Convergence of Trajectory Following Dynamic Programming algorithms for multistage stochastic problems without finite support assumptions

Maël Forcier*, Vincent Leclère*

June 3, 2022

Abstract

We introduce a class of algorithms, called Trajectory Following Dynamic Programming (TFDP) algorithms, that iteratively refines approximation of cost-to-go functions of multistage stochastic problems with independent random variables. This framework encompasses most variants of the Stochastic Dual Dynamic Programming algorithm.

Leveraging a Lipschitz assumption on the expected cost-to-go functions, we provide a new convergence and complexity proof that allows random variables with non-finitely supported distributions. In particular, this leads to new complexity results for numerous known algorithms. Further, we detail how TFDP algorithms can be implemented without the finite support assumption, either through approximations or exact computations.

1 Introduction

Multistage stochastic programming problems (MSP) are optimization under uncertainty problems where decisions are taken sequentially during stages. Between stages some part of the uncertainty is revealed. These problems have numerous applications, in, for example finance, energy and supply chain (see e.g. [Dup02, WZ05, GZ13] and references therein). Unfortunately, MSP problems are known to be #P-hard ([SN05, Sha06, HKW16]) and numerically challenging especially when the number of stages grow.

More precisely, considering a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, we define a sequence of random variables, called noises, $(\xi_t)_{t \in [T]}$, where $[T]$ stands for $\{1, \ldots, T\}$ and $T$ is the horizon of the problem. Assuming that each $\xi_t$ has finite support of size $n_t$, an MSP problem admits an equivalent deterministic formulation with $O(n_T)$ variables. There are multiple algorithms (for a recent introduction to the topic we recommend [RW21]), each with various extensions and a rich literature, that exploit the special structure of the equivalent deterministic formulation, among which L-Shaped method [YSW99, LL93], and its extension to MSP i.e. nested Bender’s decomposition [Lor80, Bir85], or progressive hedging algorithm [RW91].

However, each of these algorithms are numerically limited to small horizon $T$. For larger horizon we need some additional assumptions on the noises. If they have a limited memory (i.e. that $(\xi_t, \xi_{t+1}, \ldots, \xi_{t+\tau})$ is a Markov chain - for adequate indices) this open the door to Dynamic Programming methods, among which the Stochastic Dual Dynamic Programming (SDDP) algorithm [PP91] algorithm, and its variants (e.g [BDZ17, ZAS19, ACdC20, PWB20]). All these algorithms compute a state trajectory and then follow it to update approximations of cost-to-go functions. We call them Path Following Dynamic Programming (PFDP) algorithms.

1.1 Problem setting

We present here the general setting of multistage stochastic problem (MSP) we are considering in the paper. We also introduce three assumptions that are assumed to hold true throughout the paper.

All random variables (noises $\xi_t$ or states $x_t$) are assumed to be valued, for some adequate integer $n_t$, in $\mathbb{R}^{n_t}$ endowed with its Borel $\sigma$-algebra. To model the constraint of our stochastic problem, we consider for $t \in [T]$, the following Borel-measurable set-valued applications $X_t : \mathbb{R}^{n_{t-1}} \times \Xi_t \rightrightarrows \mathbb{R}^{n_t}$ where $\Xi_t := \text{supp}(\xi_t) \subseteq \mathbb{R}^n$. We further assume, for simplicity, that the first noise is deterministic, that is $\Xi_1 = \{\xi_1\}$. For notational consistency

*CERMICS, Ecole des Ponts, France
we introduce \( x_0 \in \mathbb{R}^{n_0} \) as a parameter, and \( x_t \) as the random variable with support \( \{ x_0 \} \). We define recursively a sequence of reachable set by

\[
X^*_0 = \{ x_0 \} \quad \text{(1a)}
\]
\[
X^*_t = \bigcup_{x_{t-1} \in X^*_{t-1}} \bigcup_{\xi_t \in \Xi_t} X_t(x_{t-1}, \xi_t) \quad \forall t \in [T]. \quad \text{(1b)}
\]

Finally, we consider a sequence of loss functions \( (\ell_t)_{t \in [T]} \) where \( \ell_t : \mathbb{R}^{n_t} \times \Xi_t \to \mathbb{R} \cup \{ +\infty \} \).

**Assumption 1** (Compatibility of constraints). We make the following assumptions, for all \( t \in [T] \),

i) \( \ell_t \) is a proper normal integrand;

ii) for all \( x_t \in X^*_t \), the random variable \( \ell_t(x_t, \xi_t) \) is integrable (in particular \( \ell_t(x_t, \xi_t) < +\infty \ P\)-almost surely);

iii) for all \( x_{t-1} \in X^*_{t-1} \) and almost all \( \xi_t \in \Xi_{t-1} \), \( X_t(x_{t-1}, \xi_t) \) is a non-empty compact subset of \( \mathbb{R}^{n_t} \).

Finally, we say that \( (x_t)_{t \in [1:T]} \) is an *admissible policy* if it is a sequence of random variables such that, for all \( t \in [T] \), \( x_t \in X_t(x_{t-1}, \xi_t) \) \( P\)-almost surely, and \( x_t \) is measurable with respect to \( \sigma(\{\xi_r\}_{r \in [t]}) \). We denote \( X^a \) the set of admissible policies. Then, the multistage stochastic problem (MSP) consists in minimizing over the set of admissible policies the sum of losses, that is

\[
\min_{x \in X^a} \mathbb{E} \left[ \sum_{t=1}^{T} \ell_t(x_t, \xi_t) \right] \quad \text{(MSP)}
\]

**Assumption 1** ensures that (MSP) is well-posed and admits an optimal solution. It also guarantees that we are in a relatively complete recourse setting in the sense that any sequence of variable \((x_r)_{r \leq t}\) satisfying \( x_r \in X_r(x_{r-1}, \xi_r) \), for \( r \leq t \) can be completed into an admissible policy \((x_r)_{r \leq T}\) such that \( \mathbb{E} \left[ \sum_{t=1}^{T} \ell_t(x_t, \xi_t) \right] < +\infty \).

As we are considering Dynamic Programming methods, the following *stagewise independence* assumption is assumed to hold true.

**Assumption 2** (Stagewise independence). \( (\xi_t)_{t \in [T]} \) is a sequence of independent exogeneous random variables, *i.e.* such that the law of \( \xi_t \) is independent of all decisions variables.

Leveraging **Assumption 2**, we can rewrite Problem (MSP) in the following equivalent nested form

\[
\min_{x_t \in X^*_{t-1}} \ell_1(x_1, \xi_1) + \mathbb{E} \left[ \min_{x_2 \in X^*_{t-1}} \ell_2(x_2, \xi_2) + \mathbb{E} \left[ \cdots + \mathbb{E} \left[ \min_{x_T \in X^*_{T-1}} \ell_T(x_T, \xi_T) \right] \right] \right] \quad \text{(2)}
\]

which can be tackled by Dynamic Programming. To this end we introduce the following (backward) Bellman operators. For a measurable proper l.s.c function \( \tilde{V} : \mathbb{R}^{n_1} \to \mathbb{R} \cup \{ +\infty \} \), we denote the Bellman operator of Problem (MSP) applied to \( \tilde{V} \) by

\[
\tilde{B}_t(\tilde{V}) = \begin{cases} \mathbb{R}^{n_t} \times \Xi_{t+1} & \to \mathbb{R} \cup \{ +\infty \} \\ (x_t, \xi_{t+1}) & \mapsto \min_{x_{t+1} \in X_{t+1}(x_t, \xi_{t+1})} \ell_{t+1}(x_{t+1}, \xi_{t+1}) + \tilde{V}(x_{t+1}) \end{cases} \quad \text{(3a)}
\]

Further, note that for \( \tilde{V} \) l.s.c. and finite valued on \( X^*_t \), \( \tilde{B}_t(\tilde{V}) \) is also a normal integrand. We then define,

\[
\tilde{B}_t(\tilde{V}) : x_t \mapsto \mathbb{E} \left[ \tilde{B}_t(\tilde{V})(x_t, \xi_{t+1}) \right] \quad \text{(3b)}
\]

With this notation we define by induction the expected cost-to-go functions \( V_t : \mathbb{R}^{n_t-1} \to \mathbb{R} \)

\[
V_T := 0 \quad \text{(4a)}
\]
\[
V_t := \tilde{B}_t(V_{t+1}) \quad \forall t \in \{0, \ldots, T-1\}. \quad \text{(4b)}
\]

Finally, as the law of \( \xi_1 \) is a dirac on \( \xi_1 \), the value of Problem (MSP) is simply \( V_0(x_0) \).
Remark 1 (Stepwise control). For notational simplicity we chose to consider loss function \( \ell_t \) that only depends on the next state \( x_t \). However, it is worth keeping in mind that these loss functions are often defined as the partial minimum of another normal integrand, i.e.

\[
\ell_t(x_t, \xi) = \inf_{y \in \mathbb{R}^m} \tilde{\ell}(x_t, y, \xi).
\]

In theory, the same problem can be tackled by extending the state vector \( x \) to also contains the step decisions \( y \). However, this is misleading: the theoretical complexity is exponential in the dimension of \( x \), which is in line with the curse of dimensionality of Dynamic Programming. Thus extending the state to include \( y \) falsely seems to imply an increase in the number of iterations required by trajectory following algorithms to converge. For example, in long term electricity management problem it is standard to have decisions \( y \) of dimension a few thousands (thermal generation, transmission on lines...) while the actual state \( x \) (hydroelectric storage) is of dimension a few dozen at most.

We end the presentation of our setting with a non-trivial assumption.

Assumption 3 (Lipschitz). For \( t \in [T] \), we assume that:

i) \( X^+_t \) has a diameter smaller than \( D_t < +\infty \);

ii) the expected cost-to-go function \( V_t \) is \( L_t \)-Lipschitz.

Both part of Assumption 3 are strong requirement, needed for the convergence results, while still being natural in most settings. Part i) is satisfied for example if Assumption 1 holds, \( X_t(x_{t-1}, \cdot) \) is Lipschitz for all \( x_{t-1} \in X^+_{t-1} \), and all \( \Xi_t \) are bounded. Part ii) is satisfied under Assumption 1 in the linear case, or through an extended relatively complete recourse assumption (see [GLP15]) which requires that state \( x_t \) that are slightly outside of \( X^+_t \) are still admissible.

1.2 Review of known convergence results

The SDDP algorithm, and its brethren, called in this paper Trajectory Following Dynamic Programming, relies on the dynamic programming equation \( (4) \). The main idea consists in iteratively refining lower (and sometimes upper) approximations of the expected cost-to-go functions \( V_t \). More precisely, at each iteration, they decide, in a forward phase, trial points at which the approximations should be refined. Then, in a backward phase, they construct cuts, that are functions that under approximate \( V_t \). These cuts are as close as possible to the true expected cost-to-go functions around the trial points. The lower approximations are finally defined as the maximum of computed cuts. This is detailed in Section 2.

To our knowledge, almost all prior works make the following assumption or consider an approximated problem which satisfies this assumption.

Assumption (FSN) (Finitely supported noise). The support of the random process \( (\xi_t)_{t \in [T]} \) is finite.

The first proven convergence result of SDDP algorithm is due to Philpott and Guan [PG08]. In this paper, the authors consider the linear setting. Using Assumption (FSN) they prove that the number of (affine) cuts that can be generated is finite. Then, leveraging the fact that each scenario is sampled an infinite number of time, they prove the almost-sure convergence in a finite number of iterations, without any bound on this number. Later convergence results by [GLP15] (then reformulated and adapted to the risk-averse setting in [Gui16]) showed convergence in a non-linear, convex setting. Again, the proof argues that each scenario are selected randomly an infinite number of time. A technical lemma coupled with Borel-Cantelli’s yields almost-sure asymptotic convergence.

Instead of random sampling, some deterministic sampling, have been proposed. The problem-child algorithm [BDZ17], which maintains both an upper and a lower approximation, proved convergence by showing that the gap between upper and lower bound is non-increasing with the iteration. This algorithm has been extended to convex-concave framework, using saddle-cuts [BDZ18], e.g. allowing for stagewise-dependent objective uncertainty in [DDB20], or risk-averse problem [GTW19]. In other cases, deterministic sampling are considered as a first step for proving the convergence of the randomized version [PG08, Lan20, ZS20].

The above papers all rely on affine, often called Benders’, cuts. Some variants of SDDP, handled by our framework, uses other types of cuts and also have proven asymptotic convergence. Zou et. al. presented a version of SDDP for binary variables, which has an asymptotic convergence proven in [ZAS19] for the convex case, although

---

1 Cost dependence on \( x_{t-1} \) is not considered here simply for notational convenience.

2 We do not necessarily require the knowledge of the diameters or Lipschitz constants.
the proof can be directly adapted to the Lipschitz case. In addition to traditional Benders’ cuts it relies on integer,
Lagrangian and strengthened Benders’ cut, recalled in Appendix A. Stochastic Lipschitz Dynamic Programming
(SLDP) by Ahmed et. al. [ACdC20] uses concave \( L_1 \) cuts instead of affine cuts for any Lipschitz \( V_t \). MIDAS by
Philpott et al. [PWB20] uses step-function cuts for quasi-monotonous \( V_t \) and also fall in this category.

Other works have been dedicated to improve the numerical efficiency of the algorithm. Some methods alleviate
the computational burden of each iteration, like Guigues in [Gui20] which considers inexact cuts, or Bandarra
and Guigues in [BG21] that present cut selection strategies, which delete some cuts from the representation of
\( V_t \). Other methods, like regularization approaches [AP18, VAdOS19, BFFdO20], try to reduce the number of
required iterations. To our knowledge, if they sometimes preserve asymptotic convergence, none of these approaches
provably reduce the number of iterations required to reach an \( \varepsilon \)-solution. These extensions are either handled by
our framework or discussed in Section 2.2.

It is worth noting that the convergence arguments that relies on each possible scenarios being sampled an infinite
number of time are mainly theoretical arguments: due to the sheer number of scenarios, in most applications, the
algorithms sample only a very small subset of scenarios (and probably never twice the same).

In two recent papers [Lan20, ZS19] new approaches were developed, focusing on the state space akin to the
complexity proof of Kelley’s cutting plane algorithm. They independently obtained the first explicit bound on the
number of iterations required to obtain an \( \varepsilon \)-solution. To this end, they fix the error \( \varepsilon \) and define some saturated
points in the state space. These points are such that the gap between the approximated value and the true value
is controlled. Then, leveraging Lipschitz continuity of the value function they control the error in a ball around
the saturated points. As the reachable sets are compact, only a finite number of such balls exists. They then
each provide a deterministic algorithm with proven convergence, and use it as a proxy to bound the expected
number of iterations. Interestingly, the complexity of the deterministic algorithm is polynomial in the horizon \( T 
while the sampled algorithm complexity is exponential in \( T \) as it requires a given event to happen at each stage
simultaneously.

All the convergence proofs recalled here rely on reachable set compactness, relatively complete recourse and
finitely supported noise assumption. They then fall into two categories: either they directly use the Lipschitz
continuity\(^3\) of \( V_t \), or argues that there exists only a finite number of possible cuts (e.g., [PC08, ZAS19]). Our
framework cover all these variants, but the convergence proof presented in Section 2 which is built on [Lan20], does
not require the finitely supported noise Assumption [FSN]. It is instead replaced, for the randomized algorithms, by
a dedicated nested Hoeffding lemma (see Appendix D.1). This is another step toward understanding the practical
convergence of these TFDP algorithms.

Further, without finitely supported noise Assumption [FSN] the standard approach consists in first discretizing
the noise and then solving the discretized problem. A common method consists in sampling the problem through
the Sample Average Approximation (SAA) approaches. The statistical guarantees of this approach are discussed by
Shapiro in [Sha11]. Other sampling strategies are numerically discussed in [HdMDMF11, Löh16]. While never
used, to our knowledge, in the context of TFDP algorithms, there are ways of discretizing the noise distribution in
order to guarantee that the value of the discretized model under (or over) estimate the value of the true problem,
especially in the convex setting, see [BW86, Kuhl06, MAB14, MP18].

An alternative approach could consist in finding a finitely supported, stagewise independent distribution that
minimize the nested-distance [PP12], to provide a good representation of the true problem. Other approaches exist,
like [Fra96, CS05], which uses convexity tools and information relaxation to construct bounds. These approaches
seem more relevant for problems with non-independent noises.

Finally, SDDP has been extended to various problem settings to handle risk aversion (e.g. [STdCS13, GTW19,
DM20]), infinite horizon (e.g. [SD20]), partially observable problems [DMP20]. We briefly discuss extensions to
risk averse setting in the last section, other extensions are outside the scope of this paper.

1.3 Contributions and structure of the paper

Our main contributions are the following:

- we provide a flexible framework (including inexact or regularized computations) for TFDP algorithms for
  finite horizon, risk neutral problems, that encompass at least 14 variants of SDDP summed up in Table 1;
- we provide geometric tools to extend those algorithms to non-finitely supported uncertainties, without sam-
  pling or approximations in the linear case;

\(^3\)Actually MIDAS has a slightly milder requirement (see [PWB20 Eq. (17)]).
• we give a convergence speed result with an upper bound on the (expected) number of iterations to reach an $\varepsilon$-solution for these algorithms (which is new for most of those variants) that does not require the finite support assumption;

• we explain how to adapt those results to the minimax case. Some risk averse or robust cases are seen as a special case.

The remains of the paper is as follows. Section 2 introduces the general framework, discusses some extensions, and the classical ways of obtaining exact or approximated cuts. Section 3 details how to leverage polyhedral geometry to implement an exact SDDP algorithm for non-finitely supported noise in the linear case. Section 4 provides the main convergence and complexity results. Finally, Section 5 briefly reviews extensions to some robust and risk averse settings. Technical proofs and definitions can be found in the appendix.

