MULTICUT DECOMPOSITION METHODS WITH CUT SELECTION FOR MULTISTAGE STOCHASTIC PROGRAMS

Vincent Guigues (corresponding author)
School of Applied Mathematics, FGV
Praia de Botafogo, Rio de Janeiro, Brazil
vguigues@fgv.br

Michelle Bandarra
School of Applied Mathematics, FGV
Praia de Botafogo, Rio de Janeiro, Brazil
michelle.bandarra@mirow.com.br

Abstract. We introduce a variant of Multicut Decomposition Algorithms (MuDA), called CuSMuDA (Cut Selection for Multicut Decomposition Algorithms), for solving multistage stochastic linear programs that incorporates strategies to select the most relevant cuts of the approximate recourse functions. We prove the convergence of the method in a finite number of iterations and use it to solve six portfolio problems with direct transaction costs under return uncertainty and six inventory management problems under demand uncertainty. On all problem instances CuSMuDA is much quicker than MuDA: between 5.1 and 12.6 times quicker for the porfolio problems considered and between 6.4 and 15.7 times quicker for the inventory problems.

Keywords: Stochastic programming; Stochastic Dual Dynamic Programming; Multicut Decomposition Algorithm; Portfolio selection; Inventory management.

AMS subject classifications: 90C15, 91B30.

1. Introduction

Multistage stochastic optimization problems are common in many areas of engineering and in finance. However, solving these problems is challenging and in general requires decomposition techniques. Two popular decomposition methods are Approximate Dynamic Programming (ADP) (Powell 2011) and sampling-based variants of the Nested Decomposition (ND) algorithm (Birge 1985, Birge and Louveaux 1997) and of the Multicut Nested Decomposition (MuND) algorithm (Birge and Louveaux 1988, Gassmann 1990). The introduction of sampling within ND was proposed by (Pereira and Pinto 1991) and the corresponding method is usually called Stochastic Dual Dynamic Programming (SDDP). Several enhancements and extensions of SDDP have been proposed such as CUPPS (Chen and Powell 1999), ReSA (Hindsberger and Philpott 2001), AND (Birge and Donohue 2001), DOASA (Philpott and Guan 2008), risk-averse variants in (Guigues and Römisch 2012, Guigues and Römisch 2012, Philpott and de Matos 2012, Guigues 2013, Shapiro et al. 2013, Shapiro et al. 2013, Kozmik and Morton 2015), cut formulas for nonlinear problems in (Guigues 2016), regularizations in (Asamov and Powell 2015) for linear problems and SDDP-REG in (Guigues et al. 2017) for nonlinear problems. Convergence of these methods was proved in (Philpott and Guan 2008) for linear problems, in (Girardeau et al. 2015) for nonlinear risk-neutral problems, and in (Guigues 2016) for risk-averse nonlinear problems. All these algorithms compute lower approximations of the cost-to-go functions expressed as a supremum of affine functions called optimality cuts. Typically, at each iteration, a fixed (moderate to large) number of cuts is added for each cost-to-go function. Therefore cut selection or cut pruning strategies may be helpful to speed up the convergence. In stochastic optimization, the problem of cut selection for lower approximations of the cost-to-go functions associated to each node of the scenario tree was discussed for the first time in (Ruszczyński 1993) where only the active cuts are selected. Pruning strategies of basis (quadratic) functions have been proposed in (Gaubert et al. 2011) and (McEneaney et al. 2008) for max-plus based approximation methods which, similarly to SDDP, approximate the cost-to-go functions of a nonlinear optimal control problem by a supremum of basis functions. More precisely, in (Gaubert et al. 2011), a fixed number of cuts is eliminated and cut selection is done solving a combinatorial optimization problem. For SDDP, in (Shapiro et al. 2013a) it is suggested at some iterations to eliminate redundant

cuts (a cut is redundant if it is never active in describing the lower approximate cost-to-go function). This procedure is called *test of usefulness* in (Pfeiffer et al. 2012). This requires solving at each stage as many linear programs as there are cuts. In (Pfeiffer et al. 2012) and (Philpott et al. 2012), only the cuts that have the largest value for at least one of the trial points computed are considered relevant. This strategy is called the *Territory algorithm* in (Pfeiffer et al. 2012) and Level 1 cut selection in (Philpott et al. 2012). It was presented for the first time in 2007 at the ROADEF congress by David Game and Guillaume Le Roy (GDF-Suez), see (Pfeiffer et al. 2012).

Sampling can also be incorporated into MuND algorithm. It was used for instance recently in (Zhang et al. 2016) to solve a water allocation problem under uncertainty. This algorithm builds many more cuts than SDDP per iteration (more than 10 times more in typical implementations) and therefore each iteration takes more time but much less iterations are needed to satisfy some stopping criterion. Therefore cut selection strategies could also be useful. However, to the best of our knowledge, the combination of Multicut decomposition methods with cut selection strategies, refereed to as CuSMuDA (Cut Selection for Multicut Decomposition Algorithms) in the sequel, has not been proposed so far. In this context, the objective of this paper is to study CuSMuDA and to propose cut selection strategies more efficient than the aforementioned ones. More precisely, instead of selecting all the cuts that are the highest at the trial points, we introduce a set of selectors that select some subset of these cuts. The selectors have to satisfy an assumption (Assumption (H3), see Section 3.2) to ensure the convergence of CuSMuDA in a finite number of iterations. We obtain a family of cut selection strategies; a given strategy corresponding to a choice of selectors along the iterations. In this family, the most economic (in terms of memory) cut selection strategy satisfying (H3), called Multicut Limited Memory Level 1, selects at each trial point only one cut, namely the oldest cut. This strategy is the Multicut variant of the Limited Memory Level 1 cut selection proposed in (Guigues 2017). The least economic strategy, i.e., the one that keeps the largest amount of cuts, is the Multicut variant of Level 1. "Between" these two strategies, using the flexibility offered by the selectors (as long as Assumption (H3) is satisfied by these selectors), we obtain a (large) family of cut selection strategies. We also prove the convergence of CuSMuDA, that uses cut selection strategies from this family, in a finite number of iterations. This proof extends the theory in (Guigues 2017) in two aspects: (i) first the stochastic case is considered, i.e., SDDP and multicut SDDP are considered whereas the deterministic case, i.e., DDP, was considered in (Guigues 2017) and (ii) second more general cut selection strategies are dealt with. Item (ii) requires an additional technical discussion to show that we do not cycle, see Lemma 1.

Numerical experiments on six instances of a portfolio problem and six instances of an inventory problem show that the corresponding variant of CuSMuDA is much quicker than sampling-based MuND combined with Multicut Level 1 (between 10.2 and 25.6 times quicker) and much quicker than MuND with sampling in the forward pass (between 5.1 and 15.7 times quicker).

The outline of the study is as follows. The class of problems considered and assumptions are discussed in Section 2. In Subection 3.1 we recall sampling-based MuND while in Subsection 3.2 CuSMuDA is described. Section 4 proves the convergence of CuSMuDA in a finite number of iterations. Finally, numerical experiments are presented in Section 5.

2. Problem formulation and assumptions

We are interested in solution methods for linear stochastic dynamic programming equations: the first stage problem is

$$\mathcal{Q}_1(x_0) = \left\{ \begin{array}{l} \inf_{x_1} c_1^T x_1 + \mathcal{Q}_2(x_1) \\ A_1 x_1 + B_1 x_0 = b_1, x_1 \geq 0 \end{array} \right. \tag{1}$$

for x_0 given and for $t = 2, \ldots, T$, $\mathcal{Q}_t(x_{t-1}) = \mathbb{E}_{\xi_t}[\mathfrak{Q}_t(x_{t-1}, \xi_t)]$ with

$$\mathfrak{Q}_t(x_{t-1}, \xi_t) = \left\{ \begin{array}{l} \inf_{x_t \in \mathbb{R}^n} c_t^T x_t + \mathcal{Q}_{t+1}(x_t) \\ A_t x_t + B_t x_{t-1} = b_t, x_t \geq 0, \end{array} \right. \tag{2}$$

with the convention that $\mathcal{Q}_{T+1}$ is null and where for $t = 2, \ldots, T$, random vector ξ_t corresponds to the concatenation of the elements in random matrices A_t, B_t which have a known finite number of rows and random vectors b_t, c_t (it is assumed that ξ_1 is not random). For convenience, we will denote

$$X_t(x_{t-1}, \xi_t) := \{x_t \in \mathbb{R}^n : A_t x_t + B_t x_{t-1} = b_t,\ x_t \geq 0\}.$$

We make the following assumptions:

(H1) The random vectors $\xi_2, \ldots, \xi_T$ are independent and have discrete distributions with finite support.

(H2) The set $X_1(x_0, \xi_1)$ is nonempty and bounded and for every $x_1 \in X_1(x_0, \xi_1)$, for every $t = 2, \ldots, T$, for every realization $\tilde{\xi}_2, \ldots, \tilde{\xi}_t$ of $\xi_2, \ldots, \xi_t$, for every $x_\tau \in X_\tau(x_{\tau-1}, \tilde{\xi}_\tau), \tau = 2, \ldots, t-1$, the set $X_t(x_{t-1}, \tilde{\xi}_t)$ is nonempty and bounded.

We will denote by $\Theta_t = \{\xi_{t1}, \ldots, \xi_{tM_t}\}$ the support of ξ_t for stage t with $p_{ti} = \mathbb{P}(\xi_t = \xi_{ti}) > 0, i = 1, \ldots, M_t$ and with vector ξ_{tj} being the concatenation of the elements in $A_{tj}, B_{tj}, b_{tj}, c_{tj}$.

3. Algorithms

3.1. **Multicut stochastic decomposition.** The multicut decomposition method approximates function $\mathfrak{Q}_t(\cdot, \xi_{tj})$ at iteration k for $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, by a piecewise affine lower bounding function $\mathfrak{Q}_t^k(\cdot, \xi_{tj})$ which is a maximum of affine functions $\mathcal{C}_{tj}^i$ called cuts:

$$\mathfrak{Q}_t^k(x_{t-1}, \xi_{tj}) = \max_{1 \le i \le k} \mathcal{C}_{tj}^i(x_{t-1}) \text{ with } \mathcal{C}_{tj}^i(x_{t-1}) = \theta_{tj}^i + \langle \beta_{tj}^i, x_{t-1} \rangle$$

where coefficients $\theta_{tj}^i, \beta_{tj}^i$ are computed as explained below and where the usual scalar product in $\mathbb{R}^n$ is denoted by $\langle x, y \rangle = x^T y$ for $x, y \in \mathbb{R}^n$. These approximations provide the lower bounding functions

$$\mathcal{Q}_t^k(x_{t-1}) = \sum_{j=1}^{M_t} p_{tj} \mathfrak{Q}_t^k(x_{t-1}, \xi_{tj}) \tag{3}$$

of $\mathcal{Q}_t$. Since $\mathcal{Q}_{T+1}$ is the null function, we will also define $\mathcal{Q}_{T+1}^k \equiv 0$. The steps in MuDA are described below.

Step 1: Initialization. For $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, take $\mathfrak{Q}_t^0(\cdot, \xi_{tj}) \equiv -\infty$ as initial approximations (if known lower bounding affine functions for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ are available, they can be taken for $\mathfrak{Q}_t^0(\cdot, \xi_{tj})$ modifying the algorithm correspondingly). Set the iteration count k to 1 and $\mathcal{Q}_{T+1}^0 \equiv 0$.

Step 2: Forward pass. We generate a sample $\tilde{\xi}^k = (\tilde{\xi}_1^k, \tilde{\xi}_2^k, \ldots, \tilde{\xi}_T^k)$ from the distribution of $(\xi_1, \xi_2, \ldots, \xi_T)$, with the convention that $\tilde{\xi}_1^k = \xi_1$ (here and in what follows, the tilde symbol will be used to represent realizations of random variables: for random variable ξ, $\tilde{\xi}$ is a realization of ξ). Using approximation $\mathfrak{Q}_t^{k-1}(\cdot, \xi_{tj})$ of $\mathfrak{Q}_t(\cdot, \xi_{tj})$ (computed at previous iterations), we solve the problem

$$\left\{ \begin{array}{l} \inf_{x_t \in \mathbb{R}^n} x_t^T \tilde{c}_t^k + \mathcal{Q}_{t+1}^{k-1}(x_t) \\ x_t \in X_t(x_{t-1}^k, \tilde{\xi}_t^k) \end{array} \right. \tag{4}$$

for $t = 1, \ldots, T$, where $x_0^k = x_0$ and $\mathcal{Q}_{t+1}^{k-1}$ is given by (3) with k replaced by $k-1$. Let x_t^k be an optimal solution of the problem.

