithm 3.1, Lagrangian cuts can be determined as

$$\mathscr{Q}_{t,NC}(x_{t-1}) \geq \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1}, \tag{16.2}$$

with

$$\alpha_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \Big(\mathcal{L}_t(\pi_t^{ikj}; x_{t-1}^{ik}, \xi_{tj}) - (\pi_t^{ikj})^\top x_{t-1}^{ik} \Big),$$

$$\beta_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \pi_t^{ikj}.$$

These cuts can be proven to be valid and, in particular, tight, as defined in Lemma 3.3. The key aspect behind this tightness property is that for $x_{t-1} \in \{0,1\}^{n_t}$ the value functions $Q_{t,NC}(\cdot)$ coincide with their lower convex envelopes at all x_{t-1}. Therefore, Lagrangian cuts recovering the latter are also tight for the former.

Moreover, if only dual basic solutions are considered, the cuts (16.2) are also finite in the sense of Lemma 3.3. Therefore, almost sure finite convergence of SDDiP to an optimal policy of (MSNCP) is guaranteed [244].

If the state variables x_t are bounded and general integer or even continuous, they can be componentwise approximated by a (weighted) sum of binary variables in order to apply SDDiP [244]:

$$x_{tj} \approx \sum_{k=1}^{K_{tj}} 2^{k-1} \beta_{tj} \lambda_{tkj},$$

with discretization precision β_{ti} (for integer x_t, $\beta_t = 1$), binary variables $\lambda_{tkj}, k = 1, \dots, K_{tj}$, and $K_{tj} \in \mathbb{N}$ for all $j = 1, \dots, n_t$. Under some recourse assumptions, it can be proven that for a sufficiently fine binary expansion, an approximately optimal policy for (MSNCP) is computed. However, it may be challenging to choose an appropriate precision in advance in practice.

A more generalized framework to generate Lagrangian cuts in SDDiP is presented in [78] based on ideas from [40]. Additionally, the theory behind Lagrangian cuts and their properties is explored in more detail in [79].

SDDiP is applied in the case studies [110], [182] and [244]. In the latter, additional non-convex functions occur in (MSNCP), which are linearized.

Non-convex Lipschitz cuts. As long as the value functions are assured to be Lipschitz continuous (*e.g.* because the *complete continuous recourse* [244] property is satisfied), the requirement of binary state variables can be dropped. This is exploited in the *stochastic Lipschitz dynamic programming* (SLDP) method proposed by Ahmed et al. in [1], which enhances SDDiP to general MILPs. In contrast to the Lagrangian cuts (16.2), here, two types of non-convex, but Lipschitz continuous cuts are derived to approximate $\mathscr{Q}_{t,NC}(\cdot)$: Reverse-norm cuts, which are constructed by using Lipschitz constants, and augmented Lagrangian cuts, which are based on (16.2), but contain an additional penalization term $-\mu\|x_{t-1} - x^i_{t-1}\|$, where μ denotes some user-controlled parameter and $\|\cdot\|$ some arbitrary norm.

This idea is further refined by Zhang and Sun in [241] who propose a new framework to solve multistage non-convex stochastic MINLPs as part of their complexity analysis of SDDP-like algorithms, see Sect. 4. The first key ingredient of their framework is to consider Lipschitz regularizations of the value functions, see Sect. 17.2. This ensures that the considered value functions are Lipschitz continuous without the requirement of restricting recourse assumptions for (MSNCP). The second idea is to construct nonlinear *generalized conjugacy cuts* by solving conjugate dual problems, similar to the approach in SLDP. Whereas of theoretical interest, this method has not been applied in computational experiments yet. In particular, it is not clear how to solve the conjugate dual problems efficiently in general. Moreover, the framework requires the costly solution of MINLP subproblems in each iteration.

Based on concepts from SDDiP and [241], Füllner and Rebennack present a new framework to solve multistage (stochastic) non-convex MINLPs [77]. Here, the original MINLP is outer approximated by MILPs using piecewise linear relaxations, which are iteratively improved in an outer loop. In an inner loop, those MILPs are solved by an SDDP- and NBD-like decomposition scheme, which combines the Lipschitz regularization approach from [241] with binary approximation to generate non-convex cuts. In contrast to SDDiP, the binary approximation is applied only temporarily to derive linear cuts in the lifted binary space, which are then projected back to the original state space. The pointwise maximum of this projection yields a Lipschitz continuous non-convex cut for the value functions. The projection is computationally important, as it allows to construct cuts which are guaranteed to be valid also for the outer loop MINLPs. The binary approximation is dynamically refined within the algorithm, instead of a static choice in advance. Another key difference compared to the approach from [241] is that it is not required to solve MINLPs in each iteration to derive cuts. The cut projection closure for a non-convex and discontinuous value function is illustrated in Figure 14.

Similar to SLDP [1], however, it is required to introduce a potentially large number of auxiliary variables and constraints to express the non-convex approximations by mixed-integer linear constraints. While the framework in [77] is presented for de-

terministic problems, the inner loop decomposition method can be enhanced to the stochastic case [79]. Therefore, by appropriate modifications of the refinement and stopping criteria, also the larger framework may be enhanced to stochastic problems.

Finally, the use of more general, possibly non-convex *dual functions* to approximate $\mathscr{Q}_t(\cdot)$ is proposed in [30].

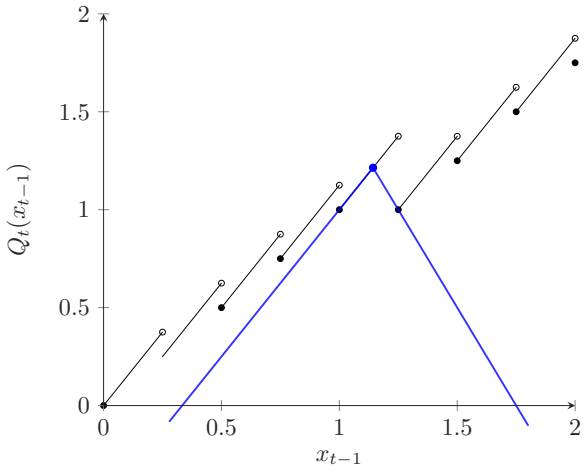


Fig. 14: Non-convex and discontinuous value function with tight non-convex cut.

Scaled cuts. A recently proposed hybrid between using classical linear cuts and non-convex cuts is the usage of so-called *scaled cuts* [236]. These cuts are linear, thus computationally preferrable, but derived by scaling non-convex approximations in such a way that they are guaranteed to asymptotically recover $\overline{\text{co}}(()(\cdot)\mathscr{Q}_t(\cdot))$. This is sufficient to guarantee convergence for mixed-integer linear programs. However, even if linear, so far scaled cuts have only been applied for two-stage stochastic programs, as computing them comes with a lot of additional effort.

17. Infeasible Subproblems [relaxing Assumption 9]. Under relatively complete recourse (see Assumption 9), it is guaranteed that any subproblem occurring in the DPE (2.4)-(2.6) and their approximations (2.10) has a feasible solution. As we also assume boundedness, for each of these subproblems there exists some optimal point with finite optimal value. Moreover, all value functions are finite-valued.

In some practical applications, Assumption 9 may not be satisfied. For instance, variable bounds may prevent equality constraints from being satisfied for all x_{t-1} and all realizations of ξ_t, as is illustrated by a toy example in [91]. In such a case, the primal subproblems become infeasible and the corresponding dual problems become unbounded. Different measures can be taken to cope with infeasibilities.

17.1. Feasibility Cuts. One approach is to approximate the effective domains $\text{dom}(\mathscr{Q}_t)$ of $\mathscr{Q}_t(\cdot)$ by cutting away states $x_{t-1}^{ik} \in \mathcal{X}_t$ leading to infeasible subproblems on stage t. This can be achieved by generating so called *feasibility cuts* in addition to the *optimality cuts* derived in Section 3. These cuts have the form $(\beta_t^f)^\top x_{t-1} \leq \alpha_t^f$, with cut gradient β_t^f, cut intercept α_t^f and the superscript f signifying the cut as a feasibility cut. They can be derived as follows [91].

Consider some stage-t subproblem

$$\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k) = \begin{cases} \min\limits_{x_t} & \left(c_t(\xi_t^k)\right)^\top x_t + \underline{\mathcal{V}}_{t+1}^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_t^k) \\ & (\beta_{t+1}^{fr})^\top x_t \le \alpha_{t+1}^{fr}, \quad r \in \Gamma_{t+1}^f \end{cases} \tag{17.1}$$

in the forward pass of SDDP. This problem may already contain some feasibility cuts, which are indexed by $r \in \Gamma_{t+1}^f$. To assess feasiblity of problem (17.1) and construct a feasibility cut if required, we consider the auxiliary feasibility problem

$$v_t^f(x_{t-1}^{ik}, \xi_t^k) := \begin{cases} \min\limits_{x_t, y_t^+, y_t^-, z_t} & e^\top y_t^+ + e^\top y_t^- + e^\top z_t \\ \text{s.t.} & W_t(\xi_t^k)x_t + Iy_t^+ - Iy_t^- = h_t(\xi_t^k) - T_{t-1}(\xi_t^k)x_{t-1}^{ik} \quad (\sigma_t) \\ & (\beta_{t+1}^{fr})^\top x_t + Iz_t \le \alpha_{t+1}^{fr}, \quad r \in \Gamma_{t+1}^f \quad (\omega_t^r) \\ & x_t \ge 0 \\ & y_t^+, y_t^-, z_t \ge 0. \end{cases}$$

Here, slack variables y_t^+, y_t^- and z_t are introduced to (17.1) to ensure feasibility. The symbol I denotes the identity matrix and e denotes a vector of ones. If we have $v_t^f(x_{t-1}^{ik}, \xi_t^k) = 0$, the subproblem (17.1) is feasible, otherwise, it is infeasible.

By strong duality of linear programs, $v_t^f(x_{t-1}^{ik}, \xi_t^k)$ can be expressed as

$$v_t^f(x_{t-1}^{ik}, \xi_t^k) = \left(h_t(\xi_t^k) - T_{t-1}(\xi_t^k)x_{t-1}^{ik}\right)^\top \sigma_t + \sum_{r \in R_{t+1}^f} (\alpha_{t+1}^{fr})^\top \omega_t^r \tag{17.2}$$

with optimal dual vectors σ_t^{ik} and $\omega_t^{ikr}, r \in R_{t+1}^f$. Then, in case of infeasibility it follows that the term in (17.2) is larger than 0.

To avoid the observed infeasibility on stage t in future iterations, the stage-$(t-1)$ trial point x_{t-1}^{ik} should removed from the feasible set on stage $t-1$. This can be achieved by adding the feasibility cut

$$-(\sigma_t^{ik})^\top T_{t-1}(\xi_t^k)x_{t-1} + (\sigma_t^{ik})^\top h_t(\xi_t^k) + \sum_{r \in R_{t+1}^f} (\omega_t^{ikr})^\top \alpha_{t+1}^{fr} \le 0 \tag{17.3}$$

to stage $t-1$. By defining

$$\alpha_{t-1}^f := -(\sigma_t^{ik})^\top h_t(\xi_t^k) - \sum_{r \in R_{t+1}^f} (\omega_t^{ikr})^\top \alpha_{t+1}^{fr}$$

and

$$\beta_{t-1}^f := -(\sigma_t^{ik})^\top T_{t-1}(\xi_t^k),$$

the cut (17.3) can be expressed in the previously stated form.

An important question when using feasibility cuts in SDDP is how to proceed, once an infeasible subproblem has been detected and a new feasibility cut (17.3) has been generated. For example, it is possible to stop the forward pass and traverse the stages in backward direction until the root node of the scenario tree is reached. Alternatively, the current subproblem can be resolved to obtain a new trial point x_{t-1}^{ik} and the forward pass can be continued. For SDDP, no assessment and comparison of these strategies has been conducted so far.

Another drawback is that feasibility cuts do not necessarily prevent infeasibilities when the obtained policy is simulated outside of SDDP [91]. For this reason, most commonly, the construction of feasibility cuts is circumvented in SDDP.

17.2. Penalization. Another common approach is to artificially enforce relatively complete recourse for a problem at hand, even if it is not satisfied initially. This can be achieved by using *soft-constraints*, that is, introducing slack variables to relax certain constraints and then penalizing their violation in the objective function. In some applications, this may even be practically justifiable, *e.g.*, in load balance equations in power optimization slack variables can be used to model load shedding or curtailment. However, a reasonable choice of the penalty parameters is not trivial and may distort the expected value functions [91].

Lipschitz Regularization. A specific penalization approach is to consider *Lipschitz regularizations*, also called *Pasch-Hausdorff envelopes* of the value functions. More precisely, let $\|\cdot\|$ denote some norm, $\sigma_t > 0$ some constant and z_t a local stage-t copy of x_{t-1}. Then, by allowing z_t to deviate from the incumbent x_{t-1}^{ik} and penalizing such deviations in the objective, for all $t = 2, \dots, T$ and the approximate value functions (2.10) we obtain the approximate Lipschitz-regularized value functions

$$\underline{Q}_t^{R;i+1}(x_{t-1}^{ik}, \xi_t; \|\cdot\|) := \min_{z_t \geq 0} \left\{ \underline{Q}_t^{i+1}(z_t, \xi_t) + \sigma_t \|z_t - x_{t-1}^{ik}\| \right\}. \tag{17.4}$$

These functions are proven to be Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$ with Lipschitz constant σ_t. Moreover, for sufficiently large σ_t for all $t \in [T]$, it can be shown that by considering the regularized problems still the original (MSLP) is solved to optimality [72, 241]. However, choosing σ_t in a sufficient way is an open challenge in practice. If σ_t is chosen too small, it may even happen that the Lipschitz-regularized value function will be constant $-\infty$, given that the subproblem associated with $\underline{Q}_t^{i+1}(\cdot, \xi_t)$ for some fixed ξ_t is unbounded for any z_t.