1.4 Notation

For ease of reference we recall here some notational convention used throughout the paper. $[n]$ is the set of integer between 1 and $n$. Bold font (e.g. $x_t$) denotes a random variable, normal font of the same character (e.g. $x_t$) an element of its support. $k$ and $\kappa$ are indices of iteration, $t$ and $\tau$ are indices of time-step. $L$ represent Lipschitz-constant, $\gamma$ errors allowed in solving the subproblems, $\varepsilon$ characterize the quality of the solution obtained. $x$ denote a state, $X$ a set of state (e.g. $X_t^r$ is the set of reachable state) and $\mathcal{X}$ a set-valued application representing set of states (e.g. $\mathcal{X}_t(x, \xi)$ is the set of admissible next state, $\mathcal{X}^{\gamma}_t(V)(x, \xi)$ is the set of $\gamma$-optimal next state...), $\xi$ a noise, $\ell_t$ is the loss function at time $t$, $V$ a cost-to-go function, $f$ a cut. Overline is used for upper-approximation, underline for lower-approximation, hat (as in $\hat{B}$) represent functions parametrized by $\xi$, and their counterpart without hat (as in $B$) being their expectation. $B$ (resp. $F$) represent backward (resp. forward) Bellman operators. $\text{ri}(X)$ is the relative interior of $X$. 
<table>
<thead>
<tr>
<th>Algorithm’s name</th>
<th>Paper</th>
<th>Node selection: Choice $\xi^k_t$</th>
<th>$\mathcal{F}_t$</th>
<th>$V^k_t$</th>
<th>$\nabla^k_t$</th>
<th>Hypothesis</th>
<th>Complexity known</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDDP</td>
<td>PP91</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>$V_t$</td>
<td>Convex</td>
<td>✓</td>
</tr>
<tr>
<td>EDDP</td>
<td>Lan20</td>
<td>Explorative</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>$V_t$</td>
<td>Convex</td>
<td>✓</td>
</tr>
<tr>
<td>APSDDP</td>
<td>SS22</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Adaptive partition</td>
<td>$V_t$</td>
<td>Linear</td>
<td>×</td>
</tr>
<tr>
<td>SDDiP</td>
<td>ZAS19</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Lagrangian or integer cuts</td>
<td>$V_t$</td>
<td>Mixed Integer Linear</td>
<td>×</td>
</tr>
<tr>
<td>MIDAS</td>
<td>PWB20</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Step cuts</td>
<td>$V_t$</td>
<td>Monotonic Mixed Integer</td>
<td>×</td>
</tr>
<tr>
<td>SLDI</td>
<td>ACdC20</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Reverse norm cuts</td>
<td>$V_t$</td>
<td>Non-Convex</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>BDZ17</td>
<td>Problem child</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>Epigraph as convex hull</td>
<td>Convex</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>BDZ18</td>
<td>Problem child</td>
<td>Exact</td>
<td>Benders $\times$ Epigraph</td>
<td>Hypograph $\times$ Benders</td>
<td>Convex-Concave</td>
<td>×</td>
</tr>
<tr>
<td>R DD</td>
<td>GTW19</td>
<td>Deterministic</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>Epigraph as convex hull</td>
<td>Robust</td>
<td>×</td>
</tr>
<tr>
<td>NDDP</td>
<td>Gui20</td>
<td>Random sampling</td>
<td>Inexact</td>
<td>Inexact Lagrangian cuts</td>
<td>$V_t$</td>
<td>Convex</td>
<td>×</td>
</tr>
<tr>
<td>TDP</td>
<td>ACT20</td>
<td>Problem child</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>Min of quadratic</td>
<td>Convex</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>ZS19</td>
<td>Random or Problem</td>
<td>Regularized</td>
<td>Generalized conjugacy cuts</td>
<td>Norm cuts</td>
<td>Mixed Integer Convex</td>
<td>✓</td>
</tr>
<tr>
<td>NDDP</td>
<td>ZS20</td>
<td>Random or Problem</td>
<td>Regularized</td>
<td>Benders cuts</td>
<td>Norm cuts</td>
<td>Distributionally Robust</td>
<td>✓</td>
</tr>
<tr>
<td>DSDDP</td>
<td>LCC$^+$20</td>
<td>Random sampling</td>
<td>Exact</td>
<td>Benders cuts</td>
<td>Fenchel transform</td>
<td>Linear</td>
<td>×</td>
</tr>
</tbody>
</table>

Table 1: Synthesis of algorithms following the same framework
2 Trajectory Following Dynamic Programming framework

Various extensions of the Stochastic dual dynamic programming (SDDP) algorithm have been developed for different set of assumptions. In this section, we first present a generic algorithmic framework for TFDP algorithms (see Algorithm 2) for risk-neutral multistage stochastic program that encompasses multiple known algorithms (see Table 4). These algorithms consider under and over approximation of the expected cost-to-go functions. The under-approximations are defined as the maximum of basic functions called cuts (some classical cuts are recalled in Appendix [X]). The upper-approximations are more diverse and not always computed. The remains of the section details how to obtain cuts, in particular Section 2.3 presents the finitely supported case, while Section 2.4 builds approximated cuts in the convex case.

2.1 Algorithm

The flexible framework of Algorithm 2 defines improving lower approximations \( \mathcal{V}^k_t \) (resp. upper approximation \( \mathcal{V}^1_t \)) of the expected cost-to-go functions \( V_t \). Each iteration \( k \) of the algorithm consists in a forward phase to determine where to refine the approximations, followed by a backward phase to actually refine the approximations.

During the forward phase, we generate a trajectory \( x^k_t, \ldots, x^k_{T-1} \). Each \( x^k_t \) is chosen as an (almost) optimal decision at time \( t \) starting from state \( x^k_{t-1} \), knowing that the random variable \( \xi_t \) takes the value \( \xi^k_t \), and considering that the cost-to-go is given by the lower approximation \( \mathcal{V}^k_t \). This is encapsulated in the forthcoming notion of forward Bellman operator. We denote \( \gamma\text{-arg min}_{x \in X} f(x) \) the set of \( x \in X \) such that \( f(x) \leq \inf_{x \in X} f(x) + \gamma \). We now define, for any Lipschitz (on \( X_1 \)) function \( \tilde{V} \), the set \( \mathcal{X}^k_{\gamma,t}(\tilde{V}) \) of \( \gamma \)-optimal solution of the parametrized stage \( t \) problem with cost-to-go function \( \tilde{V} \), that is,

\[
\mathcal{X}^k_{\gamma,t}(\tilde{V}) : (x, \xi) \mapsto \gamma \text{-arg min}_{y \in X} \ell_t(y, \xi) + \tilde{V}(y).
\]

Since \( \ell_t + \tilde{V} \) is a normal integrand, [RW09, Corollary 14.33] guarantees measurability of \( \mathcal{X}^k_{\gamma,t}(\tilde{V}) \). Further, as \( \mathcal{X}^k_{\gamma,t}(\tilde{V}) \) is compact, there exists ([RW09 Cor 14.6]) a measurable selection of \( \xi \mapsto \mathcal{X}^k_{\gamma,t}(\tilde{V})(x, \xi) \) for all \( x \in X_1 \).

The following definition mathematically formalizes the selection choice.

**Definition 2** (Forward operator). We say that \( \mathcal{F}_t \) is a \( \gamma^k_{t+1} \)-forward operator if, for all functions \( \tilde{V} : \mathbb{R}^{n+1} \rightarrow \mathbb{R} \cup \{+\infty\} \), Lipschitz on \( X_{t+1} \) and \( x \in X_t \), \( \mathcal{F}_t(\tilde{V})(x, \cdot) \) is a measurable selection of \( \mathcal{X}^k_{\gamma,t}(\tilde{V})(x, \xi) \).

During the backward phase, we refine the approximations \( \mathcal{V}^k_t \) and \( \mathcal{V}^1_t \), they are both assumed to be Lipschitz on \( X_t \). Further, the lower approximations \( \mathcal{V}^k_t \) is defined as the maximum of a finite number of cuts: \( \mathcal{V}^k_t = \max_{k \leq k} f^k_t \).

For Algorithm 2 to converge we make the following assumption on the approximations computed.

**Assumption 4** (Admissible approximations). The computed cuts \( f^k_t \) of \( B_t(\mathcal{V}^k_{t+1}) \) at \( x^k_t \) satisfy:

i) \( f^k_t(x^k_t) \geq B_t(\mathcal{V}^k_{t+1})(x^k_t) - \gamma_t \)

ii) \( f^k_t \) is valid, i.e. \( f^k_t \leq B_t(\mathcal{V}^k_{t+1}) \)

iii) \( \mathcal{V}^k_t \) is \( \mathcal{L}_t \)-Lipschitz

On the other hand the upper approximation \( \mathcal{V}^k_t \), not necessarily computed, shall satisfy the following properties:

iv) \( \mathcal{V}^k_t(x^k_t) \leq B_t(\mathcal{V}^k_{t+1})(x^k_t) + \gamma_t \) (tightness)

v) \( \mathcal{V}^k_t \geq B_t(\mathcal{V}^k_{t+1}) \) (validity)

vi) \( \mathcal{V}^k_t \leq \mathcal{V}^k_{t-1} \) (monotonicity)

vii) \( \mathcal{V}^k_t \) is \( \mathcal{L}_t \)-Lipschitz

4 In some common cases, the upper approximations are chosen as \( \mathcal{V}^k_t = V_t \) but never evaluated.

5 This choice is comparable to selecting a stage solver which always return the same solution among the set of optimal solutions.
Data: Random variables $\xi_t$, cost function at each step $\ell_t$, constraints set-valued function $X_t$, initial state $x_0$, $\gamma^k$-forward operators $F_t$.

1. $V_i^0 = -\infty$ and $\overline{V}_i^0 = +\infty$ for $t \in [T]$;
2. for $k \in \mathbb{N}$ do
   /* Forward phase */
   3. Set $x_0^k = x_0$;
   4. for $t = 1 : T - 1$ do
      5. Choose $\xi_t^k \in \text{supp}(\xi_t)$ ;
      6. Let $x_t^k = F_{t-1}(V_{t-1}^{k-1})(x_{t-1}^k, \xi_t^k)$ ;
   end
   /* Backward phase */
   7. Set $V_T^k = \overline{V}_T^k = 0$;
   8. for $t = T - 1 : -1 : 1$ do
      9. Find a $L_t$-Lipschitz on $X_t^*$, valid and $\gamma_t$-tight cut $f_t^k$ of $B_t(V_{t+1}^k)$ at $x_t^k$, i.e. such that
         $f_t^k(x_t^k) \geq B_t(V_{t+1}^k)(x_t^k) - \gamma_t$ and $f_t^k \leq B_t(V_{t+1}^k)$;
      10. Set $V_t^k = \max(V_{t-1}^k - f_t^k)$;
      11. Define $\overline{V}_t^k$ satisfying Assumption 4, Items iv) to vii);
   end
end

Algorithm 1: A general framework for TFDP algorithms

For the algorithm to be well-defined we need to guarantee the existence of cuts and upper approximation satisfying previous assumption, as formally assumed now:

**Assumption 5.** For every $t \in [T]$ and $k \in \mathbb{N}^+$, there exists at least one cut $f_t^k$ of $B_t(V_{t+1}^k)$ satisfying Assumption 4.

This assumption is for example ensured through relatively complete recourse in the linear setting [PP91], through extended relatively complete recourse in the convex setting [GLP15], through relatively complete continuous recourse in the binary setting [ZAS19], and relatively complete recourse and Lipschitz assumption in the Lipschitz setting of [ACJC20].

Remark 3 (Asymmetry of upper and lower approximations). The framework is not symmetrical in its treatment of the upper and lower cost-to-go approximations. Indeed, Line 4 should not be done with the upper approximations as it would restrict the exploration of the state space. For example, assume that $\overline{V}_t^k$ are (slightly Lipschitz-regularized) indicator functions of a single point, then the forward phase would always produce the same trajectory, and the upper bound would not be updated.

Further, multiple TFDP algorithm do not actually compute $\overline{V}_t$, simply setting it to the true expected cost-to-go $V_t$ (for iterations bounds).

We have a first monotonicity result.

**Lemma 4.** Under Assumptions 4 to 5, for all $k \in \mathbb{N}$, $t \in [T - 1]$ and $x \in \mathbb{R}^n$, we have

\[
V_t^{k-1}(x) \leq V_t^k(x) \leq B_t(V_{t+1}^k)(x) \leq V_t(x) \leq B_t(\overline{V}_{t+1}^k)(x) \leq \overline{V}_t^k(x) \leq V_t^{k-1}(x)
\] (5a)

\[
V_t(x) \leq B_t(\overline{V}_{t+1}^k)(x) \leq \overline{V}_t^k(x) \leq V_t^{k-1}(x)
\] (5b)

In particular, the gap can only decrease

\[
0 \leq V_t^k(x) - \overline{V}_t^k(x) \leq V_t^{k-1}(x) - \overline{V}_t^{k-1}(x)
\] (6)

**Proof.** Direct by double induction on $t$ and $k$ and monotonicity of the Bellman operator. \hfill \Box

Remark 5 (The standard SDDP algorithm). The most common TFDP algorithm is the stochastic dual dynamic programming (SDDP). It was originally designed by Pinto and Pereira [PP91] for multistage stochastic linear problems. In SDDP, the value of the noise $\xi_t^k$, chosen in Line 4 is drawn randomly on $\text{supp}(\xi_t)$ which is assumed to be finite. The lower approximations are defined as maximum of affine cuts. For each $\xi \in \text{supp}(\xi_t)$, computing

\[\overline{V}_t^k \equiv -\infty\] for $t \in [T]$ still provide an admissible policy through the forward Bellman operators which has interesting properties, see [LCC+20].
\( \bar{B}_t(V^{k-1}_{t+1})(x^k_t, \xi) \) consists in solving a LP, and standard linear programming duality yields a subgradient \( \bar{\alpha}^k_t(\xi) \in \partial B_t(V^{k-1}_{t+1})(x^k_t, \xi) \) and value \( \bar{\beta}^k_t(\xi) = \bar{B}_t(V^{k-1}_{t+1})(x^k_t, \xi) \). Taking the expectation, we set \( \alpha^k_t = \mathbb{E}[\bar{\alpha}^k_t(\xi)] \) and \( \beta^k_t = \mathbb{E}[\bar{\beta}^k_t(\xi)] \), to define the so-called Benders’ cut

\[
 f^k_t : x_t \mapsto \alpha^k_t \top(x_t - x^k_t) + \beta^k_t.
\]

Under relatively complete recourse assumption, the cuts can be assumed to be \( L_t \)-Lipschitz. Further, in this simple setting, all errors are null: \( \gamma^k_t = \tau^k_t = \gamma^k_F = 0 \). Finally, no upper bound are computed and the complexity results of Section 4 are obtained by taking \( \nabla^k_t = V^k_t \).

Algorithm 1 is a flexible framework, and some lines remain to be detailed, which we now discuss.

**Node selection choice in Line 5** Most TFDP algorithms choose \( \xi^k_t \) by drawing it randomly according to the law of the random variable \( \xi^k_t \). The forward phase can then be seen as a Monte Carlo method for finding a trajectory \( x^k_1, \ldots, x^k_T \). Then, it is also possible to choose \( \xi^k_t \) thanks to quasi Monte-Carlo methods.

Another way of choosing \( \xi^k_t \) consists in picking the \( \xi \in \text{supp}(\xi_t) \) that maximizes a certain criterion. In [BDZ17], Bauke, Downward and Zakeri suggested to chose \( \xi^k_t \) such that \( x^k_t \) maximizes the gap between the upper and lower approximations, i.e., \( V^k_t(x^k_t) - V^k_t(x^k_{t-1}) \). They called this choice of \( \xi^k_t \) the problem child node selection. In [Lan20], Lan presented the Explorative Dual Dynamic Programming algorithm, where \( \xi^k_t \) is chosen so that \( x^k_t \) is the most distinguishable point, i.e. such that \( x^k_t \) is far from the previous computed points, see Eq. (46b), we speak of explorative node selection.

The proofs of convergence are harder to derive when \( \xi^k_t \) is chosen randomly, and the best upper bound known on the number of iterations of these algorithms are exponential in the horizon \( T \). In comparison, when \( \xi^k_t \) is chosen deterministically as the problem child or as the most distinguishable point, the number of iterations is bounded by a polynomial in \( T \). However, random sampling is often more efficient in practice (and easier to implement). We discuss the complexity results in Section 4.

**Forward operator choice in Line 6** In most algorithms, we assume that \( \gamma^k_t = 0 \) for all \( t \in [T-1] \), thus \( \mathcal{F}_{t-1}(V)(x, \cdot) \) is a measurable selection of \( \arg\min_{y \in X_t(x)} \ell_t(y, \cdot) + \bar{V}(y) \). There has also been proposition to use inexact cuts [Gin20] to alleviate computational burden of each iteration.

Further, there have been various propositions to regularize the SDDP algorithm, see [APIS18, VAdOS19, GLT20]. They mostly boil down to choosing a different forward operator, e.g., by adding a regularization term, which can be seen as \( \gamma^F_t \)-forward operator with \( \gamma^F_t \neq 0 \). For example, one can choose \( \mathcal{F}_{t-1}(V)(x, \xi) \) as a proximity operator

\[
 \text{prox}_{\ell_t(\cdot, \xi) + \bar{V}(\cdot, \cdot)}(\tilde{y}) := \arg\min_{y \in X_t(x)} \ell_t(y, x) + \bar{V}(y) + \alpha \|y - \tilde{y}\|_2^2.
\]

In that case, if \( X_t(x, \xi) \) has a finite diameter \( D \), for \( y = \mathcal{F}_{t-1}(V)(x, \xi) \), we have \( \ell_t(y, x) + \bar{V}(y) \leq \min_{y' \in X_t(x)} \ell_t(y', x) + \bar{V}(y') + \alpha D \). Then, \( \mathcal{F}_t \) is an \( \alpha D \)-forward operator.

Finally, it is important that the algorithm use a single \( \gamma^F_t \)-forward operator. Indeed, if the set of \( \gamma^F_t \)-optimal solutions \( \mathcal{X}^F_{\gamma^F_t}(V)(x, \xi) \) is not reduced to a single point, the convergence results only holds for the points selected by the forward operator. This remark is not only theoretical and have implications in practice: to be safe one should use the same solver (and parameters) during the training phase and exploitation phase of the algorithm. For example, consider a problem with two equivalent storage and that only one of them is required to provide an optimal solution. Consider two forward operators, the first one, \( \mathcal{F}^1_{t-1} \), prefers using the first storage while the second, \( \mathcal{F}^2_{t-1} \), prefers using the second storage. Now assume that the algorithm ran until convergence with \( \mathcal{F}^2_{t-1} \) yielding the approximations \( \bar{V}^\infty \). Then, \( \bar{V}^\infty \) correctly evaluates the value of the first storage, but has no information on the second. Consequently, a trajectory given by \( \mathcal{F}^2_{t-1}(\bar{V}^\infty) \) might be far from optimal. A discussion of this fact, and practical answers, can be found in [Dow18 §2.7].

**Cuts \( f^k_t \) choice in Line 10** We need to compute cuts \( f^k_t \) to approximate \( B_t(V^{k-1}_{t+1}) \) in the neighborhood of \( x^k_{t-1} \). Recall that in Eq. (3a), \( B_t \) is defined as an expectation of parametric Bellman operators \( B_t(V^{k-1}_{t+1}) = \mathbb{E}[\bar{B}_t(V^{k-1}_{t+1})(\cdot, \xi)] \) Eq. (3a). Then, we can compute the average cut \( f^k_t \) thanks to parametric cuts \( f^k_{i, \xi} \). In the finitely supported case in Section 2.3, we show that we can compute the average cut \( f^k_t \) directly by taking \( f^k_t = \mathbb{E}[\hat{f}^k_{i, \xi}] \) whereas in the convex, non finitely-supported case, we present in Section 2.4 methods to approximate \( \mathbb{E}[f^k_{i, \xi}] \). Finally, exact methods for linear problems are developed in Section 3. Furthermore, depending on the problem structure, there exist several types of parametrized cuts \( f^k_{i, \xi} \) in the literature. We recall them in Appendix A.
In the finitely supported case, instead of computing an average cut \( f \) work of Algorithm 1 by considering that the cut \( t \) compute, for each time step \( k \) iterations, another idea is to delete some cuts. For example, we can decide to delete only the dominated cuts, i.e., results are still valid in this setting. Unfortunately, finding which cut is dominated is time-consuming which do not the cuts that do not affect the values of the approximations \( \tau^k \).

Cut selection. After many iterations, the number of cuts can slow down the new iterations. To speed up SDDP iterations, another idea is to delete some cuts. For example, we can decide to delete only the dominated cuts, i.e., the cuts that do not affect the values of the approximations \( \tau^k \). The monotonicity property and the complexity results are still valid in this setting. Unfortunately, finding which cut is dominated is time-consuming which do not.