Step 3: Backward pass. The backward pass builds cuts for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ at x_{t-1}^k computed in the forward pass. For $k \ge 1$ and $t = 1, \ldots, T$, we introduce the function $\underline{\mathfrak{Q}}_t^k : \mathbb{R}^n \times \Theta_t \to \mathbb{R}$ given by

$$\underline{\mathfrak{Q}}_t^k(x_{t-1}, \xi_t) = \left\{ \begin{array}{l} \inf_{x_t \in \mathbb{R}^n} c_t^T x_t + \mathcal{Q}_{t+1}^k(x_t) \\ x_t \in X_t(x_{t-1}, \xi_t), \end{array} \right. \tag{5}$$

with the convention that $\Theta_1 = \{\xi_1\}$, and we set $\mathcal{Q}_{T+1}^k \equiv 0$. For $j = 1, \ldots, M_T$, we solve the problem

$$\mathfrak{Q}_T(x_{T-1}^k, \xi_{Tj}) = \left\{ \begin{array}{l} \inf\limits_{x_T \in \mathbb{R}^n} c_{Tj}^T x_T \\ A_{Tj} x_T + B_{Tj} x_{T-1}^k = b_{Tj}, x_T \ge 0, \end{array} \right. \quad \text{with dual} \left\{ \begin{array}{l} \sup_\lambda \lambda^T (b_{Tj} - B_{Tj} x_{T-1}^k) \\ A_{Tj}^T \lambda \le c_{Tj}. \end{array} \right. \tag{6}$$

Let λ_{Tj}^k be an optimal solution of the dual problem above. We get

$$\mathfrak{Q}_T(x_{T-1}, \xi_{Tj}) \ge \langle \lambda_{Tj}^k, b_{Tj} - B_{Tj} x_{T-1} \rangle$$

and compute $\theta^k_{Tj} = b^T_{Tj}\lambda^k_{Tj}$ and $\beta^k_{Tj} = -B^T_{Tj}\lambda^k_{Tj}$. Then for $t = T-1$ down to $t = 2$, knowing $\mathcal{Q}^k_{t+1} \leq \mathcal{Q}_{t+1}$, we solve the problem below for $j = 1, \ldots, M_t$,

$$(7) \quad \underline{\mathfrak{Q}}^k_t(x^k_{t-1}, \xi_{tj}) = \left\{ \begin{array}{l} \inf\limits_{x_t} c^T_{tj}x_t + \mathcal{Q}^k_{t+1}(x_t) \\ x_t \in X_t(x^k_{t-1}, \xi_{tj}) \end{array} \right. = \left\{ \begin{array}{l} \inf\limits_{x_t, f} c^T_{tj}x_t + \sum\limits_{\ell=1}^{M_{t+1}} p_{t+1\ell} f_\ell \\ A_{tj}x_t + B_{tj}x^k_{t-1} = b_{tj}, x_t \geq 0, \\ f_\ell \geq \theta^i_{t+1\ell} + \langle \beta^i_{t+1\ell}, x_t \rangle, i = 1, \ldots, k, \ell = 1, \ldots, M_{t+1}. \end{array} \right.$$

Observe that due to (H2) the above problem is feasible and has a finite optimal value. Therefore $\underline{\mathfrak{Q}}^k_t(x^k_{t-1}, \xi_{tj})$ can be expressed as the optimal value of the corresponding dual problem

$$(8) \qquad \underline{\mathfrak{Q}}^k_t(x^k_{t-1}, \xi_{tj}) = \left\{ \begin{array}{l} \sup\limits_{\lambda, \mu} \lambda^T(b_{tj} - B_{tj}x^k_{t-1}) + \sum\limits_{i=1}^{k} \sum\limits_{\ell=1}^{M_{t+1}} \mu_{i\ell}\theta^i_{t+1\ell} \\ A^T_{tj}\lambda + \sum_{i=1}^k \sum_{\ell=1}^{M_{t+1}} \mu_{i\ell}\beta^i_{t+1\ell} \leq c_{tj}, \\ p_{t+1\ell} = \sum_{i=1}^k \mu_{i\ell}, \ \ell = 1, \ldots, M_{t+1}, \\ \mu_{i\ell} \geq 0, \ i = 1, \ldots, k, \ell = 1, \ldots, M_{t+1}. \end{array} \right.$$

Let $(\lambda^k_{tj}, \mu^k_{tj})$ an optimal solution of dual problem (8). Using the fact that $\mathcal{Q}^k_{t+1} \leq \mathcal{Q}_{t+1}$, we get

$$\mathfrak{Q}_t(x_{t-1}, \xi_{tj}) \geq \underline{\mathfrak{Q}}^k_t(x_{t-1}, \xi_{tj}) \geq \langle \lambda^k_{tj}, b_{tj} - B_{tj}x_{t-1} \rangle + \langle \mu^k_{tj}, \theta^k_{t+1} \rangle$$

and we compute

$$\theta^k_{tj} = \langle \lambda^k_{tj}, b_{tj} \rangle + \langle \mu^k_{tj}, \theta^k_{t+1} \rangle \text{ and } \beta^k_{tj} = -B^T_{tj}\lambda^k_{tj}.$$

In these expressions, vector θ^k_{t+1} has components $\theta^i_{t+1\ell}, \ell = 1, \ldots, M_{t+1}, i = 1, \ldots, k$, arranged in the same order as components $\mu^k_{tj}(\ell, i)$ of μ^k_{tj}.

Step 4: Do $k \leftarrow k+1$ and go to Step 2.

3.2. **Multicut stochastic decomposition with cut selection.** We now describe a variant of MuDA that stores all cut coefficients $\theta^i_{tj}, \beta^i_{tj}$, and trial points x^i_{t-1}, but that uses a reduced set of cuts $\mathcal{C}^i_{tj}$ to approximate functions $\mathfrak{Q}_t(\cdot, \xi_{tj})$ when solving problem (4) in the forward pass and (7) in the backward pass. Let $\mathcal{S}^k_{tj}$ be the set of indices of the cuts selected at the end of iteration k to approximate $\mathfrak{Q}_t(\cdot, \xi_{tj})$. In the end of the backward pass of iteration k, the variant of MuDA with cut selection computes approximations $\mathcal{Q}^k_t$ of $\mathcal{Q}_t$ given by (3) now with $\mathfrak{Q}^k_t(\cdot, \xi_{tj})$ given by

$$(9) \qquad \mathfrak{Q}^k_t(x_{t-1}, \xi_{tj}) = \max_{\ell \in \mathcal{S}^k_{tj}} \mathcal{C}^\ell_{tj}(x_{t-1}),$$

where the set $\mathcal{S}^k_{tj}$ is a subset of the indices of the cuts that have the largest value for at least one of the trial points computed so far. More precisely, sets $\mathcal{S}^k_{tj}$ are initialized taking $\mathcal{S}^0_{tj} = \{0\}$ and $\mathcal{S}^1_{tj} = \{1\}$. For $t \in \{2, \ldots, T\}$, sets $\mathcal{S}^k_{tj}$ are computed as follows. For $i = 1, \ldots, k$, $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, let I^{ik}_{tj} be the set of cuts for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ computed at iteration k or before that have the largest value at x^i_{t-1}:

$$(10) \qquad I^{ik}_{tj} = \operatorname*{arg\,max}_{\ell=1,\ldots,k} \mathcal{C}^\ell_{tj}(x^i_{t-1}),$$

where the cut indices in I^{ik}_{tj} are sorted in ascending order. With a slight abuse of notation, we will denote the ℓ-th smallest element in I^{ik}_{tj} by $I^{ik}_{tj}(\ell)$. For instance, if $I^{ik}_{tj} = \{2, 30, 50\}$ then $I^{ik}_{tj}(1) = 2, I^{ik}_{tj}(2) = 30$, and $I^{ik}_{tj}(3) = 50$. A cut selection strategy is given by a set of selectors $\mathcal{S}_{tj}(m), m = 1, 2, \ldots,$ where $\mathcal{S}_{tj}(m)$ is a subset of $\{1, 2, \ldots, m\}$ giving the indices of the cuts to select in I^{ik}_{tj}, through the relation

$$\mathcal{S}^k_{tj} = \bigcup_{i=1}^{k} \left\{ I^{ik}_{tj}(\ell) : \ell \in \mathcal{S}_{tj}(|I^{ik}_{tj}|) \right\},$$

where $|I^{ik}_{tj}|$ is the cardinality of set I^{ik}_{tj}. We require the selectors to satisfy the following assumption:

(H3) for $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, for every $m \geq 1$, $\mathcal{S}_{tj}(m) \subseteq \mathcal{S}_{tj}(m+1)$.

Example 3.1 (Multicut Level 1 and Territory Algorithm)**.** *The strategy $\mathcal{S}_{tj}(m) = \{1, 2, \ldots, m\}$ selects all cuts that have the highest value for at least one trial point. In the context of SDDP, this strategy was called Level 1 in (Philpott et al. 2012) and Territory Algorithm in (Pfeiffer et al. 2012). For this strategy, we have $\mathcal{S}^k_{Tj} = \{1, \ldots, k\}$ for all j and $k \geq 1$, meaning that no cut selection is needed for the last stage T. This comes from the fact that for all $k \geq 2$ and $1 \leq k_1 \leq k$, cut $\mathcal{C}^{k_1}_{Tj}$ is selected because it is one of the cuts with the highest value at $x^{k_1}_{T-1}$. Indeed, for any $1 \leq k_2 \leq k$ with $k_2 \neq k_1$, since $\lambda^{k_2}_{Tj}$ is feasible for problem (6) with x^k_{T-1} replaced by $x^{k_1}_{T-1}$, we get*

$$\mathcal{C}^{k_1}_{Tj}(x^{k_1}_{T-1}) = \mathfrak{Q}_T(x^{k_1}_{T-1}, \xi_{Tj}) \geq \langle \lambda^{k_2}_{Tj}, b_{Tj} - B_{Tj} x^{k_1}_{T-1} \rangle = \mathcal{C}^{k_2}_{Tj}(x^{k_1}_{T-1}).$$

Example 3.2 (Multicut Limited Memory Level 1)**.** *The strategy that eliminates the largest amount of cuts, called Multicut Limited Memory Level 1 (MLM Level 1 for short), consists in taking a singleton for every set $\mathcal{S}_{tj}(m)$. For (H3) to be satisfied, this implies $\mathcal{S}_{tj}(m) = \{1\}$. This choice corresponds to the Limited Memory Level 1 cut selection introduced in (Guigues 2017) in the context of DDP. For that particular choice, at a given point, among the cuts that have the highest value only the oldest (i.e., the cut that was first computed among the cuts that have the highest value at that point) is selected.*

The computation of $\mathcal{S}^k_{tj}$, i.e., of the cut indices to select at iteration k, is performed in the backward pass (immediately after computing cut $\mathcal{C}^k_{tj}$) using the pseudo-code given in the left and right panels of Figure 1 for the Multicut Level 1 and MLM Level 1 cut selection strategies respectively.

In this pseudo-code, we use the notation I^i_{tj} in place of I^{ik}_{tj}. We also store in variable m^i_{tj} the current value of the highest cut for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ at x^i_{t-1}. At the end of the first iteration, we initialize $m^1_{tj} = \mathcal{C}^1_{tj}(x^1_{t-1})$. After cut $\mathcal{C}^k_{tj}$ is computed at iteration $k \geq 2$, these variables are updated using the relations

$$\begin{cases} m^i_{tj} & \leftarrow & \max(m^i_{tj}, \mathcal{C}^k_{tj}(x^i_{t-1})), \ i = 1, \ldots, k-1, \\ m^k_{tj} & \leftarrow & \max(\mathcal{C}^\ell_{tj}(x^k_{t-1}), \ell = 1, \ldots, k). \end{cases}$$

We also use an array of Boolean called `Selected` using the information given by variables I^i_{tj} whose ℓ-th entry is `True` if cut ℓ is selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ and `False` otherwise. This allows us to avoid copies of cut indices that may appear in $I^{i_1 k}_{tj}$ and $I^{i_2 k}_{tj}$ with $i_1 \neq i_2$.

4. Convergence analysis

Theorem 1 proves that CuSMuDA converges in a finite number of iterations. We will make the following assumption:

(H4) The samples in the forward passes are independent: $(\tilde{\xi}^k_2, \ldots, \tilde{\xi}^k_T)$ is a realization of $\xi^k = (\xi^k_2, \ldots, \xi^k_T) \sim (\xi_2, \ldots, \xi_T)$ and $\xi^1, \xi^2, \ldots,$ are independent.

The convergence proof is based on the following lemma:

Lemma 1. *Assume that all subproblems in the forward and backward passes of CuSMuDA are solved using an algorithm that necessarily outputs an extreme point of the feasible set (for instance the simplex algorithm). Let assumptions (H1), (H2), (H3), and (H4) hold. Then almost surely, there exists $k_0 \geq 1$ such that for every $k \geq k_0$, $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, we have*

$$\mathfrak{Q}^k_t(\cdot, \xi_{tj}) = \mathfrak{Q}^{k_0}_t(\cdot, \xi_{tj}) \text{ and } \mathcal{Q}^k_t = \mathcal{Q}^{k_0}_t. \tag{11}$$

Proof. Let Ω_1 be the event on the sample space Ω of sequences of forward scenarios such that every scenario is sampled an infinite number of times. By Assumption (H4), this event Ω_1 has probability one.

Consider a realization $\omega \in \Omega$ of CuSMuDA in Ω_1 corresponding to realizations $(\tilde{\xi}^k_{1:T})_k$ of $(\xi^k_{1:T})_k$ in the forward pass. To simplify, we will drop ω in the notation. For instance, we will simply write $x^k_t, \mathcal{Q}^k_t$ for realizations $x^k_t(\omega)$ and $\mathcal{Q}^k_t(\cdot)(\omega)$ of $x^k_t, \mathcal{Q}^k_t$ given realization $\omega \in \Omega$ of CuSMuDA.

We show by induction on t that the number of different cuts computed by the algorithm is finite and that after some iteration k_t the same cuts are selected for functions $\mathfrak{Q}_t(\cdot, \xi_{tj})$. Our induction hypothesis $\mathcal{H}(t)$ for $t \in \{2, \ldots, T\}$ is that the sets $\{(\theta^k_{tj}, \beta^k_{tj}) : k \in \mathbb{N}\}, j = 1, \ldots, M_t$, are finite and there exists some finite k_t such that for every $k > k_t$ we have

$$\{(\theta^\ell_{tj}, \beta^\ell_{tj}) : \ell \in S^k_{tj}\} = \{(\theta^\ell_{tj}, \beta^\ell_{tj}) : \ell \in S^{k_t}_{tj}\} \text{ and } x^k_{t-1} = x^{k_t}_{t-1}, \tag{12}$$

Multicut Level 1	MLM Level 1
$I_{tj}^k = \{k\}$, $m_{tj}^k = \mathcal{C}_{tj}^k(x_{t-1}^k)$.	$I_{tj}^k = \{1\}$, $m_{tj}^k = \mathcal{C}_{tj}^1(x_{t-1}^k)$.
For $\ell = 1, \dots, k-1$,	**For** $\ell = 1, \dots, k-1$,
If $\mathcal{C}_{tj}^k(x_{t-1}^\ell) > m_{tj}^\ell$	**If** $\mathcal{C}_{tj}^k(x_{t-1}^\ell) > m_{tj}^\ell$
$I_{tj}^\ell = \{k\}$, $m_{tj}^\ell = \mathcal{C}_{tj}^k(x_{t-1}^\ell)$	$I_{tj}^\ell = \{k\}$, $m_{tj}^\ell = \mathcal{C}_{tj}^k(x_{t-1}^\ell)$
Else if $\mathcal{C}_{tj}^k(x_{t-1}^\ell) = m_{tj}^\ell$	**End If**
$I_{tj}^\ell = I_{tj}^\ell \cup \{k\}$	
End If	
If $\mathcal{C}_{tj}^\ell(x_{t-1}^k) > m_{tj}^k$	**If** $\mathcal{C}_{tj}^{\ell+1}(x_{t-1}^k) > m_{tj}^k$
$I_{tj}^k = \{\ell\}$, $m_{tj}^k = \mathcal{C}_{tj}^\ell(x_{t-1}^k)$	$I_{tj}^k = \{\ell+1\}$, $m_{tj}^k = \mathcal{C}_{tj}^{\ell+1}(x_{t-1}^k)$
Else if $\mathcal{C}_{tj}^\ell(x_{t-1}^k) = m_{tj}^k$	**End If**
$I_{tj}^k = I_{tj}^k \cup \{\ell\}$	
End If	
End For	**End For**
For $\ell = 1, \dots, k$,	**For** $\ell = 1, \dots, k$,
`Selected`$[\ell]$=`False`	`Selected`$[\ell]$=`False`
End For	**End For**
For $\ell = 1, \dots, k$	**For** $\ell = 1, \dots, k$
For $m = 1, \dots, \lvert I_{tj}^\ell \rvert$	**For** $m = 1, \dots, \lvert I_{tj}^\ell \rvert$
`Selected`$[I_{tj}^\ell[m]]$=`True`	`Selected`$[I_{tj}^\ell[m]]$=`True`
End For	**End For**
End For	**End For**
$\mathcal{S}_{tj}^k = \emptyset$	$\mathcal{S}_{tj}^k = \emptyset$
For $\ell = 1, \dots, k$	**For** $\ell = 1, \dots, k$
If `Selected`$[\ell]$=`True`	**If** `Selected`$[\ell]$=`True`
$\mathcal{S}_{tj}^k = \mathcal{S}_{tj}^k \cup \{\ell\}$	$\mathcal{S}_{tj}^k = \mathcal{S}_{tj}^k \cup \{\ell\}$
End If	**End If**
End For	**End For**

