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 portfolio problems considered and between 6.4 and 15.7 times quicker for the inventory problems.

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 pruned 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 a cut selection strategy 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. The most economic of this family of cut selection strategies satisfying (H3), called limited memory Multicut Level 1, selects at each trial point only one cut, namely the oldest cut. 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 Subsection 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.

We use the following notation and terminology:
- The tilde symbol will be used to represent realizations of random variables: for random variable $\xi$, $\tilde{\xi}$ is a realization of $\xi$.
- The usual scalar product in $\mathbb{R}^n$ is denoted by $\langle x, y \rangle = x^T y$ for $x, y \in \mathbb{R}^n$.
- $|I|$ is the cardinality of the set $I$.

2. Problem formulation and assumptions

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

$$Q_1(x_0) = \begin{cases} \inf_{x_1} c_1^T x_1 + Q_2(x_1) \\ A_1 x_1 + B_1 x_0 = b_1, x_1 \geq 0 \end{cases}$$

for $x_0$ given and for $t = 2, \ldots, T$, $Q_t(x_{t-1}) = E_{\xi_t}[Q_{t+1}(x_t)]$ with

$$Q_t(x_{t-1}, \xi_t) = \begin{cases} \inf_{x_t \in \mathbb{R}^n} c_t^T x_t + Q_{t+1}(x_t) \\ A_t x_t + B_t x_{t-1} = b_t, x_t \geq 0, \end{cases}$$

with the convention that $Q_{T+1}$ is null and where for $t = 2, \ldots, T$, random vector $\xi_t$ 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 $\xi_1$ 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 $\Theta_t = \{ \xi_1, \ldots, \xi_M t \}$ the support of $\xi_t$ for stage $t$ with $p_{ti} = \mathbb{P}(\xi_t = \xi_{ti}) > 0, i = 1, \ldots, M_t$ and with vector $\xi_{tj}$ being the concatenation of the elements in $A_t, B_t, b_t, c_t$.

3. Algorithms

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

$$\Omega_t^k(\cdot, \xi_{tj}) = \max_{1 \leq i \leq k} C_{tj}^i(x_{t-1})$$

with $C_{tj}^i(x_{t-1}) = \theta_{tj}^i + \langle \beta_{tj}^i, x_{t-1} \rangle$

where coefficients $\theta_{tj}^i, \beta_{tj}^i$ are computed as explained below. These approximations provide the lower bounding functions

$$Q_t^k(x_{t-1}) = \sum_{j=1}^{M_t} p_{tj} \Omega_t^k(x_{t-1}, \xi_{tj})$$

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

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

**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 $\tilde{\xi}_1^k = \xi_1$. With approximations $\Omega_t^{k-1}(\cdot, \xi_{tj})$ at hand (computed at previous iterations), we solve for $t = 1, \ldots, T$, the problem

$$\begin{cases}
\inf_{x_t \in \mathbb{R}^n} x_t^T \tilde{\xi}_t^k + Q_{t+1}^{k-1}(x_t) \\
x_t \in X_t(x_{t-1}, \tilde{\xi}_t^k)
\end{cases}$$

where $x_0^k = x_0$ and $Q_{t+1}^{k-1}$ is given by (3) with $k$ replaced by $k - 1$. We denote by $x_t^k$ an optimal solution.

**Step 3: Backward pass.** The backward pass builds cuts for $\Omega_t(\cdot, \xi_{tj})$ at $x_{t-1}^k$ computed in the forward pass. For the description of this pass and the convergence analysis of the method, it is convenient to introduce for $k \geq 1$ and $t = 1, \ldots, T$, the function $\tilde{\Omega}_t^k : \mathbb{R}^n \times \Theta_t \rightarrow \mathbb{R}$ given by

$$\tilde{\Omega}_t^k(x_{t-1}, \xi_t) = \begin{cases}
\inf_{x_t \in \mathbb{R}^n} x_t^T c_t^k + Q_{t+1}^{k-1}(x_t) \\
x_t \in X_t(x_{t-1}, \xi_t),
\end{cases}$$

with the convention that $\Theta_1 = \{\xi_1\}$. We set $Q_{T+1}^k \equiv 0$ and solve for $j = 1, \ldots, M_T$, the problem

$$\Omega_T(x_{T-1}^k, \xi_{Tj}) = \begin{cases}
\inf_{x_T \in \mathbb{R}^n} c_{Tj}^T x_T \\
A_{Tj} x_T + B_{Tj} x_{T-1}^k = b_{Tj}, x_T \geq 0,
\end{cases}$$

with dual

$$\begin{cases}
\sup_{\lambda} \lambda^T (b_{Tj} - B_{Tj} x_{T-1}^k) \\
A_{Tj}^T \lambda \leq c_{Tj}.
\end{cases}$$

Denoting by $\lambda_{Tj}^k$ an optimal solution of the dual problem above, we get

$$\Omega_T(x_{T-1}, \xi_{Tj}) \geq (\lambda_{Tj}^k, b_{Tj} - B_{Tj} x_{T-1})$$
and compute $\theta_{t,j}^k = b_{t,j}^T \lambda_{t,j}^k$ and $\beta_{t,j}^k = -B_{t,j}^T \lambda_{t,j}^k$. Then for $t = T - 1$ down to $t = 2$, knowing $Q_{t+1}^k \leq Q_{t+1}$, we solve for $j = 1, \ldots, M_t$, the problem

$$
\Omega_t^k(x_{t-1}, \xi_{t,j}) = \begin{cases} 
\inf_{x_t} c_{t,j}^T x_t + Q_{t+1}^k(x_t) \\
x_t \in X_t(x_{t-1}, \xi_{t,j}) 
\end{cases}
$$

$$
= \begin{cases} 
\inf_{x_t} c_{t,j}^T x_t + \sum_{\ell=1}^{M_t+1} p_{t+1,\ell} f_{\ell} \\
A_{t,j} x_t + B_{t,j} x_{t-1} = b_{t,j}, x_t \geq 0, \\
f_{\ell} \geq \theta_{t+1,\ell} + \langle \beta_{t+1,\ell}^i, x_t \rangle, i = 1, \ldots, k, \ell = 1, \ldots, M_{t+1}.
\end{cases}
$$