Upper approximations \( \tau^k \) choice in Line 12

In most TFDLP algorithms, no upper bound function is computed. In that case, we just set \( \tau^k_t \equiv V_t \) in the convergence proof. However, some algorithms rely on the computation of these upper bounds, for example for computing a problem-child node selection. In the convex case, assume that we have, for \( t \in [T] \), some points \( (x^t_{\xi}, \pi^t_{\xi}) \) that are in the epigraph of \( V_t \). Now define \( \tau^k_t \) such that 
\[
\text{epi}(\tau^k_t) = \text{Conv} \left( (x^t_{\xi}, \pi^t_{\xi}) \right) + \{ (x, z) \in \mathbb{R}^{n_t} \times \mathbb{R} | \sum \|x\|_1 = z \} \subseteq \text{epi}(V_t).
\]

Then, \( \tau^k_t \) is an upper-approximation \( \tau^k_t \) of \( V_t \) on \( X_t \). Computing points \( (x^t_{\xi}, \pi^t_{\xi}) \) in the epigraph of \( V_t \) can be done either throughout the algorithm as in the problem-child approach \cite{BDZ17}, or in batch backward in time for a given set of trajectories as suggested by \cite{PdMF13}. Upper approximation functions can also be obtained through duality see \cite{LCC20, CL21}.

2.2 Extensions of the framework

Although we tried to present a general framework, for the sake of simplicity, Algorithm 1 does not integrate every variants of SDDP. We now discuss how this framework can be extended and if the complexity results and proofs are still valid with these new extensions.

Multiple forward phases. In practice, SDDP is often implemented with multiple forward phases, i.e., at iteration \( k \) we compute \( N \) forward phases \( (x^i_t)_{t \in [T \ldots]} \), in parallel. Consequently, in the backward phase we compute, for each time step \( t \in [T - 1] \), \( N \) tight and valid cuts \( (f^k_{t, i})_{i \in [N]} \). This variation is included in the framework of Algorithm 1 by considering that the cut \( f^k_t \) is the maximum over \( i \in [N] \) of all cuts \( f^k_{t, i} \). The complexity results follow directly (in iteration number).

Multicut. In the finitely supported case, instead of computing an average cut \( f^k_t \) of the expected cost-to-go function \( V_t \), it is possible to store for each \( \xi \in \text{supp}(\xi_t) \) a cut \( f^k_{t, \xi} \) of the cost-to-go function \( V_t(\cdot, \xi) \). Unlike the single cut case where \( V^k_t(\cdot) = \max_{x \in k} f^k_t(\cdot) \), in the multicut case, we compute approximation function as \( V^k_t(\cdot) = \sum_{\xi \in \text{supp}(\xi_t)} P(\xi) \max_{x \in k} f^k_{t, \xi}(\cdot) \). Up to a slight reinterpretation, by considering a global cut \( f_t(\cdot) = \sum_{\xi \in \text{supp}(\xi_t)} P(\xi) \max_{x \in k} f^k_{t, \xi}(\cdot) \), this variation is covered by our framework.

However, with continuous random variables, the notion of multiple cuts is not well-defined.

Cut computation in forward. Another variation of SDDP consists in computing the cuts during the forward phase (and no backward phase). In this variant, the cuts do not approximate \( B_t(V^k_{t+1}) \) and \( B_t(V^{k+1}_{t+1}) \) in the neighborhood of \( x^t_k \), but approximate \( B_t(V^k_{t+1}) \) and \( B_t(V^{k+1}_{t+1}) \) in the same neighborhood. Although this variant is not handled by the framework, all proofs can be adapted in a straightforward manner. More precisely, we only need to adapt the forthcoming proof of Lemma 39. In particular, in the proof of Lemma 39 we obtain directly Eq. (87c) and Eq. (88c), without using the monotonicity, because we approximate \( B_t(V^k_{t+1}) \) and \( B_t(V^{k+1}_{t+1}) \).

Cut selection. After many iterations, the number of cuts can slow down the new iterations. To speed up SDDP iterations, another idea is to delete some cuts. For example, we can decide to delete only the dominated cuts, i.e., the cuts that do not affect the values of the approximations \( V^k_t \). The monotonicity property and the complexity results are still valid in this setting. Unfortunately, finding which cut is dominated is time-consuming which do not.

![Figure 1: An example of upper and lower approximations](image-url)
make this method numerically efficient. Instead, we often use some heuristics to delete cuts which are probably dominated. However, these heuristics do not guarantee that we have the monotonicity property of approximations. Then, the complexity and convergence results seems harder to obtain. See [BG21] for an asymptotic convergence result on SDDP with cut selection.

**Adaptive partition based methods** In [SL15], Song and Luedtke presented the adaptive partition based methods (APM) to solve 2-stage linear problems by partitioning the set of scenarios. It was then adapted to the multistage case in [SS22] where Siddig and Song proposed an adaptive partition based SDDP, in both case under the finitely supported noise Assumption [FSN]. The idea of APM is to replace the expected cost-to-go function $V = \mathbb{E}[\hat{V}(\cdot, \xi)]$ by a partitioned expected cost-to-go function $V_P = \sum_{\xi \in P} P[\xi \in P] V(\cdot, \mathbb{E}[\xi | \xi \in P])$ where $P$ is a partition of the uncertainty set $\Xi$. A partition $P$ is said to be tight at $\hat{x}$, if $V_P(\hat{x}) = V(\hat{x})$, valid if $V_P(x) \leq V(x)$ for all $x \in \mathbb{R}^n$ and adapted to $\hat{x}$ if it is valid and tight at $\hat{x}$. Then, when $P$ is a partition adapted to $\hat{x}$, we can see the partitioned expected cost-to-go function $V_P$ as a valid and tight cut of $V$ at $\hat{x}$. Such cuts represent the tangeant cone of $\text{epi}(B_i(V_{t+1}))$ at $\hat{x}$ where Benders’ cut represent a single tangent plane (see [FL21] §3.2]). APM methods were extended to general distribution in [RPM21]. In [FL21], the authors provided a necessary and sufficient condition for a partition to be adapted (without Assumption [FSN]) as well as a geometric method to obtain a valid and adapted partition. In particular, APM SDDP algorithm of [SS22] is a TFDP algorithm falling in the framework of Algorithm [1] It can be adapted to the non-finitely supported case through the discussion in Section [3].

### 2.3 Cuts with finitely supported distribution

We now focus on finding a cut in Line 10 of Algorithm 1. More precisely, we want to approximate $B_i(V_t(x))$ in the neighborhood of $x_{t-1}$. Recall that $B_i$ is defined as an expectation of parametric Bellman operators $B_i$ (see Eq. (3)). When the distribution of $\xi_t$ is finitely supported, computing a cut of $\hat{B}_i(\hat{V})(\cdot, \xi)$ for each elements $\xi \in \text{supp}(\xi_t)$ automatically yields a cut for $B_i(\hat{V})$.

**Proposition 6.** Assume that $\xi_t$ is finitely supported with $p_\xi := P[\xi_t = \xi]$, for all $\xi \in \text{supp}(\xi_t)$, then

$$B_i(\hat{V})(x) = \sum_{\xi \in \text{supp}(\xi_t)} p_\xi \hat{B}_i(\hat{V})(x, \xi)$$

(8a)

For every $\xi \in \text{supp}(\xi_t)$, assume that $\hat{f}_\xi$ is a valid and $\gamma_{\xi}$-tight cut of function $\hat{B}_i(\hat{V})(\cdot, \xi)$ at $\hat{x}$, then we have

$$f := \sum_{\xi \in \text{supp}(\xi_t)} p_\xi \hat{f}_\xi$$

is a valid and $\gamma_{\xi}$-tight cut of $B_i(\hat{V})$ at $\hat{x}$ with $\gamma_{\xi} := \sum_{\xi \in \text{supp}(\xi_t)} p_\xi \gamma_{\xi}$

(8b)

In this finitely-supported distribution setting, it remains to find cuts of the function $\hat{B}_i(\hat{V})(\cdot, \xi)$. There exist several tight and valid cuts depending on the structure of $\ell_t$ and $X_t$. We present classical cuts of the literature in Appendix A where we detail under which conditions these cuts are tight and valid and show how to compute them.

### 2.4 Approximated cuts in the convex case

In this section, we now turn to obtaining approximated cuts leveraging convexity. We present a method based on the inequalities of Jensen and Edmundson-Madanski, adapting the results of Birge and Wets [BW86] to our setting, see also [KM70] 4.7.

We start by recalling two well-known useful convex inequalities.

**Proposition 7** (Jensen’s and Edmundson-Madanski inequalities). Let $g: \mathbb{R}^l \mapsto \mathbb{R}$ be a convex function and $\xi$ be a random variable. Assume that there exists a polytope $\Xi \subset \mathbb{R}^l$ containing $\text{supp}(\xi)$.

For any $\xi \in \Xi$ we denote $S_\xi$ the set of barycentric coordinates of $\xi$, that is the set of coefficients $\{\sigma_\xi(v)\} v \in \text{Vert} \Xi(\xi) \in [0, 1]|\text{Vert} \Xi|\}$ such that $\xi = \sum_{v \in \text{Vert} \Xi(\xi)} \sigma_\xi(v) \xi(v)$ and $\sum_{v \in \text{Vert} \Xi(\xi)} \sigma_\xi(v) = 1$. Let $\xi \mapsto (\sigma_\xi(v)) v \in \text{Vert} \Xi$ be any mea-
Moreover, if \( g \) is Lipschitz with constant \( L \) and \( \Xi \) has a diameter \( D \), the gap is at most \( LD \):

\[
\sum_{v \in \text{Vert}(\Xi)} \mathbb{E}[\sigma_{\Xi,v}(\xi)]g(v) \leq g(\mathbb{E}[\xi]) + LD \tag{10}
\]

**Proof.** Left-hand side of Eq. (9) is the classical Jensen inequality. Let \( \xi \in \Xi \), as \( \langle \sigma_{\Xi,v} \rangle_{v \in \text{Vert}(\Xi)} = \text{barycentric coordinates} \) in \( \Xi \), we have, by convexity of \( g \), \( g(\xi) \leq \sum_{v \in \text{Vert}(\Xi)} \sigma_{\Xi,v}(\xi)g(v) \). Taking the expectation leads to the right-hand side of Eq. (9) called Edmundson-Madanski inequality.

Assume now that \( \Xi \) has diameter \( D \). Since \( \Xi \) is convex, \( \mathbb{E}[\xi] \in \Xi \), thus for all \( v \in \text{Vert} \Xi \), \( \| \mathbb{E}[\xi] - v \| \leq D \). Further, as \( g \) is Lipschitz, we have \( \| g(v) - g(\mathbb{E}[\xi]) \| \leq LD \). Taking the convex combination yields Eq. (10). \( \square \)

These inequalities can be refined. Let \( \mathcal{P} \) be a finite collection of almost surely disjoint polyhedra covering \( \text{supp}(\xi) \), i.e. \( \text{supp}(\xi) \subset \bigcup_{P \in \mathcal{P}} P \) and \( \mathbb{P}[P \cap P'] = 0 \) if \( P \neq P' \in \mathcal{P} \). Then, by the law of total expectation, \( \mathbb{E}[g(\xi)] = \sum_{P \in \mathcal{P}} \mathbb{P}[P] \mathbb{E}[g(\xi)|P] \). Applying Jensen and Edmundson-Madanski inequalities to each term of this sum, we get

\[
\sum_{P \in \mathcal{P}} \mathbb{P}[P]g(\mathbb{E}[\xi|P]) \leq \mathbb{E}[g(\xi)] \leq \sum_{P \in \mathcal{P}} \mathbb{P}[P] \sum_{v \in \text{Vert}(P)} \mathbb{E}[\sigma_{P,v}(\xi)]g(v) \tag{11}
\]

In particular, if all polyhedra \( P \in \mathcal{P} \) have a diameter smaller than \( d \), the gap can be bounded by \( Ld \).

We now get back to the problem of Line 10 of Algorithm 1 where we want to approximate \( B_t(\sum_{k} x^{k}_{t-1}) \) in the neighborhood of \( x^{k}_{t-1} \). Recall that \( B_t \) is defined as an expectation of parametric Bellman operators \( \tilde{B}_t \) (see Eq. (3)). Unlike in Section 2.3 where the random variable where finitely supported, we cannot write the expected cut as a finite sum of parametric cuts. However, the Jensen and Edmundson-Madanski inequalities allow us to derive approximate cuts and upper bound functions.

**Proposition 8.** Assume that \( \ell_t \) is a jointly convex function with Lipschitz constant \( L \). Let \( \mathcal{P} \) be a finite collection of almost surely disjoint polyhedra covering \( \text{supp}(\xi) \), such that any \( P \in \mathcal{P} \) has a diameter smaller than \( d \in \mathbb{R}_+ \). Denote for each \( P \in \mathcal{P} \), \( p_P := \mathbb{P}[P] \) and \( \xi_P := \mathbb{E}[\xi|P] \).

For every \( P \in \mathcal{P} \), assume that \( \tilde{f}_{P} \) is a valid and \( \gamma_{t,P} \)-tight cut of the parametric function \( \tilde{B}_t(\tilde{V})(\cdot,\xi_P) \) at \( \tilde{x} \), then by defining \( \gamma_t := Ld + \sum_{P \in \mathcal{P}} p_P \gamma_{t,P} \), we have

\[
\tilde{f} := \sum_{P \in \mathcal{P}} p_P \tilde{f}_{P} \text{ is a valid and } \gamma_t \text{-tight cut of } B_t(\tilde{V}) \text{ at } \tilde{x} \tag{12}
\]

For every \( P \in \mathcal{P} \) and \( v \in \text{Vert} P \), assume that \( \tilde{T}_{v,P} \) satisfies \( \tilde{T} \geq \tilde{B}_t(\tilde{V})(\cdot,v) \) and \( \tilde{T}(\tilde{x}) \leq \tilde{B}_t(\tilde{V})(\tilde{x},v) + \gamma_{t,P} \), then by defining \( \gamma_t := Ld + \sum_{P \in \mathcal{P}} p_P \sum_{v \in \text{Vert}(P)} \mathbb{E}[\sigma_{P,v}(\xi)]\gamma_{t,P} \), we have

\[
\tilde{T} := \sum_{P \in \mathcal{P}} p_P \sum_{v \in \text{Vert}(P)} \mathbb{E}[\sigma_{P,v}(\xi)]T_{P,v} \text{ satisfies } \tilde{T} \geq B_t(\tilde{V}) \text{ and } \tilde{T}(\tilde{x}) \leq B_t(\tilde{V})(\tilde{x}) + \gamma_t \tag{13}
\]
This result can also be adapted to “saddle” cost functions, i.e. functions that are convex in some coordinates of $\xi$ and concave in the other coordinates of $\xi$, by using both inequalities according to the sign of convexity, see e.g. [Kuh06, §4].

3  Exact SDDP in the linear case with generic distributions

In this section, we consider the particular case of multistage linear stochastic programming i.e. Problem (MSP)

$$\begin{align*}
\text{Problem (MSP)}
\end{align*}$$

where, for all $t \in [T]$, $\xi_t = (A_t, B_t, b_t, c_t)$, $\ell_t(x_t, \xi_t) = c_t^\top x_t$ is linear and $X_t(x_{t-1}, \xi_t) = \{x_t \in \mathbb{R}^n \mid A_t x_t + B_t x_{t-1} = b_t, x_t \geq 0\}$ is a polyhedron.

In this linear case, we present an algorithm that compute a partition adapted to a fixed $\hat{x}$, i.e., a valid partition tight at $\hat{x}$. In other words, we can discretize our problem without getting an approximation error. To do so we leverage the polyhedral geometry for linear stochastic problem tools developed in [FGL21] and [FL21]. For the sake of completeness, an introduction to those tools can be found in Appendix [3.1] (see [DLRS10] for a more complete reference).

We start by reformulating the stage problem (3a) as a standard two-stage linear program in Section 3.1. Section 3.2 formally defines the notion of adapted partition that allows to compute cuts. Section 3.3 leverages the geometric tools to construct an explicit adapted partition for the multistage linear setting.

3.1  Expected cost-to-go function in standard form

We make the following assumptions:

3.1. Expected cost-to-go function in standard form

1. $A_t$ has a finitely supported distribution;
2. $c_t$ and $(B_t, b_t)$ are independent$^9$
3. the lower expected cost-to-go function $V_t^k$ defined as the maximum of affine cuts, i.e., we have $(\alpha_t^l)_{l \leq k}$ and $(\beta_t^l)_{l \leq k}$ such that

$$V_t^k(x_t) = \max_{l \leq k} \alpha_t^l x_t + \beta_t^l$$

(14)

Under Assumption (LS) Bellman operator defined in (3a) applied to $V_t^k$ reads

$$\begin{align*}
\hat{B}_{t-1}(V_t^k)(x_{t-1}, \xi_t) &= \min_{x_t, z} \left[ c_t^\top x_t + z \right] \\
&= \min_{x_t, z^+, z^-, r} \left[ \alpha_t^l x_t + z^+ - z^- \right] \\
s.t. &A_t x_t + B_t x_{t-1} = b_t, \quad \alpha_t^k x_t + \beta_t^k \leq z, \quad \forall k \leq k
\end{align*}$$

(15a)

$$\begin{align*}
\text{s.t.} &A_t x_t + B_t x_{t-1} = b_t, \quad \alpha_t^k x_t + \beta_t^k + r = z^+ - z^-, \quad \forall k \leq k
\end{align*}$$

(15b)

$$\begin{align*}
x_t \geq 0, \quad x_t, z^+, z^-, r \geq 0
\end{align*}$$

(15c)

To simplify notation, we denote

$$Q(x, W, q, T, h) := \min_y \{q^\top y \mid T x + W y = h, y \geq 0\}.$$ 

(16)

Then, for any $t \in [T]$ and $k \in \mathbb{N}$, setting

$$\begin{align*}
x := x_t, \quad y := (x_t, z^+, z^-, r), \quad (A_t, B_t, c_t, b_t) := \begin{pmatrix} A_t & 0 & 0 & 0 \\ \alpha_t^1 & -1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_t^k & -1 & 1 & 1 \end{pmatrix}, \quad (c_t, b_t) := \begin{pmatrix} c_t \1 \alpha_t^l \\ -1 \\ 0 \end{pmatrix}, \quad (T, h) := \begin{pmatrix} B_t \\ 0 \\ 0 \\ -\beta_t^l \end{pmatrix}
\end{align*}$$

(17)

, we obtain

$$\mathbb{E}[Q(x, W, q, T, h)] = \mathbb{E}[\hat{B}_{t-1}(V_t^k)(x_{t-1}, \xi_t)].$$

(18)

Under Assumption (LS) we have that i) $W$ is finitely supported, and ii) $q$ and $(T, h)$ are independent.

$^9$Independence can be replaced by finite support assumption on one of the random variables. More generally, we can consider a finitely supported random variable $M_t$ such that $c_t$ and $(B_t, b_t)$ are independent conditionally to $M_t$. 

13
3.2 Partition and cuts

Let \( n, m, p \in \mathbb{N}^* \) be integers and \( W, q, T, h \) be integrable random variables taking values respectively in \( \mathbb{R}^{p \times m}, \mathbb{R}^m, \mathbb{R}^{p \times n}, \mathbb{R}^p \). We define the uncertainty set \( \Xi \) as the support of \((W, q, T, h)\). We make the following assumption:

**Assumption (FSW)** (Finitely supported \( W \)). The random variables \((W, q, T, h)\) satisfies:

1. \( W \) is finitely supported,
2. \( q \) and \((T, h)\) are independent.

Note that if the random variables are defined thanks to [17] then Assumption [LS] implies Assumption [FSW].