FIGURE 1. Pseudo-codes for the computation of set $\mathcal{S}_{tj}^k$ for fixed $t \in \{2, \dots, T\}, k \geq 2, j = 1, \dots, M_t$, and two cut selection strategies.

for every $j = 1, \dots, M_t$. We will denote by $\mathcal{I}_{tj}^{ik}$ the set $\{I_{tj}^{ik}(\ell) : \ell \in \mathcal{S}_{tj}(|I_{tj}^{ik}|)\}$. We first show, in items **a** and **b** below that $\mathcal{H}(T)$ holds.

a. Observe that λ_{Tj}^k defined in the backward pass of CuSMuDA is an extreme point of the polyhedron $\{\lambda : A_{Tj}^\top \lambda \leq c_{Tj}\}$. This polyhedron is a finite intersection of closed half spaces in finite dimension (since A_{Tj} has a finite number of rows and columns) and therefore has a finite number of extreme points. It follows that λ_{Tj}^k can only take a finite number of values, same as $(\theta_{Tj}^k, \beta_{Tj}^k) = (\langle \lambda_{Tj}^k, b_{Tj} \rangle, -B_{Tj}^\top \lambda_{Tj}^k)$, and there exists $\bar{k}_T$ such that for every $k > \bar{k}_T$ and every j, each cut $\mathcal{C}_{Tj}^k$ is a copy of a cut $\mathcal{C}_{Tj}^{k'}$ with $1 \leq k' \leq \bar{k}_T$ (no new cut is computed for functions $\mathfrak{Q}_T(\cdot, \xi_{Tj})$ for $k > \bar{k}_T$).

Now recall that x_{T-1}^k computed in the forward pass is a solution of (4) with $t = T-1$ and this optimization problem can be written as a linear problem adding variables $f_1, f_2, \dots, f_{M_T}$ replacing in the objective $\mathcal{Q}_T^{k-1}(x_{T-1})$ by $\sum_{\ell=1}^{M_T} p_{T\ell} f_\ell$ and adding the linear constraints $f_\ell \geq \theta_{T\ell}^i + \langle \beta_{T\ell}^i, x_{T-1} \rangle$, $i = 1, \dots, k-1$, $\ell = 1, \dots, M_T$. On top of that, for iterations $k > \bar{k}_T$, since functions $\mathfrak{Q}_T^k(\cdot, \xi_{Tj})$ are made of a collection of cuts taken from the finite and fixed set of cuts $\mathcal{C}_{Tj}^\ell, \ell \leq \bar{k}_T$, the set of possible functions $(\mathfrak{Q}_T^k(\cdot, \xi_{Tj}))_{k \geq 1}$ and therefore of possible functions $(\mathcal{Q}_T^k)_{k \geq 1}$ is finite. It follows that there is a finite set of possible polyhedrons for the feasible set of (4) (with $t = T-1$) rewritten as a linear program as we have just explained, adding variables $f_1, f_2, \dots, f_{M_T}$. Since these polyhedrons have a finite number of extreme points (recall that there is a finite number of different linear constraints), there is only a finite number of possible trial points $(x_{T-1}^k)_{k \geq 1}$. Therefore we can assume without loss of generality that $\bar{k}_T$ is such that for iterations $k > \bar{k}_T$, trial point x_{T-1}^k is also a copy of a trial point $x_{T-1}^{k'}$ with $k' \leq \bar{k}_T$.

b. We show that for every $i \geq 1$ and $j = 1, \ldots, M_T$, there exists $1 \leq i' \leq \bar{k}_T$ and $k_{Tj}(i') \geq \bar{k}_T$ such that for every $k \geq \max(i, k_{Tj}(i'))$ we have

$$\left\{(\theta^{\ell}_{Tj}, \beta^{\ell}_{Tj}) : \ell \in \mathcal{I}^{ik}_{Tj}\right\} = \left\{(\theta^{\ell}_{Tj}, \beta^{\ell}_{Tj}) : \ell \in \mathcal{I}^{i'k_{Tj}(i')}_{Tj}\right\}, \tag{13}$$

which will show $\mathcal{H}(T)$ with $k_T = \max_{1 \leq j \leq M_T, 1 \leq i' \leq \bar{k}_T} k_{Tj}(i')$. Let us show that (13) indeed holds. Let us take $i \geq 1$ and $j \in \{1, \ldots, M_T\}$. If $1 \leq i \leq \bar{k}_T$ define $i' = i$. Otherwise due to **a.** we can find $1 \leq i' \leq \bar{k}_T$ such that $x^i_{T-1} = x^{i'}_{T-1}$ which implies $I^{ik}_{Tj} = I^{i'k}_{Tj}$ for every $k \geq i$. Now consider the sequence of sets $(I^{i'k}_{Tj})_{k \geq \bar{k}_T}$. Due to the definition of $\bar{k}_T$, the sequence $(|I^{i'k}_{Tj}|)_{k > \bar{k}_T}$ is nondecreasing and therefore two cases can happen:

(A) there exists $k_{Tj}(i')$ such that for $k \geq k_{Tj}(i')$ we have $I^{i'k}_{Tj} = I^{i'k_{Tj}(i')}_{Tj}$ (the cuts computed after iteration $k_{Tj}(i')$ are not active at $x^{i'}_{T-1}$). In this case $\mathcal{I}^{i'k}_{Tj} = \mathcal{I}^{i'k_{Tj}(i')}_{Tj}$ for $k \geq k_{Tj}(i')$, $\mathcal{I}^{ik}_{Tj} = \mathcal{I}^{i'k}_{Tj} = \mathcal{I}^{i'k_{Tj}(i')}_{Tj}$ for $k \geq \max(k_{Tj}(i'), i)$ and (13) holds.

(B) The sequence $(|I^{i'k}_{Tj}|)_{k \geq \bar{k}_T}$ is unbounded. Due to Assumption (H3), the sequence $(|\mathcal{S}_{Tj}(m)|)_m$ is nondecreasing. If there exists $k_{Tj} > \bar{k}_T$ such that $\mathcal{S}_{Tj}(k) = \mathcal{S}_{Tj}(k_{Tj})$ for $k \geq k_{Tj}$ then if $k_{Tj}(i')$ is the smallest k such that $|I^{i'k}_{Tj}| \geq k_{Tj}$ then for every $k \geq k_{Tj}(i')$ we have $\mathcal{I}^{i'k}_{Tj} = \mathcal{I}^{i'k_{Tj}(i')}_{Tj}$ and for $k \geq \max(k_{Tj}(i'), i)$ we deduce $\mathcal{I}^{ik}_{Tj} = \mathcal{I}^{i'k_{Tj}(i')}_{Tj}$ and (13) holds. Otherwise the sequence $(|\mathcal{S}_{Tj}(m)|)_m$ is unbounded and an infinite number of cut indices are selected from the sets $(I^{i'k}_{Tj})_{k \geq i'}$ to make up sets $(\mathcal{I}^{i'k}_{Tj})_{k \geq i'}$. However, since there is a finite number of different cuts, there is only a finite number of iterations where a new cut can be selected from $I^{i'k}_{Tj}$ and therefore there exists $k_{Tj}(i')$ such that (13) holds for $k \geq \max(k_{Tj}(i'), i)$.

c. Now assume that $\mathcal{H}(t+1)$ holds for some $t \in \{2, \ldots, T-1\}$. We want to show $\mathcal{H}(t)$. Consider the set $\mathcal{D}_{tjk}$ of points of form

$$(\langle \lambda, b_{tj} \rangle + \sum_{\ell=1}^{M_{t+1}} \sum_{i \in S^k_{t+1\ell}} \mu_{i\ell} \theta^i_{t+1\ell}, -B^\top_{tj} \lambda) \tag{14}$$

where (λ, μ) is an extreme point of the set $\mathcal{P}_{tjk}$ of points (λ, μ) satisfying

$$\mu \geq 0, p_{t+1\ell} = \textstyle\sum_{i \in S^k_{t+1\ell}} \mu_{i\ell},\ \ell = 1, \ldots, M_{t+1},\ A^\top_{tj}\lambda + \sum_{\ell=1}^{M_{t+1}} \sum_{i \in S^k_{t+1\ell}} \mu_{i\ell} \beta^i_{t+1\ell} \leq c_{tj}. \tag{15}$$

We claim that for every $k > k_{t+1}$, every point from $\mathcal{D}_{tjk}$ can be written as a point from $\mathcal{D}_{tjk_{t+1}}$, i.e., a point of form (14) with k replaced by k_{t+1} and (λ, μ) an extreme point of the set $\mathcal{P}_{tjk_{t+1}}$. Indeed, take a point from $\mathcal{D}_{tjk}$, i.e., a point of form (14) with (λ, μ) an extreme point of $\mathcal{P}_{tjk}$ and $k > k_{t+1}$. It can be written as a point from $\mathcal{D}_{tjk_{t+1}}$, i.e., a point of form (14) with k replaced by k_{t+1} and $(\lambda, \hat{\mu})$ in the place of (λ, μ) where $(\lambda, \hat{\mu})$ is an extreme point of $\mathcal{P}_{tjk_{t+1}}$ obtained replacing the basic columns $\beta^i_{t+1\ell}$ with $i \in S^k_{t+1\ell}, i \notin S^{k_{t+1}}_{t+1\ell}$ associated with μ by columns $\beta^{i'}_{t+1\ell}$ with $i' \in S^{k_{t+1}}_{t+1\ell}$ such that $(\theta^i_{t+1\ell}, \beta^i_{t+1\ell}) = (\theta^{i'}_{t+1\ell}, \beta^{i'}_{t+1\ell})$ (this is possible due to $\mathcal{H}(t+1)$).

Since $\mathcal{P}_{tjk_{t+1}}$ has a finite number of extreme points, the set $\mathcal{D}_{tjk}$ has a finite cardinality and recalling that for CuSMuDA $(\theta^k_{tj}, \beta^k_{tj}) \in \mathcal{D}_{tjk}$, the cut coefficients $(\theta^k_{tj}, \beta^k_{tj})$ can only take a finite number of values. This shows the first part of $\mathcal{H}(t)$. Therefore, there exists $\bar{k}_t$ such that for every $k > \bar{k}_t$ and every j, each cut $\mathcal{C}^k_{tj}$ is a copy of a cut $\mathcal{C}^{k'}_{tj}$ with $1 \leq k' \leq \bar{k}_t$ (no new cut is computed for functions $\mathfrak{Q}_t(\cdot, \xi_{tj})$ for $k > \bar{k}_t$). As for the induction step $t = T$, this clearly implies that after some iteration, no new trial points are computed and therefore we can assume without loss of generality that $\bar{k}_t$ is such that for $k > \bar{k}_T$ trial point x^k_{t-1} is a copy of some x^ℓ_{t-1} with $1 \leq \ell \leq \bar{k}_t$.

Finally, we can show (12) proceeding as in **b.**, replacing T by t. This achieves the proof of $\mathcal{H}(t)$.

Gathering our observations, we have shown that (11) holds with $k_0 = \max_{t=2,\ldots,T} k_t$. □ □

Remark 4.1. *For Multicut Level 1 and MLM Level 1 cut selection strategies corresponding to selectors $\mathcal{S}_{tj}$ satisfying respectively $\mathcal{S}_{tj}(m) = \{1, \ldots, m\}$ and $\mathcal{S}_{tj}(m) = \{1\}$, integers $k_0, \tilde{k}_t$ defined in Lemma 1 and its proof satisfy $k_0 \leq \max_{t=2,\ldots,T} \tilde{k}_t$. For other selectors $\mathcal{S}_{tj}$ this relation is not necessarily satisfied (see **b.**-(B) of the proof of the lemma).*

Theorem 1. *Let Assumptions (H1), (H2), (H3), and (H4) hold. Then Algorithm CuSMuDA converges with probability one in a finite number of iterations to a policy which is an optimal solution of* (1)-(2).

Proof. Let Ω_1 be defined as in the proof of Lemma 1 and let Ω_2 be the event such that k_0 defined in Lemma 1 is finite. Note that $\Omega_1 \cap \Omega_2$ has probability 1. Consider a realization of CuSMuDA in $\Omega_1 \cap \Omega_2$ corresponding to realizations $(\tilde{\xi}^k_{1:T})_k$ of $(\xi^k_{1:T})_k$ in the forward pass and let $(x^*_1, x^*_2(\cdot), \ldots, x^*_T(\cdot))$ be the policy obtained from iteration k_0 on which uses recourse functions $\mathcal{Q}^{k_0}_{t+1}$ instead of $\mathcal{Q}_{t+1}$. Recall that policy $(x^*_1, x^*_2(\cdot), \ldots, x^*_T(\cdot))$ is optimal if for every realization $\tilde{\xi}_{1:T} := (\xi_1, \tilde{\xi}_2, \ldots, \tilde{\xi}_T)$ of $\xi_{1:T} := (\xi_1, \xi_2, \ldots, \xi_T)$, we have that $x^*_t(\xi_1, \tilde{\xi}_2, \ldots, \tilde{\xi}_t)$ solves

$$\mathfrak{Q}_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \tilde{\xi}_t) = \inf_{x_t} \{\tilde{c}_t^T x_t + \mathcal{Q}_{t+1}(x_t) \ : \ x_t \in X_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \tilde{\xi}_t)\} \tag{16}$$

for every $t = 1, \ldots, T$, with the convention that $x^*_0 = x_0$. We prove for $t = 1, \ldots, T$,

$\overline{\mathcal{H}}(t)$: for every $k \geq k_0$ and for every sample $\tilde{\xi}_{1:t} = (\xi_1, \tilde{\xi}_2, \ldots, \tilde{\xi}_t)$ of $(\xi_1, \xi_2, \ldots, \xi_t)$, we have $\underline{\mathfrak{Q}}^k_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \tilde{\xi}_t) = \mathfrak{Q}_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \tilde{\xi}_t)$.