When Lipschitz-regularized subproblems (17.4) are solved in the forward pass of SDDP (line 10 of Algorithm 3.1), then subgradients of $\underline{Q}_t^{R;i+1}(\cdot; \|\cdot\|)$ can be derived in the backward pass (line 18) by solving a dual problem (line 16), where the dual variables are bounded by σ_t in the dual norm $\|\cdot\|^*$ to $\|\cdot\|$ [241].

18. No Block-diagonal Structure [relaxing Assumption 7]. A key element of dynamic programming methods is that in the multistage decision process only subsequent stages are linked in the constraints, as it allows one to express (MSLP) using the DPE (2.4)-(2.6). In the single-problem formulation (2.3) of (MSLP), this coincides with a block-diagonal structure, see Assumption 7.

In some cases, it may be relevant to include constraints spanning multiple stages instead. One example is the incorporation of emission quotas that are not allowed to be exceeded for a given time horizon in energy optimization problems [14, 185, 187].

In order to apply SDDP, the considered (MSLP) has to be reformulated to a problem satisfying Assumption 7. This can be achieved by aggregating stages [58], even though this changes the structure, solution and interpretability of (MSLP). An alternative approach is augmenting the state space. For emission quotas, for instance, instead of summing emissions over several stages and comparing them with the upper bound, at a given stage the remaining emission allowances can be considered as an additional state variable [185], see Section 9.

19. Infinite Horizon [relaxing Assumption 1]. So far, we considered problems (MSLP) with a finite time horizon $T < \infty$ (Assumption 1). In some practical

applications, however, repeated decisions have to be modeled without a clear bound on the horizon. Considering such infinite-horizon problems is for instance common for Markov decision processes [19]. In such a case, to ensure that v^* is finite, a geometric discount factor $\delta < 1$ is introduced for the cost at each stage.

Since SDDP performs a forward and a backward pass through all stages in each iteration, it is not directly applicable to such problems, as no iteration would ever be completed. Therefore, often different solution methods are utilized in such a setting, see for example [9]. Still, recently there has been some focus on enhancing the SDDP idea to problems with infinite time horizon.

One approach, called Benders squared or B^2, is based on limiting each iteration of SDDP to a finite horizon of τ stages, but to dynamically increase τ per iteration, *e.g.*, by 1, until convergence is reached [152]. By presuming that the uncertainty occurs in the RHS and is not only stagewise independent, but also i.i.d. for all stages $t \in [T]$, almost sure convergence to an approximately optimal policy is assured. The reason is that under this special assumption, $\mathscr{Q}_t(\cdot)$ are the same for all stages, so cuts computed at stage t cannot only be incorporated at stage $t-1$, but at *all* stages [152].

A different option to adapt SDDP to infinite horizon problems exists if such problems possess some kind of periodical behavior. This idea is put forward by Shapiro and Ding [211]. Assume that for some period $m \in \mathbb{N}$, the distributions of ξ_t as well as the data c_t, W_t, h_t and T_{t-1} are the same for $t = \tau$ and $t = \tau + m$ for all $\tau = 2, \dots$ Then, under Assumption 9, the functions $\mathscr{Q}_t(\cdot)$ and $\mathscr{Q}_{t+m}(\cdot)$ are equivalent as well. This means that it is sufficient to derive cuts for $\mathscr{Q}_{t+m}(\cdot)$ at stages $t = 2, \dots, m+1$ in order to obtain valid cuts for all stages.

In the forward pass of SDDP, it is proposed to only consider a finite number of T stages starting from stage 1 (see line 8 of Algorithm 3.1), with $T \geq m+1$ in order to determine at least one trial point for each of the differing expected value functions. In the case of $T > m+1$, multiple candidate trial points exist, at which cuts can be constructed in the backward pass. Before starting the backward pass, the used trial points can be chosen from such a candidate set randomly or by some heuristic.

For both approaches, B^2 and periodic SDDP, for discount factors δ close to 1, the influence of late stages on v^* may be substantial, and thus policy evaluation and upper bound determination may become very challenging and computationally costly. Still, Shapiro and Ding [211] propose some proxies based on some finite, but sufficiently large T. However, they do not provide a convergence proof.

A big advantage of SDDP for periodical problems is that it can also be applied to increase the performance for problems with a finite, but very large number of stages, given that they satisfy some notion of periodicity. The authors present an example where for a 60-month horizon, exploiting the periodical structure of the problem, instead of a 60-stage problem only a 13-stage problem has to be solved [211]. This can make even large problems amenable to SDDP and computationally tractable. It is also considered to mitigate the so-called *end-of-horizon effect*, which we discuss in Section 9.

On a different note, the policy graph approach introduced by Dowson [60] to model (MSLP) provides a natural extension to infinite-horizon problems, as it allows for cyclic graphs. Solving such problems, similarly to [152], relies on a discount factor and a truncation after a finite number of nodes in the graph. Then, approximate convergence can be proven.

20. Random Horizon [relaxing Assumption 1]. Another way to relax Assumption 1 is to assume that the horizon T is random. For simplicity, we discuss this

aspect for the linear case only, even though it is presented in [96] for the convex case.

Consider (MSLP) from Sect. 2.3, satisfying Assumptions 2 to 8, but with T not being fixed. Instead, we take the following assumption:

ASSUMPTION 13. *The time horizon T is a discrete random variable taking values in $\{2, \dots, \overline{T}\}$ with known $\overline{T} \in \mathbb{N}$.*

Then, the horizon T induces the Bernoulli process $(\boldsymbol{D}_t)_{t \in [\overline{T}]}$ with realizations

$$D_t = \begin{cases} 0, & \text{if the optimization period ended at } t \text{ or before} \\ 1, & \text{otherwise,} \end{cases}$$

and therefore T can be written as

$$T = \min \left\{ t \in [1, \overline{T}] \ : \ D_t = 0 \right\}.$$

Under stagewise independence (Assumption 2), the decisions $\boldsymbol{x}_t(\cdot)$ are functions of $\boldsymbol{\xi}_t$, $\boldsymbol{D}_t$, $\boldsymbol{D}_{t-1}$ and x_{t-1} (see $x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)$). In other words, $\boldsymbol{x}_t$ is $\overline{\mathscr{F}}_t$-measurable with $\overline{\mathscr{F}}_t$ the sigma-algebra $\sigma(\boldsymbol{\xi}_t, \boldsymbol{D}_j, j \leq t)$ [96].

As shown in [96], for (MSLP) with this type of random horizon, the following DPE equations can be derived. Importantly, the state space is augmented by D_{t-1}:

$$Q_t(x_{t-1}, D_t, D_{t-1}, \xi_t) = \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} D_{t-1} \big(c_t(\xi_t)\big)^\top x_t + \mathscr{Q}_{t+1}(x_t, D_t),$$

where

$$\mathscr{Q}_{t+1}(x_t, D_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}, \boldsymbol{D}_t | D_{t-1}} \left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1}) \right]$$

and $\mathscr{Q}_{\overline{T}+1}(x_{\overline{T}}, D_{\overline{T}}) \equiv 0$. For the first stage, we obtain

$$v^* = \min_{x_1 \in \mathcal{X}_1(x_0, \xi_1)} c_1^\top x_1 + \mathscr{Q}_2(x_1, D_1).$$

These DPE are the same as those that would be obtained for a problem with a fixed number of stages $\overline{T} \in \mathbb{N}$, but an objective function including the stagewise dependent stochastic process $(\boldsymbol{D}_t)_{t \in [\overline{T}]}$. As $(\boldsymbol{D}_t)_{t \in [\overline{T}]}$ can be modeled by an inhomogeneous Markov chain with two states, SDDP for processes with Markov chains can be applied [96], see Section 14.4.

21. Performance Improvements. Apart from extensions to different problem classes, a lot of research on SDDP has focused on improving its computational performance, because standard SDDP may suffer from various performance issues.

As shown in Section 4.2, its worst-case iteration complexity is exponential in the number of stages T and the dimension n_t of the state space, the latter being a well-known drawback of cutting-plane methods in general. Whereas SDDP is successfully applied to various large-scale problems in practice, see Section 9, with the optimality gap closed in reasonable time, especially for problems with a large state space it may empirically fail to converge. For instance, Ávila et al. [6] report instances for which the lower bounds $\underline{v}^i$ already start to stall at a gap of about 22%.

In addition to the high number of iterations required, also the computational effort in each iteration can become substantial, even if the number of subproblems solved per iteration has linear complexity, see Section 4.2. The reason is that with each iteration of SDDP, the subproblems (2.10) become larger, as additional cuts

are included. This can increase the computational effort per iteration significantly, especially if many iterations are required until convergence is achieved.

In this section, we give an overview on modifications of SDDP to address these issues and improve its performance.

21.1. Speeding up SDDP Iterations. We start with techniques attempting to speed up the SDDP iterations by reducing the computational effort.

21.1.1. Cut Elimination and Selection. As mentioned before, with each added cut, the subproblems (2.10) become larger, and thus potentially harder to solve. However, computational results indicate that SDDP tends to generate a large number of similar or redundant cutting planes, which do not contribute much to the approximation quality in later iterations [6, 213]. Therefore, the computational burden of SDDP may be reduced if only a subset of all cuts is taken into account. However, this requires careful elimination of cuts which are dominated and do not contribute to the solution process, or careful selection of decisive cuts, as otherwise the performance of SDDP may even become worse.

Cut Elimination. One way to reduce the number of cuts is to eliminate some cuts permanently. This can be done by repeatedly solving an auxiliary problem after a specified number of iterations, checking feasibility of the system

$$\begin{cases} \theta_{t+1} \leq \alpha_{t+1}^{\tilde{r}} + (\beta_{t+1}^{\tilde{r}})^\top x_t \\ \theta_{t+1} \geq \alpha_{t+1}^{r} + (\beta_{t+1}^{r})^\top x_t, \quad r \in \Gamma_{t+1} \setminus \{\tilde{r}\} \\ x_t \in X_t \end{cases}$$

for each $\tilde{r} \in \Gamma_{t+1}$, where X_t is assumed to be a compact set [213]. If this system is infeasible, then the cut $\theta_{t+1} \geq \alpha_{t+1}^{\tilde{r}} + (\beta_{t+1}^{\tilde{r}})^\top x_t$ is redundant and can be eliminated. The drawback of this method is that the auxiliary problem has to be solved for all cuts in the system.

A different approach is to permanently store all cuts for each stage t, but only select a subset of those cuts to be considered in each iteration i. Selection techniques based on this approach are introduced in [8, 53].

Selecting Last Cuts. In this naive strategy, only the $\Gamma \in \mathbb{N}$ most recently added cuts are selected. Although on average, late cuts may provide a better approximation of $\mathscr{Q}_t(\cdot)$ than early ones, this strategy does not guarantee that all important cuts are considered.

Level of Dominance. This strategy is a heuristic in order to consider only non-dominated cuts, but avoid the computational effort of the above cut elimination approach. Using the most basic approach, only cuts are selected, which yield the highest function value at one of the trial solutions considered so far within the algorithm. This is called *Level 1 Dominance* [53]. A similar approach is proposed in [163], but there cuts are permanently removed if they are dominated.

Let x_t^ℓ be the trial solution corresponding to the ℓ-th cut, $\ell \in \Gamma_{t+1}$, and $\phi^r(x_t^\ell)$ the corresponding function value of cut r. Then, the values $v(\ell) := \max_{r \in \Gamma_{t+1}} \left\{\phi^r(x_t^\ell)\right\}$ and $r(\ell) := \arg\max_{r \in \Gamma_{t+1}} \left\{\phi^r(x_t^\ell)\right\}$ can be saved in a list and be updated every time a new cut is constructed. Similarly, a *Level H Dominance* strategy can be used, selecting the $H \in \mathbb{N}$ highest cuts for all trial solutions. Using this strategy, only previous trial points are taken into consideration, though. Therefore, cuts may be excluded which provide a significant benefit at not yet visited feasible states.

Another challenge is that this strategy draws a lot of resources to store all the required cut information – especially, since the number of visited trial points increases

significantly in the course of SDDP. Memory requirements can even be relevant for Level 1, especially if the maximum function value at the trial solutions is attained by several cuts. As a resort, in [94], the *Limited Memory Level 1* strategy is introduced, selecting only the oldest of such cuts. In [8] a more general cut selection strategy is applied to SDDP and almost sure convergence is proven.

Dynamic Cut Selection. A dynamic, but also computationally more expensive strategy is to select cuts dynamically within the SDDP framework. In [53] it is proposed to remove all cuts at the beginning of each iteration. Then, for each stage t, each scenario k, and each function $\phi^r(\cdot), r \in \Gamma_{t+1}$, the forward pass subproblem (2.10) is solved. If the current cut yields the highest value at the obtained trial solution, it is added to the subproblem, and the next cut is considered. This way, only those cuts are selected that contribute to the optimal solution in the current iteration. On the other hand, the additional loop in the forward pass may slow down the convergence speed. The computational effort can be reduced by some additional heuristics [53].

A similar approach is considered in [33]. Here, cuts are iteratively added as long as they induce a substantial change in the current optimal value and up to a predefined maximum number of cuts. Instead of iterating over all cuts, in each step, the cut with the highest value at the current trial point is chosen as a candidate for selection.

Numerical results for sampling about 5,000 scenarios and computing 10,000 cuts in SDDP indicate that all cut selection techniques can significantly speed-up the classical SDDP method [53]. For example, the Level 1 strategy is reported to be ten times faster than SDDP without cut selection. For dynamic cut selection, the reported speed-up is much smaller. It is also shown that the cut selection strategies do not have a significant impact on the quality of the determined policies and bounds. In [8], Limited Memory Level 1 is identified as more efficient than pure Level 1.

21.1.2. Sampling Schemes. SDDP allows to use a variety of different sampling schemes which affect its computational performance.