To alleviate notation, for a measurable set \( P \subset \Xi \), we shorten \( \mathbb{P}[(W, q, T, h) \in P] \) as \( \mathbb{P}[P] \), and similarly for conditional expectation.

**Definition 9.** We define the value function \( V : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\} \)

\[
V(x) := \mathbb{E}[Q(x, W, q, T, h)].
\]  

Let \( \mathcal{P} \) be a partition of \( \Xi \). We define the partitioned value function \( V_\mathcal{P} : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\} \) as

\[
V_\mathcal{P}(x) := \sum_{P \in \mathcal{P}} \mathbb{P}[P]Q\left(x, \mathbb{E}[W|P], \mathbb{E}[q|P], \mathbb{E}[T|P], \mathbb{E}[h|P]\right)
\]

where \( Q \) is defined in Eq. [16], and we made a slight abuse of notation with,

\[
Q\left(x, \mathbb{E}[W|P], \mathbb{E}[q|P], \mathbb{E}[T|P], \mathbb{E}[h|P]\right) = Q\left(x, \mathbb{E}[W|P], \mathbb{E}[q|P], \mathbb{E}[T|P], \mathbb{E}[h|P]\right).
\]

We say that

- \( \mathcal{P} \) is valid if \( V_\mathcal{P}(x) \leq V(x) \) \( \forall x \in \mathbb{R}^n \)
- \( \mathcal{P} \) is tight at \( \hat{x} \) if \( V_\mathcal{P}(\hat{x}) = V(\hat{x}) \)
- \( \mathcal{P} \) is adapted to \( \hat{x} \) if \( \mathcal{P} \) is valid and tight at \( \hat{x} \)

For any partition \( \mathcal{P} \), if we know how to compute probabilities and conditional expectations on each element of the partition, we know by Section 2.3 how to compute a tight and valid cut for \( V_\mathcal{P} \). If \( \mathcal{P} \) is a valid and adapted partition, then the cut is still valid for the true value function \( V \).

**Proposition 10.** Let \( \mathcal{P} \) be an adapted partition to \( \hat{x} \). For all \( P \in \mathcal{P} \), let \( \hat{f}_P \) be a valid and tight cut of \( Q(, \mathbb{E}[W, q, T, h|P]) \) at \( \hat{x} \). Then \( f := \sum_{P \in \mathcal{P}} \mathbb{P}[P] \hat{f}_P \) is a valid and tight cut of \( V \) at \( \hat{x} \).

**Proof.** We have

\[
f(x) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] \hat{f}_P(x) \quad (23a)
\]

\[
\leq \sum_{P \in \mathcal{P}} \mathbb{P}[P] Q(x, \mathbb{E}[W, q, T, h|P]) \quad \text{since } \hat{f}_P \text{ is valid} \quad (23b)
\]

\[
= V_\mathcal{P}(x) \leq V(x) \quad \text{since } \mathcal{P} \text{ is valid.} \quad (23c)
\]

Thus, \( f \) is valid. Moreover,

\[
f(\hat{x}) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] \hat{f}_P(\hat{x}) \quad (24a)
\]

\[
= \sum_{P \in \mathcal{P}} \mathbb{P}[P] Q(\hat{x}, \mathbb{E}[W, q, T, h|P]) \quad \text{since } \hat{f}_P \text{ is tight} \quad (24b)
\]

\[
= V_\mathcal{P}(\hat{x}) = V(\hat{x}) \quad \text{since } \mathcal{P} \text{ is tight.} \quad (24c)
\]

Thus, \( f \) is tight. \( \square \)

Thus, an adapted partition provides a valid and tight cut. We now detail how to obtain such a partition.
3.3 Explicit valid and adapted partition

In this section, we provide an explicit adapted partition for linear stochastic problems in standard form.

By Assumption [1], the admissible primal set \( \{ y \in \mathbb{R}^m \mid Ty + Wy = h, y \geq 0 \} \) is non-empty and compact. Then, by strong duality, we can rewrite \( Q \) defined in Eq. (19) as

\[
Q(x, W, q, T, h) = \max_{\lambda \in \mathbb{R}^p} \left\{ (h - Tx)^\top \lambda \mid W^\top \lambda \leq q \right\}
\]

(25)

We now define the dual admissible set \( D_{W, q} \) as

\[
D_{W, q} := \{ \lambda \in \mathbb{R}^p \mid W^\top \lambda \leq q \}
\]

(26)

We recall that the normal fan \( \mathcal{N}(E) \) of a set \( E \) is the finite collection of all normal cones \( N_E(x) \):

\[
\mathcal{N}(E) := \{ N_E(x) \mid x \in E \}
\]

(27a)

where \( N_E(x) := \{ \phi \mid \forall x' \in E, \phi^\top (x' - x) \leq 0 \} \)

(27b)

When \( W \) and \( q \) are fixed, the value of \( Q(\tilde{x}, W, q, T, h) \) depends on which normal cone \( h - T\tilde{x} \) belongs to. Thus, we finally define

\[
E_{N, \tilde{x}} := \{ (T, h) \mid h - T\tilde{x} \in \text{ri}(N) \}
\]

(28a)

\[
\mathcal{R}_{\tilde{x}, W, q} := \{ E_{N, \tilde{x}} \mid N \in \mathcal{N}(D_{W, q}) \}
\]

(28b)

We begin with the finitely supported \( q \) case as a warm-up.

**Remark 11** (Finitely supported \( q \)). We can show that, when \( W \) and \( q \) are fixed, \( \mathcal{R}_{\tilde{x}, W, q} \) is an adapted partition to \( \tilde{x} \) (see [FL21, Thm 3]). If \( \text{supp}(W) \) and \( \text{supp}(q) \) are finite, we can extend this result to show that \( \{ \{ W \} \times \{ q \} \times \mathcal{R}_{\tilde{x}, W, q} \mid W \in \text{supp}(W), q \in \text{supp}(q) \} \) is an adapted partition to \( \tilde{x} \):

\[
\mathbb{E} \left[ Q(\tilde{x}, W, q, T, h) \right] = \sum_{(W, q) \in \text{supp}(W, q)} \mathbb{E} \left[ Q(\tilde{x}, W, q, T, h) \mid W = W, q = q \right] \mathbb{P} \left[ W = W, q = q \right]
\]

(29a)

\[
= \sum_{(W, q) \in \text{supp}(W, q)} \sum_{R \in \mathcal{R}_{\tilde{x}, W, q}} Q \left( \tilde{x}, \mathbb{E} \left[ W, q, T, h \mid (T, h) \in R, W = W, q = q \right] \right) \mathbb{P} \left[ (T, h) \in R, W = W, q = q \right]
\]

(29b)

We now extend this result to the case where \( q \) has a non-finitely supported distribution. This extension relies on a partition \( S_W \) of \( \mathbb{R}^{p \times m} \) such that \( q \mapsto \mathcal{R}_{\tilde{x}, W, q} \) is constant on each \( S \in S_W \). Actually, this partition \( S_W \) is the collection of relative interiors of secondary cones (see [DLRS10, Definition 5.2.1]) which are classical objects from polyhedral geometry. We also give a more elementary, but equivalent, construction of \( S_W \) in Appendix [B1].

To any constraint matrix \( W \), we associate its secondary fan \( \Sigma \text{-fan}(W) \) (which is the normal fan of the secondary polytope defined, e.g., in [DLRS10, Definition 5.1.7]), a well-studied finite collection of closed cones associated with \( W \). Let \( S_W \) be the collection of relative interiors of the elements of \( \Sigma \text{-fan}(W) \):

\[
S_W := \{ \text{ri}(S) \mid S \in \Sigma \text{-fan}(W) \}.
\]

(30)

In particular, the elements of \( S_W \) are relatively open (convex) cones of \( \mathbb{R}^m \). Further, note that [DLRS10] provides constructive representation of \( \Sigma \text{-fan}(W) \) and thus of \( S_W \), which paves the way toward explicit computation of \( S_W \).

**Lemma 12.** Let \( W \in \mathbb{R}^{p \times m} \) and \( S \in S_W \). For every \( q, q' \in S \) we have \( \mathcal{R}_{\tilde{x}, W, q} = \mathcal{R}_{\tilde{x}, W, q'} \). Consequently, instead of considering an infinite number of \( \mathcal{R}_{\tilde{x}, W, q} \) parametrized by \( q \), we can consider a finite number of \( \mathcal{R}_{\tilde{x}, W, S} \) parametrized by \( S \in S_W \) where

\[
\mathcal{R}_{\tilde{x}, W, S} := \mathcal{R}_{\tilde{x}, W, q} \text{ for an arbitrary } q \in S.
\]

(31)

**Proof.** The proof is given in Appendix [B1].

□
We now leverage this reduction to a finite number of $R_{x,W,S}$ to define an adapted partition. We start by showing in Theorem 13 that, for given $x$ and $W$, the cost-to-go function $Q(x,W,q,T,h)$ is piecewise linear in $q$ and $(T,h)$. More precisely, it is an adaptation of the basis decomposition theorem of Walkup and Wets [WW69] (see also Sturmfels and Thomas [ST97] or [DLRS10, Theorem 1.2.2] for a more modern presentation).

**Theorem 13.** Let $W \in \mathbb{R}^{l \times m}$ and $\hat{x} \in \mathbb{R}^n$. Then, for every $S \in S_W$ and $R \in R_{x,W,S}$, there exists a basis $B \subset [m]$ such that

$$\forall q \in S, \quad \forall(T,h) \in R, \quad Q(\hat{x},W,q,T,h) = q_B^T W_B^{-1}(h - T\hat{x})$$

(32)

**Proof.** The proof is given in Appendix B.2.

We deduce the following lemma which gives regions where we can interchange the function $Q$ with the expectation. This lemma can be seen as an exact quantization result.

**Lemma 14.** Let $W \in \mathbb{R}^{l \times m}$. Assume that $(T,h)$ and $q$ are independent random variables, then, for all $S \in S_W$ and $R \in R_{x,W,S}$,

$$Q(\hat{x},W,\mathbb{E}[q,T,h] \mid q \in S, (T,h) \in R) = \mathbb{E}[Q(\hat{x},W,q,T,h) \mid q \in S, (T,h) \in R]$$

(33)

**Proof.** Let $B$ be the basis in Theorem 13 by independence of $(T,h)$ and $q$, we have

$$\mathbb{E}[Q(\hat{x},W,q,T,h) \mid q \in S, (T,h) \in R] = \mathbb{E}[q_B^T W_B^{-1}(h - T\hat{x}) \mid q \in S, (T,h) \in R]$$

(34a)

$$= \mathbb{E}[q_B^T \mid q \in S] W_B^{-1} \mathbb{E}[(h - T\hat{x}) \mid (T,h) \in R]$$

(34b)

$$= Q(\hat{x},W,\mathbb{E}[q \mid q \in S], \mathbb{E}[T,h] \mid (T,h) \in R)$$

(34c)

Indeed, by convexity of $S$ (resp. $R$), we have $\mathbb{E}[q \mid q \in S] \in S$ (resp. $\mathbb{E}[T,h] \mid (T,h) \in R] \in R$).

By summing over all $W \in \text{supp}(W)$, all $S \in S_W$ and $R \in R_{x,W,S}$ and applying this lemma for every term of the sum, we can now deduce an explicit adapted partition.

**Theorem 15** (Adapted partition for general second stage cost $q$). Assume that $q$ and $\xi$ are independent conditionally to $W$ and that $\text{supp}(W)$ is finite. We define $P_\hat{x}$ the following partition

$$P_\hat{x} = \{\{W\} \times S \times R \mid W \in \text{supp}(W), S \in S_W, R \in R_{x,W,S}\}$$

(35)

then $P_\hat{x}$ is an adapted partition to $\hat{x}$.

**Proof.** The proof, based on Lemma 14, can be found in Appendix B.2.

**Remark 16.** As we saw in Section 3.2, this explicit adapted partition provides a new method to find tight and valid cut in Line 10 of Algorithm 1 without having an approximation error, i.e., $\gamma_1 = 0$, in the linear case (under Assumption (LS) and Assumption (LS)). Moreover, this explicit adapted partition allows to extend the scope of APM methods. In [FL21], a method to find an explicit adapted partition for deterministic $W$ and $q$ is presented and the authors state, in Remark 12, that the partition $P_\hat{x}$ is adapted without proving it formally. This section provides a formal proof of this statement.

Finally, Siddig and Song presented an algorithm combining ideas of APM methods and SDDP in the finitely supported case. This explicit adapted partition paves the way to an APM-SDDP algorithm with non-finitely supported random variables.

## 4 Complexity results

In this section, we give convergence and complexity results for various instances of Algorithm 1. In Section 4.1 we first define the notion of effective iteration and deduce an upper bound on the number of effective iterations required by Algorithm 1 to get an $\varepsilon$-solution. We then distinguish between deterministic and randomized selection processes for the choice in Line 9 of the algorithm. For deterministic selection processes, namely the problem-child and explorative node selections, we show in Section 4.2 that all iterations are effective. Finally, when the node selection is randomized, we show in Section 4.3 the existence of a positive probability for an iteration to be effective. We then deduce a complexity bound on the expected number of iterations.
4.1 Bounding the number of effective iterations

We first recall that the value of Problem \(\text{MSP}\) can be written in a more concise form, by using the nested problem in Eq. (2) and the definition of expected cost-to-go function in Eq. (4), and keeping in mind that \(\xi_1\) is deterministic:

\[
\text{val}(\text{MSP}) = \min_{x_1 \in X_1(x_0, \xi_1)} \ell_1(x_1, \xi_1) + V_1(x_1)
\] (36)

Our aim is to show that, for some iteration \(k\), the solution \(x_k\) is a \(\varepsilon\)-solution of Eq. (36), and the lower-bound \(V_0(x_0)\) is \(\varepsilon\)-tight. Unfortunately, Assumptions 1 to 5 are not enough to ensure convergence of Algorithm 1, we need a further assumption on the node selection process.

Regardless of node selection, we define the notion of effective iteration. Recall that \(\gamma_{t}^{F}, \gamma_{t}, \gamma_{t}^{F}, \gamma_{t}^{D}\) are errors in forward Bellman operator and approximation update (see Algorithm 1) at time \(t \in [T]\), and \(L_{t}\) (resp. \(\widetilde{L}_{t}\)) are Lipschitz bounds on the cuts (resp. upper-approximation) at time \(t\). In the remains of the section we consider a sequence \((\nabla_{t}, L_{t}, x_{t}^{k})_{t \in [T], k \in \mathbb{N}}\) produced by Algorithm 1.

**Definition 17** (effective iteration). For every \(t \in [T-1]\), let \(\delta_t > 0\) and \(\eta_t \geq 0\). By backward induction, we define

\[
\begin{align*}
\varepsilon_{T-1} &:= \frac{\gamma_{T-1}^{F}}{2} + \frac{\gamma_{T-1}}{2} \\
\varepsilon_t &:= \varepsilon_{t+1} + (\frac{L_{t+1} + L_{t}}{2})(\delta_{t+1} + \eta_{t+1}) + \gamma_{t+1}^{F} + \gamma_{t}^{F} + \gamma_{t} + \eta_{t} + \gamma_{t}^{D} \quad \forall t \in [T-2] \\
\varepsilon_0 &:= \varepsilon_{1} + (\frac{L_{1} + L_{t}}{2})(\delta_{1} + \eta_{1}) + \gamma_{1}^{F} \\
\end{align*}
\] (37)

For \(t \in [T-1]\) and \(k \in \mathbb{N}\), we say that

- \(x_k^t\) is \(\varepsilon_t\)-saturated, if \(\nabla_{t}(x_k^t) - \nabla_{t}(x_k^t) \leq \varepsilon_t\) and
- \(x_k^t\) is \(\delta_t\)-distinguishable if \(\|x_k^t - x_k^t\| > \delta_t\) for all \(\kappa < k\) such that \(x_k^t\) is \(\varepsilon_t\)-saturated.

We say that an iteration \(k \in \mathbb{N}\) is effective if it generates either a \(\varepsilon_0\)-saturated point, which is also a \(\varepsilon_0\)-solution to Problem \(\text{MSP}\), or a new \(\varepsilon_t\)-saturated and \(\delta_t\)-distinguishable point for at least one \(t \in [T]\), i.e.,

\[
\begin{align*}
x^t_1 & \text{ is } \varepsilon_t\text{-saturated and } \ell_1(x^t_1, \xi_t) + V_1(x^t_1) - \text{val}(\text{MSP}) \leq \varepsilon_0 \quad (38a) \\
\text{OR} \quad & \exists t \in [T-1], \ x^t_k \text{ is } \varepsilon_t\text{-saturated and } \delta_t\text{-distinguishable.} \quad (38b)
\end{align*}
\]

We now give an upper bound on the number of effective iterations of Algorithm 1 to find an \(\varepsilon_0\)-optimal solution.

**Theorem 18** (bound on effective iterations number). Let Assumptions 4 to 5 be satisfied and \(t \in [T-1]\), assume that \(\delta_t \in [0, D_t]\) and \(\eta_t \in \mathbb{R}_+\) are given and \(\varepsilon_t\) defined by (37). Let

\[
\mathcal{K} := \sum_{t=1}^{T-1} \left(\frac{D_t}{\delta_t} + 1\right)^{n_t}
\] (39)

After at most \(\mathcal{K} + 1\) effective iterations we have a \(\varepsilon_1\)-lower bound of Problem \(\text{MSP}\):

\[
\nabla_{0}(x_0) = \ell_1(x^1_1, \xi_1) + V_1(x^1_1) \geq \text{val}(\text{MSP}) - \varepsilon_1
\] (40)

Further, there exists, among those \(\mathcal{K} + 1\) effective iterations, at least one such that \(x^t_1\) is an \(\varepsilon_0\)-solution to Problem \(\text{MSP}\):

\[
\ell_1(x^t_1, \xi_t) + V_1(x^t_1) \leq \text{val}(\text{MSP}) + \varepsilon_0
\] (41)

**Proof.** For \(t \in [T-1]\), there are at most \((\frac{D_t}{\delta_t} + 1)^{n_t}\) disjoint balls\footnote{We consider here balls for the euclidean norm \(\| \cdot \|_2\), but the result is still valid with the \(p\)-norm \(\| \cdot \|_p\) for every \(p \in [1, +\infty]\).} of diameter \(\delta_t\) in a ball of diameter \(D_t + \delta_t\) (see [ZS19 A.3.2]). In particular, we cannot compute more than \((\frac{D_t}{\delta_t} + 1)^{n_t}\), \(\delta_t\)-distinguishable points at step \(t\). Thus, after \(\mathcal{K} = \sum_{t=1}^{T-1} (\frac{D_t}{\delta_t} + 1)^{n_t}\) effective iterations, for all \(t \in [T]\), it is impossible to compute a new \(\delta_t\)-distinguishable
point. Then, as the iteration $k$ is effective and we cannot have (38b), we have (38a) and in particular $x^k_1$ is an $\varepsilon_0$-solution. Moreover, $x^1_1$ is $\varepsilon_1$-saturated. Then,

$$
\ell_1(x_1^k, \xi_1) + V_1^k(x_1^k) \geq \ell_1(x_1^k, \xi_1) + V_1^k(x_1^k) - \varepsilon_1 \quad (42a)
$$

$$
\geq \ell_1(x_1^k, \xi_1) + V_1(x_1) - \varepsilon_1 \quad (42b)
$$

$$
\geq \min_{x_1 \in X(V^k(x_1), \xi_1)} \ell_1(x_1, \xi_1) + V_1(x_1) - \varepsilon_1 \quad (42c)
$$

$$
= \text{val} (\text{MSP}) - \varepsilon_1 \quad (42d)
$$

\[ \Box \]

**Remark 19.** Finally, although the theorems of this section state that we find an $\varepsilon_0$-optimal solution at stage 1, we have no guarantee that the approximations $V_i^k$ converges to $V_i$. We cannot hope that these approximations converge to the true expected cost-to-go functions far from the optimal and reachable trajectories. Nevertheless, by considering the sets of points that are $\delta_i$-close to every optimal and reachable trajectories, we could hope to have a convergence of strategies generated by $F(V_i^k)$ on those sets. If we add a finite diameter of the support of $\xi$, and Lipschitz assumptions for $\xi$, we are confident that the proof can be adapted. However, the general case looks harder and might require different ideas for proving complexity results for the convergence of strategies at every stages.