We show $\overline{\mathcal{H}}(1), \ldots, \overline{\mathcal{H}}(T)$ by induction. $\overline{\mathcal{H}}(T)$ holds since $\underline{\mathfrak{Q}}^k_T = \mathfrak{Q}_T$ for every k. Now assume that $\overline{\mathcal{H}}(t+1)$ holds for some $t \in \{1, \ldots, T-1\}$. We want to show $\overline{\mathcal{H}}(t)$. Take an arbitrary $k \geq k_0$ and a sample $\tilde{\xi}_{1:t-1} = (\xi_1, \tilde{\xi}_2, \ldots, \tilde{\xi}_{t-1})$ of $(\xi_1, \xi_2, \ldots, \xi_{t-1})$. We have for every $j = 1, \ldots, M_t$, that

$$\begin{aligned} \underline{\mathfrak{Q}}^k_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj}) &= \begin{cases} \inf \ c_{tj}^T x_t + \mathcal{Q}^k_{t+1}(x_t) \\ x_t \in X_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj}) \end{cases} \\ &= c_{tj}^T x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}) + \mathcal{Q}^k_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})). \end{aligned} \tag{17}$$

Now we check that for $j = 1, \ldots, M_t$, we have

$$\mathcal{Q}^k_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})) = \mathcal{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})). \tag{18}$$

Indeed, if this relation did not hold, since $\mathcal{Q}^k_{t+1} \leq \mathcal{Q}_{t+1}$, we would have

$$\mathcal{Q}^{k_0}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})) = \mathcal{Q}^k_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})) < \mathcal{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})).$$

From the definitions of $\mathcal{Q}^k_{t+1}, \mathcal{Q}_{t+1}$, there exists $m \in \{1, \ldots, M_{t+1}\}$ such that

$$\mathfrak{Q}^{k_0}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m}) < \mathfrak{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m}).$$

Since the realization of CuSMuDA is in Ω_1, there exists an infinite set of iterations such that the sampled scenario for stages $1, \ldots, t$, is $(\tilde{\xi}_{1:t-1}, \xi_{tj})$. Let ℓ be one of these iterations strictly greater than k_0. Using $\overline{\mathcal{H}}(t+1)$, we have that

$$\underline{\mathfrak{Q}}^\ell_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m}) = \mathfrak{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m})$$

which yields

$$\mathfrak{Q}^{k_0}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m}) < \underline{\mathfrak{Q}}^\ell_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m})$$

and at iteration $\ell > k_0$ we would construct a cut for $\mathfrak{Q}_{t+1}(\cdot, \xi_{t+1m})$ at $x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}) = x^\ell_t$ with value

$$\underline{\mathfrak{Q}}^\ell_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}), \xi_{t+1m})$$

strictly larger than the value at this point of all cuts computed up to iteration k_0. Due to Lemma 1, this is not possible. Therefore, (18) holds, which, plugged into (17), gives

$$\underline{\mathfrak{Q}}^k_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj}) = c_{tj}^T x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}) + \mathcal{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})) \geq \mathfrak{Q}_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj})$$

(recall that $x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}) \in X_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj})$). Since $\underline{\mathfrak{Q}}^k_t \leq \mathfrak{Q}_t$, we have shown $\underline{\mathfrak{Q}}^k_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj}) = \mathfrak{Q}_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj})$ for every $j = 1, \ldots, M_t$, which is $\overline{\mathcal{H}}(t)$. Therefore, we have proved that for $t = 1, \ldots, T$, for every realization $(\tilde{\xi}_{1:t-1}, \xi_{tj})$ of $\xi_{1:t}$, $x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})$ satisfies $c_{tj}^T x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj}) + \mathcal{Q}_{t+1}(x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})) = \mathfrak{Q}_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}), \xi_{tj})$, meaning that for every $j = 1, \ldots, M_t$, $x^*_t(\tilde{\xi}_{1:t-1}, \xi_{tj})$ is an optimal solution of (16) written with $\tilde{\xi}_{1:t} = (\tilde{\xi}_{1:t-1}, \xi_{tj})$ and completes the proof. $\square$

Remark 4.2. *The convergence proof above also shows the almost sure convergence in a finite number of iterations for cut selection strategies that would always select more cuts than any cut selection strategy satisfying Assumption (H3). It also shows the convergence of Level H cut selection from (Philpott et al. 2012) which keeps the H cuts having the largest values at each trial point.*

5. Application to portfolio selection and inventory management

5.1. Portfolio selection.

5.1.1. *Model.* We consider a portfolio selection problem with direct transaction costs over a discretized horizon of T stages. The direct buying and selling transaction costs are proportional to the amount of the transaction (Ben-Tal et al. 2000, Best and Hlouskova 2007).[1] Let $x_t(i)$ be the dollar value of asset $i = 1, \ldots, n+1$ at the end of stage $t = 1, \ldots, T$, where asset $n+1$ is cash; $\xi_t(i)$ is the return of asset i at t; $y_t(i)$ is the amount of asset i sold at the end of t; $z_t(i)$ is the amount of asset i bought at the end of t, $\eta_t(i) > 0$ and $\nu_t(i) > 0$ are respectively the proportional selling and buying transaction costs at t. Each component $x_0(i), i = 1, \ldots, n+1$, of x_0 is known. The budget available at the beginning of the investment period is $\sum_{i=1}^{n+1} \xi_1(i)x_0(i)$ and $u(i)$ represents the maximal amount that can be invested in asset i.

For $t = 1, \ldots, T$, given a portfolio $x_{t-1} = (x_{t-1}(1), \ldots, x_{t-1}(n), x_{t-1}(n+1))$ and ξ_t, we define the set $X_t(x_{t-1}, \xi_t)$ as the set of $(x_t, y_t, z_t) \in \mathbb{R}^{n+1} \times \mathbb{R}^n \times \mathbb{R}^n$ satisfying

$$x_t(n+1) = \xi_t(n+1)x_{t-1}(n+1) + \sum_{i=1}^{n} \Big((1-\eta_t(i))y_t(i) - (1+\nu_t(i))z_t(i) \Big), \tag{19}$$

and for $i = 1, \ldots, n$,

$$x_t(i) = \xi_t(i)x_{t-1}(i) - y_t(i) + z_t(i), \tag{20a}$$

$$y_t(i) \leq \xi_t(i)x_{t-1}(i), \tag{20b}$$

$$x_t(i) \leq u(i) \sum_{j=1}^{n+1} \xi_t(j)x_{t-1}(j), \tag{20c}$$

$$x_t(i) \geq 0, y_t(i) \geq 0, z_t(i) \geq 0. \tag{20d}$$

Constraints (19) are the cash flow balance constraints and define how much cash is available at each stage. Constraints (20a) define the amount of security i held at each stage t and take into account the proportional transaction costs. Constraints (20b) preclude selling an amount larger than the amount held. Constraints (20c) prevent the position in security i at time t from exceeding a proportion $u(i)$. Constraints (20d) prevent short-selling and enforce the non-negativity of the amounts bought and sold.

With this notation, the following dynamic programming equations of a risk-neutral portfolio model can be written: for $t = T$, setting $\mathcal{Q}_{T+1}(x_T) = \mathbb{E}[\sum_{i=1}^{n+1} \xi_{T+1}(i)x_T(i)]$ we solve the problem

$$\mathfrak{Q}_T\left(x_{T-1}, \xi_T\right) = \left\{ \begin{array}{l} \sup \ \mathcal{Q}_{T+1}(x_T) \\ (x_T, y_T, z_T) \in X_T(x_{T-1}, \xi_T), \end{array} \right. \tag{21}$$

while at stage $t = T-1, \ldots, 1$, we solve

$$\mathfrak{Q}_t\left(x_{t-1}, \xi_t\right) = \left\{ \begin{array}{l} \sup \ Q_{t+1}\left(x_t\right) \\ (x_t, y_t, z_t) \in X_t(x_{t-1}, \xi_t), \end{array} \right. \tag{22}$$

where for $t = 2, \ldots, T$, $\mathcal{Q}_t(x_{t-1}) = \mathbb{E}[\mathfrak{Q}_t\left(x_{t-1}, \xi_t\right)]$. With this model, we maximize the expected return of the portfolio taking into account the transaction costs, non-negativity constraints, and bounds imposed on the different securities.

[1] This portfolio problem was solved using SDDP and SREDA (Stochastic REgularized Decomposition Algorithm) in (Guigues et al. 2017).

5.1.2. *CuSMuDA for portfolio selection.* We assume that the return process (ξ_t) satisfies Assumption (H1).[2] In this setting, we can solve the portfolio problem under consideration using MuDA and CuSMuDA. For the sake of completeness, we show how to apply MuDA to this problem, including the stopping criterion (CuSMuDA follows, incorporating one of the pseudo-codes from Figure 1). In this implementation, N independent scenarios $\tilde{\xi}^k, k = (i-1)N+1, \ldots, iN$, of $(\xi_1, \xi_2, \ldots, \xi_T)$ are sampled in the forward pass of iteration i to obtain N sets of trial points (note that the convergence proof of Theorem 1 still applies for this variant of CuSMuDA). At the end of iteration i, we end up with approximate functions $\mathfrak{Q}_t^i(x_{t-1}, \xi_{tj}) = \max_{1 \le \ell \le iN} \langle \beta_{tj}^\ell, x_{t-1} \rangle$ for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ (observe that for this problem the cuts have no intercept).

Forward pass of iteration i. We generate N scenarios $\tilde{\xi}^k = (\tilde{\xi}_1^k, , \tilde{\xi}_2^k, \ldots, \tilde{\xi}_T^k), k = (i-1)N+1, \ldots, iN$, of $(\xi_2, \ldots, \xi_T)$ and solve for $k = (i-1)N+1, \ldots, iN$, and $t = 1, \ldots, T-1$, the problem

$$\begin{cases} \inf\limits_{x_t, f} \displaystyle\sum_{\ell=1}^{M_{t+1}} p_{t+1\ell} f_\ell \\ x_t \in X_t(x_{t-1}^k, \tilde{\xi}_t^k), \\ f_\ell \ge \langle \beta_{t+1\ell}^m, x_t \rangle, m = 1, \ldots, (i-1)N, \ell = 1, \ldots, M_{t+1}, \end{cases}$$

starting from $(x_0^k, \tilde{\xi}_1^k) = (x_0, \xi_1)$.[3] Let x_t^k be an optimal solution. For $t = T$ and $k = (i-1)N+1, \ldots, iN$, we solve

$$\begin{cases} \inf \ -x_T^T \mathbb{E}[\xi_{T+1}] \\ x_T \in X_T(x_{T-1}^k, \tilde{\xi}_T^k), \end{cases}$$

with optimal solution x_T^k.

In the end of the forward pass we compute the empirical mean $\overline{\texttt{Cost}}^i$ and standard deviation σ^i of the cost on the sampled scenarios of iteration i:

$$\overline{\texttt{Cost}}^i = -\frac{1}{N} \sum_{k=(i-1)N+1}^{iN} \mathbb{E}[\xi_{T+1}]^T x_T^k, \quad \sigma^i = \sqrt{\frac{1}{N} \sum_{k=(i-1)N+1}^{iN} \Big(-\mathbb{E}[\xi_{T+1}]^T x_T^k - \overline{\texttt{Cost}}^i \Big)^2}. \tag{23}$$

This allows us to compute the upper end z_{sup}^i of a one-sided confidence interval on the mean cost of the policy obtained at iteration i given by

$$z_{\text{sup}}^i = \overline{\texttt{Cost}}^i + \frac{\sigma^i}{\sqrt{N}} \Phi^{-1}(1-\alpha) \tag{24}$$

where $\Phi^{-1}(1-\alpha)$ is the $(1-\alpha)$-quantile of the standard Gaussian distribution.

Backward pass of iteration i. For $k = (i-1)N+1, \ldots, iN$, we solve for $t = T$, $j = 1, \ldots, M_T$,

$$\begin{cases} \inf \ -x_T^T \mathbb{E}[\xi_{T+1}] \\ x_T \in X_T(x_{T-1}^k, \xi_{Tj}) \end{cases} \tag{25}$$

and for $t = T-1, \ldots, 2$, $j = 1, \ldots, M_t$,

$$\begin{cases} \inf\limits_{x_t, f} \displaystyle\sum_{\ell=1}^{M_{t+1}} p_{t+1\ell} f_\ell \\ x_t \in X_t(x_{t-1}^k, \xi_{tj}), \\ f_\ell \ge \langle \beta_{t+1\ell}^m, x_t \rangle, m = 1, \ldots, iN, \ell = 1, \ldots, M_{t+1}. \end{cases} \tag{26}$$

For stage t problem above with realization ξ_{tj} of ξ_t (problem (25) for $t = T$ and (26) for $t < T$), let λ_{tj}^k be the optimal Lagrange multipliers associated to the equality constraints and let $\mu_{1tj}^k \ge 0, \mu_{2tj}^k \ge 0$

[2]It is possible (at the expense of the computational time) to incorporate stagewise dependant returns within the decomposition algorithms under consideration, for instance including in the state vector the relevant history of the returns as in (Infanger and Morton 1996, Guigues 2014).

[3]We use minimization instead of maximization subproblems. In this context, the optimal mean income is the opposite of the optimal value of the first stage problem.