Number of Forward Samples per Iteration. In standard SDDP, see Section 3, $|\mathcal{K}|$ scenarios are sampled in each iteration, with $|\mathcal{K}| \ll |\mathcal{S}|$ and $\mathcal{K} \subset \mathcal{S}$ (line 6 of Algorithm 3.1). Philpott and Guan even propose a method with only $|\mathcal{K}| = 1$ for all iterations [171]. This strategy may be particularly efficient in earlier iterations in order to obtain a rough approximation of $\mathscr{Q}_t(\cdot)$ fast without wasting too much effort in regions which are likely to be far from optimal. On the other hand, if the current policy is already reasonably good, it should be beneficial to generate more than one new cut per stage and iteration [53]. Moreover, if $|\mathcal{K}| = 1$, then it is not possible to apply a statistical stopping criterion, see Section 7.

Therefore, instead of fixing $|\mathcal{K}|$, a *scenario incrementation* strategy in which $|\mathcal{K}|$ is gradually increased is a promising approach [212]. It is tested in [53].

Subsampling Trial Points. In the *reduced sampling method* (ReSa) [109] the forward pass follows the same principle as in SDDP by sampling scenarios $\xi_t^k, k \in \mathcal{K}$, for $\mathcal{K} \subset \mathcal{S}$. In the backward pass, however, to reduce the number of subproblems to be solved, only a subsample $\tilde{\mathcal{K}} \subset \mathcal{K}$ is considered and only $|\tilde{\mathcal{K}}|$ cuts are generated (see line 14 of Algorithm 3.1). Considering more samples in the forward pass without additional effort in the backward pass is helpful to compute accurate statistical upper bounds in an efficient way.

A similar approach is applied in the *abridged nested decomposition* (AND) method [57]. It is claimed that SDDP is not well-designed for bushier scenario trees with large values q_t because solving $|\mathcal{K}|q_t$ subproblems per stage may quickly become computationally costly, especially for large $|\mathcal{K}|$. On the other hand, a large $|\mathcal{K}|$ may be required

to get reliable statistical upper bounds and to incorporate information on sufficiently many scenarios in the trajectories $(x_t^{ik})_{k\in\mathcal{K}}$. As a remedy, an alternative sampling scheme is proposed. In the forward pass, on each stage $t \in [T]$ a set $\mathcal{K}_t$ of realizations is sampled and trial points x_t^{ik} are computed. On stage $t+1$, however, only a few *branching values* are used as parameters (in the forward and backward pass), which can either be sampled from or be a convex combination of all $x_t^{ik}, k \in \mathcal{K}_t$. The latter idea allows to compute trial points which contain information on a large set of scenarios, while keeping the computational effort in the backward pass at a minimum. The main drawback of AND is that the special structure of the forward pass allows no direct estimate of an upper bound [109].

Sampling in the Cut Generation Process. The computational effort in the backward pass can be reduced if the subproblems (2.10) are not solved for all noise terms ξ_{tj}, $j = 1, \dots, q_t$, in each iteration, but only for a subset (see line 15 of Algorithm 3.1). The remaining elements that are required to compute a valid cut can then be used from previous iterations where the corresponding noise ξ_{tj} was sampled.

Even more, if the uncertainty is restricted to the RHS h_t of (MSLP), then the dual feasible set does not depend on it. Therefore, optimal dual multipliers which correspond to dual extreme points can be re-used between different realizations $j = 1, \dots, q_t$. This allows for the following procedure: Assume that in each iteration i, for each stage $t \in [T]$ only one noise term $\hat{\xi}_t^i$ is sampled and used to compute optimal dual multipliers $\hat{\pi}_t^i$ and (scenario-specific) cut intercepts $\hat{\alpha}_t^i$ as in (21.2.1). For each stage $t = 2, \dots, T$, all dual multipliers and intercepts obtained up to iteration i are then stored in a set $\mathcal{D}_t^i$ defined by $\mathcal{D}_t^i = \mathcal{D}_t^{i-1} \cup \left\{ \left(\hat{\pi}_t^i, \hat{\alpha}_t^i, \hat{\xi}_t^i \right) \right\}$.

For any $\xi_{tj}, j = 1, \dots, q_t$, and a given incumbent x_{t-1}^i, the dual multipliers used to compute a new cut in line 19 of Algorithm 3.1 can then be determined as

$$\left(\hat{\pi}_t^{ij}, \hat{\alpha}_t^{ij}, \hat{\xi}_t^{ij} \right) = \underset{(\hat{\pi}_t, \hat{\alpha}_t, \hat{\xi}_t) \in \mathcal{D}_t^i}{\arg\max} \left\{ \hat{\alpha}_t - \hat{\pi}_t^\top T_{t-1} x_{t-1}^i + \hat{\pi}_t^\top \left(h_t(\xi_{tj}) - h_t(\hat{\xi}_t) \right) \right\}. \tag{21.1}$$

Hence, not necessarily optimal dual multipliers for ξ_{tj} are used, but the ones in $\mathcal{D}_t^i$ providing the best approximation for realization ξ_{tj} at x_{t-1}^i.

Let $\pi_{tj}^i = \hat{\pi}_t^{ij}$ and $\alpha_{tj}^i = \hat{\alpha}_t^{ij} + (\hat{\pi}_t^{ij})^\top \left(h_t(\xi_{tj}) - h_t(\hat{\xi}_t^j) \right)$ for all $j = 1, \dots, q_t$. Then, a cut can be defined by using subgradient formula (3.7) and taking expectations as in formula (3.4). Note that our description slightly differs from the presentation in the literature, as we adapted it to our cut formulas in Section 3.3.

This idea for the cut generation process is used in two algorithms related, which mainly differ by *when* cuts are constructed. The CUPPS (*convergent cutting-plane and partial-sampling*) method [41] only contains a forward pass, in which both trial points are computed and cuts are generated using some sample $\xi_t^{k'}$. It has the drawback that the obtained cuts are not necessarily tight. First, the dual multipliers obtained from formula (21.1) are not necessarily optimal for all $j = 1, \dots, q_t$. Second, no backward pass is used, and thus new information in form of cuts for stage $t+1$ are not taken into account when deriving a new cut for stage t.

The *dynamic outer approximation sampling algorithm* (DOASA) [171] contains a forward pass and a backward pass. In the former, a trajectory of trial points $(x_t^{ik})_{k\in\mathcal{K}}$ is computed as in SDDP (note that in [171] $|\mathcal{K}| = 1$ is chosen, but this is not mandatory). In the backward pass, cuts are constructed using a backward sample $\xi_t^{k'}$ and formula (21.1). It is proven that this generalization of SDDP also exhibits almost sure finite convergence [171].

21.1.3. Inexact SDDP. Recall Lemma 3.3 (b), stating that the cuts generated in the backward pass of SDDP are *tight* for $\underline{\mathscr{Q}}_t^{i+1}(\cdot)$ at the incumbent x_{t-1}^{ik}. This result is premised on using optimal dual multipliers in the cut formula, *i.e.*, solving the LP subproblem or its dual to global optimality in line 16 of Algorithm 3.1 (*exact* solution). Whereas such an exact solution is the standard assumption in the literature on SDDP, computationally it may be more efficient to solve subproblems only approximately, especially early in the solution process when the cut approximations are suboptimal anyway [95].

We first introduce the notion of inexact cuts.

DEFINITION 21.1 (ε-inexact cut). *For any $t = 2, \dots, T$, $\varepsilon > 0$ and a trial point x_{t-1}^{ik}, let $\phi_t : \mathbb{R}^{d_{a(n)}} \to \mathbb{R}$ be an affine function satisfying*

$$\mathscr{Q}_t(x_{t-1}) \geq \underline{\mathscr{Q}}_t^{i+1}(x_{t-1}) \geq \phi_t(x_{t-1}) \quad \textit{(validity)}$$

for all $x_{t-1} \in \mathcal{X}_{t-1}$ and

$$\underline{\mathscr{Q}}_t^{i+1}(x_{t-1}^{ik}) - \phi_t(x_{t-1}^{ik}) \leq \varepsilon \quad \textit{(ε-tightness).}$$

Then, $\phi_t(\cdot)$ defines an ε-inexact cut *at x_{t-1}^{ik} [95].*

Importantly, inexact cuts still yield valid lower approximations of $\mathscr{Q}_t(\cdot)$ for all $t = 2, \dots, T$. We now address how inexact cuts can be determined.

Linear Problems. For any iteration i in SDDP, any $t = 2, \dots, T$ and any trial point x_{t-1}^{ik}, consider the linear subproblem (2.10). For simplicity, we assume that $X_t = \{x_t \in \mathbb{R}^{n_t} \ : \ x_t \geq 0\}$.

For some $\varepsilon > 0$, let π_{tjk}^i be an ε-optimal feasible solution for the dual problem of (2.10) given ξ_{tj} and let θ_{tjk}^i be the corresponding dual objective value for $j = 1, \dots, q_t$. Then, analogously to Section 3.3, an ε-inexact cut can be defined by [95]

$$\mathscr{Q}_t(x_{t-1}) \geq \phi_{tk}^i(x_{t-1}) := \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1},$$

with intercept and subgradient defined by

$$\begin{aligned} \alpha_{tk}^i &= \sum_{j=1}^{q_t} p_{tj} \big(\theta_{tjk}^i - (\beta_{tkj}^i)^\top x_{t-1}^{ik}\big), \\ \beta_{tk}^i &= -\sum_{j=1}^{q_t} p_{tj} (\pi_{tkj}^i)^\top T_{t-1,j}. \end{aligned}$$

Nonlinear Differentiable Problems. Consider a multistage stochastic convex program (MSCP) as introduced in Section 15, that is, satisfying Assumptions 10 to 12. Moreover, recall the definitions of the Lagrangian function (15.5), the dual function (15.6) and the Lagrangian dual problem (15.7).

Then, an ε-inexact cut can be derived using a pair of approximate primal-dual solutions as follows [95]. Let $\bar{x}_{tj}$ be an ε-optimal feasible primal solution for problem (15.2) given some noise realization $\xi_{tj}, j = 1, \dots, q_t$, and some trial point $\bar{x}_{t-1}$, and let $\bar{\pi}_{tj}$ be an ε-optimal feasible solution for the corresponding Lagrangian dual (15.7).

We define

$$\eta(\varepsilon) := \ell(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}\xi_{tj}) := \max_{x_t \in X_t} \nabla_{x_t} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj})^\top (\bar{x}_{tj} - x_t). \tag{21.2}$$

Assume that $f_t(x_t, \xi_{tj})$ takes finite values for all $x_t \in X_t$ and that the term in (21.2) is finite. Then, an ε-inexact cut can be defined by

$$\mathscr{Q}_t(x_{t-1}) \geq \phi^i_{tk}(x_{t-1}) := \alpha^i_{tk} + (\beta^i_{tk})^\top x_{t-1},$$

with intercept and subgradient defined by

$$\alpha^i_{tk} = \sum_{j=1}^{q_t} p_{tj}\big(L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}) - \eta(\varepsilon) - (\beta^i_{tkj})^\top x^{ik}_{t-1}\big),$$

$$\beta^i_{tk} = \sum_{j=1}^{q_t} p_{tj} \nabla_{x_{t-1}} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}).$$

We refer to [95] for a convergence analysis of SDDP using inexact cuts, both for the linear and the nonlinear convex case. In particular, it is shown that the obtained dual solutions are almost surely bounded and that the error terms $\eta(\varepsilon^i_t)$ vanish as i approaches $+\infty$.

Non-differentiable Problems. Using SDDP with inexact cuts is generalized to non-differentiable problems in [98]. In this paper, inexact cuts are derived using two different approaches. In the first approach, it is assumed that the objective and constraint functions have saddle-point representations. The second approach is more general, but requires the introduction of additional variables and constraints.

More precisely, consider a multistage stochastic convex program (MSCP) as introduced in Section 15 and assume that it is satisfying Assumptions 10 and 11 except for the differentiability properties. Using a local copy z_t of the state variable x_{t-1}, the approximate value functions can be reformulated as

$$Q_{t,C}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t, z_t} & f_t(x_t, \xi_t) + \mathscr{Q}_{t+1,C}(x_t) \\ \text{s.t.} & g_t(z_t, x_t, \xi_t) \leq 0 \\ & x_t \in X_t \\ & x_{t-1} = z_t. \end{cases} \tag{21.3}$$

Assume that this modified subproblem satisfies a slater condition analogous to Assumption 12. Additionally, consider the Lagrangian dual problem

$$\max_{\pi_t} \mathcal{L}_t(\pi_t; x_{t-1}, \xi_t). \tag{21.4}$$

with dual function

$$\mathcal{L}_{t,C}(\pi_t; x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t \in X_t} & f_t(x_t, \xi_t) + \mathscr{Q}_{t+1,C}(x_t) + \pi_t^\top (x_{t-1} - z_t) \\ \text{s.t.} & g_t(z_t, x_t, \xi_t) \leq 0 \\ & x_t \in X_t, \end{cases}$$

which is obtained by relaxing the copy constraint.

Given a trial point $\bar{x}_{t-1}$ and a noise realization $\xi_{tj}, j = 1, \ldots, q_t$, let $\bar{x}_{tj}$ denote an ε_P-optimal feasible solution of problem (21.3) and let $\bar{\pi}_{tj}$ be an ε_D-optimal feasible solution of problem (21.4). Then, an $(\varepsilon_P + \varepsilon_D)$-inexact cut is defined by function

$$\phi^i_{tk}(x_{t-1}) := \sum_{j=1}^{q_t} p_{tj}\Big(f_t(\bar{x}_{tj}, \xi_{tj}) - (\varepsilon_P + \varepsilon_D) + \bar{\pi}^\top_{tj}(x_{t-1} - \bar{x}_{t-1})\Big).$$

For more details and a convergence analysis we refer to [98].

21.2. Reducing the Number of SDDP Iterations. We now consider techniques with the attempt to reduce the required number of iterations of SDDP until convergence is reached.