For a class of specific (deterministic) implementations of Algorithm [I] each iteration is effective, in which case we can directly bound the number of iterations required to obtain an $\varepsilon_0$-optimal solution.

### 4.2 Deterministic node selection

In this section, we present sufficient condition for an iteration to be effective. Consequently, for two algorithms with deterministic node selections (namely problem-child node selection [BDZ17] and explorative node selection [Lan20]), we show that each iteration is effective, yielding a complexity result.

We first define the distance to the set of $\varepsilon_i$-saturated points.

**Definition 20.** Let $t \in [T - 1]$ and $k \geq 1$.

We denote $y^k_t$ the random variable

$$
y^k_t := F_{t-1}(V^{k-1}((x^k_{t-1}, \xi_t)) \quad (43)
$$

We denote by $d^k_t$ the distance function to the set of $\varepsilon_1$-saturated points until iteration $k$ :

$$
d^k_t(x) := \min_{\kappa < k | x^{\kappa}_t \text{ is } \varepsilon_1 \text{-saturated}} \|x - x^{\kappa}_t\| \quad (44)
$$

In particular, $x^k_t$ is $\delta_i$-distinguishable if and only if $d^k_t(x^k_t) > \delta_i$.

The following technical lemma, whose proof can be found in Appendix [C], shows that if the new state $x^k_t$ (resulting from the choice of $\xi^k_t$) is either i) far enough from the set of saturated points, or ii) yielding a large enough gap, then iteration $k$ is effective.

**Lemma 21.** Let Assumptions [I] to [B] be satisfied and assume that, for all $t \in [T - 1]$, $\delta_i \in [0, D_i]$, $\eta_t \in \mathbb{R}_+$ are given and $\varepsilon_i$ defined by (37). Let $k \in \mathbb{N}^*$. If, for all $t \in [T - 1]$, at least one of the following inequalities is satisfied

$$
E[V^{k-1}_{t}(y^k_t) - V^{k-1}_{t}(y^k_t)] \leq V^{k-1}_{t}(x^k_t) - V^{k-1}_{t}(x^k_t) + (\mathcal{I}_t + L)\eta_t \quad (45a)
$$

$$
E[d^k_t(y^k_t)] \leq d^k_t(x^k_t) + \eta_t \quad (45b)
$$

then, iteration $k$ is effective.

In [Lan20], Lan suggested to choose $x^k_t$ as the most distinguishable point in a new algorithm called Explorative Dual Dynamic Programming (EDDP). We then speak of *explorative* node selection. The following lemma shows that both these selections lead to effective iterations.
Lemma 22. Let Assumptions 1 to 5 hold and assume that, for all \( t \in [T] \) \( \delta_t \in [0, D_t] \), \( \eta_t = 0 \), are given and \( \varepsilon_t \) is defined by (37).

We say that we have a problem-child node selection if for all \( k \in \mathbb{N}^* \), and \( t \in [T-1] \), \( \xi^k_t \) is chosen such that it maximizes the current gap, i.e.,

\[
\xi^k_t \in \arg\max_{\xi \in \text{supp}(\xi_t)} V^{k-1}_t \left( F_{t-1}(V^{k-1}_t)(x^{k-1}_{t-1}, \xi) \right) - V^{k-1}_t \left( F_{t-1}(V^{k-1}_t)(x^{k-1}_{t-1}, \xi) \right)
\]

We say that we have an explorative node selection if for all \( k \in \mathbb{N}^* \), and \( t \in [T-1] \), \( \xi^k_t \) is chosen such that \( x^k_t \) maximizes the distance to previous \( \varepsilon_t \)-saturated points, i.e.,

\[
\xi^k_t \in \arg\max_{\xi \in \text{supp}(\xi_t)} d^k_t \left( F_{t-1}(V^{k-1}_t)(x^{k-1}_{t-1}, \xi) \right)
\]

Then, with a problem-child or an explorative node selection method, each iteration of Algorithm 1 is effective.

Proof. It is a consequence of Lemma 21. Indeed, as \( \eta_t = 0 \), \( x^k_t \) = \( F_{t-1}(V^{k-1}_t)(x^{k-1}_{t-1}, \xi^k_t) \) and \( y^k_t = F_{t-1}(V^{k-1}_t)(x^{k-1}_{t-1}, \xi^k_t) \), and since the maximum is greater than the expected value, Eq. (45a) implies Eq. (46a) and Eq. (45b) implies Eq. (46b).

Lemma 22 implies that every iteration of these deterministic node selection method is effective. Coupled with Theorem 18 we easily obtain complexity bounds, for example as follows.

Corollary 23. Let Assumptions 1 to 5 hold and assume that every iteration of Algorithm 1 is effective (e.g., problem-child or explorative node selection). Further, for simplicity, let the total error be \( \gamma := \sum_{t=1}^{T-1} \gamma_t + \gamma^F_t \), and choose \( n, D, L \) such that, for all \( t \in [T-1] \), \( n_t \leq n \), \( D_t = D \), \( T_t = L_t = L \). Then, for every \( \varepsilon > \gamma \), sufficiently small (e.g., such that \( \varepsilon \leq 2DL + \gamma \)), Algorithm 1 finds an \( \varepsilon \)-first stage solution \( x^k_t \) within at most \( K_\varepsilon \) iterations where

\[
K_\varepsilon := \left( \frac{2DL}{\varepsilon - \gamma} \right)^n (T-1)^{n+1}
\]

Proof. We set \( \delta_t := \frac{\varepsilon - \gamma}{2L(T-1)} \) and \( \eta_t := 0 \) for all \( t \in [T-1] \). Then, as \( \varepsilon \leq 2DL + \gamma \) we have \( \delta_t \leq D = D_t \). Moreover, \( \varepsilon_0 \), defined in Eq. (37), satisfies

\[
\varepsilon_0 = \sum_{t=1}^{T-1} \left[ (L_t + L_t)(\delta_t + \eta_t) + \gamma_t + \gamma^F_t \right] = (T-1)2Ld + \gamma = \varepsilon.
\]

With this setting, we have that \( K \), as defined in (39), satisfies

\[
K \leq (T-1) \left( \frac{D}{\delta} + 1 \right)^n = T \left( \frac{2DL(T-1)}{\varepsilon - \gamma} + 1 \right)^n
\]

Now, as \( \varepsilon \) is assumed to be small enough to have \( 2DL/(\varepsilon - \gamma) \leq 1 \) (i.e. \( \varepsilon \leq 2DL + \gamma \)) we get

\[
K \leq (T-1) \left( \frac{2DL(T-1)}{\varepsilon - \gamma} \right)^n = K_\varepsilon.
\]

By assumption all iterations are effective and Theorem 18 ends the proof.

Remark 24. Note that the maximum in (46a) (resp. (46b)) is easily obtained under finite noise Assumption (FSN). Indeed, we can compute the gap (resp. the distance) for every \( \xi_t \) in the support of \( \xi_t \) and keep \( \xi^k_t \) maximizing the gap.

However, without finite noise Assumption (FSN), we just need to find a \( \xi^k_t \) leading to a gap worse than the expected gap (see Lemma 21), and not necessarily a maximizer. This paves the way for a deterministic node selection, with non finitely supported random variables.
4.3 Randomized algorithms

When the choice of $\xi_t^k$ is made randomly, there is no guarantee that the iteration will be effective. However, through a technical, yet necessary to deal with dependence issue, nested Hoeffding lemma’s shown in Appendix D.1, we show that there is a positive probability $p$ for an iteration to be effective. Then, by comparing the time to obtain an effective iteration to a geometric random variable of probability of success $p$ in Appendix D.2, we deduce a bound on the expected number of iteration required to get an $\varepsilon$-optimal solution.

Remark 25 (Notational difficulty of randomized algorithm on stochastic problem). We are now considering a stochastic algorithm for solving the MSP Problem. Thus, there are two sources of randomness: the intrinsic $(\xi_t)_{t \in [T]}$ and the node selection $\xi_t^k = \xi_t^k$. To distinguish both, we denote in bold random variables that are $(\xi_t)_{t \in [T]}$ measurable, with a tilde random variables that are $(\xi_t^k)_{t \in [T], k \in \mathbb{N}^*}$ measurable (and with both if they are neither).

For example the trajectory determined during the forward phase $(\tilde{x}_t^k)_{t \in [T]}$ only depends on the past node selections, whereas the tentative points $\tilde{y}_k^T$ depends both on the past node selections and the actual realization of $\xi_t$.

Under Assumption (FSN), this discussion is usually avoided by representing the dependence on $(\xi_t)_{t \in [T]}$ with a (finite) scenario tree, and indexing the variables by the tree nodes.

Let $(A^k)_{k \in \mathbb{N}}$ be the filtration such that $A^k := \sigma(\xi_t^k)_{t \in [T-1], k \in [k]}$ and $\mathcal{A}^\infty = \bigcup_{k \in \mathbb{N}} A^k$. In particular, a random variable measurable with $A^k$ knows all node selection up to iteration $k$, that include, for example, $\tilde{V}^k$ for all $t \in [T]$.

Lemma 26. Let Assumptions 1 to 3 be satisfied and assume that, for all $t \in [T-1]$, $\delta_t \in (0, D]$, $\eta_t \in \mathbb{R}_+$ are given and $\varepsilon_t$ defined by (37). Further, assume that in Algorithm 1 Line 7 we draw $\xi_t^k$ randomly according to the distribution of $\xi_t$, and independently of all other $\xi_t^k$ as well as $(\xi_t)_{t \in [T]}$.

Then, for all iteration $k \in \mathbb{N}$ of Algorithm 1 and all event $A^{k-1} \in A^{k-1}$ such that $P[A^{k-1}] = 0$, we have

$$\Pr[\text{Iteration k is effective} \mid A^{k-1}] \geq \prod_{t=1}^{T} \left(1 - e^{-\frac{2\varepsilon_t^2}{D}}\right)$$

Proof. Let $\mathcal{A}^\infty := \sigma(\xi_t^k)_{t \in [T-1], k \in \mathbb{N}^*}$ and $k \in \mathbb{N}^*$. By Lemma 21, we have

$$\Pr[\text{Iteration k is effective} \mid A^{k-1}] \geq \Pr[\forall t \in [T-1], E[d_t^k(\tilde{y}_k^T) \mid \mathcal{A}^\infty] < d_t^k(x_t^k) + \eta_t \mid A^{k-1}]$$

For $t \in [T-1]$, let $A^k_t := \sigma(A^{k-1}_t, (\xi_t^k)_{t \in [t]})$. We have that $\sigma(d_t^k(\tilde{y}_k^T)) \subset \sigma(A^{k-1}_t, \xi_t)$ from which we deduce that $E[d_t^k(\tilde{y}_k^T) \mid \mathcal{A}^\infty] = E[d_t^k(\tilde{y}_k^T) \mid A^{k-1}_t]$. We define the events $E^k_t := \{\omega \in \Omega \mid E[d_t^k(\tilde{y}_k^T) \mid A^{k-1}_t] < d_t^k(x_t^k) + \eta_t\}$. Thus, $\Pr[\text{Iteration k is effective} \mid A^{k-1}] \geq \Pr[\bigcap_{t=1}^{T-1} E^k_t \mid A^{k-1}]$. By applying Lemma 41 with the random variables $(\xi_k^k)_{k \in \mathbb{N}, t \in [T-1]}$, the filtration $(A^k_t)_{k \in \mathbb{N}, t \in [T-1]}$ and the measurable function $d_t^k(x_t^k) \rightarrow d_t^k(x_t^k)$ taking its value in $[0, D_t]$, we have $\Pr[\bigcap_{t=1}^{T-1} E^k_t \mid A^{k-1}] \geq \prod_{t=1}^{T} \left(1 - e^{-\frac{2\varepsilon_t^2}{D}}\right)$ which gives Eq. (49). $\square$

We now give a complexity results for all TFDP algorithms (following framework of Algorithm 1) where the choice of $\xi_t^k$ is made randomly.

Theorem 27. Let Assumptions 1 to 3 be satisfied and assume that in Line 7 we draw $\xi_t^k$ randomly according to the distribution of $\xi_t$, and independently from the previous $\xi_t^k$.

Further, for simplicity, let the total error be $\gamma_t : = \gamma_t^\Sigma + \tau_t + \gamma_t^F$ and choose $n, D, L$ such that, for all $t \in [T-1]$, $n_t \leq n$, $D_t = D$, $L_t = L$.

Then, for $\varepsilon > \gamma_t$, sufficiently small (e.g., such that $\varepsilon \leq 4DL + \gamma_t$), the expected number of iterations of Algorithm 1 required to find an $\varepsilon$-solution $x_t^k$ to problem (36), i.e., such that $\ell_1(x_t^k, \xi_t) + V_2(x_t^k) \leq \text{val}(\text{MSP}) + \varepsilon$ is bounded by $\left(1 - \frac{4DL(T-1)}{\varepsilon - \gamma_t^\Sigma}\right)^n + 2^{(T-1)^2}$. Then, as $\varepsilon \leq 4DL + \gamma_t$ we have $\eta_t = \delta_t \leq D = D_t$. Moreover, $\varepsilon_0$, defined in Eq. (37), satisfies
\[ \varepsilon_0 = \sum_{t=1}^{T-1} \left[ (L_t + L_0)(\delta_t + \eta_t) + \gamma_t + \tau_t + \gamma_t^F \right] = (T-1)2L \times 2 \frac{\varepsilon - \gamma_\Sigma}{4L(T-1)} + \gamma_\Sigma = \varepsilon. \]

Let \( \tilde{K} \) the (random) number of iterations needed to compute \( K_\varepsilon := \sum_{t=1}^{T-1} 1 + \frac{D_k}{\delta_t} \) effective iterations, then by Theorem 18 Algorithm 1 finds an \( \varepsilon \)-solution after at most \( \tilde{K} \) iterations. Let \( p := \prod_{t=1}^{T} \left( 1 - \exp \left( -\frac{2n^2}{D_t^2} \right) \right) \), by Lemma 26 for \( A^{k-1} \in A^{k-1} \), we have \( \mathbb{P} \left[ \text{Iteration } k \text{ is effective} \mid A^{k-1} \right] \geq p \). Thus, by Lemma 42 we have \( E \left[ \tilde{K} \right] \leq K_p \).

Moreover, since as \( x \mapsto \frac{1}{\varepsilon - \gamma_\Sigma} \) is an increasing function on \( (0, \frac{1}{2}] \), then for all \( x \in (0, 1] \), we have \( \frac{1}{1-e^{-\varepsilon}} \leq \frac{1}{1-e^{-\gamma_\Sigma}} \times \frac{1}{2} \leq 1.6 \times \frac{1}{2} \). Thus, as \( \frac{2n^2}{D_t^2} \leq 1 \), we have that \( \frac{1}{p} \leq \prod_{t=1}^{T-1} 1.6 \times \frac{D_t^2}{2n^2} \leq \left( \frac{4DL(T-1)}{\varepsilon - \gamma_\Sigma} \right)^2(T-1) \). We then obtain \( E \left[ \tilde{K} \right] \leq K_p \leq \left( \frac{4DL}{\varepsilon - \gamma_\Sigma} \right)^n (T-1)^{n+1} \times \left( \frac{4DL(T-1)}{\varepsilon - \gamma_\Sigma} \right)^2(T-1) = (T-1) \left( \frac{4DL(T-1)}{\varepsilon - \gamma_\Sigma} \right)^n+2(T-1) \).

**Remark 28** (Stochastic dominance and comparison with finitely supported noise). The proof of Theorem 27 actually give more information on the (random) number of iteration \( \tilde{K} \) after which we obtain an \( \varepsilon \)-solution: \( \tilde{K} \) is stochastically dominated by a random variable with a negative binomial distribution representing the number of trials to obtain \( K_\varepsilon \) successes with probability of success \( p \) (see Lemma 12).

Further, under finitely supported noise assumption Assumption (FSN) the probability of choosing the problem child \( \xi_t^k \) (cf Lemma 22) for each \( t \in [T-1] \) is lower bounded by \( \prod_{t=1}^{T-1} \frac{1}{|\text{supp}(\xi_t)|} \). Then, Lemma 26 still holds after replacing the right hand side probability of success by \( \prod_{t=1}^{T-1} \frac{1}{|\text{supp}(\xi_t)|} \). We can then deduce more complexity bounds under Assumption (FSN). For example, in [Lan201], assuming that \( |\text{supp}(\xi_t)| \leq N \), for all \( t \in [T-1] \), the probability of having an effective iteration is bounded by \( \frac{1}{N^{1/2}} \).

### 5 Extension to risk-averse setting

We now briefly discuss extensions involving a maximization problem in the dynamic programming equation, arising for example from multistage risk-averse, robust or distributionally robust problems. Algorithm 1 can be adapted to such problems, by changing the definitions of the Bellman operators.

Further, in the risk neutral case, Algorithm 1 is not symmetrical in its treatment of lower and upper approximations. As noted in Remark 3 for a minimization problem, in Algorithm 1 the forward phase Line 6 should be done using the lower approximations \( \tilde{V}_t^k \). More generally, one should use an outer approximation (that is under approximation for min sub-problems and upper approximations for max sub-problems) during the forward phase to be able to explore the state space. Thus, for those min-max problems the computation of upper approximations \( \tilde{V}_t^k \) is not optional.