Number of stocks	Stocks
$n = 4$	AAPL (Apple Inc.), XOM (Exxon Mobil Corp.), MSFT (Microsoft Corp.), JNJ (Johnson & Johnson).
$n = 5$	AAPL, XOM, MSFT, JNJ, WFC (Wells Fargo).
$n = 6$	AAPL, XOM, MSFT, JNJ, WFC, GE (General Electric)

TABLE 1. Subset of stocks chosen.

re

be the optimal Lagrange multipliers associated with respectively the constraints $y_t(i) \leq \xi_{tj}(i)x_{t-1}(i)$ and $x_t(i) \leq u(i) \sum_{\ell=1}^{n+1} \xi_{tj}(\ell)x_{t-1}(\ell)$. We compute

$$\beta_{tj}^k = \left(\lambda_{tj}^k - \langle u, \mu_{2tj}^k\rangle \mathbf{e} - \begin{bmatrix} \mu_{1tj}^k \\ 0 \end{bmatrix}\right) \circ \xi_{tj},$$

where $\mathbf{e}$ is a vector in $\mathbb{R}^{n+1}$ of ones and where for vectors x, y, the vector $x \circ y$ has components $(x \circ y)(i) = x(i)y(i)$.

Stopping criterion. In the end of the backward pass of iteration i, we solve

$$\begin{cases} \inf_{x_1, f} \sum_{\ell=1}^{M_2} p_{2\ell} f_\ell \\ x_1 \in X_1(x_0, \xi_1), \\ f_\ell \geq \langle \beta_{2\ell}^m, x_1\rangle, m = 1, \ldots, iN, \ell = 1, \ldots, M_2, \end{cases}$$

whose optimal value provides a lower bound z_{inf}^i on the optimal value $\mathcal{Q}_1(x_0)$ of the problem. Given a tolerance $\varepsilon > 0$, the algorithm stops either when $z_{\text{inf}}^i = 0$ and $z_{\text{sup}}^i \leq \varepsilon$ or when

$$\left|z_{\text{sup}}^i - z_{\text{inf}}^i\right| \leq \varepsilon \max(1, |z_{\text{sup}}^i|). \tag{27}$$

In the expression above, we use $\varepsilon \max(1, |z_{\text{sup}}^i|)$ instead of $\varepsilon |z_{\text{sup}}^i|$ in the right hand side to account for the case $z_{\text{sup}}^i = 0$.

5.1.3. *Numerical results.* We compare three methods to solve the porfolio problem presented in the previous section: MuDA with sampling that we have just described (denoted by `MuDA` for short), CuSMuDA with Multicut Level 1 cut selection (denoted by `CuSMuDA CS 1` for short) and CuSMuDA with MLM Level 1 cut selection (denoted by `CuSMuDA CS 2` for short). The implementation was done in Matlab run on a laptop with Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz. All subproblems in the forward and backward passes were solved numerically using Mosek Optimization Toolbox (Andersen and Andersen 2013).

We fix $u(i) = 1, i = 1, \ldots, n$, while x_0 has components uniformly distributed in $[0, 10]$. For the stopping criterion, we use $N = 200, \alpha = 0.025$, and $\varepsilon = 0.05$ in (27). Below, we generate various instances of the portfolio problem as follows. For fixed T (number of stages [days for our experiment]) and n (number of risky assets), the distributions of $\xi_t(1:n)$ have $M_t = M$ realizations with $p_{ti} = \mathbb{P}(\xi_t = \xi_{ti}) = 1/M$, and $\xi_1(1:n), \xi_{t1}(1:n), \ldots, \xi_{tM}(1:n)$ chosen randomly among historical data of daily returns of n of the assets of the S&P 500 index for the period 18/5/2009-28/5/2015. For fixed n, the n stocks chosen are given in Table 1.[4] The daily return $\xi_t(n+1)$ of the risk-free asset is 0.1% for all t.

Transaction costs are assumed to be known with $\nu_t(i) = \mu_t(i)$ obtained sampling from the distribution of the random variable $0.08 + 0.06\cos(\frac{2\pi}{T}U_T)$ where U_T is a random variable with a discrete distribution over the set of integers $\{1, 2, \ldots, T\}$. For our experiments the values $(5, 4), (5, 5), (5, 6), (8, 4), (8, 5)$, and $(8, 6)$ of the pair (T, n) are considered.

Checking the implementations. If all algorithms are correctly implemented the lower and upper bounds z_{inf}^K and z_{sup}^K computed at the last iteration K of the algorithms should be close to the optimal value $\mathcal{Q}_1(x_0)$ of the problem. If we do not know the optimal value $\mathcal{Q}_1(x_0)$, we can however check on Figure 2 that

[4]They correspond to the first n stocks listed in our matrix of stock prices downloaded from Wharton Research Data Services (WRDS: https://wrds-web.wharton.upenn.edu/wrds/).

	MuDA	CuSMuDA CS 1	CuSMuDA CS 2
$T = 5, n = 4, M = 20$	8.97	13.45	1.31
$T = 5, n = 5, M = 20$	17.03	29.41	2.38
$T = 5, n = 6, M = 20$	45.87	64.22	5.25
$T = 8, n = 4, M = 10$	7.94	16.27	1.56
$T = 8, n = 5, M = 10$	43.39	73.98	4.30
$T = 8, n = 6, M = 10$	63.47	110.55	5.05

TABLE 2. Computational time (in minutes) for solving instances of the portfolio problem of Section 5.1.1 with `MuDA`, `CuSMuDA CS 1`, and `CuSMuDA CS 2`.

the bounds for the 3 algorithms are very close to each other at the last iteration. In Figure 2 we also observe, as expected, that the lower bounds increase and the upper bounds tend to decrease along iterations. Note that few iterations are needed to satisfy the stopping criterion for all instances. However, recall that at each iteration, `MuDA` computes $NM = 200 \times 20 = 4\,000$ cuts for each stage $t = 2, \ldots, T$, meaning that, for instance, for $T = 5, n = 6$, where 5 iterations are needed for `MuDA` to converge (see Figure 2), $5(T-1)MN = 80\,000$ cuts are computed by `MuDA`.

Computational time. The computational time for solving several instances of the portfolio problem is given in Table 2. On these runs, we observe a huge variation in the computational time. On all these runs, `CuSMuDA CS 2` is much quicker than both `MuDA` (between 5.1 and 12.6 times quicker) and `CuSMuDA CS 1` (between 10.3 and 21.9 times quicker). To understand these results, let us analyze the proportion of cuts selected by the two cut selection strategies.

Proportion of cuts selected. An important aspect of the cut selection strategies that directly impacts the computational time is the proportion of cuts selected along the iterations of the algorithm.

Selecting a small number of cuts, especially when many cuts have been computed, allows us to solve the problems in the backward and forward passes much more quickly and can yield an important reduction in CPU time. This is the case of `CuSMuDA CS 2` which surprisingly succeeds in solving the problem selecting at each stage and iteration a very small number of cuts, as can be seen in Figures 3-5. More precisely, the mean proportion of cuts selected (across iterations) is below 5% for all stages with `CuSMuDA CS 2` (see Figure 3). This means that a large number of cuts have the same value at some trial points and for each such trial point if `CuSMuDA CS 1` selects all these cuts `CuSMuDA CS 2` only selects the oldest of these cuts.

On the contrary, if nearly all cuts are selected then the time spent to select the cuts may not be compensated by the (small) reduction in CPU time for solving the problems in the backward and forward passes. This is the case of `CuSMuDA CS 1` (see Figure 3) where for all stages the mean proportion of cuts selected is above 55% and all cuts are selected at the last stage (as expected, recall Example 3.1) but also for some instances at other stages (stages 3 and 4 for the instance $(T, n, M) = (5, 4, 20)$, stage 4 for the instances $(T, n, M) = (5, 5, 20), (5, 6, 20), (8, 5, 20)$, stages 6 and 7 for the instance $(T, n, M) = (8, 4, 10)$).

It is also interesting to note that in general, fewer cuts are selected for the first stages, especially for `CuSMuDA CS 1`. This makes sense since at the last stage, the cuts for functions $\mathfrak{Q}_T(\cdot, \xi_{Tj})$ are exact (and as we recalled, all of them are selected with `CuSMuDA CS 1`) but not necessarily for stages $t < T$ because for these stages approximate recourse functions are used and the approximation errors *propagate* backward. Therefore it is expected that at the early stages old cuts (computed at the first iterations using crude approximations of the recourse functions) will probably not be selected.

Similar conclusions can be drawn from Figures 4-5 which depict the evolution of the mean proportions of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Additionally, we see that for each stage t, the proportions of cuts selected for functions $\mathfrak{Q}_t(\cdot, \xi_{tj}), j = 1, \ldots, M$, are quite similar.

One last comment is now in order. We have already observed that on all experiments, the Multicut Level 1 and Territory Algorithm cut selection strategies, i.e., `CuSMuDA CS 1`, correctly select all cuts at the final stage. However, a crude implementation of the Multicut Level 1 pseudo-code given in Figure 1 resulted

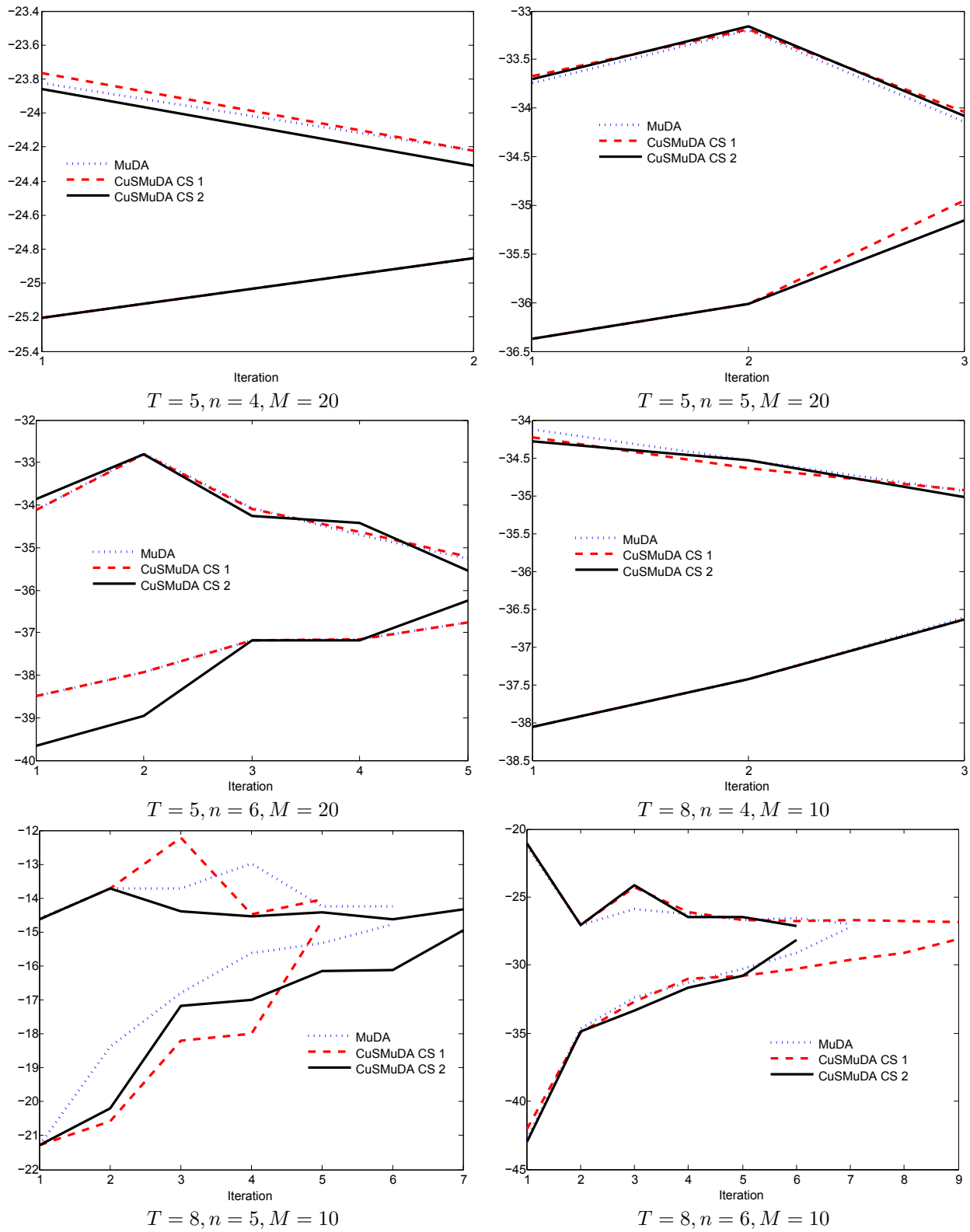


FIGURE 2. Evolution of the upper bounds z^i_{sup} and lower bounds z^i_{inf} along the iterations of the algorithms.

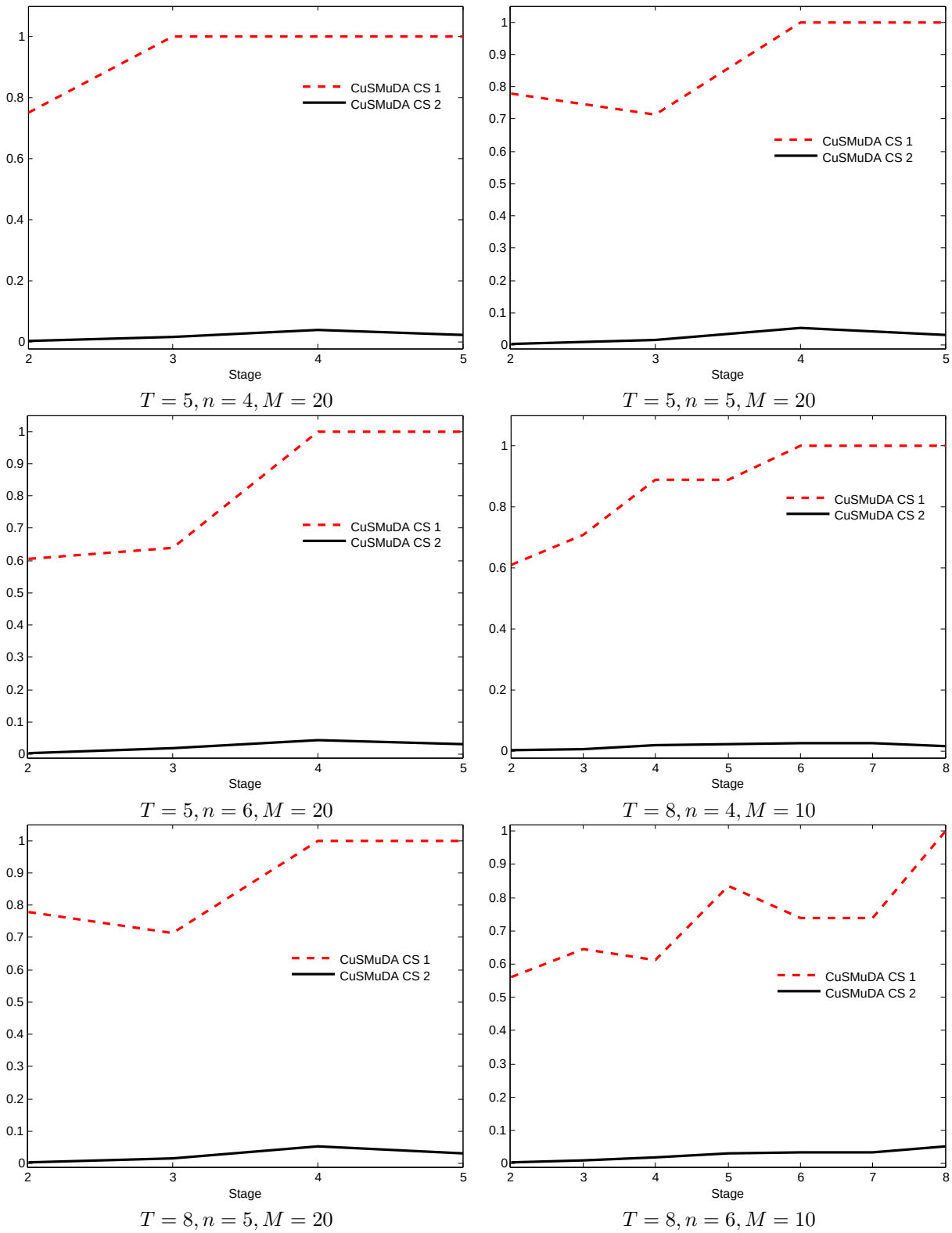


FIGURE 3. Mean proportion of cuts (over the iterations of the algorithm) selected for stages $t = 2, \ldots, T$ by `CuSMuDA CS 1` and `CuSMuDA CS 2`.