21.2.1. Multi-cut SDDP. In the backward pass of SDDP, for any $t \in [T]$ and any $x_{t-1}^{ik}, k \in \mathcal{K}$, subproblems (2.10) are solved for all noise realizations ξ_{tj}, $j = 1, \ldots, q_t$ (line 16 of Algorithm 3.1). By taking expected values, a cut (3.5) is derived (line 18). Such cuts are then incorporated into the stage-$(t-1)$ subproblem using a single variable $\theta_t \in \mathbb{R}$ by

$$\phi_{tk}^i(x_{t-1}) = (\beta_{tk}^i)^\top x_{t-1} + \alpha_{tk}^i \leq \theta_t,$$

see Section 3.3. This is referred to as a *single-cut* approach.

A different approach, called multi-cut, that is well-studied for (nested) Benders decomposition [28, 81, 151], is to not aggregate the dual information, but to generate a separate cut for each noise realization $\xi_{tj}, j = 1, \ldots, q_t$. This means that a pendant to line 18 is included in the loop over j in Algorithm 3.1. In the stage-t subproblem, this requires to introduce variables $\theta_{t,\ell}$ and cut approximations $\underline{\mathcal{V}}_{t+1,\ell}^{i+1}(\cdot)$ for all $\ell = 1, \ldots, q_t$. In this case, we obtain cuts

$$\phi_{tkj}^i(x_{t-1}) := (\beta_{tkj}^i)^\top x_{t-1} + \alpha_{tkj}^i \leq Q_t(x_{t-1}, \xi_{tj}), \qquad j = 1, \ldots, q_t,$$

where, analogously to the derivation in Section 3.3, β_{tkj}^i denotes a subgradient of $\underline{Q}_t^{i+1}(\cdot, \xi_{tj})$ at x_{t-1}^{ik} for $k \in \mathcal{K}, j = 1, \ldots, q_t$, and α_{tkj}^i is defined by

$$\alpha_{tkj}^i := \underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_t) - (\beta_{tkj}^i)^\top x_{t-1}^{ik}.$$

The expectation is then taken in the objective function instead of the cut formula:

$$\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj}) = \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} \big(c_t(\xi_{tj})\big)^\top x_t + \sum_{\ell=1}^{q_{t+1}} p_{t+1,\ell} \underline{\mathcal{V}}_{t+1,\ell}^{i+1}(x_t).$$

This way, more specific information about the value functions is incorporated into the subproblems, hopefully leading to fewer iterations. Moreover, it can be shown that multi-cut SDDP has the same convergence properties as SDDP [8]. On the downside, the number of decision variables and cuts grows significantly compared to the single-cut approach, especially if q_t is large, which increases the computational effort for each iteration. Therefore, so far multi-cut SDDP has rarely been considered in the literature. It should be most promising when q_t is only of moderate size. For the two-stage case, a rule of thumb is that a single-cut approach should be preferred if the number of realizations is considerably larger than the number of first-stage constraints [27]. Note that in principle also a trade-off between single-cut and multi-cut is possible by partially aggregating cuts [24, 29]. Another approach to reduce the computational burden of multi-cut SDDP is to combine it with cut selection strategies, see Section 21.1.1, as proposed in [8].

We return to Example 3.4 to illustrate the multi-cut approach.

EXAMPLE 21.2. *(Continuation of Example 3.4) Using multi-cut SDDP, at stage 3, instead of $\mathcal{Q}_3(\cdot)$, the functions $Q_3(\cdot, \xi_3)$ are separately approximated by cuts for $\xi_3 \in \{1, 2, 4\}$. These value functions are displayed in Figure 15. Each of them consists of only two linear pieces, so two cuts are required to represent them exactly. In contrast, $\mathcal{Q}_3(\cdot)$ consists of four linear segments. Therefore, multi-cut SDDP should need less iterations than single-cut SDDP to achieve convergence.*

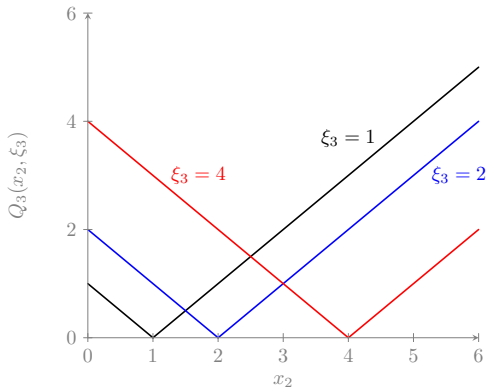


Fig. 15: Stage-3 value functions for Example 3.4.

21.2.2. Batch Learning and Experience Replay. While SDDP is used in stochastic programming, dynamic programming or optimal control, its methodology also shares some characteristics with Q-learning algorithms, which are studied in reinforcement learning, see Remark 3.1. This can be exploited by translating established performance enhancing techniques from reinforcement learning to SDDP.

As one such technique, Ávila et al. [6] propose to use a *batch learning* technique called *experience replay* in SDDP. The motivation of this is the following: In SDDP, the cut approximations $\underline{\mathcal{V}}_t(\cdot)$ of the expected value functions $\mathscr{Q}_t(\cdot)$ are generated recursively in a backward pass through the stages $t = T, \dots, 2$. This means that approximation errors at later stages are propagated to earlier stages by means of the cut approximations $\underline{\mathcal{V}}_t(\cdot)$, which then leads to loose cuts at these earlier stages and so on. However, this implies that errors are accumulated at early stages. The authors identify this as a driver for the slow convergence of SDDP, as it favors over-exploring of suboptimal regions and the generation of redundant cuts throughout the iterations.

Experience replay addresses this issue by revisiting previous trial points x_t^i and updating the cut approximations $\underline{\mathcal{V}}_t(\cdot)$ at these points. This seems counterintuitive at first glance because cuts are generated at already visited points instead of improving the approximation of $\mathscr{Q}_t(\cdot)$ at regions of $\mathcal{X}_t$ that have not been visited yet. However, by taking into account all the information currently available to update $\underline{\mathcal{V}}_t(\cdot)$ at x_t^i, it avoids that on earlier stages $\tau < t$ unnecessarily poor approximations of $\mathscr{Q}_t(\cdot)$ at x_t^i are used for several more iterations.

More precisely, the proposed SDDP method works as follows. A predefined number of iterations of standard SDDP are executed and the corresponding trial points x_t^i are stored in a replay memory M_t for all $t \in [T-1]$ in line 10 of Algorithm 3.1. When the sizes of the replay memories reach a predefined cardinality Z, then the experience replay step is initiated. This step performs a backward pass through the stages $t = T-1, \dots, 2$. For each stage t, first, a batch $B_t \subseteq M_t$ of trial points is selected from the replay memory (also a full batch $B_t = M_t$ is possible). For each trial point $\tilde{x}_t^\ell$ from this batch, with $\ell = 1, \dots, |B_t|$, the previously generated cut is removed from $\underline{\mathcal{V}}_{t+1}^{i+1}(\cdot)$ and a new cut is constructed by solving the associated subproblems (2.10) (including the experience replay updates from following stages) for $\tilde{x}_t^\ell$. With these cuts, $\underline{\mathcal{V}}_{t+1}^{i}(\cdot)$ is updated and then, the previous stage is explored. After the replay step, a standard iteration of SDDP is started again.

It is shown that experience replay manages to improve the convergence behavior of SDDP, and also the out-of-sample performance of the obtained policies [6]. However, experience replay comes at an increased computational effort, as every Z iterations an additional backward pass solving $q_t|B_t|$ subproblems for each stage $t = T, \dots, 2$ has to be performed. For full batches, this adds up to $q_t|\mathcal{K}|Z$ LPs per stage. For this reason, the authors suggest to parallelize both standard SDDP iterations as well as the experience replay. They report computational results which indicate that batch learning is better exploiting parallelism than standard SDDP.

21.2.3. Regularization. As Kelley's cutting-plane method [118, 153], SDDP exhibits an iteration complexity which is exponential in the dimension n_t of the state variables, see Section 4.2. An unfavorable characteristic of cutting-plane methods, and also of SDDP, in this regard is *zig-zagging* behavior. This means that trial points x_t^i and x_t^{i+1} computed in subsequent iterations can be located far away from each other in different regions of the state space, and that with each new cut the minimum of the subproblems (2.10) is again attained in the respective other region. In particular, this implies that these regions of $\mathcal{X}_t$ experience very tight, but almost redundant approximations $\underline{\mathcal{V}}_t(\cdot)$ of $\mathscr{Q}_t(\cdot)$, while other regions are not properly explored and thus the approximation quality at the true optimum improves very slowly.

In convex and nonsmooth optimization, regularization techniques called *bundle methods* are shown to entail faster convergence than classical cutting-plane methods [129], as they mitigate zig-zagging by stabilizing subsequent trial points around a *stability center* (also called incumbent). Hence, it looks promising to translate these regularization techniques to SDDP.

A common regularization approach, which is predominantly used in two-stage stochastic programming [199, 202], is convex quadratic regularization. Here, some quadratic deviation of x_t from a stability center $\widehat{x}_t$ is penalized in the objective function for stabilization. An application of quadratic regularization to SDDP is not straightforward because using a separate stability center for each scenario $s \in \mathcal{S}$ is computationally infeasible due to the exponential growth of $|\mathcal{S}|$ in T [5].

Therefore, Asamov and Powell [5] propose a regularization technique for linear problems, in which stability centers are considered part of the state variable, and thus are the same for all realizations of $\xi_{tj}, j = 1, \dots, q_t$. Then, in the forward pass subproblems (2.10) (see line 10 of Algorithm 3.1), the objective function is modified to

$$c_t^\top x_t + \underline{\mathcal{V}}_{t+1}^i(x_t) + \frac{\gamma^i}{2}(x_t - \widehat{x}_t^{i-1})^\top H_t (x_t - \widehat{x}_t^{i-1}), \tag{21.5}$$

with a positive semidefinite matrix H_t and some sequence $(\gamma^i)_{i\in\mathbb{N}}$ satisfying $\gamma^i \geq 0$ for all i and $\lim_{i\to\infty} \gamma^i = 0$. The stability centers $\widehat{x}_t^{i-1}$ are chosen as the previous forward pass solution, *i.e.*, the solution is stabilized around a "known" region of the domain of $\mathscr{Q}_t(\cdot)$. This idea is generalized to nonlinear problems and improved in [97] by considering weighted averages of several previous forward pass solutions.

Using objective (21.5), a convex, continuous and linearly constrained quadratic programming problem has to be solved in each forward pass step of SDDP, hopefully, reducing the required number of iterations. Importantly, only the forward pass of SDDP is changed, while the backward pass remains the same. In particular, only LPs have to be solved in the backward pass. As the cuts are still finite (see Lemma 3.3), almost sure finite convergence is assured. In computational tests, it is shown that this method exhibits faster convergence than SDDP, in particular for a high state

dimension n_t [5]. This speed-up is especially important for regularized DDP, see the numerical experiments in [97]. DDP (Dual Dynamic Programming) is the corresponding deterministic counterpart of SDDP (when ξ_t is deterministic for all $t \in [T]$).

Whereas the above approach stabilizes the solution around a "known" region of the domain of $\mathscr{Q}_t(\cdot)$, in a sampling setting, it is not clear whether this is always beneficial. For the current sample ξ_t^k a region may be identified and used for stabilization, which is no appropriate indicator for all $\xi_{tj}, j = 1, \ldots, q_t$. Additionally, as pointed out in [234], the condition $\lim_{i\to\infty} \gamma^i = 0$ may evoke that the regularization is diminished and the proposed method in [5] reduces to standard SDDP *before* convergence is obtained, although regularization may be particularly important close to the optimal solution. Therefore, this is claimed to be detrimental to convergence speed [234].

Van Ackooij et al. [234] also address that convergence of proximal bundle methods usually requires the stability centers to be feasible, which is not guaranteed for SDDP subproblems where the feasible set changes with x_{t-1}^i. Therefore, they propose to combine SDDP with a level bundle method, which does not face this requirement.

For stage t and scenario ξ_t^k, trial solutions x_t^{ik} are obtained by solving

$$\begin{cases} \min\limits_{x_t} & \psi_t(x_t) \\ \text{s.t.} & x_t \in \mathbb{X}_t(x_{t-1}^{ik}; \ell_t) \end{cases} \tag{21.6}$$

with $\psi_t(x_t) : \mathbb{R}^{n_t} \to \mathbb{R}$ a given convex function, *e.g.*, $\psi_t(x_t) := x_t^\top x_t$, and

$$\mathbb{X}_t(x_{t-1}^{ik}; \ell_t) := \begin{cases} \arg\min\limits_{x_t \geq 0} & \max\left\{c_t^\top x_t + \underline{\mathcal{V}}_{t+1}^i(x_t), \ell_t\right\} \\ \text{s.t.} & W_t x_t = h_t - T_{t-1} x_{t-1}^{ik}. \end{cases} \tag{21.7}$$

If the maximum in (21.7) is attained by the first term, then x_t^{ik} obtained by solving (21.6) is an ordinary SDDP trial point, referred to as a *normal iterate*. Otherwise, problem (21.6) reduces to a typical level bundle method subproblem, yielding a regularized *level iterate* x_t^{ik}.

The determination of a good level ℓ_t and of an efficient regularization for SDDP are still open questions, and heuristics are proposed in [234] to choose ℓ_t.

In [23] a non-convex regularization scheme is suggested instead of the previous convex schemes. The main motivation behind this is that the existing schemes tend to heavily penalize the exploration of the state space right from the beginning, even if it is well-known that early trajectories in SDDP are far from optimal. By using specific concave penalty functions it is possible to penalize large deviations from the stability centers much less than for the quadratic regularization from [5], but at the same time to heavily penalize small deviations. One example for such a penalty function is the the minimax concave penalty

$$\sum_{j=1}^{n_t} \lambda \int_0^{|x_{tj} - \widehat{x}_{tj}^i|} \max\left\{1 - \frac{v}{\alpha\lambda}, 0\right\} \mathrm{d}v \tag{21.8}$$

with parameters λ, α, which is known from statistical learning [240].