**Minimax problems.** Baecke, Downward and Zakeri, in [BDZ12], presented a convergent problem-child algorithm to solve stochastic minimax dynamic programs. Although our framework of Algorithm 1 do not handle such minimax problem, we can extend it to do so. More precisely, we consider a problem where the decision maker chooses \( x_t \in \mathcal{X}_t(x_{t-1}, y_{t-1}, \xi_t) \), and then an adversary chooses \( y_t \in \mathcal{Y}_{t-1}(x_{t-1}, y_{t-1}, x_t, \xi_t) \). Thus, the Bellman operators are now defined as

\[
E_{t-1}(\tilde{V})(x_{t-1}, y_{t-1}) = \mathbb{E} \left[ \min_{x_t \in \mathcal{X}_t(x_{t-1}, y_{t-1}, \xi_t)} \max_{y_t \in \mathcal{Y}_{t-1}(x_{t-1}, y_{t-1}, x_t, \xi_t)} \ell_t(x_t, y_t, \xi_t) + \tilde{V}(x_t, y_t) \right].
\]

The reachable sets then become

\[
X_0^t = \{ x_0 \} \quad \quad Y_0^t = \{ y_0 \}
\]
\[
X_t^t = \bigcup_{x_{t-1} \in X_{t-1}^t} \bigcup_{y_{t-1} \in Y_{t-1}^t} \mathcal{X}_t(x_{t-1}, y_{t-1}, \xi_t) \quad \forall t \in [T].
\]
\[
Y_t^t = \bigcup_{x_{t-1} \in X_{t-1}^t} \bigcup_{y_{t-1} \in Y_{t-1}^t} \bigcup_{x_t \in X_t^t} \mathcal{Y}_t(x_{t-1}, y_{t-1}, x_t, \xi_t) \quad \forall t \in [T].
\]
In the forward phase, as in Algorithm 1, the $\gamma^F_t$-optimal solution $x_t^k$ should be chosen thanks to the approximation $V^t_{k-1}$. However, as we maximize over $y_t$, $y_t^k$ must be a $\gamma^F_t$-optimal solution of the step problem with the approximation $V^t_{k-1}$:

$$x_t^k = \mathcal{F}_{t-1}^{\min}(V^t_{k-1})(x_{t-1}, y_{t-1}, \xi^k_t) \in \gamma^F_t - \arg\min_{x_t \in X_t(x_{t-1}, y_{t-1}, \xi^k_t)} \max_{y_t \in Y_t(x_{t-1}, y_{t-1}, x_t, \xi^k_t)} \ell_t(x_t, y_t, \xi^k_t) + V^t_{k-1}(x_t, y_t)$$

(53a)

$$y_t^k = \mathcal{F}_{t-1}^{\max}(V^t_{k-1})(x_{t-1}, y_{t-1}, x_t^k, \xi^k_t) \in \gamma^F_t - \arg\max_{y_t \in Y_t(x_{t-1}, y_{t-1}, x_t^k, \xi^k_t)} \ell_t(x_t, y_t, \xi^k_t) + V^t_{k-1}(x_t, y_t)$$

(53b)

Assuming that the reachable sets $X_t^r$ and $Y_t^r$ have finite dimensions $d_x$ and $d_y$ and diameter $D$, and that the objective function are $L$-Lipschitz, the convergence and complexity results still hold developing on the ideas of [ZST19]. The upper bound on the number of effective iterations then becomes $K_z := \left(\frac{2DL}{\varepsilon - \gamma \Sigma}\right)^{d_x + d_y}(T-1)^{d_x + d_y + 1}$.

### Robust

Closely related, in [GTW19], Georgiou, Tsoukalas and Wiesemann presented the Robust Dual Dynamic Programming algorithm (RDDP) to solve multistage robust optimization problems. In such problems, instead of minimizing the expectation like in Eq. (MSP), we minimize considering the worst case scenario $\xi_t \in \Xi_t$. In this setting, the Bellman operator reads

$$\mathcal{B}_{t-1}(\tilde{V}) = \max_{\xi_t \in \Xi_t} \min_{x_t \in X_t(x_{t-1}, \xi_t)} \ell_t(x_t, \xi_t) + \tilde{V}(x_t).$$

(54)

Note that this robust setting can be seen as a particular case of minimax problems where we have deterministic random variables. Indeed, if we invert the order of max and min, either by changing the indices or by taking the opposite, and Eq. (54) can be written as Eq. (51) where $\xi_t$ of (54) plays the role of $y_t$ and the $\xi_t$ of (51) are deterministic parameter. The upper bound on the number of effective iterations then becomes $K_z := \left(\frac{2DL}{\varepsilon - \gamma \Sigma}\right)^{d_x + d_y}(T-1)^{d_x + d_y + 1}$.

### Risk averse

Multistage stochastic problems in the risk averse setting are MSP where the expectation is replaced by a multiperiod risk measure. In the nested coherent risk measure framework we present conditions under which Algorithm 1 can be adapted.

Let $\rho$ be a coherent risk measure (see [ADEH99], [ADE+07] or [SDR14] Def 6.4) the Bellman operator in the risk averse setting reads

$$\mathcal{B}_{t-1}(\bar{V}) : x_{t-1} \mapsto \rho\left(\min_{x_t \in X_t(x_{t-1}, \xi_t)} \ell_t(x_t, \xi_t) + \bar{V}(x_t)\right).$$

(55)

We recall a classical Fenchel representation theorem for proper, lower semicontinuous, law-invariant, coherent risk measure (see [SDR14] Thm 6.5). For every random variable $z \in L_1(\Omega, \mathcal{A}, \mathbb{P}, \mathbb{R})$, we have

$$\rho(z) = \max_{y \in \mathfrak{A}_\rho} \mathbb{E}_\mathbb{P}[yz]$$

(56)

where $\mathfrak{A}_\rho := \{y \in L_\infty(\Omega, \mathcal{A}, \mathbb{P}, \mathbb{R}) \mid \mathbb{E}[y] = 1, y \geq 0 \text{ a.s., } \mathbb{E}[yz'] \leq \rho(z), \forall z' \in L_1(\Omega, \mathcal{A}, \mathbb{P}, \mathbb{R})\}$.

With this representation, we get

$$\mathcal{B}_{t-1}(\bar{V}) = \max_{y \in \mathfrak{A}_\rho} \mathbb{E}_\mathbb{P}\left[\min_{x_t \in X_t(x_{t-1}, \xi_t)} y \ell_t(x_t, \xi_t) + y\bar{V}(x_t)\right].$$

(57)

Up to a slight change of notation, we can write this problem as a minimax problem. In particular, a sufficient condition to obtain convergence and complexity bounds for risk averse MSP is that the set $\mathfrak{A}_\rho$ has a finite dimension and a finite diameter. For example, if $\Omega$ is finite, $\mathfrak{A}_\rho$ is contained in the space of random variables in $\Omega$, isomorphic to a simplex of dimension $|\Omega| - 1$ which has finite diameter. More generally, if $\mathfrak{A}_\rho$ is contained in the convex hull of $n$ random variables $(y_k)_{k \in [n]}$, then $\mathfrak{A}_\rho$ has a finite diameter smaller than $\max_{k, k' \in [n]}(\|y_k - y_{k'}\|_\infty)$ and a finite dimension smaller than $n - 1$. Thus, we obtain complexity results similar to Corollary 23 and Theorem 27 with $K_z := \left(\frac{2DL}{\varepsilon - \gamma \Sigma}\right)^{d_x + d_y}(T-1)^{d_x + d_y + 1}$.

We now comment the particular case of the average value at risk [RU+00] with value $\alpha \in [0, 1)$, denoted $AV@R_\alpha$ and defined as:

$$AV@R_\alpha(z) := \inf_{s \in \mathbb{R}} \left\{ s + \frac{1}{1 - \alpha} \mathbb{E}_\mathbb{P}[\max(z - s, 0)] \right\}$$

(58)

22
We cannot use the dual representation Eq. (56) to derive complexity bounds as $\mathfrak{A}_{AV\otimes R}$ has, in general, non-finite dimension. However note that in Eq. (58), since $AV\otimes R_\alpha(z) \leq \frac{E_\mathcal{P}[z]}{1-\alpha}$ the infimum on $s$ over $\mathbb{R}$ can be replaced by a minimum on the compact interval $[0, \frac{1}{1-\alpha} E_\mathcal{P}[z]]$. To obtain an upper bound that does not depend on $k$ and $x_{t-1}$, we set $z = \min_{x_t \in X} (x_t - 1, \xi_t) \ell_t(x_t, \xi_t) + V^k_t(x_t)$ then $E_\mathcal{P}[z]$ is upper bounded by $\min_{x_t \in X} E[\ell_t(x_t, \xi_t) + V^k_t(x_t)]$ which has a finite value by Assumption 3. Thus, MSP with nested average value at risk measure can be handled by this framework and we can obtain complexity results similar to Corollary 23 and Theorem 27 with $K_\varepsilon := \left(\frac{2DL}{\varepsilon - \gamma} \right)^{d+1} (T - 1)^{d+2}$.

**Acknowledgements** We sincerely thank Mr Julien Weibel for interesting and useful discussions concerning Appendix D.2. This research benefited from the support of the FMJH Program Gaspard Monge for optimization and operations research and their interactions with data science.
A Cut methodologies

In this section, for the sake of completeness, we give several cuts that are used in different algorithms to solve particular multistage problems.

<table>
<thead>
<tr>
<th>Cut</th>
<th>Oracle needed</th>
<th>Setting and advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benders</td>
<td>First order</td>
<td>Convex, simple to implement</td>
</tr>
<tr>
<td>Reverse norm</td>
<td>Zeroth order and Lipschitz constant</td>
<td>Lipschitz</td>
</tr>
<tr>
<td>Step</td>
<td>Zeroth order, ( \varepsilon ) and ( \gamma )</td>
<td>Monotonic</td>
</tr>
<tr>
<td>Lagrangian</td>
<td>Solving a dual problem</td>
<td>Problem with small duality gap</td>
</tr>
<tr>
<td>Integer</td>
<td>Zeroth order</td>
<td>Binary variables</td>
</tr>
<tr>
<td>Adaptive partition</td>
<td>Adapted partition oracle</td>
<td>Linear, whole tangent cone</td>
</tr>
<tr>
<td>Generalized conjugacy</td>
<td>Conjugate computation</td>
<td>Regularisation</td>
</tr>
<tr>
<td>Saddle</td>
<td>First order and Lipschitz constant</td>
<td>Minimax problems</td>
</tr>
<tr>
<td>Fenchel conjugate</td>
<td>Compute fenchel transform of Bellman equation</td>
<td>Linear, Exact upper bound</td>
</tr>
</tbody>
</table>

Table 2: Synthesis of different cuts and oracle required

A.1 Benders cuts for convex functions

The most commonly used cuts are the Benders cuts which are affine functions. This kind of cut only works if the expected cost-to-go functions are convex.

The word cut is actually used because the graph of a Benders cut is a hyperplane which is tangent to the epigraph of the approximated function.

**Proposition 29.** Let \( F \) be a convex function and \( g \in \partial F(x) \) a subgradient of \( F \). We define the Benders cut \( f \) at \( \hat{x} \) as

\[
f(x) := F(\hat{x}) + g^\top (x - \hat{x})
\]

(59)

Then, \( f \) is valid and tight at \( \hat{x} \).

**Proof.** By definition of a subgradient, \( f(x) := F(\hat{x}) + g^\top (x - \hat{x}) \leq F(x) \), thus \( f \) is valid. By definition of \( f \), \( f(\hat{x}) = F(\hat{x}) + g^\top (\hat{x} - \hat{x}) = F(\hat{x}) \) thus \( f \) is tight at \( \hat{x} \).

We see that a first order oracle for the function \( F \), i.e., an oracle that returns the value \( F(\hat{x}) \) and a subgradient \( g \in \partial F(\hat{x}) \) for an input \( \hat{x} \), provides a direct algorithm to compute Benders cut.

A.2 Reverse norm cuts for Lipschitz functions

Stochastic Lipschitz Dynamic Programming (SDLP) presented in [ACdC20] provides an algorithm to deal with non-convex Lipschitz multistage stochastic programs. In this setting, the cost functions \( \ell_t \) are simply assumed to be Lipschitz continuous. Thus, the expected cost-to-go functions \( F_t \) is not necessarily convex and Benders cuts are not valid anymore. Ahmed, Cabral and da Costa replaced these cuts by reverse norm cuts or augmented lagrangian cuts using only the Lipschitz property of expected cost-to-go functions.

**Proposition 30.** Let \( F \) be a function with Lipschitz constant \( L \) (for norm \( \| \cdot \| \)). We define the reverse norm cut \( f \) of \( F \) at \( \bar{x} \) as

\[
f(x) := F(\bar{x}) - L\|x - \bar{x}\|
\]

(60)

Then, \( f \) is valid and 0-tight at \( \bar{x} \).

**Proof.** For any given \( x \) and \( \bar{x} \)

\[
f(x) = F(\bar{x}) - L\|x - \bar{x}\|
\]

(61a)

\[
= F(\bar{x}) - F(x) + F(x) - L\|x - \bar{x}\|
\]

(61b)

\[
\leq L\|\bar{x} - x\| + F(x) - L\|x - \bar{x}\|
\]

(61c)

\[
= F(x)
\]

(61d)

Thus, \( f \) is valid. By definition of \( f \), \( f(\bar{x}) = F(\bar{x}) - L\|\bar{x} - \bar{x}\| = F(\bar{x}) \) thus \( f \) is tight at \( \bar{x} \).
We see that a zeroth order oracle for the function $F$, i.e., an oracle that returns the value $F(\hat{x})$ for an input $\hat{x}$, together with a Lipschitz constant $L$ provides a direct algorithm to compute reverse norm cuts. Thus, SDLP integrates our framework in Algorithm 1.

We can also define the norm cut $f(x) := F(\hat{x}) + L\|x - \hat{x}\|$. These norm cuts can be used to compute $V^k_t$. The algorithm Tropical dynamic programming in [ACT20] uses this upper cuts together with Benders cuts for lower approximation $\overline{V}^k_t$ and thus integrates the framework of Algorithm 1.

### A.3 Step cuts for monotonic functions

We now look at “almost monotonic” expected cost-to-go functions.

**Proposition 31.** Let $F$ be a function such that there exists $\delta > 0$ and $\gamma \geq 0$ with

$$\forall x, y, x \leq y + \delta \mathbf{1} \implies F(x) \leq F(y) + \gamma$$

where $\mathbf{1}$ is the vector whose coefficients are all equal to 1. We assume that $F$ is upper bounded by $M$.

For a point $\hat{x}$, we define the upper increasing step cut $f$ as

$$f(x) := \begin{cases} F(\hat{x}) + \gamma & \text{if } x \leq \hat{x} + \delta \mathbf{1} \\ M & \text{otherwise} \end{cases}$$

Then, the upper increasing step cut $f$ satisfies

$$f(x) \geq F(x), \quad \forall x$$

$$f(\hat{x}) \leq F(\hat{x}) + \gamma$$

**Proof.** Since $\hat{x} \leq \hat{x} + \delta$, $f(\hat{x}) = F(\hat{x}) + \gamma$. Moreover, if $x \leq \hat{x} + \delta \mathbf{1}$, by Eq. (62) $F(x) \leq F(\hat{x}) + \gamma = f(x)$ and otherwise $f(x) = M \geq F(x)$. \[\Box\]

We could also define lower increasing step cuts for functions verifying Eq. (62). This cut methods also adapt to lower bounded decreasing functions, we will define in the same way upper and lower decreasing step cuts. However, these cuts are not Lipschitz. To integrate the framework of Algorithm 1 we could adapt these step cuts by interpolating with affine functions between the constant regions of the cuts.

In [PWB20], Philpott, Wahid and Bonnans presented an algorithm called mixed integer dynamic approximation scheme (MIDAS). This method applies to multistage mixed integer problems, given as a maximization problems. After adapting the problem by taking the opposite of the expected cost-to-go function, adding the constant $\gamma$ and choosing the right affinely interpolated step cut, the algorithm MIDAS integrates the framework of Algorithm 1 with step cuts.

### A.4 Lagrangian cuts

Lagrangian cuts were introduced for TFDP by Zou, Ahmed and Sun in [ZAS19]. These cuts are based on the Lagrangian relaxation of an optimisation problems.

**Proposition 32.** Let $F$ be a function, $H$ be a convex function and $x \mapsto Y(x)$ be a graph-convex set valued mapping see [RW09, p155] such that $F$ is defined as

$$F(x) := \inf_{y \in Y(x)} \ell(y)$$

We define the Lagrangian cut $f$ at $\hat{x}$ as

$$f(x) := \hat{\lambda}^\top x + \hat{\beta}$$

where

$$\hat{\lambda} \in \arg\max_{\lambda} \lambda^\top \hat{x} + \inf_{y,z | y \in Y(x)} \ell(y) - \lambda^\top z$$

$$\hat{\beta} = \inf_{y,z | y \in Y(x)} \ell(y) - \hat{\lambda}^\top z$$

Then, the Lagrangian cut is valid and tight at $\hat{x}$. 25
Proof. Consider \( x \in \text{dom}(Y) \). We rely on a strong duality result:

\[
F(x) = \inf_{y \in Y(x)} \ell(y) \tag{69a}
\]

\[
= \inf_{y, z | y \in Y(x) \text{ and } z = x} \ell(y) \tag{69b}
\]

\[
= \inf_{y, z | y \in Y(x)} \max_{\lambda} \ell(y) + \lambda^\top (x - z) \tag{69c}
\]

\[
= \max_{\lambda} \lambda^\top x + \inf_{y, z | y \in Y(x)} \ell(y) - \lambda^\top z \tag{69d}
\]

\[
= F(x) \tag{69e}
\]

Indeed, as \( Y \) is graph-convex, we have that \( \{(y, z) | y \in Y(z)\} \) is a non-empty convex set. Thus, we have

\[
f(\hat{x}) = \hat{\lambda}^\top \hat{x} + \inf_{y, z | y \in Y(z)} \ell(y) - \hat{\lambda}^\top z \tag{70a}
\]

\[
= \max_{\lambda} \hat{\lambda}^\top \hat{x} + \inf_{y, z | y \in Y(z)} \ell(y) - \lambda^\top z \tag{70b}
\]

\[
= F(x) \tag{70c}
\]

and \( f \) is tight at \( \hat{x} \). Moreover,

\[
f(x) = \hat{\lambda}^\top x + \inf_{y, z | y \in Y(z)} \ell(y) - \hat{\lambda}^\top z \leq \max_{\lambda} \lambda^\top x + \inf_{y, z | y \in Y(z)} \ell(y) - \lambda^\top z \tag{71a}
\]

\[
= F(x) \tag{71b}
\]

and then \( f \) is valid. \( \square \)

Note that this result is still true without the convexity assumption if we replace the tightness result by a lower \( \gamma \)-tightness result where \( \gamma \) is the duality gap.

Secondly, in this simplified setting and when the variable \( x \) takes value in a continuous space, the Lagrangian cut can be seen as a Benders cut since \( \hat{\lambda} \) is a subgradient of \( F \) at \( \hat{x} \). Nevertheless, the point of view of Lagrangian allows new ideas to compute cuts. In particular, one can define the Lagrangian cut with a different relaxation to deal with more complex setting such as integer cases as presented in [ZAS19]. We can also combine the Lagrangian cut with the reverse norm cut, such ideas are presented under the name augmented Lagrangian cuts in [ACdC20]. Thus, the algorithm SLDP from [ACdC20] integrates the framework of Algorithm 1.

Zhang and Sun in [ZS19] generalized these Lagrangian cuts by introducing the point of view of generalized conjugacy (see [RW09, Chapter 11]). They called these new cuts generalized conjugacy cuts, and proved them to be tight and valid, thanks to the Fenchel-Young inequality.

### A.5 Integer optimality cuts

The integer optimality cuts where first introduced by Laporte and Louveaux in [LL93] for 2-stage stochastic integer problems.

**Proposition 33.** Let \( F \) be a function taking value in \( \{0, 1\}^d \) and \( \hat{x} \in \{0, 1\}^d \) be a binary vector with \( \hat{S} := \{i | x_i = 1\} \).

We assume that \( F \) is lower bounded by \( \underline{M} \).

We define the integer optimality cut \( f \) as

\[
f(x) := (F(\hat{x}) - \underline{M})(\sum_{i \in \hat{S}} x_i - \sum_{i \notin \hat{S}} x_i - |\hat{S}| + 1) + \underline{M} \tag{72}
\]

Then, \( f \) is a valid and tight at \( \hat{x} \).

