# A Multicut Approach to Compute Upper Bounds for Risk-Averse SDDP

Joaquim Dias Garcia* Iago Leal de Freitas†
Raphael Chabar‡ Mario V. F. Pereira§

July 25, 2023

**Abstract**

Stochastic Dual Dynamic Programming (SDDP) is a widely used and fundamental algorithm for solving multistage stochastic optimization problems. Although SDDP has been frequently applied to solve risk-averse models with the Conditional Value-at-Risk (CVaR), it is known that the estimation of upper bounds is a methodological challenge, and many methods are computationally intensive. In practice, this leaves most SDDP implementations without a practical and clear stopping criterion. In this paper, we propose using the information already contained in a multicut formulation of SDDP to solve this problem with a simple and computationally efficient methodology.

The multicut version of SDDP, in contrast with the typical average cut, preserves the information about which scenarios give rise to the worst costs, thus contributing to the CVaR value. We use this fact to modify the standard sampling method on the forward step so the average of multiple paths approximates the nested CVaR cost. We highlight that minimal changes are required in the SDDP algorithm and there is no additional computational burden for a fixed number of iterations.

We present multiple case studies to empirically demonstrate the effectiveness of the method. First, we use a small hydrothermal dispatch test case, in which we can write the deterministic equivalent of the entire scenario tree to show that the method perfectly computes the correct objective values. Then, we present results using a

*PSR, Brazil. joaquim@psr-inc.com
†Brazil. iago.lealf@gmail.com
‡PSR, Brazil. chabar@psr-inc.com
§PSR, Brazil. mario@psr-inc.com

standard approximation of the Brazilian operation problem and a real hydrothermal dispatch case based on data from Colombia. Our numerical experiments showed that this method consistently calculates upper bounds higher than lower bounds for those risk-averse problems and that lower bounds are improved thanks to the better exploration of the scenarios tree.

# 1 Introduction

There are many problems that can be modeled using the framework of multistage stochastic programming, especially in the energy sector. For treating large problems, the main algorithm in use today is called Stochastic Dual Dynamic Programming (SDDP), introduced in [9]. It is a clever variation of traditional stochastic dynamic programming that samples paths (or trajectories) in a extremely large scenario tree and at each iteration uses linear programming duality to approximate the future cost functions, also known as cost-to-go functions, for each time step. When the problems considered are all linear, the algorithm returns both a lower bound and an upper bound that can be used to guarantee convergence to the optimal solution.

Traditionally, stochastic dynamic programming represents the relations between stages through a recursive equation known as the Bellman equation, where we optimize the expected value among all possible realizations of the random process for the next stage:

$$
\begin{aligned}
Q_t(x_{t-1}, \xi_t) = & \\
\min_{x_t, u_t} \quad & C_t(x_{t-1}, \xi_t, u_t) + \mathbb{E}_{\xi_{t+1}}[Q_{t+1}(x_t, \xi_{t+1})] \\
\text{s.t.} \quad & x_t = D_t(x_{t-1}, \xi_t, u_t) \\
& u_t \in U_t(x_{t-1}, \xi_t)
\end{aligned}
\tag{1}
$$

where $x_{t-1}$ is the initial state and $\xi_t$ is the current value of the random variable, both considered input coefficients for the optimization problem. The decision variables are the state at the end of the stage $x_t$ and the control (or action) $u_t$. $U_t$ is the set of feasible actions, $D_t$ is the state transition dynamics, and $C_t$ is the immediate cost. For the final stage $T$, we set $Q_T(\cdot, \cdot) = 0$. This is called the *risk-neutral* formulation of a multistage stochastic program.

In many practical problems, it is desirable to protect oneself against the worst outcomes of random processes. In this setting, called *risk-averse*, the expected value, $\mathbb{E}_{\xi_{t+1}}[\cdot]$, is replaced by a risk measure, $\rho_{\xi_{t+1}}[\cdot]$, typically a coherent risk measure [1]. The most common of these risk measures is the *Conditional Value-at-Risk* (CVaR) [5, 12], which is roughly equivalent to taking the average conditioned to be

above a given quantile. The CVaR is also known as Average Value-at-Risk, AVaR, and Expected Shortfall, ES. Unfortunately, in the presence of risk aversion, we lose the capacity to properly estimate an upper bound for the SDDP [10].