Observe that due to (H2) the above problem is feasible and has a finite optimal value. Therefore $\Omega_t^k(x_{t-1}, \xi_{t,j})$ can be expressed as the optimal value of the corresponding dual problem

$$
\Omega_t^k(x_{t-1}, \xi_{t,j}) = \begin{cases} 
\sup_{\lambda, \mu} \lambda^T (b_{t,j} - B_{t,j} x_{t-1}) + \sum_{i=1}^{k} \sum_{\ell=1}^{M_t+1} \mu_{i\ell} \theta_{t+1,\ell}^i \\
A_{t,j}^T \lambda + \sum_{i=1}^{k} \sum_{\ell=1}^{M_t+1} \mu_{i\ell} \beta_{t+1,\ell}^i \leq c_{t,j}, \\
p_{t+1,\ell} = \sum_{i=1}^{k} \mu_{i\ell}, \ell = 1, \ldots, M_{t+1}, \\
\mu_{i\ell} \geq 0, i = 1, \ldots, k, \ell = 1, \ldots, M_{t+1}.
\end{cases}
$$

Denoting by $(\lambda_{t,j}^k, \mu_{t,j}^k)$ an optimal solution of dual problem (8) and using the fact that $Q_{t+1}^k \leq Q_{t+1}$, we get

$$
\Omega_t(x_{t-1}, \xi_{t,j}) \geq \Omega_t^k(x_{t-1}, \xi_{t,j}) \geq \langle \lambda_{t,j}^k, b_{t,j} - B_{t,j} x_{t-1} \rangle + \langle \mu_{t,j}^k, \theta_{t+1}^k \rangle
$$

and we compute

$$
\theta_{t,j}^k = \langle \lambda_{t,j}^k, b_{t,j} \rangle + \langle \mu_{t,j}^k, \theta_{t+1}^k \rangle \text{ and } \beta_{t,j}^k = -B_{t,j}^T \lambda_{t,j}^k.
$$

In these expressions, vector $\theta_{t+1}^k$ has components $\theta_{t+1,\ell}^i, \ell = 1, \ldots, M_{t+1}, i = 1, \ldots, k$, arranged in the same order as components $\mu_{i\ell}^k(\ell, i)$ of $\mu_{t,j}^k$.

**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 $\theta_{t,j}^i, \beta_{t,j}^i$, and trial points $x_{t-1}^i$, but that uses a reduced set of cuts $C_{t,j}^i$ to approximate functions $\Omega_t(\cdot, \xi_{t,j})$ when solving problem (4) in the forward pass and (7) in the backward pass. Denoting by $S_{t,j}^k$ the indexes of the cuts selected at the end of iteration $k$ to approximate $\Omega_t(\cdot, \xi_{t,j})$, the variant of MuDA with cut selection computes in the end of the backward pass of iteration $k$ approximations $Q_{t}^k$ of $Q_t$ given by (3) now with $\Omega_t^k(\cdot, \xi_{t,j})$ given by

$$
\Omega_t^k(x_{t-1}, \xi_{t,j}) = \max_{\ell \in S_{t,j}^k} C_{t,j}^i(x_{t-1}),
$$

where the set $S_{t,j}^k$ is a subset of the indexes of the cuts that have the largest value for at least one of the trial points computed so far. More precisely, sets $S_{t,j}^k$ are initialized taking $S_{t,j}^0 = \{0\}$ and $S_{t,j}^1 = \{1\}$. For $t \in \{2, \ldots, T\}$, sets $S_{t,j}^k$ are computed as follows. For $i = 1, \ldots, k$, $t = 2, \ldots, T$, $j = 1, \ldots, M_t$, we denote by
\( I_{ij}^k \) the set of cuts for \( \Omega_i(\cdot, \xi_{ij}) \) computed at iteration \( k \) or before that have the largest value at \( x_{i-1}^k \):

\[
I_{ij}^k = \arg \max_{\ell=1, \ldots, k} C_{ij}^\ell(x_{i-1}^k),
\]

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

\[
S_{ij}^k = \bigcup_{i=1}^{k} \{ I_{ij}^k(\ell) : \ell \in S_{ij}(|I_{ij}^k|) \}.
\]

We require the selectors to satisfy the following assumption:

(H3) for \( t = 2, \ldots, T, j = 1, \ldots, M_t \), for every \( m \geq 1, S_{ij}(m) \subseteq S_{ij}(m + 1) \).

**Example 3.1** (Multicut Level 1 and Territory Algorithm). The strategy \( S_{ij}(m) = \{1, 2, \ldots, m\} \) 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 \( S_{Tj}^k = \{1, \ldots, k\} \) 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 \( C_{Tj}^{k_1} \) is selected because it is one of the cuts with the highest value at \( x_{T-1}^{k_1} \). Indeed, for any \( 1 \leq k_2 \leq k \) with \( k_2 \neq k_1 \), since \( \lambda_{Tj}^{k_2} \) is feasible for problem (6) with \( x_{T-1}^{k_2} \) replaced by \( x_{T-1}^{k_1} \), we get

\[
C_{Tj}^{k_1}(x_{T-1}^{k_1}) = \Omega_T(x_{T-1}^{k_1}, \xi_{Tj}) \geq (\lambda_{Tj}^{k_2}, b_{Tj} - B_{Tj}x_{T-1}^{k_1}) = C_{Tj}^{k_2}(x_{T-1}^{k_1}).
\]

**Example 3.2** (Multicut limited memory Level 1). The strategy that eliminates the largest amount of cuts, denoted by FMLM Level 1 for short, consists in taking a singleton for every set \( S_{ij}(m) \). For (H3) to be satisfied, this implies \( S_{ij}(m) = \{1\} \). 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.\(^1\)

The computation of \( S_{ij}^k \), i.e., of the cut indexes to select at iteration \( k \), is performed in the backward pass (immediately after computing cut \( C_{ij}^k \)) using the pseudo-code given in the left and right panes of Figure 1 for respectively the Multicut Level 1 and FMLM Level 1 cut selection strategies.