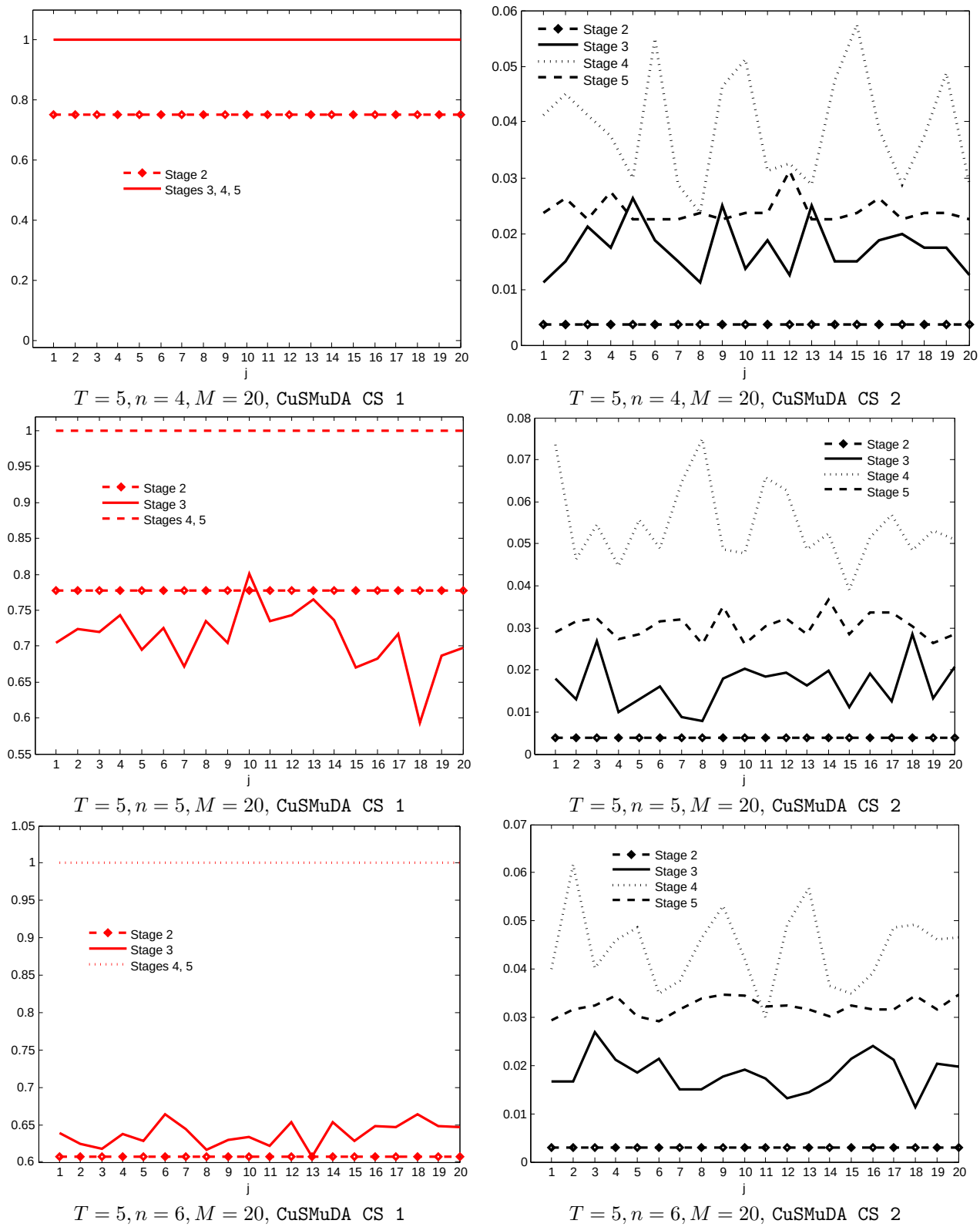
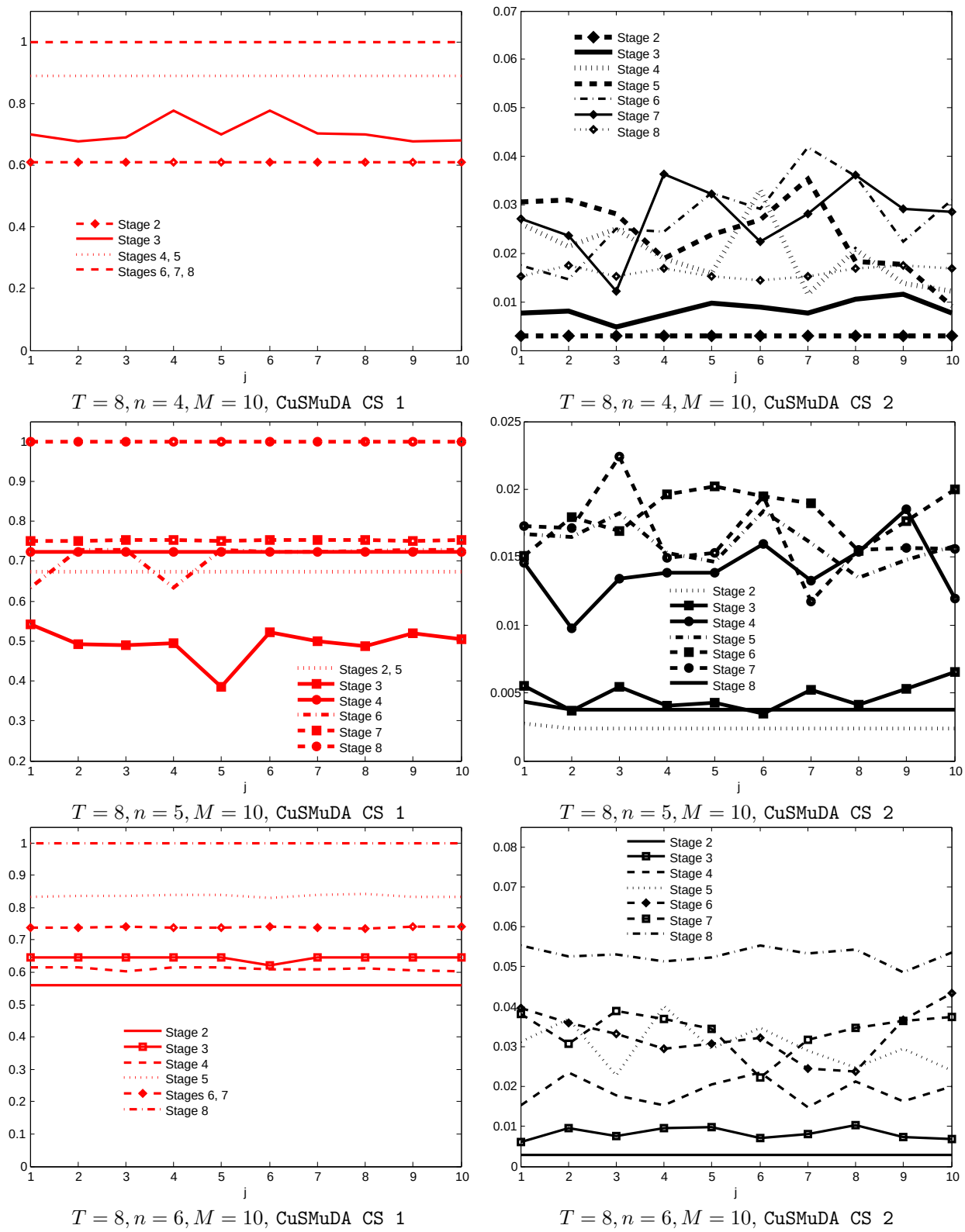


FIGURE 4. Representation for stages $t = 2, \ldots, T$, of the mean proportion of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Left plots: CuSMuDA CS 1, right plots: CuSMuDA CS 2.



$T = 8, n = 4, M = 10$, `CuSMuDA CS 1` $T = 8, n = 4, M = 10$, `CuSMuDA CS 2`

$T = 8, n = 5, M = 10$, `CuSMuDA CS 1` $T = 8, n = 5, M = 10$, `CuSMuDA CS 2`

$T = 8, n = 6, M = 10$, `CuSMuDA CS 1` $T = 8, n = 6, M = 10$, `CuSMuDA CS 2`

FIGURE 5. Representation for stages $t = 2, \ldots, T$, of the mean proportion of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Left plots: `CuSMuDA CS 1`, right plots: `CuSMuDA CS 2`.

in the elimination of cuts at the final stage. This comes from the fact that approximate solutions of the optimization subproblems are computed. Therefore two optimization problems could compute the same cuts but the solver may return two different (very close) approximate solutions. Similarly, a cut may be in theory the highest at some point (for instance cut $\mathcal{C}_{Tj}^k$ is, in theory, the highest at x_{T-1}^k) but numerically the value of another cut at this point may be deemed slightly higher, because of numerical errors. The remedy is to introduce a small error term ε_0 ($\varepsilon_0 = 10^{-6}$ in our experiments) such that the values V_1 and V_2 of two cuts $\mathcal{C}_1$ and $\mathcal{C}_2$ at a trial point are considered equal if $|V_2 - V_1| \leq \varepsilon_0 \max(1, |V_1|)$ while $\mathcal{C}_1$ is above $\mathcal{C}_2$ at this trial point if $V_1 \geq V_2 + \varepsilon_0 \max(1, |V_1|)$. Therefore the pseudo-codes of Multicut Level 1 and MLM Level 1 given in Figure 1 need to be modified. The corresponding pseudo-codes taking into account the approximation errors are given in Figure 10 in the Appendix.

5.2. **Inventory management.**

5.2.1. *Model.* We consider the T-stage inventory management problem given in (Shapiro et al. 2009). For each stage $t = 1, \ldots, T$, on the basis of the inventory level y_t at the beginning of period t, we have to decide on the quantity $x_t - y_t$ of a product to buy so that the inventory level becomes x_t. Given demand ξ_t for that product for stage t, the inventory level is $y_{t+1} = x_t - \xi_t$ at the beginning of stage $t+1$. The inventory level can become negative, in which case a backorder cost is paid. If one is interested in minimizing the average cost over the optimization period, we need to solve the following dynamic programming equations: for $t = 1, \ldots, T$, defining $\mathcal{Q}_t(y_t) = \mathbb{E}_{\xi_t}[\mathfrak{Q}_t(y_t, \xi_t)]$, the stage t problem is

$$\mathfrak{Q}_t(y_t, \xi_t) = \begin{cases} \inf c_t(x_t - y_t) + b_t(\xi_t - x_t)_+ + h_t(x_t - \xi_t)_+ + \mathcal{Q}_{t+1}(y_t) \\ y_{t+1} = x_t - \xi_t, x_t \geq y_t, \end{cases}$$

where c_t is the unit buying cost, h_t is the holding cost, and b_t the backorder cost. The optimal mean cost is $\mathcal{Q}_1(y_1)$ where y_1 is the initial stock. In what follows we present the results of numerical simulations obtained solving this problem with MuDA with sampling (denoted again by `MuDA`), CuSMuDA with Mulicut Level 1 cut selection (denoted again by `CuSMuDA CS 1`), and CuSMuDA with MLM Level 1 cut selection (denoted again by `CuSMuDA CS 2`).

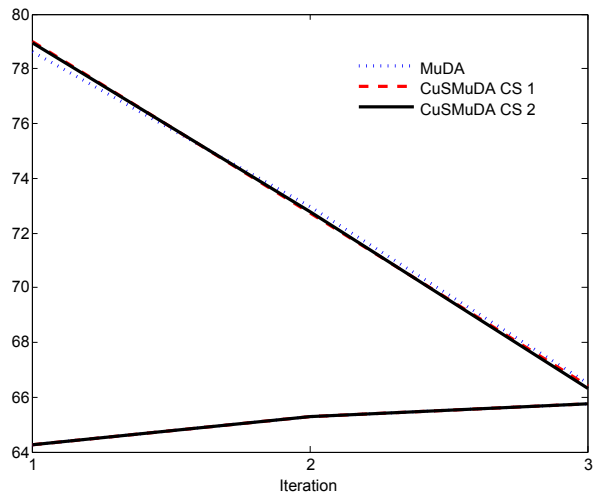
5.2.2. *Numerical results.* We consider six values for the number of stages T ($T \in \{5, 10, 15, 20, 25, 30\}$) and for fixed T, the following values of the problem and algorithm parameters are taken:

- $c_t = 1.5 + \cos(\frac{\pi t}{6}), b_t = 2.8, h_t = 0.2, M_t = M$ for all $t = 1, \ldots, T$,
- $y_1 = 10$, $p_{ti} = \frac{1}{M} = \frac{1}{20}$ for all t, i,
- $\xi_1 = \overline{\xi_1}$ and $(\xi_{t1}, \xi_{t2}, \ldots, \xi_{tM})$ corresponds to a sample from the distribution of $\overline{\xi_t}(1 + 0.1\varepsilon_t)$ for i.i.d $\varepsilon_t \sim \mathcal{N}(0, 1)$ for $t = 2, \ldots, T$, where $\overline{\xi_t} = 5 + 0.5t$,
- for the stopping criterion, in (24) $\alpha = 0.025$ and $N = 200$, and $\varepsilon = 0.05$ in (27).