**Proof.** We have that \( f(\hat{x}) = (F(\hat{x}) - \underline{M})(|\hat{S}| - 0 - |\hat{S}| + 1) + \underline{M} = F(\hat{x}) \). Thus, \( f \) is tight at \( \hat{x} \). Let \( x \in \{0, 1\}^d \) different from \( \hat{x} \), we have \( \sum_{i \in \hat{S}} x_i - \sum_{i \notin \hat{S}} x_i \leq |\hat{S}| - 1 \). Then, \( f(x) \leq \underline{M} \leq F(x) \) and thus \( f \) is valid. \( \square \)
In [ZAS19], Zou, Ahmed and Sun presented an algorithm called Stochastic dual dynamic integer programming (SDDiP), suggesting to use integer optimality cuts or Lagrangian cuts instead of classical Benders cuts. By tightness and validity of these cuts, the algorithm SDDiP integrates the framework of Algorithm 1 (with a potentially large Lipschitz constant).

B  Explicit valid and adapted partition, geometric tools and proofs

In this appendix, we give an elementary definition of the partition $S_W$. This then allow us to prove Theorem 13 and Theorem 15 of Section 3.

B.1  Another elementary definition of $S_W$

We give an alternative and equivalent definition of the partition $S_W$ without directly using the fundamental notion of secondary fan.

Let us denote by $(w_i)_{i \in [m]}$ the columns of the matrix $W \in \mathbb{R}^{l \times m}$, and choose $q \in \mathbb{R}^m$. We shall think of $(w_i)_{i \in [m]}$ as a vector configuration in $\mathbb{R}^l$, and $q$ as a height vector: for each $i \in [m]$, we draw the point $(w_i, q_i) \in \mathbb{R}^l \times \mathbb{R}$. We now consider the convex hull $E$ of the points $(w_i, q_i)$ in $\mathbb{R}^l \times \mathbb{R}$ (see Fig. 3). The geometric regular subdivision induced by the height vector $q$ is the polyhedral complex defined as the projection onto $\mathbb{R}^l$ of the lower faces of the polyhedron $E$ (i.e., the faces of $E$ that have a normal vector of the form $(\lambda, -1)$). A lower face $F$ of $E$ can be represented by the set of indices of columns of $W$ defining $F$, that is $I_F = \{i \in [m] \mid (w_i, q_i) \in F\}$. For example, if $(w_i, q_i)$ is an extreme point of $F$ then $i \in I_F$. The collection $\mathcal{I}(W^\top, q)$ of the sets $I_F$, for $F$ describing the lower faces of $E$, is known as a combinatorial regular subdivision. More details can be found in [DLRS10] Chapter 2.5.

![Figure 3: Three lifted vector configurations, their projections and the regular subdivisions $\mathcal{I}(W^\top, q)$ induced for different values of $\omega_5$. We define $\mathcal{I}_{com} := \{0, 1, 2, 3, 4, \{1, 2\}, \{2, 3\}, \{3, 4\}\}$](image)

These notions are formalized by the following definition.

**Definition 34** (Regular subdivisions). Consider a matrix $W \in \mathbb{R}^{l \times m}$. Let us denote by $(w_i)_{i \in [m]}$ the columns of the matrix $W$, and let $q \in \mathbb{R}^m$. The (combinatorial) regular subdivision of the configuration of vectors $(w_i)_{i \in [m]}$ induced by the height vector $q$ is the collection $\mathcal{I}(W^\top, q)$ of subsets of $[m]$ such that

$$\mathcal{I}(W^\top, q) := \left\{ I \subset [m] \mid \exists \lambda \in \mathbb{R}^l, \quad w_i^\top \lambda = q_i, \quad \forall i \in I, \quad w_j^\top \lambda < q_j, \quad \forall j \notin I \right\}$$  (73)
We define \( \sim_W \) the equivalence relation on \( \mathbb{R}^m \) such that \( q \sim_W q' \) iff \( I(W^\top, q) = I(W^\top, q') \).

We now give an equivalent characterization of the partition \( S_W \) defined in Eq. (30), see Chapter 5 and in particular Theorem 5.2.16 of [DLRS10] for a proof of this equivalence.

**Proposition 35.** The partition \( S_W \) corresponds to the collection of equivalence classes of the relation \( \sim_W \).

We now prove that \( q \mapsto R_{x,W,q} \) is constant on each \( S \in S_W \).

**Proof of Lemma** [12] By [DLRS10] Theorem 9.5.6, for every \( q \in \mathbb{R}^n \), we have

\[
N(D_{W,q}) = \{ W_I \mathbb{R}^J_+ \mid I \in I(W^\top, q) \}. 
\]

(74)

Thus, for \( q, q' \in S \), we have \( N(D_{W,q}) = N(D_{W,q'}) \). Further, as, by definition (28b), \( R_{x,W,q} = \{ E_{N,x} \mid N \in N(D_{W,q}) \} \), we get \( R_{x,W,q} = R_{x,W,q'} \).

**B.2 Proof of basis decomposition theorem**

The goal of this appendix is to prove Theorem [13]. We start by recalling some usual definitions and results in linear programming’s theory that can be found in any standard linear programming book, e.g. [MG07].

**Definition 36** (Basic point, reduced cost). We consider the linear problem

\[
Q(x, W, q, T, h) := \min_{y \in \mathbb{R}^m_+} \left\{ q^\top y \mid Tx + Wy = h \right\}. 
\]

(75)

We say that \( B \subset [m] \) is a basis if the submatrix \( W_B = (w_i)_{i \in B} \), where \( w_i \) is the \( i \)-th column of \( W \), is invertible. The basic point associated to basis \( B \) is the vector in \( \mathbb{R}^m \), with coordinates \( y_B := (W_B^{-1}(h - Tx))_{i \in B} \) and 0 for \( i \notin B \). A base \( B \) is said to be admissible (resp. optimal), if its associated basic point is an admissible (resp. optimal), solution of Eq. (75). Finally, the reduced cost associated to a basis \( B \) is the vector

\[
(q_j - w_j^\top W_B^{-1}q_B)_{j \in [m]} 
\]

(76)

Reduced cost is a key ingredient of the simplex method. In particular, it is well known that an admissible basis is optimal if and only if its reduced cost is non-negative. More formally, we have:

**Lemma 37.** Let \( B \subset [m] \) a basis. If \( y_B := (W_B^{-1}(h - Tx))_{i \in B} \geq 0 \) and for all \( j \in [m] \), \( q_j - w_j^\top W_B^{-1}q_B \geq 0 \).

Then, \( B \) is optimal and in particular,

\[
Q(x, W, q, T, h) = q_B^\top W_B^{-1}(h - Tx) 
\]

(77)

Finally, we recall a classic generalization of Caratheodory’s theorem for conic hull.

**Lemma 38** (Caratheodory). Let \( W \in \mathbb{R}^{l \times m} \) be a matrix and a subset of indices \( J \subset [m] \) such that \( \text{span}((w_i)_{i \in J}) := W_J \mathbb{R}^l = \mathbb{R}^l \) where \( w_i \) is the \( i \)-th column of \( W \). Consider a vector \( h \) in the conic hull of \( (w_i)_{i \in J} \), i.e. \( h \in W_J \mathbb{R}^l_+ \). Then, there exists a basis \( B \subset J \) such that \( h \) is in the conic hull of \( (w_i)_{i \in B} \), i.e. \( h \in W_B \mathbb{R}^l_+ \).

**Proof.** Let \( I \subset J \) be such that \( h \in W_I \mathbb{R}^l_+ \) and \( (w_i)_{i \in I} \) is spanning \( \mathbb{R}^l \). There exist \((\mu_i)_{i \in I} \in \mathbb{R}^I_+ \) non negative coefficients such that \( h = \sum_{i=1}^m \mu_i w_i \). Assume that \( I \) is not a basis, then \((w_i)_{i \in I} \) is not linearly independent, that is there exists a collection \((\lambda_i)_{i \in I} \in \mathbb{R}^I \) such that \( \sum_{i \in I} \lambda_i w_i = 0 \) with at least one \( \lambda_i \) different of zero, that can be assumed w.l.o.g positive.

Define \( j := \arg \min_{i \in I, \lambda_i > 0} \frac{\lambda_i}{\lambda_j} \). Then, we have \( w_j = - \sum_{i \in I \setminus \{j\}} \frac{\lambda_i}{\lambda_j} w_i \) and thus \( h = \sum_{i \in I \setminus \{j\}} (\mu_i - \mu_j \frac{\lambda_i}{\lambda_j}) w_i \).

In particular, \((w_i)_{i \in I \setminus \{j\}} \) is spanning \( \mathbb{R}^l \). We now show that each coefficient in this sum is non negative, that is \( h \in W_{I \setminus \{j\}} \mathbb{R}^l_+ \). Note that \( \lambda_j > 0 \) and for all \( i \in I, \mu_i \geq 0 \). Thus, if \( \lambda_i \leq 0 \) we have \( \mu_i - \mu_j \frac{\lambda_i}{\lambda_j} \geq 0 \). Otherwise, \( \lambda_j > 0 \), and by definition of \( j \), \( \frac{\lambda_i}{\lambda_j} \geq \frac{\mu_i}{\mu_j} \) and thus \( \mu_i - \mu_j \frac{\lambda_i}{\lambda_j} \geq 0 \). Which shows that \( h = \sum_{i \in I \setminus \{j\}} (\mu_i - \mu_j \frac{\lambda_i}{\lambda_j}) w_i \in W_{I \setminus \{j\}} \mathbb{R}^l_+ \).

By induction, we drop indices until we get a basis \( B \).

We now have all the tools required for the proof of Theorem [13].
proof of Theorem 12. Let \( S \in S_W \), \( R \in \mathcal{R}_{x,W,S} \) and \( q \in S \). Recall that \( \mathcal{R}_{x,W,q} = \{ E_{N,x} \mid N \in \mathcal{N}(D_{W,q}) \} \). Thus, there exists \( N_0 \in \mathcal{N}(D_{W,q}) \) such that \( R = E_{N_0,x} \) and \( N \in \mathcal{N}(D_{W,q}) \) a full dimensional cone such that \( N_0 \subset N \). As \( \mathcal{N}(D_{W,q}) = \{ W_i \eta_i | I \in \mathcal{I}(W^+,q) \} \), there exists \( I \in \mathcal{I}(W^+,q) \) such that \( N = W_i \eta_i \). Since \( N \) is full dimensional, \((w_i)_i \in I \) is spanning \( \mathbb{R}^l \). Finally, by definition of the regular subdivision \( \mathcal{I}(W^+,q) \) in (73), there exists \( \lambda(I) \) such that

\[
\forall i \in I, \quad w_i^T \lambda(I) = q_i
\]

(78a)

\[
\forall j \notin I, \quad w_j^T \lambda(I) < q_j.
\]

(78b) Let \( (T,h) \in E_{N_0,x} = \{ (T',h') | h' - T'x \in r_i(N) \} \), we have that \( h - Tx \in N_0 \subset N = W_i \eta_i \). By Caratheodory’s Lemma 38 as \((w_i)_i \in I \) is spanning \( \mathbb{R}^l \), there exists a basis \( B \subset I \), such that \( h - Tx \in \mathbb{W}_B \eta_B = W_B \eta_B \). In particular, we have that \( y_B = W_B^{-1}(h-Tx) \geq 0 \), thus \( B \) is an admissible basis. Moreover, as \( B \subset I \), by (78a) we have that, for all \( i \in B \), \( w_i^T \lambda(I) = q_i \), which in turn implies \( \lambda(B) = W_B^{-1} \eta_B \). Thus, for all \( j \in [m] \), we can compute the reduced cost coordinate \( q_j - w_j^T W_B^{-1} \eta_B = q_j - w_j^T \lambda(I) \geq 0 \), by (78a) and (78b). By Lemma 37, \( B \) is an optimal basis, leading to

\[
Q(x,W,q,T,h) = q_B^T W_B^{-1}(h-Tx).
\]

Note that the resulting formula does not depend on the choice of the extracted basis \( B \). Indeed, let \( B' \subset I \) be a basis. As for all \( i \in B' \), \( w_i^T \lambda(I) = q_i \), we also have \( W_{B'}^{-1} \eta_{B'} = \lambda(I) = W_B^{-1} \eta_B \). Thus, \( Q(x,W,q,T,h) = q_B^T W_B^{-1}(h-Tx) = q_{B'}^T W_{B'}^{-1}(h-Tx) \).

### B.3 Proof of explicit valid and adapted partitions

We now prove Theorem 15.

Proof of Theorem 15. We have

\[
V_{P_{x}}(\hat{x}) := \sum_{P \in P_{x}} \mathbb{P}[\{(W,q,T,h) \in P \} \mid Q(\hat{x},E[(W,q,T,h) \mid (W,q,T,h) \in P])]
\]

(79a)

\[
= \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} \sum_{R \in \mathcal{R}_{x,W,S}} \mathbb{P}[\{W = W, q \in S, (T,h) \in R \} \mid Q(\hat{x},E[(W,q,T,h) \mid W = W, q \in S, (T,h) \in R])] \quad (79b)
\]

\[
= \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} \sum_{R \in \mathcal{R}_{x,W,S}} \mathbb{P}[\{W = W, q \in S, (T,h) \in R \} \mid Q(\hat{x},W,E[(q,T,h) \mid W = W, q \in S, (T,h) \in R])] \quad (79c)
\]

\[
= \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} \sum_{R \in \mathcal{R}_{x,W,S}} \mathbb{P}[\{W = W, q \in S, (T,h) \in R \} \mid \mathbb{E}[Q(\hat{x},W,q,T,h) \mid W = W, q \in S, (T,h) \in R]] \quad (79d)
\]

\[
= \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} \sum_{R \in \mathcal{R}_{x,W,S}} \mathbb{E}[Q(\hat{x},W,q,T,h) \mid W = W, q \in S, (T,h) \in R] \quad (79e)
\]

\[
= \mathbb{E}[Q(\hat{x},W,q,T,h)] = V(\hat{x}) \quad (79f)
\]

Eq. (79a) comes from the definition of the partitioned expected cost-to-go function \( V_{P_{x}} \) (see (20)), and Eq. (79b) from the definition of \( P_{x} \). The equality (79b) = (79c) is simply the abuse of notation presented in (21). Conditioned by \( W = W \), we can use Eq. (33) to obtain (79c) = (79d). Finally, law of total expectation yields (79e) = (79f).

We now prove that \( V_{P_{x}} \leq V \). For all \( W \in \text{supp}(W) \) and \( S \in S_W \), we denote \( E_{W,S} \) (resp \( \mathbb{P}_{W,S} \)) the expectation (resp. the probability) conditional to the event \( \{W = W, q \in S\} \). By the law of total expectation, we have

\[
V_{P_{x}}(x) = \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} \mathbb{P}[\{W = W, q \in S \} \mid \mathbb{E}_{W,S}[Q(x,W,E_{W,S}[(q,T,h) \mid (T,h) \in R])] \quad (80)
\]

Now by independence of \( q \) and \( (T,h) \)

\[
Q(x,W,E_{W,S}[(q,T,h) \mid (T,h) \in R]) = Q(x,W,E_{W,S}[q],E_{W,S}[(T,h) \mid (T,h) \in R]) \quad (81)
\]

By convexity of \((T,h) \mapsto Q(x,W,q,T,h)\) and Jensen inequality, we have that

\[
Q(x,W,E_{W,S}[q],E_{W,S}[(T,h) \mid (T,h) \in R]) \leq E_{W,S}[Q(x,W,E_{W,S}[q],T,h) \mid (T,h) \in R] \quad (82)
\]
Now, for an event $A$, note that we have, by applying the law of total expectation and Lemma 14 twice, and with the abuse of notation Eq. (21),

\[
E_{W,S}[Q(x,W,E_{W,S}[q],T,h)|A]
\]

\[
= \sum_{R \in \mathcal{R}_{x,W,S}} E_{W,S}[\mathbb{1}_{(T,h) \in R} \, Q(x,W,E_{W,S}[q],T,h)|A]
\]

\[
= \sum_{R \in \mathcal{R}_{x,W,S}} P_{W,S}[\{T,h\} \in R] \, E_{W,S}[Q(x,W,E_{W,S}[q],T,h)|A \cap (T,h) \in R]
\]

\[
= \sum_{R \in \mathcal{R}_{x,W,S}} P_{W,S}[\{T,h\} \in R] \, Q(x,E_{W,S}[(W,E_{W,S}[q],T,h)|A \cap (T,h) \in R])
\]

\[
= \sum_{R \in \mathcal{R}_{x,W,S}} P_{W,S}[\{T,h\} \in R] \, E_{W,S}[Q(x,W,q,T,h)|A \cap (T,h) \in R]
\]

\[
= \sum_{R \in \mathcal{R}_{x,W,S}} P_{W,S}[\{T,h\} \in R] \, E_{W,S}[Q(x,W,q,T,h)|A \cap (T,h) \in R]
\]

\[
= E_{W,S}[Q(x,W,q,T,h)|A]
\]

By replacing $A$ by $(T,h) \in R$, for $R \in \mathcal{R}_{x,W,S}$ to Eq. (83a), we have

\[
E_{W,S}[Q(x,W,E_{W,S}[q],T,h)|(T,h) \in R] = E_{W,S}[Q(x,W,q,T,h)|(T,h) \in R]
\]

Combining (80), (81) and (82), we now get

\[
V_{P_x}(x) \leq \sum_{W \in \text{supp}(W)} \sum_{S \in S_W} P[W = W,q \in S] \sum_{R \in \mathcal{R}_{x,W,S}} P_{W,S}[(T,h) \in R] \, E_{W,S}[Q(x,W,q,T,h)|(T,h) \in R]
\]

By the law of total expectation, we see that the right term is equal to $V(x)$. Thus, $V_{P_x} \leq V(x)$. □

C  Sufficient conditions for effective iterations

In this appendix, we want to prove Lemma 21. We start with a technical lemma linking the gap at $t-1$ with the expected gap for tentative points at $t$.

Lemma 39. Let Assumptions 1 to 5 be satisfied and $t \in [T-1]$, assume that $\delta_t \in [0,D_t]$ and $\eta_t \in \mathbb{R}_+$ are given and $\xi_t$ defined by (57). Then, for all algorithms satisfying the framework of Algorithm 1, we have for $t \in [T-1]$

\[
0 \leq V_{t-1}^k(x_{t-1}) - V_{t-1}^k(x_{t-1}) \leq E\left[V_{t}^{k-1}(y_t^k) - V_{t}^{k-1}(y_t^k)\right] + \gamma_{t-1} + \tau_{t-1} + \gamma_t^F
\]

where we recall that $y_t^k := F_{t-1}(V_{t-1}^{k-1}(x_{t-1})^k, \xi_t)$.

Proof.

\[
V_{t-1}^k(x_{t-1}) \geq B_{t-1}(V_k^k)(x_{t-1}) - \gamma_{t-1}
\]

(backward phase: $\gamma_{t-1}$-tight cut) (87a)

\[
= E\left[\min_{x \in \mathcal{X}_t(x_{t-1}^k, \xi_t)} \ell_t(x, \xi_t) + V_{t}^{k-1}(x)\right] - \gamma_{t-1}
\]

(definition of $B_t$) (87b)

\[
\geq E\left[\min_{x \in \mathcal{X}_t(x_{t-1}^k, \xi_t)} \ell_t(x, \xi_t) + V_{t}^{k-1}(x)\right] - \gamma_{t-1}
\]

(monotonicity of approx.) (87c)

\[
\geq E[\ell_t(y_t^k, \xi_t) + V_{t}^{k-1}(y_t^k)] - \gamma_t^F - \gamma_{t-1}
\]

(definition of $F_{t-1}$) (87d)
Let $t \in [T-1]$. We first prove that if one of the inequalities Eqs. [45a] and [45b] is satisfied then, $x_{t-1}^k$ is $\epsilon_{t-1}$-saturated as soon as $x_{t-1}^k$ is not $\delta_t$-distinguishable. Recall that $d_t^k(x) := \min_{0 < k < T} \|x - x_t^k\|$.