In this pseudo-code, we use the notation \( I_{ij}^i \) in place of \( I_{ij}^k \). We also store in variable \( m_{ij}^i \) the current value of the highest cut for \( \Omega_i(\cdot, \xi_{ij}) \) at \( x_{i-1}^i \). At the end of the first iteration, we initialize \( m_{ij}^1 = C_{ij}^1(x_{i-1}^1) \).

---

\(^1\)FMLM Level 1 stands for First Multicut Limited Memory Level 1 referring to the fact that at any trial point, among the cuts that have the highest value at that point, the first computed cut is selected.
After cut $C_{ij}^k$ is computed at iteration $k \geq 2$, these variables are updated using the relations

$$
\begin{align*}
m_{ij}^k &\leftarrow \max(m_{ij}^k, C_{ij}^k(x_{i-1}^k)), \quad i = 1, \ldots, k - 1, \\
m_{ij}^k &\leftarrow \max(C_{ij}^k(x_{i-1}^k), \ell = 1, \ldots, k).
\end{align*}
$$

We also use an array $\text{Selected}$ of Boolean using the information given by variables $I_{ij}^k$ whose $\ell$-th entry is $\text{True}$ if cut $\ell$ is selected for $\Omega_{t}(\cdot, \xi_{ij})$ and $\text{False}$ otherwise. This allows us in particular to avoid copies of cut indexes that may appear in $I_{ij}^{1,k}$ and $I_{ij}^{2,k}$ with $i_1 \neq i_2$.

4. **Convergence analysis**

We show in Theorem 1 below the convergence of CuSMuDA in a finite number of iterations. We will make the following assumption:

(H4) The samples in the forward passes are independent: $(\tilde{\xi}_2^k, \ldots, \tilde{\xi}_T^k)$ is a realization of $\xi^k = (\xi_2^k, \ldots, \xi_T^k) \sim (\xi_2, \ldots, \xi_T)$ and $\xi^k_1, \xi^k_2, \ldots$, 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
\begin{equation}
\Omega_t^k(\cdot, \xi_{ij}) = \Omega_{t-1}^k(\cdot, \xi_{ij}) \quad \text{and} \quad \Omega_t^k = \Omega_{t-1}^{k_0}.
\end{equation}

Proof. By Assumption (H4) the event $\Omega_1$ such that the sequence $(\xi_{i,t})_k$ goes through any possible realization
of $(\xi_{i,t})_k$ an infinite number of times has probability 1.

Consider a realization of CuSMuDA in $\Omega_1$ corresponding to realizations $(\xi_{i,t})_k$ of $(\xi_{i,t})_k$ in the forward
pass.

$1^0$. We show by induction that the set of possible trial points $\{x_t^k : k \geq 1\}$ is finite for every $t = 1, \ldots, T - 1$.

$1^0$.a. $x_t^k$ is an extreme point of the polyhedron $X_1(x_0, \xi)$ and this polyhedron has a finite number of extreme
points since $A_1$ has a finite number of rows and columns. Therefore $x_t^k$ can only take a finite number of values.

$1^0$.b. Assume now that $x_t^{k-1}$ can only take a finite number of values for some $t \in \{2, \ldots, T\}$. Recall that
$x_t^k$ is an extreme point of one of the sets $X_t(x_{t-1}^k, \xi_t)$ where $j \in \{1, \ldots, M_t\}$. Since each matrix $A_{ij}$ has
a finite number of rows and columns, each set $X_t(x_{t-1}^k, \xi_t)$ has a finite number of extreme points. Using the
induction hypothesis and $M_t < +\infty$, there is a finite number of these sets and therefore $x_t^k$ can only take a
finite number of values. This shows $1^0$.

$2^0$. 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 $\Omega_t(\cdot, \xi_{ij})$. Our induction hypothesis $H(t)$
for $t \in \{2, \ldots, T\}$ is that the sets $\{(\theta_t^k, \beta_t^k) : k \in \mathbb{N}\}, j = 1, \ldots, M_t$, are finite and there exists some finite $k_t$
such that for every $k > k_t$ we have
\begin{equation}
\{(\theta_t^k, \beta_t^k) : \ell \in S_{t,j}^k\} = \{(\theta_t^k, \beta_t^k) : \ell \in S_{t,j}^{k_t}\}
\end{equation}
for every $j = 1, \ldots, M_t$. We will denote by $I_j^{k_t}$ the set $\{I_i^{k_t}(\ell) : \ell \in S_{t,j}(\|I_i^{k_t}\|)\}$.

$2^0$.a. Observe that $\lambda_{t,j}^k$ defined in the backward pass of CuSMuDA is an extreme point of the polyhedron
$\{\lambda : A_T^T\lambda \leq c_T\}$. This polyhedron is a finite intersection of closed half spaces in finite dimension (since $A_T$
has a finite number of rows and columns) and therefore has a finite number of extreme points. It follows that
$\lambda_{t,j}^k$ can only take a finite number of values, same as $(\theta_t^k, \beta_t^k) = (\langle \lambda_{t,j}^k, b_{t,j} \rangle, -B_{t,j}^T \lambda_{t,j}^k)$, and there exists $k_T$
such that for every $k > k_T$ and every $j$, each cut $C_{t,j}^k$ is a copy of a cut $C_{t,j}^{k_T}$ with $1 \leq k' \leq k_T$ (no new cut is
computed for functions $\Omega_T(\cdot, \xi_{T,j})$ for $k > k_T$) and $x_{t-1}^k$ is a copy of some $x_{t-1}^{k'}$ with $1 \leq \ell \leq k_T$.