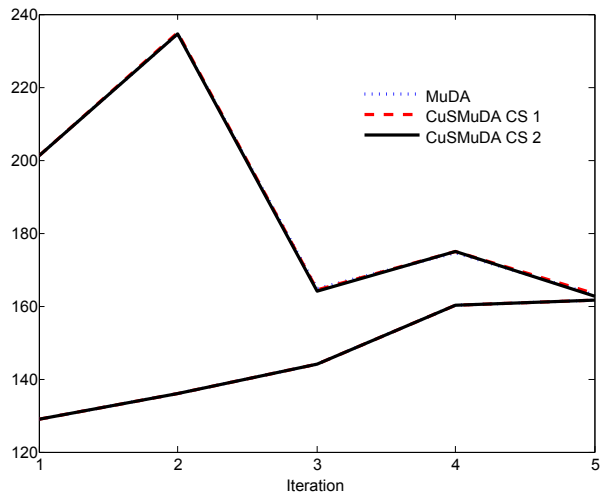
Checking the implementations. As for the portfolio problem, for each value of $T \in \{10, 15, 20, 25, 30\}$, we plot in Figure 6 the evolution of the upper and lower bounds along the iterations of the three algorithms. These bounds are very close for the three algorithms at the final iteration. For these runs, the bounds are in fact very close for all iterations, not only the last one. For $T = 5$ only one iteration was necessary for all three algorithms, yielding the bounds $[25.25, 25.61], [25.25, 25.56]$, and $[25.25, 25.62]$, for respectively `MuDA`, `CuSMuDA CS 1`, and `CuSMuDA CS 2` at the end of this iteration.

Computational time. Table 3 shows the CPU time needed to solve our six instances of inventory management problems. As for the portfolio problem and for the same reasons, `CuSMuDA CS 2` is much quicker than both `MuDA` (between 6.4 and 15.7 times quicker) and `CuSMuDA CS 1` (between 10.2 and 25.6 times quicker).

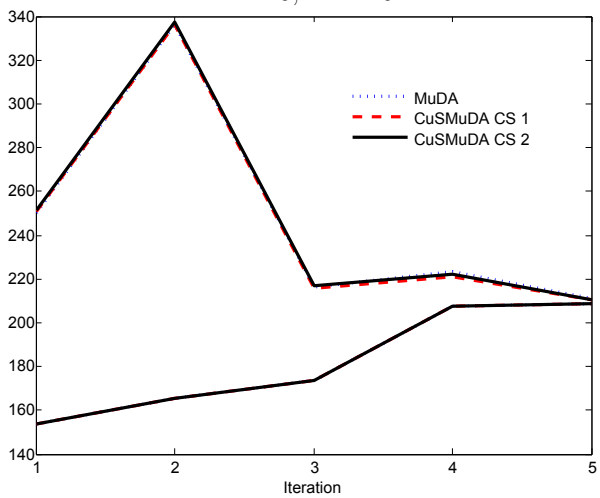
Proportion of cuts selected. The proportions of cuts selected exhibit similar patterns to those observed on the portfolio problems. In Figure 7 which reports for all instances the mean proportion of cuts selected for all stages, we observe again that for all stages on average at least 40% of cuts are selected for `CuSMuDA CS 1` and at most 20% of cuts are selected for `CuSMuDA CS 2`. Figures 8 and 9 show the evolution of the mean proportions of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. The conclusions are similar to those of Section 5.1:



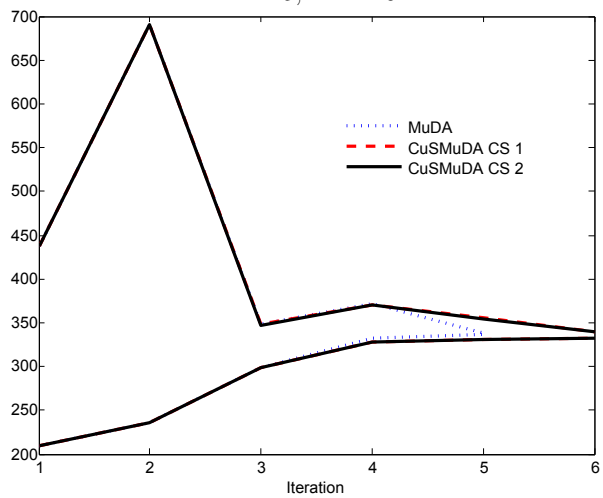
$T = 10, M = 20$



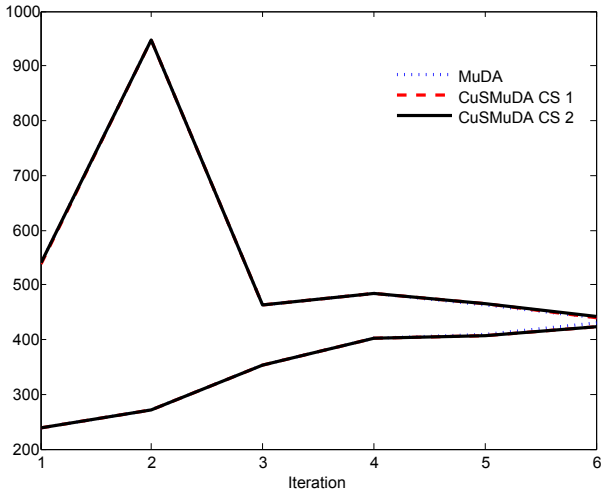
$T = 15, M = 20$



$T = 20, M = 20$



$T = 25, M = 20$



$T = 30, M = 20$

FIGURE 6. Evolution of the upper bounds z^i_{sup} and lower bounds z^i_{inf} along the iterations of the algorithms.

	MuDA	CuSMuDA CS 1	CuSMuDA CS 2
$T = 5, M = 20$	1.57	2.90	0.22
$T = 10, M = 20$	26.33	43.07	1.68
$T = 15, M = 20$	106.52	178.08	8.07
$T = 20, M = 20$	158.10	245.24	19.43
$T = 25, M = 20$	189.27	405.58	22.64
$T = 30, M = 20$	363.86	575.53	56.62

TABLE 3. Computational time (in minutes) for solving instances of the inventory problem of Section 5.2.1 with `MuDA`, `CuSMuDA CS 1`, and `CuSMuDA CS 2`.

(1) for most stages, the proportion of cuts selected with `CuSMuDA CS 1` is very small, below 2% (see the right plots of Figures 8 and 9).
(2) With `CuSMuDA CS 1` for the last stage but also for other stages on some instances (stages $10-14$ for $T = 15$, stages $10-12$ for $T = 20$, stages $25-29$ for $T = 30$) all cuts are selected at all iterations and at several other stages a large proportion (above 80%) of cuts is selected.
(3) For each stage t, the proportions of cuts selected for functions $\mathfrak{Q}_t(\cdot, \xi_{tj}), j = 1, \ldots, M$, are quite similar.

For $T = 5$, where one iteration is enough to satisfy the stopping criterion for all algorithms, `CuSMuDA CS 1` selects all cuts for all stages while `CuSMuDA CS 2` selects for all stages in the backward pass only 0.5% of the cuts, i.e., $(0.5/100)\times 200\times 20 = 20$ cuts per stage. The observations above explain why `CuSMuDA CS 2` is much quicker.

6. Conclusion

We proposed CuSMuDA, a combination of a limited memory variant of the Level 1 cut selection strategy (MLM Level 1) with Multicut Decomposition algorithms to solve multistage stochastic linear programs. The convergence of the method in a finite number of iterations was proved for a class of cut selection strategies that contains MLM Level 1. Compared to traditional Multicut Decomposition methods, CuSMuDA significantly reduced the computational time on all six instances of a portfolio problem and six instances of an inventory problem.

It would be interesting to test CuSMuDA and the limited memory variant of Level 1 (both for MuDA and SDDP) on other types of stochastic programs and to extend the analysis to nonlinear stochastic programs.

Acknowledgments

The second author's research was partially supported by an FGV grant, CNPq grants 307287/2013-0 and 401371/2014-0, and FAPERJ grant E-26/201.599/2014. The authors wish to thank Vincent Leclère for helpful discussions.

References

[1] E. D. Andersen and K.D. Andersen. *The MOSEK optimization toolbox for MATLAB manual. Version 7.0*, 2013. https://www.mosek.com/.
[2] T. Asamov and W. Powell. Regularized decomposition of high-dimensional multistage stochastic programs with markov uncertainty. *Available at:* `https://arxiv.org/abs/1505.02227`, 2015.
[3] A. Ben-Tal, T. Margalit, and A. Nemirovski. Robust modeling of multi-stage portfolio problems. *in: H. Frenk, K. Roos, T. Terlaky, S. Zhang, Eds., High Performance Optimization, Kluwer Academic Publishers*, pages 303–328, 2000.
[4] M. Best and J. Hlouskova. An algorithm for portfolio optimization with variable transaction costs, part 1: Theory. *Journal of Optimization Theory and Applications*, 135:563–581, 2007.
[5] J. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, 1997.
[6] J.R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Oper. Res.*, 33:989–1007, 1985.
[7] J.R. Birge and C. J. Donohue. The Abridged Nested Decomposition Method for Multistage Stochastic Linear Programs with Relatively Complete Recourse. *Algorithmic of Operations Research*, 1:20–30, 2001.
[8] J.R. Birge, C.J. Donohue, D.F. Holmes, and O.G. Svintsitski. A parallel implementation of the nested decomposition algorithm for multistage stochastic linear programs. *Mathematical Programming*, 75:327–352, 1996.

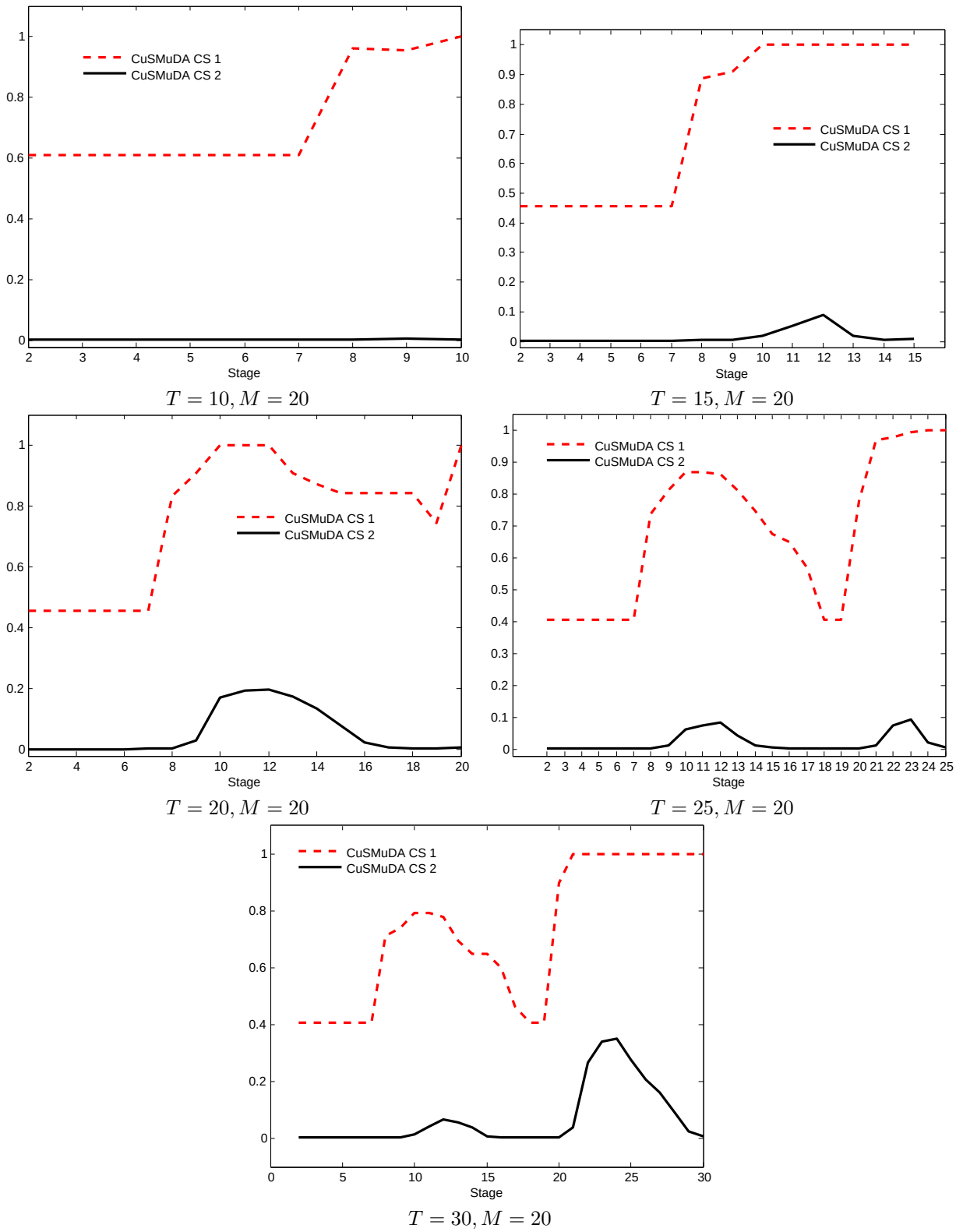


FIGURE 7. Mean proportion of cuts (over the iterations of the algorithm) selected for stages $t = 2, \ldots, T$ for `CuSMuDA CS 1` and `CuSMuDA CS 2`.