Assume now that $x_{t-1}^k$ is not $\delta_{t+1}$-distinguishable, then $d_t^k(x_t^k) \leq \delta_t$ and there exists $j < k$ such that $x_{t-1}^j$ is $\epsilon_{t-1}$-saturated and $\|x_t^j - x_t^k\| \leq \delta_t$. If Eq. [45a] is satisfied, we have

$$
\mathbb{E}[V_t^{k-1}(y_t^j) - V_t^{k-1}(y_t^k)] \leq V_t^{k-1}(x_t^j) - V_t^{k-1}(x_t^k) + (\lambda_t + \lambda_t)\eta_t
$$

Eq. (45a) (88a)

Similarly, if Eq. [45b] is satisfied, we define $j(x)$ such that

$$
\hat{x} \in \arg \min_{j \in \{1, \ldots, T\}} \|x_t^j - F_{t-1}(V_t^{j-1}(x_{t-1}^j, x_t^j))\|
$$

In particular, $d_t^k(F_{t-1}(V_t^{k-1}(x_{t-1}^k, x_t^k)) = \|x_t^j - F_{t-1}(V_t^{k-1}(x_{t-1}^k, x_t^j))\|$ and thus $\mathbb{E}[d_t^k(y_t^k)] = \mathbb{E}[[x_t^j - y_t^k]]$

$$
\mathbb{E}[V_t^{k-1}(y_t^j) - V_t^{k-1}(y_t^k)]
\leq \mathbb{E}[(\lambda_t + \lambda_t)\eta_t]
\leq (\lambda_t + \lambda_t)\eta_t
$$

Then, in both cases, $\mathbb{E}[V_t^{k-1}(y_t^j) - V_t^{k-1}(y_t^k)] \leq (\lambda_t + \lambda_t)\eta_t$. By Lemma 39, we have

$$
\mathbb{E}[V_t^{k-1}(x_{t-1}^k) - V_t^{k-1}(x_{t-1}^k)] \leq \mathbb{E}[V_t^{k-1}(y_t^k) - V_t^{k-1}(y_t^k)] + \gamma_{t-1} + \gamma_t + \gamma^F
$$

Thus, $x_t^k$ is $\epsilon_t$-saturated as soon as $x_{t+1}^k$ is not $\delta_t$-distinguishable.

We now prove by backward induction on $t$ that iteration $k$ is effective. We first prove that, for all $k \in \mathbb{N}^*$, $x_{T-1}^k$ is $\epsilon_{T-1}$-saturated.

$$
\mathbb{V}_{T-1}(x_{T-1}^k) - \mathbb{V}_{T-1}(x_{T-1}^k) \leq \mathbb{B}_{T-1}(\mathbb{V}_{T-1}(x_{T-1}^k)) + \gamma_{T-1} + \gamma_{T-1} + \gamma^F
$$

(91a)

(91b)
Let $t \geq 2$ such that, for every $\tau \geq t$, $x^k_\tau$ is $\varepsilon_\tau$-saturated. If $x^k_t$ is $\delta_t$-distinguishable, then iteration $k$ is effective. Otherwise, $x^k_t$ is not $\delta_t$-distinguishable and by the previous paragraph, it implies that $x^k_{t-1}$ is $\varepsilon_{t-1}$-saturated. Eventually, assume that $x^k_t$ is $\delta_1$-distinguishable, then iteration $k$ is effective. Otherwise, there exists $j < k$ such that $\|x^j_t - x^k_t\| \leq \delta_1$ and $x^j_t$ is $\varepsilon_1$ saturated. We get

$$V_1(x^k_t) \leq \tilde{V}_1^j(x^k_t)$$

$$= \tilde{V}_1^j(x^k_t) - \tilde{V}_1^j(x^j_t) + \tilde{V}_1^j(x^j_t)$$

$$\leq L_1 \|x^k_t - x^j_t\| + \tilde{V}_1^j(x^j_t)$$

$$\leq L_1 \delta_1 + \varepsilon_1 + \tilde{V}_1^j(x^j_t)$$

$$= (L_1 + L_1) \delta_1 + \varepsilon_1 + \tilde{V}_1^j(x^j_t)$$ (93a)

Then,

$$\ell_1(x^k_t, \xi_1) + V_1(x^k_t) \leq (L_1 + L_1) \delta_1 + \varepsilon_1 + \ell_1(x^k_t, \xi_1) + \tilde{V}_1^{k-1}(x^k_t)$$

$$\leq (L_1 + L_1) \delta_1 + \varepsilon_1 + \gamma_1^F + \min_{x_1 \in X_1(x_0)} \ell_1(x_1, \xi_1) + \tilde{V}_1^{k-1}(x^k_t)$$

$$\leq \varepsilon_0 + \min_{x_1 \in X_1(x_0)} \ell_1(x_1, \xi_1) + V_1(x_1)$$ (94a)

Thus, in all the covered cases, iteration $k$ is effective. 

\[ \square \]

**D Probabilistic lemmas**

In this appendix, we present useful probabilistic lemmas to prove the convergence of SDDP with randomized choice of $\xi^k_t$.

**D.1 A nested Hoeffding lemma**

**Lemma 40.** Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space, $X$ and $Y$ be two independent random variables taking values respectively in the euclidean spaces $\mathbb{X}$ and $\mathbb{Y}$.

Let $r > 0$ be a positive real and $f : \mathbb{X} \times \mathbb{Y} \rightarrow \mathbb{R}$ be a measurable function such that $0 \leq f(X, Y) \leq r$ almost surely.

Then for every $\eta > 0$ and $A \in \sigma(X)$ such that $\mathbb{P}[A] > 0$, we have

$$\mathbb{P}\left[f(X, Y) > \mathbb{E}[f(X, Y)|\sigma(X)] - \eta \right] \geq \mathbb{P}[A] > 1 - \mathbb{e}^{-2r^2}$$ (95)

**Proof.** Recall that the Hoeffding lemma states that if $Z$ is a real random variable such that there exists $a, b \in \mathbb{R}$ with $a \leq Z \leq b$ almost surely then for every $\eta > 0$ we have

$$\mathbb{P}\left[Z - \mathbb{E}[Z] \leq -\eta \right] \leq \mathbb{e}^{-\frac{2\eta^2}{(b-a)^2}}$$ (96)

By taking the complementary event, we have

$$\mathbb{P}\left[Z > \mathbb{E}[Z] - \eta \right] \geq 1 - \mathbb{e}^{-\frac{2\eta^2}{(b-a)^2}}$$ (97)

Then for every $x \in \mathbb{X}$, by applying the Hoeffding lemma to the random variable $Z = f(x, Y)$, $a = 0$ and $b = r$, we have

$$\mathbb{P}\left[f(x, Y) > \mathbb{E}[f(x, Y) - \eta \left] \geq 1 - \mathbb{e}^{-\frac{2\eta^2}{r^2}}\right.\right.$$ (98)
Let $A \in \sigma(X)$ and $B \subset X$ such that $A = X^{-1}(B)$

\[
\mathbb{P}\left[ \{f(X, Y) > \mathbb{E}[f(X, Y)|\sigma(X)] - \eta\} \cap A \right] = \int_{\Omega} \mathbb{1}\{f(X(\omega), Y(\omega)) > \mathbb{E}[f(X, Y)|\sigma(X)](\omega) - \eta\} \mathbb{1}_{\omega \in A} \mathbb{P}(d\omega) 
\]

(99a)

= \int \int X = (\ldots, X_n, \ldots, X_n) \mathbb{1}_{\omega \in \mathbb{E}[f(X, Y)|\sigma(X)](\omega) - \eta} \mathbb{1}_{\mathbb{R}^n} \mathbb{P}(d\omega) (99b)

= \int \int X = (\ldots, X_n, \ldots, X_n) \mathbb{1}_{\omega \in \mathbb{E}[f(X, Y)|\sigma(X)](\omega) - \eta} \mathbb{1}_{\mathbb{R}^n} \mathbb{P}(d\omega) (99c)

\[
\leq \int \int \mathbb{1}_{\mathbb{E}[f(X, Y)|\sigma(X)](\omega) - \eta} \mathbb{1}_{\mathbb{R}^n} \mathbb{P}(d\omega) (99d)
\]

\[
\geq \int \int \mathbb{1}_{\mathbb{E}[f(X, Y)|\sigma(X)](\omega) - \eta} \mathbb{1}_{\mathbb{R}^n} \mathbb{P}(d\omega) (99e)
\]

\[
= (1 - e^{-2\eta^2}) \mathbb{P}[B] (99f)
\]

\[
= (1 - e^{-2\eta^2}) \mathbb{P}[A] (99g)
\]

Thus, by dividing by $\mathbb{P}[A]$, we get $\mathbb{P}\left[ f(X, Y) > \mathbb{E}[f(X, Y)|\sigma(X)] - \eta \right] \geq 1 - e^{-2\eta^2}$. \hfill \square

**Lemma 41.** Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space, $(X_n)_{n \in \mathbb{N}}$ be a sequence of independent random variables taking values in the euclidean space $X$ and $\mathcal{A}_n = \sigma(X_n)_{k \in \mathbb{N}}$ be its adapted filtration.

For every $n \in \mathbb{N}$, let $r_n$ and $\eta_n$ be two positive real and $f_n : X^n \to \mathbb{R}$ be a measurable function such that $0 \leq f_n(X_1, \ldots, X_n) \leq r_n$ almost-surely.

We denote by $E_n$ the event $\{\omega \mid f_n(X_1, \ldots, X_n) > \mathbb{E}[f_n(X_1, \ldots, X_n)|A_{n-1}] - \eta_n\} \in \mathcal{A}_n$. Then, for all $m \leq n \in \mathbb{N}$ and $A_{m-1} \in \mathcal{A}_{m-1}$ such that $\mathbb{P}[A_{m-1}] > 0$, we have

\[
\mathbb{P}\left[ \bigcap_{k=m}^{n} E_k \mid A_{m-1} \right] \geq \prod_{k=m}^{n} \left( 1 - e^{-\frac{2\eta^2}{r_n^2}} \right) (100)
\]

**Proof.** For every $n \in \mathbb{N}^*$, $\eta > 0$ and $A_{n-1} \in \mathcal{A}_{n-1}$ such that $\mathbb{P}[A] > 0$, by the previous lemma applied to $X = (X_1, \ldots, X_n)$, $Y = X_n$, $f = f_n$, $\eta = \eta_n$ and $r = r_n$, we have

\[
\mathbb{P}\left[ E_n \mid A_{n-1} \right] \geq 1 - e^{-\frac{2\eta^2}{r_n^2}} (101)
\]

Let $m \in \mathbb{N}$, we now prove our lemma by induction on $n$. If $n = m = 1$ the result is true by the Hoeffding lemma and for $n = m > 1$ the result is true by Eq. (101) with $A_{n-1} = A_{m-1}$.

Let $n \geq m$ and assume that $\mathbb{P}\left[ \bigcap_{k=m}^{n} E_k \cap A_{m-1} \right] > 0$ and $\mathbb{P}\left[ \bigcap_{k=m}^{n} E_k \mid A_{m-1} \right] \geq \prod_{k=m}^{n} \left( 1 - e^{-\frac{2\eta^2}{r_n^2}} \right)$.

\[
\mathbb{P}\left[ \bigcap_{k=m}^{n+1} E_k \mid A_{m-1} \right] = \mathbb{P}\left[ E_{n+1} \mid \bigcap_{k=m}^{n} E_k \cap A_{m-1} \right] \mathbb{P}\left[ \bigcap_{k=m}^{n} E_k \mid A_{m-1} \right] (102a)
\]

\[
\geq \left( 1 - e^{-\frac{2\eta^2}{r_n^2}} \right) \prod_{k=m}^{n} \left( 1 - e^{-\frac{2\eta^2}{r_n^2}} \right) (102b)
\]

where we underestimate the first factor thanks to Eq. (101) and $\bigcap_{k=m}^{n} E_k \in \mathcal{A}_n$ and the second factor thanks to the induction hypothesis. In particular, $\mathbb{P}\left[ \bigcap_{k=m}^{n+1} E_k \cap A_{m-1} \right] > 0$ and induction ends the proof. \hfill \square

33
D.2 Stochastic dominance by geometric random variables

Recall that a real random variable \( X \) is (first-order) stochastically dominated by a real random variable \( Y \) if the cumulative density function of \( X \) is smaller than the cumulative density function of \( Y \). If \( X \) and \( Y \) are integer random variables, \( X \) is stochastically dominated by \( Y \) is equivalent to \( \Pr[X \geq n] \leq \Pr[Y \geq n] \), for all \( n \in \mathbb{N}^* \).

We now present a lemma where we leverage this notion to bound the number of effective iteration in randomized algorithm in Theorem 27.

**Lemma 42.** Let \((\Omega, A, \mathbb{P})\) be a space of probability, \((X_n)_{n \in \mathbb{N}}\) be a sequence of independent and identically distributed random variables and \(A_n = \sigma(X_k)_{k \in \mathbb{N}^*},\) be its adapted filtration. Let \((Y_n)_{n \in \mathbb{N}}\) be a sequence of (not necessarily independent neither identically distributed) binary random variables, i.e. taking values in \([0, 1]\), such that \(\sigma(Y_n) \subseteq A_n\). Assume that there exists \(p \in (0, 1)\) such that for all \(n \in \mathbb{N}^*\) and all \(A_n \in A_n\) such that \(\Pr[A_n] > 0\), we have

\[
\Pr[Y_{n+1} = 1 | A_n] \geq p
\]

(103)

For \(m \in \mathbb{N}\), we define the stopping time \(\tau_m := \inf\{n \in \mathbb{N} | \sum_{k=1}^{n} Y_i \geq m\}\). Let \(B_{m,p}\) be a random variable with a negative binomial distribution representing the number of trials to obtain \(m\) successes with probability of success \(p\), i.e. \(\Pr[B_{m,p} = n] = \binom{n-1}{m-1} p^m (1-p)^{n-m}, \) for all \(n \geq m\).

Then, \(\tau_m\) is stochastically dominated by \(B_{m,p}\) i.e.

\[
\Pr[\tau_m \geq n] \leq \Pr[B_{m,p} \geq n], \quad \forall n \in \mathbb{N}^*
\]

(104)

In particular,

\[
\mathbb{E}[\tau_m] \leq \mathbb{E}[B_{m,p}] = \frac{m}{p}
\]

(105)

**Proof.** Let \((\tilde{Y}_n)_{n \in \mathbb{N}^*}\), a sequence of independent and identically distributed Bernoulli random variables with parameter \(p\). For all \(n \in \mathbb{N}^*\), we define the random variables \(S_n := \sum_{k=1}^{n} Y_n\) and \(\tilde{S}_n := \sum_{k=1}^{n} \tilde{Y}_n\). We first show by induction on \(n\) that \(\tilde{S}_n\) is stochastically dominated by \(S_n\), i.e. for all \(a \in \mathbb{N}^*\), we have \(\Pr[S_n \geq a] \geq \Pr[\tilde{S}_n \geq a]\).

Indeed, for \(n = 1\) we have \(S_1 = Y_1\) and \(\tilde{S}_2 = Y_2\) then \(\Pr[S_1 \geq 0] = \Pr[\tilde{S}_1 \geq 0] = 1\) and \(\Pr[S_1 = 1] = \Pr[Y_1 = 1] \geq p = \Pr[\tilde{S}_1] \geq 1\). Finally, for all \(a \geq 2\), \(\Pr[S_1 = 2] = \Pr[\tilde{S}_1 = 2] = 0\).

We now assume that there exists \(n \in \mathbb{N}^*\) such that for all \(a \in \mathbb{N}^*\), \(\Pr[S_n \geq a] \geq \Pr[\tilde{S}_n \geq a]\). We then have

\[
\Pr[S_{n+1} \geq a] = \Pr[S_{n+1} \geq a, S_n = a - 1] + \Pr[S_{n+1} \geq a, S_n = a] + \Pr[S_{n+1} > a, S_n = a - 1] + \Pr[S_{n+1} > a, S_n = a]
\]

(106a)

\[
= \Pr[Y_{n+1} = 1 | S_n = a - 1] \Pr[S_n = a - 1] + \Pr[S_n = a] + \Pr[S_{n+1} > a] \geq p \Pr[S_n = a - 1] + \Pr[S_n = a] \quad \text{by assumption (103)}
\]

(106b)

\[
= p \Pr[S_n = a - 1] + \Pr[S_n = a]\)

(106c)

\[
= p \Pr[S_n = a - 1] + (1 - p) \Pr[S_n = a] \quad \text{by induction assumption (106d)}
\]

(106e)

\[
= p \Pr[S_n = a - 1] + (1 - p) \Pr[S_n = a] \quad \text{by induction assumption (106f)}
\]

(106f)

\[
= \Pr[\tilde{S}_{n+1} \geq a, \tilde{Y}_{n+1} = 1] + \Pr[\tilde{S}_{n+1} \geq a, \tilde{Y}_{n+1} = 0] \quad \text{by induction assumption (106g)}
\]

(106g)

\[
= \Pr[\tilde{Y}_{n+1} = 1 | \tilde{S}_n \geq a - 1] \Pr[\tilde{S}_n \geq a - 1] + \Pr[\tilde{Y}_{n+1} = 0 | \tilde{S}_n \geq a] \Pr[\tilde{S}_n \geq a]
\]

(106h)

\[
= \Pr[\tilde{S}_n \geq a - 1, \tilde{Y}_{n+1} = 1] + \Pr[\tilde{S}_n \geq a, \tilde{Y}_{n+1} = 0]
\]

(106i)

\[
= \Pr[\tilde{S}_{n+1} \geq a, \tilde{Y}_{n+1} = 1] + \Pr[\tilde{S}_{n+1} \geq a, \tilde{Y}_{n+1} = 0]
\]

(106j)

\[
= \Pr[\tilde{S}_{n+1} \geq a]
\]

(106k)

Then, by induction, \(\tilde{S}_n\) is stochastically dominated by \(S_n\). For \(m \in \mathbb{N}^*\), we recall that we had \(\tau_m = \inf\{n \in \mathbb{N} | S_n \geq m\}\), similarly we define \(\tau_{m} := \inf\{n \in \mathbb{N} | \tilde{S}_n \geq m\}\). As \(\tilde{S}_n\) is stochastically dominated by \(S_n\), it is easy to see that the stopping time \(\tau_m\) is stochastically dominated by the stopping time \(\tau_{m}\). Indeed, \(\Pr[\tau_m \geq a] = \Pr[S_a < m] = 1 - \Pr[S_a \geq m + 1] \leq 1 - \Pr[\tilde{S}_a \geq m + 1] = \Pr[\tilde{S}_a < m] = \Pr[\tilde{\tau}_m \geq a]\). Finally, the random variable \(\tilde{\tau}_1\) and the random variables \(\tilde{\tau}_{k+1} - \tilde{\tau}_k\), for all \(k \in \mathbb{N}^*\), are independent and identically distributed geometric random variables with probability of success \(p\). Thus, \(B_{m,p} := m \tilde{\tau}_1 + \sum_{k=1}^{m-1} (\tilde{\tau}_{k+1} - \tilde{\tau}_k)\) is a random variable with negative binomial distribution representing the number of trials to obtain \(m\) successes with probability of success \(p\) and \(\tau_m\) is stochastically dominated by \(B_{m,p}\). \(\square\)
References