$2^0$.b. We show that for every $i \geq 1$ and $j = 1, \ldots, M_T$, there exists $1 \leq i' \leq k_T$ and $k_T(i') \geq k_T$ such that
for every $k \geq \max(i, k_T(i'))$ we have
\begin{equation}
\{(\theta_t^k, \beta_t^k) : \ell \in I_j^{k_T}\} = \{(\theta_t^k, \beta_t^k) : \ell \in I_j^{k_T, i'(k_T, i')}\},
\end{equation}
which will show $\mathcal{H}(T)$ with $k_T = \max_{1 \leq j \leq M_T, 1 \leq \nu \leq k_T} k_{Tj}(i')$. Let us show that (13) indeed holds. Let us take $i \geq 1$ and $j \in \{1, \ldots, M_T\}$. If $1 \leq i \leq \tilde{k}_T$ define $i' = i$. Otherwise due to 2.0.a. we can find $1 \leq i' \leq \tilde{k}_T$ such that $x_{t-1}^i = x_{t-1}^{i'}$ which implies $I_{Tj}^k = I_{Tj}^{i'}$ for every $k \geq i$. Now consider the sequence of sets $(I_{Tj}^k)_{k \geq \tilde{k}_T}$.

Due to the definition of $\tilde{k}_r$, the sequence $(|I_{Tj}^k|)_{k \geq \tilde{k}_T}$ is nondecreasing and therefore two cases can happen:

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

(B) The sequence $(|I_{Tj}^k|)_{k \geq \tilde{k}_T}$ is unbounded. Due to Assumption (H3), the sequence $(|S_{Tj}(m)|)_{m}$ is nondecreasing. If there exists $k_{Tj}$ such that $S_{Tj}(k) = S_{Tj}(k_{Tj})$ for $k \geq k_{Tj}$ then $I_{Tj}^{i'_{k_{Tj}}} = I_{Tj}^{i'_{k_{Tj}}}$ for $k \geq k_{Tj}$, $I_{Tj}^{i'} = I_{Tj}^{i'_{k_{Tj}}}$ for $k \geq \max(k_{Tj}, i)$ and (13) holds with $k_{Tj}(i') = k_{Tj}$. Otherwise the sequence $(|S_{Tj}(m)|)_{m}$ is unbounded and an infinite number of cut indexes are selected from the sets $(I_{Tj}^k)_{k \geq \nu}$ to make up sets $(I_{Tj}^k)_{k \geq \nu}$. 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 $I_{Tj}^k$ and therefore there exists $k_{Tj}(i')$ such that $I_{Tj}^{i'_{k_{Tj}}} = I_{Tj}^{i'_{k_{Tj}(i')}}$ for $k \geq k_{Tj}(i')$, $I_{Tj}^{i'} = I_{Tj}^{i'_{k_{Tj}(i')}}$ for $k \geq \max(k_{Tj}(i'), i)$, and (13) holds.

2.0.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

$$
(\lambda, b_{tj}) + \sum_{\ell=1}^{M_{t+1}} \sum_{i \in S_{t+1}^k} \mu_i \theta_{t+1, \ell} - B_{tj}^T \lambda 
$$

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

$$
\mu \geq 0, p_{t+1, \ell} = \sum_{i \in S_{t+1}^k} \mu_i \delta_{t+1, \ell}, \ell = 1, \ldots, M_{t+1},
$$

$$
A_{tj}^T \lambda + \sum_{\ell=1}^{M_{t+1}} \sum_{i \in S_{t+1}^k} \mu_i \beta_{t+1, \ell} \leq c_{tj}.
$$

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 $(\lambda, \mu)$ 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 $(\lambda, \mu)$ 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 $(\lambda, \mu)$ in the place of $(\lambda, \mu)$ where $(\lambda, \mu)$ is an extreme point of $\mathcal{P}_{tjk_{t+1}}$, obtained replacing the basic columns $\beta_{t+1, \ell}$ with $i \in S_{t+1}^k, i \notin S_{t+1}^{k_{t+1}}$ associated with $\mu$ by columns $\beta_{t+1, \ell}$ with $i' \in S_{t+1}^{k_{t+1}}$ such that $(\theta_{t+1, \ell}, \beta_{t+1, \ell}) = (\theta_{t+1, \ell}, \beta_{t+1, \ell})$ (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 $(\theta_{tj}^k, \beta_{tj}^k) \in \mathcal{D}_{tjk}$, the cut coefficients $(\theta_{tj}^k, \beta_{tj}^k)$ can only take a finite number of values. This shows the first part of $\mathcal{H}(t)$. Therefore, there exists $\tilde{k}_t$ such that for every $k > \tilde{k}_t$ and every $j$, each cut $C_{tj}^k$ is a copy of a cut $C_{tj}^{k'}$ with $1 \leq k' \leq \tilde{k}_t$ (no new cut is computed for functions $\Omega_{t}^j(\xi_{tj})$ for $k > \tilde{k}_t$ and $x_{t-1}^k$ is also a copy of some $x_{t-1}^\ell$ with $1 \leq \ell \leq \tilde{k}_t$.}
Finally, we can show (12) proceeding as in $2^0 \cdot b$, 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 FMLM Level 1 cut selection strategies corresponding to selectors $S_{ij}$ satisfying respectively $S_{ij}(m) = \{1,\ldots,m\}$ and $S_{ij}(m) = \{1\}$, integers $k_0, \hat{k}$, defined in Lemma 1 and its proof satisfy $k_0 \leq \max_{t=2,\ldots,T} \hat{k}_t$. For other selectors $S_{ij}$ this relation is not necessarily satisfied (see $2^0 \cdot b$.-(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 $\Omega_1$ be defined as in the proof of Lemma 1 and let $\Omega_2$ 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 $(\xi_{1:T})_k$ of $(\xi_{1:T})_k$ in the forward pass and let $(x_1^*, x_2^*, \ldots, x_T^*)$ be the policy obtained from iteration $k_0$ on which uses recourse functions $Q_{t+1}^{k_0}$ instead of $Q_{t+1}$. Recall that policy $(x_1^*, x_2^*, \ldots, x_T^*)$ is optimal if for every realization $\xi_{1:T} := (\xi_1, \xi_2, \ldots, \xi_T)$ of $\xi_{1:T} := (\xi_1, \xi_2, \ldots, \xi_T)$, we have that $x_t^*(\xi_1, \xi_2, \ldots, \xi_t)$ solves