By using this penalty function for regularization in the forward pass of SDDP, in early iterations exploration of the state space is incentivized. Additionally letting $\lambda \to 0$ in SDDP, for later iterations, then more and more focus is put on exploitation close to the current stability center. It is shown that even for the non-convex regularization using (21.8) the SDDP subproblems remain tractable, as they can be reformulated as equivalent MILPs.

Computational results in [23] indicate that with this regularization, the iteration times of SDDP increase, but on the other hand much less iterations are required until SDDP terminates due to bound stalling, and that the lower bounds at termination exceed that of standard SDDP and the quadratic regularization (21.5).

An alternative stabilization approach is proposed in [15] based on the concept of Chebyshev centers of polyhedrons. Here, in the forward pass of SDDP, the subproblems (2.10) are modified such that the computed trial states are defined as Chebyshev centers of the polyhedrons given by previously constructed cuts and an appropriate upper bound. It can be shown that this approach is equivalent to modifying the cut formula to

$$-(\beta_{t+1}^r)^\top x_t + \theta_{t+1} \geq \alpha_{t+1}^r + \bar{\sigma}_t \|(1, c_t + \beta_{t+1}^r)\|, \quad r \in \Gamma_{t+1}. \tag{21.9}$$

The authors use the Euclidean norm $\|\cdot\|_2$ in (21.9), however, different choices are possible as well.

Geometrically, the additional term in (21.9) changes the cut intercept, thus lifting the cut. For $\bar{\sigma}_t = 0$, the usual SDDP trial point x_t^i is determined, whereas for $\bar{\sigma}_t > 0$ an offset in the objective compared to the standard SDDP subproblem is considered, yielding a different iterate. To actually improve the performance of SDDP, choosing $\bar{\sigma}_t$ appropriately is crucial, yet not trivial. Adversely, if $\bar{\sigma}_t$ is chosen too large, basically any feasible point can become the new trial solution. Moreover, to ensure convergence, it has to be ensured that $\bar{\sigma}_t$ converges to zero in the course of the algorithm. In [15] heuristics are used to determine $\bar{\sigma}_t$, but it is not clear whether they guarantee performance gains for SDDP.

21.3. Parallelization. The performance of SDDP cannot only be improved by modifications of the algorithm itself, but also by its implementation and computational execution. Since several computational steps in SDDP are independent of each other, a performance improvement can be achieved by parallelization.

Different parallelization strategies have been proposed for SDDP. They can be classified with respect to how the workload is distributed among different processors and how the processors are synchronized. Based on this observation, Ávila et al. [6] present a taxonomy of parallelization strategies, which we follow in this section.

Parallelization by Scenario. This is the predominant parallelization strategy for SDDP in the literature. Mostly, a *synchronized* version is proposed. In the forward pass, for all $t \in [T]$, the subproblems (2.10) are solved for $|\mathcal{K}|$ different scenarios, which are sampled independently. The uncertain data ξ_t^k and the trial solutions x_{t-1}^k in each of those problems do only depend on scenario k. Therefore, the different scenarios $\xi^k, k \in \mathcal{K}$, can be assigned to different processors. Assuming P different processors, each processor is assigned $\frac{P}{|\mathcal{K}|}$ scenarios and solves all corresponding subproblems. A master process is then used to aggregate the objective values and compute the upper bound estimate (3.9). This means that there is a synchronization point for all processors at the end of the forward pass.

In the backward pass, a similar approach is followed. The subproblems are again distributed among the processors by scenarios, in such way that for a specific stage $t = T, \dots, 2$ and a scenario-based trial point x_{t-1}^k, the subproblems for all noise realizations $\xi_{tj}, j = 1, \dots, q_t$, are solved by the same processor. Evenly distributing the problems between processors, this way each processor solves $\frac{P}{|\mathcal{K}|} q_t$ subproblems. However, it is also possible to let the master process assign new scenarios to processes once they become idle instead of using a fixed assignment scheme [173].

After solving all associated subproblems, each processor then generates a cut for

$\mathscr{Q}_t(\cdot)$ and sends it to the master process. When cut generation is finished for all $k \in \mathcal{K}$, the processors are synchronized so that all of them can proceed with the same set of cuts on stage $t-1$. As stated in [106], this synchronization can be partially relaxed to avoid waiting for single slow processors. Instead, the master process can assign stage-$(t-1)$ subproblems to available processors even if not all cuts have been generated for stage t yet. Numerical results show that such partial relaxation can improve the computational performance of SDDP. However, the number of cuts to wait for to achieve an optimal trade-off between faster iterations and better approximation of $\mathscr{Q}_t(\cdot)$ is problem-dependent.

Even more, an *asynchronous* approach can be used where processors immediately get back to stage $t-1$ after generating their cuts at stage t, using all cuts currently available without waiting for other processes to finish [61].

A major shortcoming of parallelization by scenario is that using more processors becomes more beneficial the more scenarios $|\mathcal{K}|$ are sampled in the forward pass. However, as discussed in Section 21.1.2, it is often favorable to only consider one or a few scenarios per iteration, especially in earlier iterations. Choosing large $|\mathcal{K}|$ may lead to the accumulation of similar trial points and the generation of redundant cuts [6]. Therefore, exploiting the potential performance gains of additional processors may wrongly incentivize to sample more scenarios than reasonable, thus not accelerating but slowing down the solution process. Additionally, Ávila et al. [6] report computational results indicating that (synchronized) parallelization by scenario scales poorly when increasing the number of samples $|\mathcal{K}|$ due to the combination of long waiting times between processors and low quality cuts.

Parallelization by Node. Using parallelization by node, the strategy is to draw only one or a few samples in the forward pass, as this is often computationally preferable. Then, the forward pass is not necessarily parallelized. In the backward pass, the work is distributed among the processors by nodes of the recombining tree (cf. Section 2.1). That means that even for the same $k \in \mathcal{K}$ and the associated trial point x_{t-1}^k, the subproblems (2.10) for different realizations $\xi_{tj}, j = 1, \dots, q_t$, may be solved by different processors. The processors are synchronized at each stage to generate aggregated cuts (given that a single-cut approach is used).

In [6], the authors report clear computational benefits using parallelization by node compared to parallelization by scenario, and also better scaling properties. However, these results require that the processors can access a shared memory, otherwise the computational overhead is too large. Another drawback is that distributing subproblems for different ξ_{tj}, but the same x_{t-1}^{ik} among different scenarios prevents the exploitation of warm starting techniques.

Parallelization by node can also be used in an asynchronous way, as proposed by Machado et al. [143] in their *asynchronous SDDP* method. In this method, the subproblems of all stages $t = 1, \dots, T$ are solved simultaneously. More precisely, in each *step*, for all stages $t = 1, \dots, T$ and scenarios $k \in \mathcal{K}$, the subproblems for all realizations $\xi_{tj}, j = 1, \dots, q_t$, are solved. Once a processor is finished, it constructs a new cut for $\mathscr{Q}_t(\cdot)$ using all available information. If a required processor has not finished yet, multipliers π_{tkj} from previous steps are re-used. The generated cut can then be incorporated in stage $t-1$ in the next step. Additionally, each processor generates a new trial point which can be used at stage t in the next step. In contrast to SDDP iterations, this approach requires several steps to propagate information through all stages. Therefore, an ordinary forward pass can only be observed implicitly over several stages. This has to be considered in the computation of upper bounds.

Independent of the applied strategy, parallelizing SDDP in practice comes with

considerable challenges, such as communication overhead, problem-dependent performance and lack of reproducibility of results. Therefore, its potential to speed up SDDP in general is naturally limited [6].

21.4. Aggregation Techniques. Aggregating information in (MSLP) is another tool with potential to speed up the SDDP solution process.

One approach is to aggregate the variables and constraints of several time periods in a single stage, thus solving a problem with a smaller horizon T. This is straightforward for NBD [58], where each node of the aggregated problem is a subtree of the original scenario tree, even though only few time periods can be aggregated to keep the subproblems tractable. However, it cannot be directly generalized to the sampling and stagewise independent setting in SDDP. The main issue is that it is difficult to model the uncertainty appropriately, without violating non-anticipativity [58].

An alternative approach is to aggregate realizations of $\boldsymbol{\xi}_t$ on each stage (or a subset of stages) [215]. To this end, for some stage t, the support Ξ_t is partitioned into clusters $C_t^\ell, \ell = 1, \dots, L_t$, with $L_t \in \mathbb{N}$. Instead of solving subproblems associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj})$ for all $j = 1, \dots, q_t$ in the backward pass of SDDP (line 16 of Algorithm 3.1), subproblems associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \bar{\xi}_t^\ell)$ are solved for clusters $\ell = 1, \dots, L_t$, with $\bar{\xi}_t^\ell := \sum_{j \in C_t^\ell} \frac{p_{tj}}{\bar{p}_t^\ell} \xi_{tj}$, and $\bar{p}_t^\ell$ the probability of cluster C_t^ℓ. This should be beneficial in early iterations where policies are still far away from optimal and a fine information structure unnecessarily slows down the solution process.

By using subgradients and intercepts associated with clusters $C_t^\ell, \ell = 1, \dots, L_t$, in line 18 of Algorithm 3.1 *coarse cuts* can be generated for $\mathscr{Q}_t(\cdot)$. Given that W_t and c_t are deterministic, these cuts are valid underestimators for $\mathscr{Q}_t(\cdot)$ by Jensen's inequality [215]. They are not guaranteed to be tight, though.

The authors in [215] discuss several different refinement strategies, such as refinements within the SDDP backward pass (the partition at stage t is refined as soon as a coarse cut does not improve the approximation of $\mathscr{Q}_t(\cdot)$ at the trial point x_{t-1}^{ik}) or refinements outside of SDDP. In the latter case, SDDP is performed on a coarse recombining tree, which is iteratively refined once the algorithm has stopped. Computational results show that this latter approach performs significantly better than the first one due to less computational overhead. However, identifying *when* SDDP should be best stopped to perform a refinement remains a challenging task.

22. Outlook. In this tutorial-type review, we give an overview on the motivation, theory, strengths and weaknesses, extensions and applications of SDDP.

While many proposals have been made in the last 30 years on how to extend SDDP and on how to improve its performance, there still remain open research questions, leaving room for future improvement. Among the most crucial topics are the following.

1. *Stopping.* To this date, in many applications SDDP is stopped heuristically, *e.g.*, based on a fixed number of iterations or stabilization of lower bounds, which leaves the task to define a reasonable stopping criterion to the user. Recently, there has been some pioneering work on developing deterministic upper bounding techniques and stopping criterions, but these are still limited, as they require significant computational effort.
2. *Upper bounds in risk-averse SDDP.* Developing efficient upper bounding techniques is especially relevant to risk-averse variants of SDDP, where the commonly used nested risk measures do not allow for employment of their pendants from risk-neutral SDDP. Lately, different risk measures have been proposed, which avoid this issue. However, such risk measures usually hamper

interpretability. Therefore, it can still be regarded an open question how risk should be optimally measured in SDDP in order to obtain a computationally tractable problem and at the same time to properly reflect the true risk preferences of a decision-maker.

3. *Distributionally robust SDDP.* Recently, the consideration of distributional uncertainty in SDDP has gained more interest. However, while distributionally robust optimization is a flourishing research area, incorporating it into SDDP is still in its early stages, with potential for further improvements.
4. *Non-convex extensions.* In many applications, nonlinear functions or integer variables are required to appropriately model the problem at hand. As the (expected) value functions become non-convex in this case, traditional cutting-plane techniques fail to approximate them correctly. Starting with SDDiP, recently, there has been a trend to extend the NBD and SDDP frameworks to non-convex problems. Lagrangian-type cuts, which are possibly non-convex, show theoretical potential in approximating non-convex functions. However, their construction is computationally costly and subject to rather strong technical assumptions, such that especially large-scale non-convex problems remain computationally intractable. Consequently, in the future, the trade-off between computationally efficient cut generation techniques and best possible approximations of the value functions needs to be further explored.
5. *Regularization.* As a descendant of Kelley's cutting-plane method, SDDP has a computational complexity which grows exponentially in the dimension of the state variables. Therefore, it can become computationally intractable for problems with high-dimensional state space. This is aggravated by common reformulations, *e.g.*, in case of stagewise dependent uncertainty, that artificially augment the state space. For Kelley's method, regularization methods have proven helpful in accelerating the solution process. Whereas some first attempts have been made to regularize SDDP, an efficient regularization remains an open challenge.
6. *Reinforcement learning techniques.* As the case of batch learning shows, SDDP can benefit from acceleration techniques that are well-known and established in reinforcement learning, but have not been translated to SDDP setting yet. By exploiting its affinity to Q-learning, there should be a lot of potential to improve the computational performance of SDDP in practice.
7. *Decision-dependent uncertainty.* The only standard assumption for SDDP that has not been relaxed in the literature yet, is to allow for stagewise-dependent stochastic processes modeling the uncertainty in (MSLP). This topic has still to be studied.

Acknowledgments. The authors thank the three anonymous referees and the associate editor for their detailed and insightful feedback which helped to improve earlier versions of this manuscript substantially.

REFERENCES

[1] S. Ahmed, F. G. Cabral, and B. Freitas Paulo da Costa. Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793, 2022.
[2] D. Arjoon, Y. Mohamed, Q. Goor, and A. Tilmant. Hydro-economic risk assessment in the eastern Nile River basin. *Water Resources and Economics*, 8:16–31, 2014.

[3] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

[4] N. V. Arvanitidis and J. Rosing. Composite representation of a multireservoir hydroelectric power systems. *IEEE Transactions on Power Apparatus and Systems*, 89(2):319–326, 1970.

[5] T. Asamov and W. B. Powell. Regularized decomposition of high-dimensional multistage stochastic programs with Markov uncertainty. *SIAM Journal on Optimization*, 28(1):575–595, 2018.