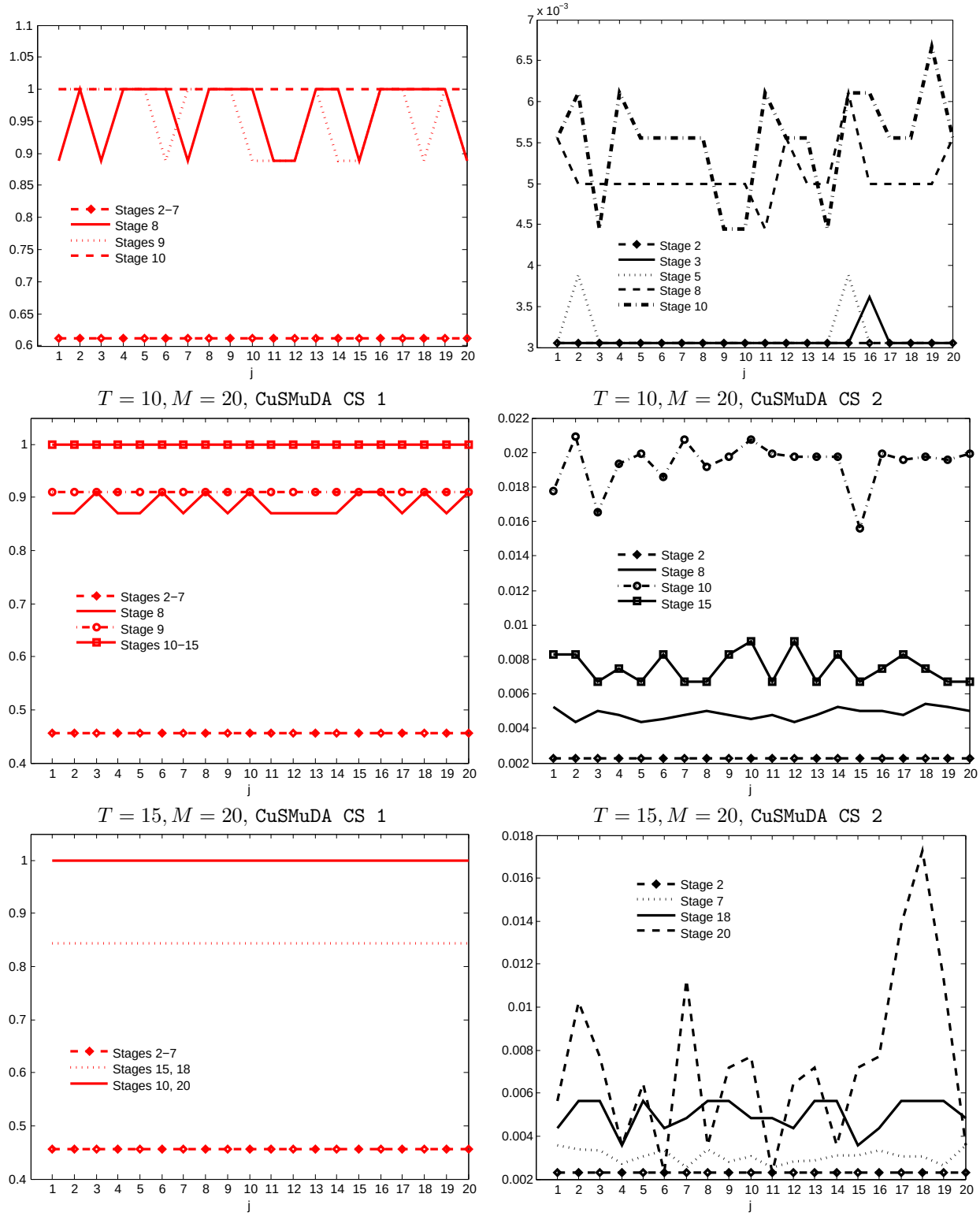
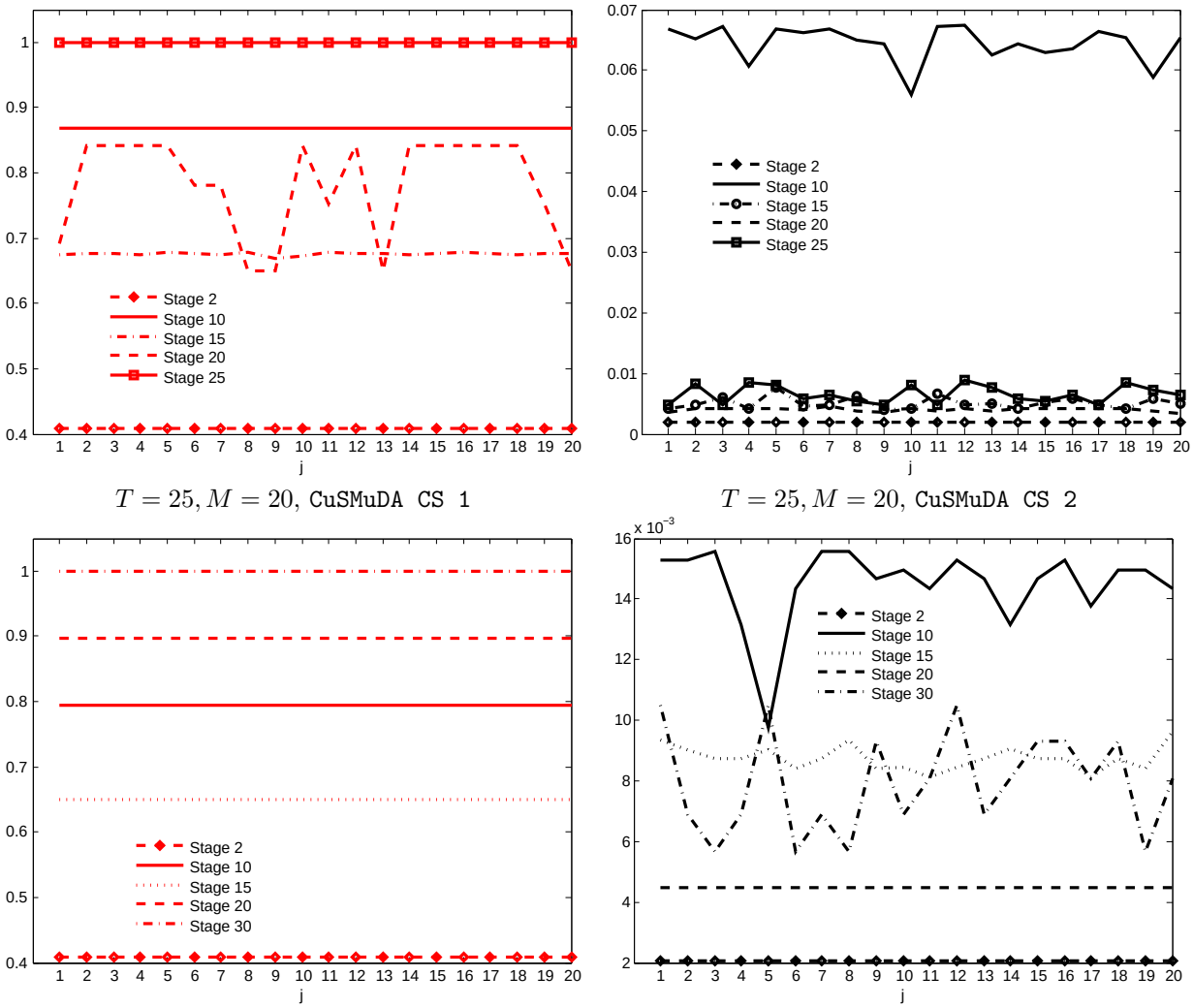


FIGURE 8. Representation for some stages $t \in \{2, \ldots, T\}$, of the mean proportion of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Left plots: `CuSMuDA CS 1`, right plots: `CuSMuDA CS 2`.



$T = 25, M = 20,$ `CuSMuDA CS 1`

$T = 25, M = 20,$ `CuSMuDA CS 2`

$T = 30, M = 20,$ `CuSMuDA CS 1`

$T = 30, M = 20,$ `CuSMuDA CS 2`

FIGURE 9. Representation for some stages $t \in \{2, \ldots, T\}$, of the mean proportion of cuts (over the iterations of the algorithm) selected for $\mathfrak{Q}_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Left plots: `CuSMuDA CS 1`, right plots: `CuSMuDA CS 2`.

[9] J.R. Birge and F.V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34:384–392, 1988.

[10] Z.L. Chen and W.B. Powell. Convergent Cutting-Plane and Partial-Sampling Algorithm for Multistage Stochastic Linear Programs with Recourse. *J. Optim. Theory Appl.*, 102:497–524, 1999.

[11] H. Gassmann. Mslip: A computer code for the multistage stochastic linear programming problem. *Math. Program.*, 47:407–423, 1990.

[12] S. Gaubert, W. McEneaney, and Z. Qu. Curse of dimensionality reduction in max-plus based approximation methods: Theoretical estimates and improved pruning algorithms. *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 1054–1061, 2011.

[13] P. Girardeau, V. Leclere, and A.B. Philpott. On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40:130–145, 2015.

[14] V. Guigues. SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203, 2014.

[15] V. Guigues. Convergence analysis of sampling-based decomposition methods for risk-averse multistage stochastic convex programs. *SIAM Journal on Optimization*, 26:2468–2494, 2016.

[16] V. Guigues. Dual dynamic programing with cut selection: Convergence proof and numerical experiments. *European Journal of Operational Research*, 258:47–57, 2017.

[17] V. Guigues and W. Römisch. Sampling-based decomposition methods for multistage stochastic programs based on extended polyhedral risk measures. *SIAM J. Optim.*, 22:286–312, 2012.
[18] V. Guigues and W. Römisch. SDDP for multistage stochastic linear programs based on spectral risk measures. *Operations Research Letters*, 40:313–318, 2012.
[19] V. Guigues, W. Tekaya, and M. Lejeune. Regularized decomposition methods for deterministic and stochastic convex optimization and application to portfolio selection with direct transaction and market impact costs. *Optimization OnLine*, 2017.
[20] M. Hindsberger and A. B. Philpott. Resa: A method for solving multi-stage stochastic linear programs. *SPIX Stochastic Programming Symposium*, 2001.
[21] G. Infanger and D. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Math. Program.*, 75:241–256, 1996.
[22] V. Kozmik and D.P. Morton. Evaluating policies in risk-averse multi-stage stochastic programming. *Mathematical Programming*, 152:275–300, 2015.
[23] N. Löhndorf, D. Wozabal, and S. Minner. Optimizing trading decisions for hydro storage systems using approximate dual dynamic programming. *Operations Research*, 61:810–823, 2013.
[24] W.M. McEneaney, A. Deshpande, and S. Gaubert. Curse of complexity attenuation in the curse of dimensionality free method for HJB PDEs. *American Control Conference*, pages 4684–4690, 2008.
[25] M.V.F. Pereira and L.M.V.G Pinto. Multi-stage stochastic optimization applied to energy planning. *Math. Program.*, 52:359–375, 1991.
[26] Laurent Pfeiffer, Romain Apparigliato, and Sophie Auchapt. Two methods of pruning benders' cuts and their application to the management of a gas portfolio. *Research Report RR-8133, hal-00753578*, 2012.
[27] A. Philpott and V. de Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218:470–483, 2012.
[28] A. Philpott, V. de Matos, and E. Finardi. Improving the performance of stochastic dual dynamic programming. *Journal of Computational and Applied Mathematics*, 290:196–208, 2015.
[29] A. B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Oper. Res. Lett.*, 36:450–455, 2008.
[30] W.P. Powell. *Approximate Dynamic Programming*. John Wiley and Sons, 2nd edition, 2011.
[31] A. Ruszczyński. Parallel decomposition of multistage stochastic programming problems. *Math. Programming*, 58:201–228, 1993.
[32] A. Shapiro. Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209:63–72, 2011.
[33] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia, 2009.
[34] A. Shapiro, W. Tekaya, J.P. da Costa, and M.P. Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224:375–391, 2013.
[35] A. Shapiro, W. Tekaya, J.P. da Costa, and M.P. Soares. Worst-case-expectation approach to optimization under uncertainty. *Oper. Res.*, 61:1435–1449, 2013.
[36] W. Zhang, H. Rahimian, and G. Bayraksan. Decomposition algorithms for risk-averse multistage stochastic programs with application to water allocation under uncertainty. *INFORMS Journal on Computing*, 28:385–404, 2016.

Appendix

Multicut Level 1	MLM Level 1
$I_{tj}^k = \{k\},\ m_{tj}^k = \mathcal{C}_{tj}^k(x_{t-1}^k)$.	$I_{tj}^k = \{1\},\ m_{tj}^k = \mathcal{C}_{tj}^1(x_{t-1}^k)$.
For $\ell = 1, \ldots, k-1$,	**For** $\ell = 1, \ldots, k-1$,
If $\mathcal{C}_{tj}^k(x_{t-1}^\ell) > m_{tj}^\ell + \varepsilon_0 \max(1, \lvert m_{tj}^\ell \rvert)$	**If** $\mathcal{C}_{tj}^k(x_{t-1}^\ell) > m_{tj}^\ell + \varepsilon_0 \max(1, \lvert m_{tj}^\ell \rvert)$
$I_{tj}^\ell = \{k\},\ m_{tj}^\ell = \mathcal{C}_{tj}^k(x_{t-1}^\ell)$	$I_{tj}^\ell = \{k\},\ m_{tj}^\ell = \mathcal{C}_{tj}^k(x_{t-1}^\ell)$
Else if $\lvert \mathcal{C}_{tj}^k(x_{t-1}^\ell) - m_{tj}^\ell \rvert \le \varepsilon_0 \max(1, \lvert m_{tj}^\ell \rvert)$	**End If**
$I_{tj}^\ell = I_{tj}^\ell \cup \{k\}$	
End If	
If $\mathcal{C}_{tj}^\ell(x_{t-1}^k) > m_{tj}^k + \varepsilon_0 \max(1, \lvert m_{tj}^k \rvert)$	**If** $\mathcal{C}_{tj}^{\ell+1}(x_{t-1}^k) > m_{tj}^k + \varepsilon_0 \max(1, \lvert m_{tj}^k \rvert)$
$I_{tj}^k = \{\ell\},\ m_{tj}^k = \mathcal{C}_{tj}^\ell(x_{t-1}^k)$	$I_{tj}^k = \{\ell+1\},\ m_{tj}^k = \mathcal{C}_{tj}^{\ell+1}(x_{t-1}^k)$
Else if $\lvert \mathcal{C}_{tj}^\ell(x_{t-1}^k) - m_{tj}^k \rvert \le \varepsilon_0 \max(1, \lvert m_{tj}^k \rvert)$	**End If**
$I_{tj}^k = I_{tj}^k \cup \{\ell\}$	
End If	
End For	**End For**

FIGURE 10. Pseudo-codes for selecting the cuts using Multicut Level 1 and MLM Level 1 taking into account approximation errors.