\begin{equation}
\mathcal{H}(t) : \text{for every } k \geq k_0 \text{ and for every sample } \xi_{1:t} := (\xi_1, \xi_2, \ldots, \xi_t) \text{ of } (\xi_1, \xi_2, \ldots, \xi_t), \text{ we have }
\end{equation}

\begin{equation}
\Omega_t(x_{t-1}^*(\xi_{1:t-1}), \xi_t) = \inf_{x_t} \{ c_t^T x_t + Q_{t+1}(x_t) : x_t \in X_t(x_{t-1}^*(\xi_{1:t-1}), \xi_t) \}
\end{equation}

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

\begin{equation}
\mathcal{H}(t) : \text{for every } k \geq k_0 \text{ and for every sample } \xi_{1:t} := (\xi_1, \xi_2, \ldots, \xi_t) \text{ of } (\xi_1, \xi_2, \ldots, \xi_t), \text{ we have }
\end{equation}

\begin{equation}
\Omega_t(x_{t-1}^*(\xi_{1:t-1}), \xi_t) = \Omega_2(x_{t-1}^*(\xi_{1:t-1}), \xi_t).
\end{equation}

We show $\mathcal{H}(1), \ldots, \mathcal{H}(T)$ by induction. $\mathcal{H}(T)$ holds since $\Omega_T^k = \Omega_T$ for every $k$. Now assume that $\mathcal{H}(t + 1)$ holds for some $t \in \{1, \ldots, T - 1\}$. We want to show $\mathcal{H}(t)$. Take an arbitrary $k \geq k_0$ and a sample $\hat{\xi}_{1:t-1} = (\hat{\xi}_1, \hat{\xi}_2, \ldots, \hat{\xi}_{t-1})$ of $(\xi_1, \xi_2, \ldots, \xi_{t-1})$. We have for every $j = 1, \ldots, M_t$, that

\begin{equation}
\Omega_t^k(x_{t-1}^*(\hat{\xi}_{1:t-1}), \xi_t) = \begin{cases} 
\inf_{x_t} c_t^T x_t + Q_{t+1}^k(x_t) \\
x_t \in X_t(x_{t-1}^*(\hat{\xi}_{1:t-1}), \xi_t) 
\end{cases}
\end{equation}

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

\begin{equation}
Q_{t+1}^k(x_t^*(\hat{\xi}_{1:t-1}, \xi_t)) = Q_{t+1}(x_t^*(\hat{\xi}_{1:t-1}, \xi_t)).
\end{equation}

Indeed, if this relation did not hold, since $Q_{t+1}^k \leq Q_{t+1}$, we would have

\begin{equation}
Q_{t+1}^k(x_t^*(\hat{\xi}_{1:t-1}, \xi_t)) = Q_{t+1}^k(x_t^*(\hat{\xi}_{1:t-1}, \xi_t)) < Q_{t+1}(x_t^*(\hat{\xi}_{1:t-1}, \xi_t)).
\end{equation}
Recalling the definitions of $Q_{t+1}^k$, $Q_{t+1}$, this implies that there exists $m \in \{1,\ldots,M_{t+1}\}$ such that

$$Q_{t+1}^k(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m}) < Q_{t+1}(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m}).$$

Since the realization of CuSMuDA is in $\Omega_1$, there exists an infinite set of iterations such that the sampled scenario for stages $1,\ldots,t$, is $(\tilde{\xi}_{1:t-1},\xi_t)$. Let $\ell$ be one of these iterations strictly greater than $k_0$. Using $\overline{\tau}(t+1)$, we have that

$$Q_{t+1}^\ell(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m}) = Q_{t+1}(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m})$$

which yields

$$Q_{t+1}^k(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m}) < Q_{t+1}(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m})$$

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

$$Q_{t+1}^\ell(x^*_t(\tilde{\xi}_{1:t-1},\xi_t),\xi_{t+1:m})$$

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

$$\widehat{Q}_{t}^k(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t) = c^T_t x^*_t(\tilde{\xi}_{1:t-1},\xi_t) + Q_{t+1}(x^*_t(\tilde{\xi}_{1:t-1},\xi_t)) \geq \Omega_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t)$$

(recall that $x^*_t(\tilde{\xi}_{1:t-1},\xi_t) \in X_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t)$). Since $\widehat{Q}_{t}^k \leq \Omega_t$, we have shown $\widehat{Q}_{t}^k(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t) = \Omega_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t)$ for every $j = 1,\ldots,M_t$, which is $\overline{\tau}(t)$. Therefore, we have proved that for $t = 1,\ldots,T$, for every realization $(\tilde{\xi}_{1:t-1},\xi_t)$ of $\xi_{1:t}$, $x^*_t(\tilde{\xi}_{1:t-1},\xi_t)$ satisfies $c^T_t x^*_t(\tilde{\xi}_{1:t-1},\xi_t) + Q_{t+1}(x^*_t(\tilde{\xi}_{1:t-1},\xi_t)) = \Omega_t(x^*_{t-1}(\tilde{\xi}_{1:t-1}),\xi_t)$, meaning that for every $j = 1,\ldots,M_t$, $x^*_t(\tilde{\xi}_{1:t-1},\xi_t)$ is an optimal solution of (16) written with $\tilde{\xi}_{1:t} = (\tilde{\xi}_{1:t-1},\xi_t)$ and achieves the proof.