[6] D. Ávila, A. Papavasiliou, and N. Löhndorf. Batch learning SDDP for long-term hydrothermal planning. *IEEE Transactions on Power Systems*, 39(1):614–627, 2024.

[7] H. Bakker, F. Dunke, and S. Nickel. A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice. *Omega*, 96, 2020.

[8] M. Bandarra and V. Guigues. Single cut and multicut stochastic dual dynamic programming with cut selection for multistage stochastic linear programs: convergence proof and numerical experiments. *Computational Management Science*, 18:125–141, 2021.

[9] R. Baucke. An algorithm for solving infinite horizon Markov dynamic programmes. Preprint, available online at http://www.optimization-online.org/DB_FILE/2018/04/6565.pdf, 2018.

[10] R. Baucke, A. Downward, and G. Zakeri. A deterministic algorithm for solving multistage stochastic programming problems. Preprint, available online at http://www.optimization-online.org/DB_FILE/2017/07/6138.pdf, 2017.

[11] R. Baucke, A. Downward, and G. Zakeri. A deterministic algorithm for stochastic minimax dynamic programmes. Preprint, available online at http://www.optimization-online.org/DB_FILE/2018/02/6449.pdf, 2018.

[12] G. Bayraksan and D. K. Love. Data-driven stochastic programming using phi-divergences. In *Tutorials in Operations Research: The Operations Research Revolution*, pages 1–19. INFORMS, 2015.

[13] R. E. Bellman. *Dynamic programming*. Princeton University Press, Princeton, New Jersey, 1957.

[14] M. M. Belsnes, A. Haugstad, B. Mo, and P. Markussen. Quota modeling in hydrothermal systems. In *IEEE Bologna Power Tech Conference Proceedings*, 2003.

[15] F. Beltrán, E. C. Finardi, G. M. Fredo, and W. de Oliveira. Improving the performance of the stochastic dual dynamic programming algorithm using Chebyshev centers. *Optimization and Engineering*, 2020.

[16] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

[17] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[18] J. M. Velásquez Bermúdez. GDDP: generalized dual dynamic programming theory. *Annals of Operations Research*, 117(1-4):21–31, 2002.

[19] D. P. Bertsekas. *Dynamic programming and optimal control, Volume II*. Athena Scientific, 4th edition edition, 2017.

[20] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

[21] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: a fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[22] A. Bhattacharya, J. P. Kharoufeh, and B. Zeng. Managing energy storage in microgrids: a multistage stochastic programming approach. *IEEE Transactions on Smart Grid*, 9(1):483–496, 2018.

[23] A. Bhattacharya, J. P. Kharoufeh, and B. Zeng. A nonconvex regularization scheme for the stochastic dual dynamic programming algorithm. *INFORMS Journal on Computing*, 35(5):1161–1178, 2023.

[24] M. Biel and M. Johansson. Dynamic cut aggregation in L-shaped algorithms. Preprint, available online at https://arxiv.org/abs/1910.13752, 2019.

[25] J. R. Birge. *Solution methods for stochastic dynamic linear programs*. Phd dissertation, Systems Optimization Laboratory, Stanford University, 1980.

[26] J. R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33(5):989–1007, 1985.

[27] J. R. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Series in Operations Research and Financial Engineering. Springer Science & Business Media, 2nd edition, 2011.

[28] J. R. Birge and F. V. Louveaux. A multicut algorithm for two-stage stochastic linear programs.

European Journal of Operational Research, 34(3):384–392, 1988.

[29] X. Blanchot, F. Clautiaux, B. Detienne, A. Froger, and M. Ruiz. The Benders by batch algorithm: Design and stabilization of an enhanced algorithm to solve multicut Benders reformulation of two-stage stochastic programs. *European Journal of Operational Research*, 309(1):202–216, 2023.

[30] S. Bolusani and T. K. Ralphs. A framework for generalized Benders' decomposition and its application to multilevel optimization. *Mathematical Programming*, pages 1–38, 2022.

[31] J. F. Bonnans, Z. Cen, and T. Christel. Energy contracts management by stochatic programming techniques. *Annals of Operations Research*, 200:199–222, 2012.

[32] R. B. S. Brandi, A. L. M. Marcato, B. H. Dias, T. P. Ramos, and I. C. da Silva Junior. A convergence criterion for stochastic dual dynamic programming: application to the long-term operation planning problem. *IEEE Transactions on Power Systems*, 33(4):3678–3690, 2018.

[33] R. B. S. Brandi, T. P. Ramos, B. H. Dias, A. L. M. Marcato, and I. Chaves da Silva Junior. Improving stochastic dynamic programming on hydrothermal systems through an iterative process. *Electric Power Systems Research*, 123:147–153, 2015.

[34] A. Brigatto, A. Street, and D. M. Valladão. Assessing the cost of time-inconsistent operation policies in hydrothermal power systems. *IEEE Transactions on Power Systems*, 32(6):4541–4550, 2017.

[35] S. Bruno, S. Ahmed, A. Shapiro, and A. Street. Risk neutral and risk averse approaches to multistage renewable investment planning under uncertainty. *European Journal on Operational Research*, 250(3):979–989, 2016.

[36] L. Cambier and D. Scieur. FAST (Finally An SDDP Toolbox). Accessed April 27, 2021, https://stanford.edu/~lcambier/cgi-bin/fast/index.php.

[37] P. Carpentier, J-.P. Chancelier, G. Cohen, M. De Lara, and P. Girardeau. Dynamic consistency for stochastic optimal control problems. *Annals of Operations Research*, 200:247–263, 2012.

[38] S. Cerisola, J. M. Latorre, and A. Ramos. Stochastic dual dynamic programming applied to nonconvex hydrothermal models. *European Journal of Operational Research*, 218(3):687–697, 2012.

[39] R. M. Chabar, M. V. F. Pereira, S. Granville, L. A. Barroso, and N. A. Iliadis. Optimization of fuel contracts management and maintenance scheduling for thermal plants under price uncertainty. In *IEEE Power Systems Conference and Exposition*, pages 923–930. IEEE, 2006.

[40] R. Chen and J. Luedtke. On generating Lagrangian cuts for two-stage stochastic integer programs. *INFORMS Journal on Computing*, 34(4):2332–2349, 2022.

[41] Z.-L. Chen and W. B. Powell. A convergent cutting-plane and partial-sampling algorithm for multistage stochastic linear programs with recourse. *Journal of Optimization Theory and Applications*, 102(3):497–524, 1999.

[42] A. Chiralaksanakul and D. P. Morton. Assessing policy quality in multi-stage stochastic programming. *Stochastic Programming E-Print Series*, 12, 2004.

[43] B. F. P. da Costa and V. Leclère. Dual SDDP for risk-averse multistage stochastic programs. *Operations Research Letters*, 51(3):332–337, 2023.

[44] B. F. P. da Costa and V. Leclère. Duality of upper bounds in stochastic dynamic programming. Preprint, available online at https://optimization-online.org/wp-content/uploads/2023/07/dual_rn.pdf, 2023.

[45] L. C. da Costa Jr., F. S. Thomé, J. Dias Garcia, and M. V. F. Pereira. Reliability-constrained power system expansion planning: a stochastic risk-averse optimization approach. *IEEE Transactions on Power Systems*, 36(1):97–106, 2021.

[46] E. L. da Silva and E. C. Finardi. Parallel processing applied to the planning of hydrothermal systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(8):721–729, 2003.

[47] G. B. Dantzig and G. Infanger. Multi-stage stochastic linear programs for portfolio optimization. *Annals of Operations Research*, 45:59–76, 1993.

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

[49] C. M. B. de Castro, A. L. M. Marcato, R. Castro Souza, I. Chaves Silva Junior, F. L. Cyrino Oliveira, and T. Pulinho. The generation of synthetic inflows via bootstrap to increase the energy efficiency of long-term hydrothermal dispatches. *Electric Power Systems Research*, 124:33–46, 2015.

[50] M. de Lara and V. Leclère. Building up time-consistency for risk-measures and dynamic optimization. *European Journal of Operational Research*, 249(1):177–187, 2016.

[51] V. L. de Matos and E. C. Finardi. A computational study of a stochastic optimization model

for long term hydrothermal scheduling. *International Journal of Electrical Power & Energy Systems*, 43(1):1443–1452, 2012.

[52] V. L. de Matos, D. P. Morton, and E. C. Finardi. Assessing policy quality in a multistage stochastic program for long-term hydrothermal scheduling. *Annals of Operations Research*, 253:713–731, 2017.

[53] V. L. de Matos, A. B. Philpott, and E. C. Finardi. Improving the performance of Stochastic Dual Dynamic Programming. *Journal of Computational and Applied Mathematics*, 290:196–208, 2015.

[54] L. Ding, S. Ahmed, and A. Shapiro. A Python package for multi-stage stochastic programming. Preprint, available online at http://www.optimization-online.org/DB_FILE/2019/05/7199.pdf, 2019.

[55] T. Ding, X. Zhang, R. Lu, M. Qu, M. Shahidehpour, Y. He, and T. Chen. Multi-stage distributionally robust stochastic dual dynamic programming to multi-period economic dispatch with virtual energy storage. *IEEE Transactions on Sustainable Energy*, 13(1):1–13, 2022.

[56] A. L. Diniz, M. E. P. Maceira, C. L. V. Vasconcellos, and D. D. J. Penna. A combined SDDP/Benders decomposition approach with a risk-averse surface concept for reservoir operation in long term power generation planning. *Annals of Operations Research*, 292:649–681, 2020.

[57] C. J. Donohue and J. R. Birge. The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse. *Algorithmic Operations Research*, 1(1):20–30, 2006.

[58] T. N. dos Santos and A. L. Diniz. A new multiperiod stage definition for the multistage Benders decomposition approach applied to hydrothermal scheduling. *IEEE Transactions on Power Systems*, 24(3):1383–1392, 2009.

[59] A. Downward, O. Dowson, and R. Baucke. Stochastic dual dynamic programming with stagewise-dependent objective uncertainty. *Operations Research Letters*, 48(1):33–39, 2020.

[60] O. Dowson. The policy graph decomposition of multistage stochastic programming problems. *Networks*, 76(1):3–23, 2020.

[61] O. Dowson and L. Kapelevich. SDDP.jl: a Julia package for stochastic dual dynamic programming. *INFORMS Journal on Computing*, 33(1):27–33, 2020.

[62] O. Dowson, D. P. Morton, and A. Downward. Bi-objective multistage stochastic linear programming. *Mathematical Programming*, 196:907–933, 2022.

[63] O. Dowson, D. P. Morton, and B. K. Pagnoncelli. Partially observable multistage stochastic programming. *Operations Research Letters*, 48(4):505–512, 2020.

[64] O. Dowson, D. P. Morton, and B. K. Pagnoncelli. Incorporating convex risk measures into multistage stochastic programming algorithms. *Annals of Operations Research*, 2022.

[65] O. Dowson, A. Philpott, A. Mason, and A. Downward. A multi-stage stochastic optimization model of a pastoral dairy farm. *European Journal of Operations Research*, 274(3):1077–1089, 2019.

[66] I. Dunning, J. Huchette, and M. Lubin. JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[67] J. Dupačová and V. Kozmík. SDDP for multistage stochastic programs: preprocessing via scenario reduction. *Computational Management Science*, 14(1):67–80, 2017.

[68] J. Dupačová and V. Kozmík. Structure of risk-averse multistage stochastic programs. *OR Spectrum*, 37:559–582, 2015.

[69] D. Duque and D. P. Morton. Distributionally robust stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(4):2841–2865, 2020.

[70] J. L. Durante, J. Nascimento, and W. B. Powell. Risk directed importance sampling in stochastic dual dynamic programming with hidden Markov models for grid level energy storage. Preprint, available online at https://arxiv.org/pdf/2001.06026.pdf, 2020.

[71] A. Eichhorn and W. Römisch. Polyhedral risk measures in stochastic programming. *SIAM Journal on Optimization*, 16(1):69–95, 2005.

[72] M.J. Feizollahi, S. Ahmed, and X. A. Sun. Exact augmented Lagrangian duality for mixed integer linear programming. *Mathematical Programming*, 161(1-2):365–387, 2017.

[73] M. L. Fisher. An applications oriented guide to Lagrangian relaxation. *Interfaces*, 15(2):10–21, 1985.

[74] B. C. Flach, L. A. Barroso, and M. V. F. Pereira. Long-term optimal allocation of hydro generation for a price-maker company in a competitive market: latest developments and a stochastic dual dynamic programming approach. *IET Generation, Transmission & Distribution*, 4(2):299–314, 2010.