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 at each trial point the H cuts having the largest values.

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 purchasing and selling transaction costs are proportional to the amount of the transaction (Ben-Tal et al. 2000, Best and Hlouskova 2007).\footnote{This portfolio problem was solved using SDDP and SREDA (Stochastic REgularized Decomposition Algorithm) in (Guigues et al. 2017).} We denote by $x_t(i)$ the dollar value of asset $i = 1,\ldots,n+1$ at the end of stage $t = 1,\ldots,T$, asset $n+1$ being 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 purchasing 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 \( \sum_{i=1}^{n+1} \xi_i^0 x_0^i \) 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 \( \xi_t, \) 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

\[
(19) \quad x_t(n + 1) = \xi_t(n + 1) x_{t-1}(n + 1) + \sum_{i=1}^n \left( (1 - \eta_t(i)) y_t(i) - (1 + \nu_t(i)) z_t(i) \right),
\]

and for \( i = 1, \ldots, n, \)

\[
(20a) \quad x_t(i) = \xi_t(i) x_{t-1}(i) - y_t(i) + z_t(i),
\]

\[
(20b) \quad y_t(i) \leq \xi_t(i) x_{t-1}(i),
\]

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

\[
(20d) \quad x_t(i) \geq 0, y_t(i) \geq 0, z_t(i) \geq 0.
\]

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) do not allow the position in security \( i \) at time \( t \) to exceed a proportion \( u(i) \). Constraints (20d) prevent short-selling and enforce the non-negativity of the amounts purchased and sold.

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

\[
(21) \quad \mathbb{Q}_T(x_{T-1}, \xi_T) = \left\{ \sup_{(x_T, y_T, z_T) \in X_T(x_{T-1}, \xi_T)} Q_{T+1}(x_T) \right\},
\]

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

\[
(22) \quad \mathbb{Q}_t(x_{t-1}, \xi_t) = \left\{ \sup_{(x_t, y_t, z_t) \in X_t(x_{t-1}, \xi_t)} Q_{t+1}(x_t) \right\},
\]

where for \( t = 2, \ldots, T \), \( Q_t(x_{t-1}) = \mathbb{E}[\mathbb{Q}_t(x_{t-1}, \xi_t)] \). 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.
5.1.2. *CuSMuDA for portfolio selection.* We assume that the return process \((\xi_t)\) satisfies Assumption (H1). In this setting, we can solve the portfolio problem under consideration using MuDA and CuSMuDA. For the sake of completeness, we detail below MuDA applied 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 \(Q^i_t(x_{t-1}, \xi_t) = \max_{1 \leq \ell \leq iN} \langle \beta^t_{\ell}, x_{t-1} \rangle\) for \(Q_t(\cdot, \xi_t)\) (observe that for this problem the cuts have no intercept).}

**Forward pass of iteration \(i\).** We generate \(N\) scenarios \(\tilde{\xi}^k = (\tilde{\xi}^k_1, \tilde{\xi}^k_2, \ldots, \tilde{\xi}^k_T), k = (i - 1)N + 1, \ldots, iN,\) of \((\xi_2, \ldots, \xi_T)\) and solve for \(k = (i - 1)N + 1, \ldots, iN,\) and \(t = 1, \ldots, T - 1,\) the problem

\[
\inf_{x_t} \sum_{\ell=1}^{M_{t+1}} p_{t+1}\ell f_{\ell} \\
x_t \in X_t(x^k_{t-1}, \tilde{\xi}^k_t), \\
f_{\ell} \geq \langle \beta^m_{t+1}\ell, x_t \rangle, m = 1, \ldots, (i - 1)N, \ell = 1, \ldots, M_{t+1},
\]

starting from \((x^k_0, \tilde{\xi}^k_1) = (x_0, \xi_1).\) Let \(x^k_t\) be an optimal solution. For \(t = T\) and \(k = (i - 1)N + 1, \ldots, iN,\) we solve

\[
\inf_{x_T} -x_T^T E[\xi_{T+1}] \\
x_T \in X_T(x^k_{T-1}, \tilde{\xi}^k_T),
\]

with optimal solution \(x^k_T.\)

In the end of the forward pass we compute the empirical mean \(\overline{\text{Cost}}^i\) and standard deviation \(\sigma^i\) of the cost on the sampled scenarios of iteration \(i:\)

\[
(23) \quad \overline{\text{Cost}}^i = -\frac{1}{N} \sum_{k=(i-1)N+1}^{iN} E[\xi_{T+1}]^T x^k_T, \quad \sigma^i = \sqrt{\frac{1}{N} \sum_{k=(i-1)N+1}^{iN} \left( -E[\xi_{T+1}]^T x^k_T - \overline{\text{Cost}}^i \right)^2}.
\]

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

\[
(24) \quad z^i_{\text{sup}} = \overline{\text{Cost}}^i + \frac{\sigma^i}{\sqrt{N}} \Phi^{-1}(1 - \alpha)
\]

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

---

3It is possible (at the expense of the computational time) to incorporate stagewise dependent 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).