[75] M. Forcier and V. Leclère. Trajectory following dynamic programming algorithms without finite support assumptions. *Journal of Convex Analysis*, 30(3):951–999, 2023.
[76] S.M. Frank and S. Rebennack. An introduction to optimal power flow: theory, formulation, and examples. 48(12):1172–1197, 2016.
[77] C. Füllner and S. Rebennack. Non-convex nested Benders decomposition. 196:987–1024, 2022.
[78] C. Füllner, X. A. Sun, and S. Rebennack. A new framework to generate lagrangian cuts in multistage stochastic mixed-integer programming. Preprint, available online at https://optimization-online.org/?p=27312.pdf, 2024.
[79] C. Füllner, X. A. Sun, and S. Rebennack. On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming. Preprint, available online at https://optimization-online.org/?p=27295.pdf, 2024.
[80] J. Dias Garcia, I. Leal de Freitas, R. Chabar, and M. V. F. Pereira. A multicut approach to compute upper bounds for risk-averse SDDP. Preprint, available online at https://arxiv.org/pdf/2307.13190.pdf, 2023.
[81] H. I. Gassmann. MSLiP: A computer code for the multistage stochastic linear programming problem. *Mathematical Programming*, 47:407–423, 1990.
[82] A. M. Geoffrion. Lagrangean relaxation for integer programming. In M. L. Balinski, editor, *Approaches to integer programming*, pages 82–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 1974.
[83] A. Georghiou, A. Tsoukalas, and W. Wiesemann. Robust Dual Dynamic Programming. *Operations Research*, 67(3):813–830, 2019.
[84] H. Gevret, N. Langrené, J. Lelong, R. Lobato, T. Ouillon, X. Warin, and A. Maheshwari. STochastic OPTimization library in C++. Technical report, EDF Lab, 2018.
[85] P. Girardeau, V. Leclère, and A. B. Philpott. On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40(1):130–145, 2015.
[86] A. Gjelsvik, M. M. Belsnes, and A. Haugstad. An algorithm for stochastic medium-term hydrothermal scheduling under spot price uncertainty. In *Proceedings of 13th power systems computation conference*, 1999.
[87] A. Gjelsvik, M. M. Belsnes, and M. Håland. A case of hydro scheduling with a stochastic price model. In E. Broch, D.K. Lysne, N. Flatabø, and E. Helland-Hansen, editors, *Procedings of the 3rd international conference on hydropower*, pages 211–218. A.A. Balkema, 1997.
[88] A. Gjelsvik, B. Mo, and A. Haugstad. Long- and medium-term operations planning and stochastic modelling in hydro-dominated power systems based on stochastic dual dynamic programming. In S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors, *Handbook of power systems*, Energy Systems. Springer-Verlag Berlin Heidelberg, 2010.
[89] A. Gjelsvik and S. W. Wallace. Methods for stochastic medium-term scheduling in hydro-dominated power systems. Technical report, Norwegian Electric Power Research Institute, Trondheim, 1996.
[90] Z. Guan and A. B. Philpott. A multistage stochastic programming model for the New Zealand dairy industry. *International Journal of Production Economics*, 134(2):289–299, 2011.
[91] V. Guigues. SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203, 2014.
[92] V. Guigues. Convergence analysis of sampling-based decomposition methods for risk-averse multistage stochastic convex programs. *SIAM Journal on Optimization*, 26(4):2468–2494, 2016.
[93] V. Guigues. DASC: a Decomposition Algorithm for multistage stochastic programs with Strongly Convex cost functions. Preprint, available online at https://arxiv.org/pdf/1711.04650.pdf, 2017.
[94] V. Guigues. Dual dynamic programing with cut selection: convergence proof and numerical results. *European Journal of Operational Research*, 258(1):47–57, 2017.
[95] V. Guigues. Inexact cuts in stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(1):407–438, 2020.
[96] V. Guigues. Multistage stochastic programs with a random number of stages: dynamic programming equations, solution methods, and application to portfolio selection. *Optimization Methods and Software*, 36(1):211–236, 2021.
[97] V. Guigues, M. A. Lejeune, and W. Tekaya. Regularized stochastic dual dynamic programming for convex nonlinear optimization problems. *Optimization and Engineering*, 21:1133–1165, 2020.
[98] V. Guigues, R. Monteiro, and B. Svaiter. Inexact cuts in SDDP applied to multistage stochastic nondifferentiable problems. *SIAM Journal on Optimization*, 31(3):2084–2110,

2021.

[99] V. Guigues and R. D. C. Monteiro. Stochastic dynamic cutting plane for multistage stochastic convex programs. *Journal of Optimization Theory and Applications*, 189:513–559, 2021.

[100] V. Guigues and W. Römisch. Sampling-based decomposition methods for multistage stochastic programs based on extended polyhedral risk measures. *SIAM Journal on Optimization*, 22(2):286–312, 2012.

[101] V. Guigues and W. Römisch. SDDP for multistage stochastic linear programs based on spectral risk measures. *Operations Research Letters*, 40(12):313–318, 2012.

[102] V. Guigues and C. Sagastizábal. The value of rolling-horizon policies for risk-averse hydrothermal planning. *European Journal of Operational Research*, 217(1):129–140, 2012.

[103] V. Guigues and C. Sagastizábal. Risk-averse feasible policies for large-scale multistage stochastic linear programs. *Mathematical Programming*, 138:167–198, 2013.

[104] V. Guigues, A. Shapiro, and Y. Cheng. Duality and sensitivity analysis of multistage linear stochastic programs. *European Journal of Operational Research*, 308(2):752–767, 2023.

[105] V. Guigues, A. Shapiro, and Y. Cheng. Risk-averse stochastic optimal control: an efficiently computable statistical upper bound. *Operations Research Letters*, 51:393–400, 2023.

[106] A. Helseth and H. Braaten. Efficient parallelization of the stochastic dual dynamic programming algorithm applied to hydropower scheduling. *Energies*, 8(12):14287–14297, 2015.

[107] A. Helseth, M. Fodstad, and B. Mo. Optimal medium-term hydropower scheduling considering energy and reserve capacity markets. *IEEE Transactions on Sustainable Energy*, 7(3):934–942, 2016.

[108] A. Helseth, B. Mo, M. Fodstad, and M. N. Hjelmeland. Co-optimizing sales of energy and capacity in a hydropower scheduling model. In *IEEE Power Tech*, pages 1–6, Eindhoven, The Netherlands, 2015.

[109] M. Hindsberger. ReSa: A method for solving multistage stochastic linear programs. *Journal of Applied Operational Research*, 6(1):2–15, 2014.

[110] M. N. Hjelmeland, J. Zou, A. Helseth, and S. Ahmed. Nonconvex medium-term hydropower scheduling by stochastic dual dynamic integer programming. *IEEE Transactions on Sustainable Energy*, 10(1), 2019.

[111] T. Homem-de-Mello and G. Bayraksan. Monte Carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1):56–85, 2014.

[112] T. Homem-de-Mello, V. L. De Matos, and E. C. Finardi. Sampling strategies and stopping criteria for stochastic dual dynamic programming: a case study in long-term hydrothermal scheduling. *Energy Systems*, 2(1):1–31, 2011.

[113] T. Homem-de-Mello and B. K. Pagnoncelli. Risk aversion in multistage stochastic programming: A modeling and algorithmic perspective. *European Journal of Operational Research*, 249(1):188–199, 2016.

[114] J. Huang, K. Zhou, and Y. Guan. A study of distributionally robust multistage stochastic optimization. Preprint, available online at https://arxiv.org/abs/1708.07930, 2017.

[115] N.A. Iliadis, V.F. Perira, S. Granville, M. Finger, P.A. Haldi, and L.A. Barroso. Benchmarking of hydroelectric stochastic risk management models using financial indicators. In *2006 IEEE power engineering society general meeting*, 2006.

[116] G. Infanger and D. P. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256, 1996.

[117] J. Kallrath, P. M. Pardalos, S. Rebennack, and M. Scheidt, editors. *Optimization in the energy industry*. Springer, 2009.

[118] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[119] A. Kiszka and D. Wozabal. Stochastic dual dynamic programming for optimal power flow problems under uncertainty. Preprint, available online at https://optimization-online.org/wp-content/uploads/2021/11/8689.pdf, 2021.

[120] V. Kozmík. On variance reduction of mean-CVaR Monte Carlo estimators. *Computational Management Science*, 12(2):221–242, 2015.

[121] V. Kozmík and D. P. Morton. Evaluating policies in risk-averse multi-stage stochastic programming. *Mathematical Programming*, 152:275–300, 2015.

[122] V. Krasko and S. Rebennack. Two-stage stochastic mixed-integer nonlinear programming model for post-wildfire debris flow hazard management: mitigation and emergency evacuation. 263(1):265–282, 2017.

[123] T. Kristiansen. Hydropower scheduling and financial risk management. *Optimal Control Applications and Methods*, 27(1):1–18, 2006.

[124] G. Lan. Complexity of stochastic dual dynamic programming. *Mathematical Programming*,

191:717–754, 2022.
[125] Y. Lan, Q. Zhai, X. Liu, and X. Guan. Fast stochastic dual dynamic programming for economic dispatch in distribution systems. *IEEE Transactions on Power Systems*, 38(4):3828–3840, 2022.
[126] V. Leclère, P. Carpentier, J.-P. Chancelier, A. Lenoir, and F. Pacaud. Exact converging bounds for stochastic dual dynamic programming via Fenchel duality. *SIAM Journal on Optimization*, 30(2):1223–1250, 2020.
[127] V. Leclère, F. Pacaud, T. Rigaut, and H. Gerard. StochDynamicProgramming.jl. Code released on GitHub https://github.com/JuliaStochOpt/StochDynamicProgramming.jl.
[128] B. Legat. StructDualDynProg.jl. Code released on GitHub https://github.com/JuliaStochOpt/StructDualDynProg.jl.
[129] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69(1-3):111–147, 1995.
[130] C. M. Leocadio, C. S. G. Richa, M. Z. Fortes, V. H. Ferreira, and B. H. Dias. Economic evaluation of a CHP biomass plant using stochastic dual dynamic programming. *Electrical Engineering*, 102:2605–2615, 2020.
[131] L. M. M. Lima, E. Popova, and P. Damien. Modeling and forecasting of Brazilian reservoir inflows via dynamic linear models. *International Journal of Forecasting*, 30(3):464–476, 2014.
[132] K. Linowsky and A. B. Philpott. On the convergence of sampling-based decomposition algorithms for multistage stochastic programs. *Journal of Optimization Theory and Applications*, 125(2):349–366, 2005.
[133] R. P. Liu and A. Shapiro. Risk neutral reformulation approach to risk averse stochastic programming. *European Journal of Operational Research*, 286(1):21–31, 2020.
[134] T. Lohmann, A. S. Hering, and S. Rebennack. Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. 255(1):243–258, 2016.
[135] N. Löhndorf. An empirical analysis of scenario generation methods for stochastic optimization. *European Journal of Operational Research*, 255(1):121–132, 2016.
[136] N. Löhndorf and A. Shapiro. Modeling time-dependent randomness in stochastic dual dynamic programming. *European Journal of Operational Research*, 273(2):650–661, 2019.
[137] N. Löhndorf and D. Wozabal. Gas storage valuation in incomplete markets. *European Journal of Operational Research*, 288(1):318–330, 2021.
[138] N. Löhndorf, D. Wozabal, and S. Minner. Optimizing trading decisions for hydro storage systems using approximate dual dynamic programming. *Operations Research*, 61(4):810–823, 2013.
[139] R. Lu, T. Ding, B. Qin, J. Ma, X. Fang, and Z. Dong. Multi-stage stochastic programming to joint economic dispatch for energy and reserve with uncertain renewable energy. *IEEE Transactions on Sustainable Energy*, 11(3):1140–1151, 2020.
[140] M. E. P. Maceira and J. M. Damázio. Use of the PAR (p) model in the stochastic dual dynamic programming optimization scheme used in the operation planning of the Brazilian hydropower system. *Probability in the Engineering and Informational Sciences*, 20(1):143–156, 2006.
[141] M. E. P. Maceira, V. S. Duarte, D. D. J. Penna, L. A. M. Moraes, and A. C. G. Melo. Ten years of application of stochastic dual dynamic programming in official and agent studies in Brazil - description of the NEWAVE program. In *16th PSCC, Glasgow, Scotland*, 2008.
[142] M. E. P. Maceira, D. D. J. Penna, A. L. Diniz, R. J. Pinto, A. C. G. Melo, C. V. Vasconcellos, and C. B. Cruz. Twenty years of application of stochastic dual dynamic programming in official and agent studies in Brazil: main features and improvements on the NEWAVE model. In *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018.
[143] F. D. R. Machado, A. L. Diniz, C. L. T. Borges, and L. C. Brandão. Asynchronous parallel stochastic dual dynamic programming applied to hydrothermal generation planning. *Electric Power Systems Research*, 191:1–14, 2021.
[144] H. Macian-Sorribes, A. Tilmant, and M. Pulibo-Velazquez. Improving operating policies of large-scale surface-groundwater systems through stochastic programming. *Water Resources Research*, 53:1407–1423, 2017.
[145] G. F. Marques and A. Tilmant. The economic value of coordination in large-scale multireservoir systems: The Parana River case. *Water Resources Research*, 49:7546–7557, 2013.
[146] Y. Mbeutcha, M. Gendreau, and G. Emiel. Benefit of PARMA modeling for long-term hydroeletric scheduling using stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 147(3):1–12, 2021.
[147] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: part I – convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[148] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.
[149] B. Mo, A. Gjelsvik, and A. Grundt. Integrated risk management of hydro power scheduling and contract management. *IEEE Transactions on Power Systems*, 16(2):216–221, 2001.
[150] B. Mo, A. Gjelsvik, A. Grundt, and K. Karesen. Optimisation of hydropower operation in a liberalised market with focus on price modelling. In *PowerTech Proceedings*, 2001.
[151] D. P. Morton. An enhanced decomposition algorithm for multistage stochastic hydroelectric scheduling. *Annals of Operations Research*, 64(1):211–235, 1996.
[152] G. Nannicini, E. Traversi, and R. Wolfler Calvo. A benders squared (b^2) framework for infinite-horizon stochastic linear programs. *Mathematical Programming Computation*, 13:645–681, 2021.
[153] Y. Nesterov. *Introductory lectures on convex optimization: a basic course*. Springer, Boston, 2004.
[154] N. Newham. *Power system investment planning using stochastic dual dynamic programming*. PhD thesis, University of Canterbury, 2008.
[155] A. Papavasiliou, Y. Mou, L. Cambier, and D. Scieur. Application of stochastic dual dynamic programming to the real-time dispatch of storage under renewable supply uncertainty. *IEEE Transactions on Sustainable Energy*, 9(2):547–558, 2018.
[156] P. M. Pardalos, S. Rebennack, M. V. F. Pereira, N. A. Iliadis, and V. Pappu, editors. *Handbook of wind power systems*. Springer, 2013.
[157] P. Parpas, B. Ustun, M.Webster, and Q. K. Tran. Importance sampling in stochastic programming: a Markov Chain Monte Carlo approach. *INFORMS Journal on Computing*, 27(2):358–377, 2015.
[158] M. V. F. Pereira, N. M. Campodónico, and R. Kelman. Application of stochastic dual DP and extensions to hydrothermal scheduling. Technical report, PSRI, 1999.
[159] M. V. F. Pereira and L. M. V. G. Pinto. Stochastic optimization of a multireservoir hydroelectric system: a decomposition approach. *Water Resources Research*, 21(6):779–792, 1985.
[160] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.
[161] M.V.F. Pereira. Optimal scheduling of hydrothermal systems - an overview. In *IFAC Symposium on Planning and Operation of Electric Energy Systems., Rio de Janeiro, Brazil, 22-25 July*, pages 1–6, 1985.
[162] S. J. Pereira-Cardenal, B. Mo, A. Gjelsvik, N. D. Riegels, K. Arnbjerg-Nielsen, and P. Bauer-Gottwein. Joint optimization of regional water-power systems. *Advances in Water Resources*, 92:200–207, 2016.
[163] L. Pfeiffer, R. Apparigliato, and S. Auchapt. Two methods of pruning Benders' cuts and their application to the management of a gas portfolio. Preprint, available online at http://www.optimization-online.org/DB_FILE/2012/11/3683.pdf, 2012.
[164] G. C. Pflug. *Probabilistic Constrained Optimization: Methodology and Applications*, chapter Some Remarks on the Value-at-Risk and the Conditional Value-at-Risk., pages 272–281. Kluwer Academic Publishers, 2000.
[165] G. C. Pflug and A. Pichler. Time-consistent decisions and temportal decomposition of coherent risk functionals. *Mathematics of Operations Research*, 41(2):682–699, 2016.
[166] G. C. Pflug and A. Ruszczyński. Measuring risk for income streams. *Computational Optimization and Applications*, 32(1):161–178, 2005.
[167] A. B. Philpott and V. L. de Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470–483, 2012.
[168] A. B. Philpott, V. L. de Matos, and E. Finardi. On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970, 2013.
[169] A. B. Philpott, V. L. de Matos, and L. Kapelevich. Distributionally robust SDDP. *Computational Management Science*, 15:431–454, 2018.
[170] A. B. Philpott, F. Wahid, and F. Bonnans. MIDAS: A mixed integer dynamic approximation scheme. *Mathematical Programming*, 181:19–50, 2020.
[171] A.B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450–455, 2008.
[172] J. Pina, A. Tilmant, and P. Côté. Optimizing multireservoir system operating policies using exogenous hydrologic variables. *Water Resources Research*, 53(11):9845–9859, 2017.
[173] R. J. Pinto, C. L. T. Borges, and M. E. P. Maceira. An efficient parallel algorithm for large scale hydrothermal system operation planning. *IEEE Transactions on Power Systems*,

28(4):4888–4896, 2013.
[174] B. T. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(3):14–29, 1969.
[175] H. Poorsepahy-Samian, V. Espanmanesh, and B. Zahraie. Improved inflow modeling in stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 142(12):1–10, 2016.
[176] W. B. Powell. *Approximate dynamic programming: solving the curses of dimensionality*. Wiley-Interscience, 2007.
[177] W. B. Powell. *Reinforcement Learning and Stochastic Optimization – A Unified Framework for Sequential Decisions*. John Wiley & Sons, 2022.
[178] G. Pritchard. Stochastic inflow modeling for hydropower scheduling problems. *European Journal of Operational Research*, 246(2):496–504, 2015.
[179] PSR. SDDP - Stochastic hydrothermal dispatch with network restrictions. Accessed April 27, 2021, https://www.psr-inc.com/softwares-en/.
[180] Quantego. QUASAR. Accessed April 27, 2021, http://quantego.com/.
[181] A. R. Queiroz and D. P. Morton. Sharing cuts under aggregated forecast when decomposing multi-stage stochastic programs. *Operations Research Letters*, 41(3):311–316, 2013.
[182] F. Quezada, C. Gicquel, and S. Kedad-Sidhoum. Combining polyhedral approaches and stochastic dual dynamic integer programming for solving the uncapacitated lot-sizing problem under uncertainty. *INFORMS Journal on Computing*, 34(2):1024–1041, 2022.
[183] H. Rahimian and S. Mehrotra. Distributionally robust optimization: a review. Preprint, available online at https://arxiv.org/pdf/1908.05659.pdf, 2019.
[184] L. Raso, P.-O. Malaterre, and J.-C. Bader. Effective streamflow process modeling for optimal reservoir operation using stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 143(4):1–11, 2017.
[185] S. Rebennack. Generation expansion planning under uncertainty with emissions quotas. 114:78–85, 2014.
[186] S. Rebennack. Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. 156:343–389, 2016.
[187] S. Rebennack, B. Flach, M.V.F. Pereira, and P.M. Pardalos. Stochastic hydro-thermal scheduling under CO_2 emissions constraints. 27(1):58–68, 2012.
[188] S. Rebennack, N.A. Iliadis, M.V.F. Pereira, and P.M. Pardalos. Electricity and co_2 emissions system prices modeling and optimization. In *PowerTech, 2009 IEEE Bucharest*, pages 1–6. IEEE, 2009.
[189] S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of power systems I*. Springer, 2010.
[190] S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of power systems II*. Springer, 2010.
[191] M. Resener, S. Rebennack, P. M. Pardalos, and S. Haffner, editors. *Handbook of optimization in electric power distribution systems*. Springer, 2020.
[192] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of risk*, 2(3):21–42, 2000.
[193] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*. Springer Berlin Heidelberg, 2009.
[194] A. W. Rosemberg, A. Street, J. Dias Garcia, D. M. Valladão, T. Silva, and O. Dowson. Assessing the cost of network simplifications in long-term hydrothermal dispatch planning models. *IEEE Transactions on Sustainable Energy*, 13(1):196–206, 2022.
[195] T. A. Rotting and A. Gjelsvik. Stochastic dual dynamic programming for seasonal scheduling in the Norwegian power system. *IEEE Transactions on Power Systems*, 7(1):273–279, February 1992.
[196] C. Rougé and A. Tilmant. Using stochastic dual dynamic programming in problems with multiple near-optimal solutions. *Water Resources Research*, 52(5):4151–4163, 2016.
[197] C. Rougé, A. Tilmant, B. Zaitchik, A. Dezfuli, and M. Salman. Identifying key water resource vulnerabilities in data-scarce transboundary river basins. *Water Resources Research*, 54(8):5264–5281, 2018.
[198] B. Rudloff, A. Street, and D. M. Valladão. Time consistency and risk averse dynamic decision models: definition, interpretation and practical consequences. *European Journal of Operational Research*, 234(3):743–750, 2014.
[199] A. Ruszczyński. Decomposition methods in stochastic programming. *Mathematical Programming*, 79(1-3):333–353, 1997.
[200] A. Ruszczyński. Risk-averse dynamic programming for Markov decision processes. *Mathematical Programming*, 125:235–261, 2010.
[201] A. Ruszczyński and A. Shapiro. Conditional risk mappings. *Mathematics of Operations*

Research, 31(3):544–561, 2006.
[202] A. Ruszczyński and A. Swietanowski. Accelerating the regularized decomposition method for two stage stochastic linear problems. *European Journal of Operational Research*, 101(2):328–342, 1997.
[203] A. Shapiro. Inference of statistical bounds for multistage stochastic programming problems. *Mathematical Methods of Operations Research*, 58(1):57–68, 2003.
[204] A. Shapiro. On complexity of multistage stochastic programs. *Operations Research Letters*, 34(1):1–8, 2005.
[205] A. Shapiro. Stochastic programming approach to optimization under uncertainty. *Mathematical Programming*, 112(1):183–220, 2008.
[206] A. Shapiro. Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209(1):63–72, 2011.
[207] A. Shapiro. Tutorial on risk neutral, distributionally robust and risk averse multistage stochastic programming. *European Journal of Operational Research*, 288(1):1–13, 2021.
[208] A. Shapiro and Y. Cheng. Dual bounds for periodical stochastic programs. *Operations Research*, 71(1):120–128, 2022.
[209] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition*. Society for Industrial and Applied Mathematics, 2014.
[210] A. Shapiro and L. Ding. Upper bound for optimal value of risk averse multistage problems. Technical report, Georgia Tech, 2016.
[211] A. Shapiro and L. Ding. Periodical multistage stochastic programs. *SIAM Journal on Optimization*, 30(3):2083–2102, 2020.
[212] A. Shapiro, W. Tekaya, J. P. da Costa, and M. P. Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.
[213] A. Shapiro, W. Tekaya, M. P. Soares, and J. P. da Costa. Worst-case-expectation approach to optimization under uncertainty. *Operations Research*, 61(6):1435–1449, 2013.
[214] P. Shinde, I. Kouveliotis-Lysikatos, and M. Amelin. Multistage stochastic programming for VPP trading in continuous intraday electricity markets. *IEEE Transactions on Sustainable Energy*, 13(2):1–12, 2022.
[215] M. Siddig and Y. Song. Adaptive partition-based SDDP algorithms for multistage stochastic linear programming with fixed recourse. *Computational Optimization and Applications*, 81:201–250, 2022.
[216] M. P. Soares, A. Street, and D. M. Valladão. On the solution variability reduction of stochastic dual dynamic programming applied to energy planning. *European Journal of Operational Research*, 258(2):743–760, 2017.
[217] A. Sorokin, S. Rebennack, P.M. Pardalos, N.A. Iliadis, and M.V.F. Pereira, editors. *Handbook of networks in power systems I*. Springer Science & Business Media, 2012.
[218] A. Sorokin, S. Rebennack, P.M. Pardalos, N.A. Iliadis, and M.V.F. Pereira, editors. *Handbook of networks in power systems II*. Springer Science & Business Media, 2012.
[219] G. Steeger, T. Lohmann, and S. Rebennack. Strategic bidding for a price-maker hydroelectric producer: stochastic dual dynamic programming and Lagrangian relaxation. 50(11):929–942, 2018.
[220] G. Steeger and S. Rebennack. Strategic bidding for multiple price-maker hydroelectric producers. 47(9):1013–1031, 2015.
[221] G. Steeger and S. Rebennack. Dynamic convexification within nested Benders decomposition using Lagrangian relaxation: an application to the strategic bidding problem. 257(2):669–686, 2017.
[222] G.M. Steeger. *Strategic bidding for price-maker hydroelectric producers*. PhD thesis, Colorado School of Mines, 2014.
[223] A. Street, L. A. Barroso, B. Flach, M. V. Pereira, and S. Granville. Risk constrained portfolio selection of renewable sources in hydrothermal electricity markets. *IEEE Transactions on Power Systems*, 24(3):1136–1144, 2009.
[224] A. Street, A. Brigatto, and D. M. Valladão. Co-optimization of energy and ancillary services for hydrothermal operation planning under a general security criterion. *IEEE Transactions on Power Systems*, 32(6):4914–4923, 2017.
[225] X. A. Sun and A. J. Conejo. *Robust optimization in electric energy systems*. International Series in Operations Research & Management Science. Springer, 2022.
[226] S. Thevenin, Y. Adulyasak, and J.-F. Cordeau. Stochastic dual dynamic programming for multiechelon lot sizing with component substitution. *INFORMS Journal on Computing*, 34(6):3151–3169, 2022.
[227] F.S. Thomé. *Representação de não-convexidade no planejamento da operação hidrotérmica*

utilizando PDDE. PhD thesis, COPPE-UFRJ, 2013.

[228] A. Tilmant, L. Beevers, and B. Muyunda. Restoring a flow regime through the coordinated operation of a multireservoir system: The case of the Zambezi River basin. *Water Resources Research*, 46(7):1–11, 2010.

[229] A. Tilmant and R. Kelman. A stochastic approach to analyze trade-offs and risks associated with large-scale water resources systems. *Water Resources Research*, 43(6), 2007.

[230] A. Tilmant and W. Kinzelbach. The cost of noncooperation in international river basins. *Water Resources Research*, 48(1):1–12, 2012.

[231] A. Tilmant, J. Lettany, and R. Kelman. Hydrological risk assessment in the Euphrates-Tigris river basin: A stochastic dual dynamic programming approach. *Water International*, 32(2):294–309, 2009.

[232] A. Tilmant, D. Pinte, and Q. Goor. Assessing marginal water values in multipurpose multireservoir systems via stochastic programming. *Water Resources Research*, 44(12):1–17, 2008.

[233] D. M. Valladão, T. Silva, and M. Poggi. Time-consistent risk-constrained dynamic portfolio optimization with transactional costs and time-dependent returns. *Annals of Operations Research*, 282:379–405, 2019.

[234] W. van Ackooij, W. de Oliveira, and Y. Song. On level regularization with normal solutions in decomposition methods for multistage stochastic programming problems. *Computational Optimization and Applications*, 74:1–42, 2019.

[235] W. van Ackooij and X. Warin. On conditional cuts for stochastic dual dynamic programming. *EURO Journal on Computational Optimization*, 8(2):173–199, 2020.

[236] N. van der Laan and W. Romeijnders. A converging Benders' decomposition algorithm for two-stage mixed-integer recourse models. *Operations Research*, 2022. Accepted.

[237] R. M. van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

[238] F. Wahid. *River optimisation: short-term hydro-bidding under uncertainty*. Optimization and control, Université Paris-Saclay; University of Auckland, New Zealand, 2017.

[239] W.W.-G. Yeh. Reservoir management and operations models: a state-of-the-art review. *Water Resources Research*, 21(12):1797–1818, 1985.

[240] C. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38(2):894–942, 2010.

[241] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985, 2022.

[242] Q.P. Zheng, S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of CO2 in power systems*. Springer Science & Business Media, 2012.

[243] X. Zhu, M.G. Genton, Y. Gu, and L. Xie. Space-time wind speed forecasting for improved power system dispatch. *TEST*, 23(1):1–25, 2014.

[244] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.

[245] L. Zéphyr and C. L. Anderson. Stochastic dynamic programming approach to managing power system uncertainty with distributed storage. *Computational Management Science*, 15:87–110, 2018.