4We 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.
**Backward pass of iteration** $i$. For $k = (i - 1)N + 1, \ldots, iN$, we solve for $t = T, j = 1, \ldots, M_T$,

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

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

\[
\begin{cases}
\inf_{x_t, f} \sum_{\ell=1}^{M_t+1} p_{t+1} f_{\ell} \\
x_t \in X_t(x_{t-1}^k, \xi_t), \\
f_{\ell} \geq \langle \beta_{t+1}^m, x_t \rangle, m = 1, \ldots, iN, \ell = 1, \ldots, M_{t+1}.
\end{cases}
\]

For stage $t$ problem above with realization $\xi_{tj}$ of $\xi_t$ (problem (25) for $t = T$ and (26) for $t < T$), denoting by $\lambda_{tj}^k$ the optimal Lagrange multipliers associated to the equality constraints and by $\mu_{1tj}^k \geq 0, \mu_{2tj}^k \geq 0$ the optimal Lagrange multipliers associated with respectively the constraints $y_t(i) \leq \xi_t(i)x_{t-1}(i)$ and $x_t(i) \leq u(i) \sum_{j=1}^{n+1} \xi_t(j)x_{t-1}(j)$, we compute

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

where $e$ 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 if iteration $i$, we solve

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

whose optimal value provides a lower bound $z_i^{\inf}$ on the optimal value $Q_1(x_0)$ of the problem. Given a tolerance $\varepsilon > 0$, the algorithm stops either when $z_i^{\inf} = 0$ and $z_i^{\sup} \leq \varepsilon$ or when

\[
|z_i^{\sup} - z_i^{\inf}| \leq \varepsilon \max(1, |z_i^{\sup}|).
\]

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

5.1.3. **Numerical results.** We compare three methods to solve the portfolio 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 FMLM Level 1 cut selection (denoted by CuSMuDA CS 2 for short). The implementation was done in Matlab run on a laptop
### Table 1. Subset of stocks chosen.

<table>
<thead>
<tr>
<th>Number of stocks</th>
<th>Stocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>n = 4</td>
<td>AAPL (Apple Inc.), XOM (Exxon Mobil Corp.), MSFT (Microsoft Corp.), JNJ (Johnson &amp; Johnson).</td>
</tr>
<tr>
<td>n = 5</td>
<td>AAPL, XOM, MSFT, JNJ, WFC (Wells Fargo).</td>
</tr>
<tr>
<td>n = 6</td>
<td>AAPL, XOM, MSFT, JNJ, WFC, GE (General Electric).</td>
</tr>
</tbody>
</table>

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 $p_{it} = \mathbb{P}(\xi_t = \xi_{ti}) = 1/M$, 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.\(^5\) The daily return (recall that stages are daily) $\xi_t(n + 1)$ 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 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 $z^K_{inf}$ and $z^K_{sup}$ computed at the last iteration $K$ of the algorithms should be close to the optimal value $Q_1(x_0)$ of the problem. If we do not know the optimal value $Q_1(x_0)$, we however check on Figure 2 that the bounds for the 3 algorithms are very close to each other at the last iteration. In Figure 2 we also observe, as it should be, 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 $NM = 200 \times 20 = 4000$ 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), $5(T - 1)MN = 80000$ 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

\(^5\)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/).
Figure 2. Evolution of the upper bounds $z_{i_{\text{sup}}}$ and lower bounds $z_{i_{\text{inf}}}$ along the iterations of the algorithms.
<table>
<thead>
<tr>
<th>$T = 5, n = 4, M = 20$</th>
<th>MuDA</th>
<th>CuSMuDA CS 1</th>
<th>CuSMuDA CS 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8.97</td>
<td>13.45</td>
<td>1.31</td>
</tr>
<tr>
<td>$T = 5, n = 5, M = 20$</td>
<td>17.03</td>
<td>29.41</td>
<td>2.38</td>
</tr>
<tr>
<td>$T = 5, n = 6, M = 20$</td>
<td>45.87</td>
<td>64.22</td>
<td>5.25</td>
</tr>
<tr>
<td>$T = 8, n = 4, M = 10$</td>
<td>7.94</td>
<td>16.27</td>
<td>1.56</td>
</tr>
<tr>
<td>$T = 8, n = 5, M = 10$</td>
<td>43.39</td>
<td>73.98</td>
<td>4.30</td>
</tr>
<tr>
<td>$T = 8, n = 6, M = 10$</td>
<td>63.47</td>
<td>110.55</td>
<td>5.05</td>
</tr>
</tbody>
</table>

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.

selected by the two cut selection strategies.

**Proportion of cuts selected.** An important aspect of the cut selection strategies with direct impact on 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 quicker and can yield an important reduction in CPU time. Such 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. Such 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 less cuts are selected for the first stages, especially for CuSMuDA CS 1. This makes sense since at the last stage, the cuts for functions $Q_T(\cdot, \xi_{Tj})$ 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 first stages old cuts (computed at the first iterations using crude approximations of the recourse functions) will likely 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 $Q_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. Addicionally, we observe on these figures that for each stage $t$, the proportions of cuts selected for functions $Q_t(\cdot, \xi_{tj}), j = 1, \ldots, M$, are quite similar.
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 $R_t(\cdot, \zeta_{ij})$ as a function of $j = 1, \ldots, M$. Left plots: CuSMuDA CS 1, right plots: 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 $\Omega_t(\cdot, \xi_{ij})$ as a function of $j = 1, \ldots, M$. Left plots: CuSMuDA CS 1, right plots: CuSMuDA CS 2.
A 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 last stage. However, a crude implementation of the Multicut Level 1 pseudo-code given in Figure 1 resulted in the elimination of cuts at the last 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 \( C^k_{Tj} \) is, in theory, the highest at \( x^k_{Tj-1} \)) 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 \( \varepsilon_0 (\varepsilon_0 = 10^{-6} \text{ in our experiments}) \) such that the values \( V_1 \) and \( V_2 \) of two cuts \( C_1 \) and \( C_2 \) at a trial point are considered equal if \( |V_2 - V_1| \leq \varepsilon_0 \max(1,|V_1|) \) while \( C_1 \) is above \( C_2 \) 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 FMLM 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 \( \xi_t \) 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 \( Q_t(y_t) = \mathbb{E}_{\xi_t}[\Omega_t(y_t, \xi_t)] \), the stage \( t \) problem is

\[
Q_t(y_t, \xi_t) = \begin{cases} 
\inf \{ c_t(x_t - y_t) + b_t(\xi_t - x_t) + h_t(x_t - \xi_t) + Q_{t+1}(y_t) \} & \text{if } y_{t+1} = x_t - \xi_t, x_t \geq y_t,
\end{cases}
\]

where \( c_t \) is the unit purchasing cost, \( h_t \) is the holding cost, and \( b_t \) the backorder cost. The optimal mean cost is \( Q_1(y_1) \) 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 Multicut Level 1 cut selection (denoted again by CuSMuDA CS 1), and CuSMuDA with FMLM 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:

- \( c_t = 1.5 + \cos\left(\frac{\pi}{T} t\right) \), \( b_t = 2.8 \), \( h_t = 0.2 \), \( M_t = M \) for all \( t = 1, \ldots, T \),
- \( y_1 = 10 \), \( p_t = \frac{1}{M_t} \) for all \( t, i \),
- \( \xi_1 = \overline{\xi}_1 \) and \( (\xi_1, \xi_2, \ldots, \xi_M) \) corresponds to a sample of 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 \( \overline{\xi}_t = 5 + 0.5 t \).
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.

<table>
<thead>
<tr>
<th>$T$ = 5, $M$ = 20</th>
<th>MuDA</th>
<th>CuSMuDA CS 1</th>
<th>CuSMuDA CS 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.57</td>
<td>2.90</td>
<td>0.22</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T$ = 10, $M$ = 20</th>
<th>26.33</th>
<th>43.07</th>
<th>1.68</th>
</tr>
</thead>
<tbody>
<tr>
<td>106.52</td>
<td>178.08</td>
<td>8.07</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T$ = 15, $M$ = 20</th>
<th>158.10</th>
<th>245.24</th>
<th>19.43</th>
</tr>
</thead>
<tbody>
<tr>
<td>189.27</td>
<td>405.58</td>
<td>22.64</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T$ = 20, $M$ = 20</th>
<th>363.86</th>
<th>575.53</th>
<th>56.62</th>
</tr>
</thead>
<tbody>
<tr>
<td>363.86</td>
<td>575.53</td>
<td>56.62</td>
<td></td>
</tr>
</tbody>
</table>

- 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 last 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. As can be seen in Table 3 where the CPU time needed to solve our six instances of inventory management problems is reported, similarly to the experiments on 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 runs of the algorithms 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. We also report on Figures 8 and 9 the evolution of the mean proportions of cuts (over the iterations of the algorithm) selected for $\Omega_t(\cdot, \xi_{tj})$ as a function of $j = 1, \ldots, M$. The conclusions are similar to those of Section 5.1:

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 $\Omega_t(\cdot, \xi_{tj}), j = 1, \ldots, M$, are quite similar.
Figure 6. Evolution of the upper bounds $z^i_{\text{sup}}$ and lower bounds $z^i_{\text{inf}}$ along the iterations of the algorithms.
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 (FMLM 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 FMLM Level 1. Compared to traditional Multicut Decomposition methods, CuSMuDA allowed us to obtain a huge reduction of the computational time on 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 instances 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

\begin{figure}
\centering
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{T_10_M_20.png}
\caption{$T = 10, M = 20$}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{T_15_M_20.png}
\caption{$T = 15, M = 20$}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{T_20_M_20.png}
\caption{$T = 20, M = 20$}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{T_25_M_20.png}
\caption{$T = 25, M = 20$}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{T_30_M_20.png}
\caption{$T = 30, M = 20$}
\end{subfigure}
\caption{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.}
\end{figure}
$T = 10, M = 20$, CuSMuDA CS 1

$T = 10, M = 20$, CuSMuDA CS 2

$T = 15, M = 20$, CuSMuDA CS 1

$T = 15, M = 20$, CuSMuDA CS 2

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

$T = 20, M = 20$, 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 $Q_t(\cdot, \xi_j)$ as a function of $j = 1, \ldots, M$. Left plots: CuSMuDA CS 1, right plots: 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 \(\Omega_t(\cdot, \xi_t)\) as a function of \(j = 1, \ldots, M\). Left plots: CuSMuDA CS 1, right plots: CuSMuDA CS 2.


**APPENDIX**

MICHELLE BANDARRA: SCHOOL OF APPLIED MATHEMATICS FGV/EMAP., 22 250-900 RIO DE JANEIRO, BRAZIL, michelle.bandarra@fgv.br

AND VINCENT GUILLES: SCHOOL OF APPLIED MATHEMATICS FGV/EMAP., 22 250-900 RIO DE JANEIRO, BRAZIL, vincent.guigues@fgv.br
\[ I_{ij}^k = \{k\}, m_{ij}^k = C_{ij}^k(x_{i-1}^k). \]

**Multicut Level 1**

For \( \ell = 1, \ldots, k-1, \)

- If \( C_{ij}^\ell (x_{i-1}^\ell) > m_{ij}^\ell + \varepsilon_0 \max(1, |m_{ij}^\ell|) \)
  \( I_{ij}^\ell = \{k\}, m_{ij}^\ell = C_{ij}^\ell(x_{i-1}^\ell) \)
- Else if \( |C_{ij}^\ell (x_{i-1}^\ell) - m_{ij}^\ell| \leq \varepsilon_0 \max(1, |m_{ij}^\ell|) \)
  \( I_{ij}^\ell = I_{ij}^\ell \cup \{k\} \)

**FMLM Level 1**

- If \( C_{ij}^{\ell+1} (x_{i-1}^{\ell+1}) > m_{ij}^{\ell+1} + \varepsilon_0 \max(1, |m_{ij}^{\ell+1}|) \)
  \( I_{ij}^{\ell+1} = \{\ell+1\}, m_{ij}^{\ell+1} = C_{ij}^{\ell+1}(x_{i-1}^{\ell+1}) \)

**Figure 10.** Pseudo-codes for selecting the cuts using Multicut Level 1 and FMLM Level 1 taking into account approximation errors.