There are many recent advancements in the literature that attempt to address the upper-bound problem through varying methods, such as statistical samplings based on bad outcomes [6, 8], inner approximations of the Bellman function [11], or dual variants of SDDP [3, 7].

In this work, we present a method for estimating the upper bound of risk-averse SDDP by modifying the sampling of scenarios in the forward pass through the weights endogenously defined by the optimal solution of the CVaR formulation of the future cost function. This method necessarily uses a multicut approximation for the future cost function and only requires the solution of the subproblem of the current stage to sample the random scenario of the next stage in the forward pass. The method does not require any additional modification on SDDP's backward step nor keeping annotations of sampled outcomes throughout iterations, which makes the method especially simple to implement.

In Section 2, we will detail the methodology briefly described above. Section 3 contains case studies with three test systems of hydrothermal dispatch: first, a small case that can be solved by a deterministic equivalent, then a standard academic representation of the Brazilian system, and finally, a realistic representation of the Colombian system. Section 4 outlines conclusions and future work.

# 2 Methodology

Throughout this paper, we consider the following definition of $\texttt{CVaR}_\alpha[Y]$ of a random variable $Y$ with distribution $F_Y(y) = P(Y \leq y)$:

$$\texttt{CVaR}_\alpha[Y] = \mathbb{E}[Y | Y \geq \texttt{VaR}_\alpha[y]] = \min_b \left\{ b + \frac{1}{1-\alpha}\mathbb{E}[(Y-b)^+] \right\}$$

where

$$\texttt{VaR}_\alpha[Y] = \min_r \{F_Y(r) \geq \alpha\}$$

Also, we consider risk-averse SDDP using a convex combination between the expected value and CVaR as a risk measure:

$$\rho[Y] = (1-\lambda)\mathbb{E}[Y] + \lambda \texttt{CVaR}_\alpha[Y]$$

where $\lambda, \alpha \in [0, 1]$ are fixed parameters. Special cases are $\lambda = 0$ in which only the expected value is considered, $\lambda =, 1$ in which only the CVaR is considered, $\alpha = 0$ in which the CVaR degenerates to the

expected value, and $\alpha = 1$ in which the CVaR represents the worst-case scenario as in a robust optimization framework.

This is a coherent risk measure [1] and can be represented as the optimal solution to a linear program [5, 12]. For the case of a discrete random variable $Y$ with possible values $\{y^l\}_{l=1}^L$ all with probability $\frac{1}{L}$, we can write:

$$\begin{aligned} \rho[Y] = \min_{\delta^l, z} \quad & \tfrac{1-\lambda}{L} \textstyle\sum_{l=1}^L y^l + \lambda z + \tfrac{\lambda}{(1-\alpha)L} \sum_{l=1}^L \delta^l \\ \text{s.t.} \quad & \delta^l \geq y^l - z, \quad \forall l = 1, ..., L \\ & \delta^l \geq 0, \quad \forall l = 1, ..., L \end{aligned} \tag{2}$$

We consider a multi-stage stochastic program whose cost function satisfies the recursion:

$$\begin{aligned} Q_t(x_{t-1}, \xi_t) = & \\ \min_{x_t, u_t} \quad & C_t(x_{t-1}, \xi_t, u_t) + \rho_{\xi_{t+1}}[Q_{t+1}(x_t, \xi_{t+1})] \\ \text{s.t.} \quad & x_t = D_t(x_{t-1}, \xi_t, u_t) \\ & u_t \in U_t(x_{t-1}, \xi_t) \end{aligned} \tag{3}$$

Which is the risk-averse version of the model (1).

Combining (2) and (3), and assuming that the possible outcomes of $\xi_{t+1}$ are $\{\xi_{t+1}^l\}_{l=1}^L$, all with probability $\frac{1}{L}$, we get the nested CVaR multicut reformulation:

$$\begin{aligned} Q_t(x_{t-1}, \xi_t) = & \\ \min_{x_t, u_t, \delta_t^l, z_t, \beta_t^l} \quad & C_t(x_{t-1}, \xi_t, u_t) + \tfrac{1-\lambda}{L} \textstyle\sum_{l=1}^L \beta_t^l + \\ & \qquad \lambda z_t + \tfrac{\lambda}{(1-\alpha)L} \textstyle\sum_{l=1}^L \delta_t^l \\ \text{s.t.} \quad & x_t = D_t(x_{t-1}, \xi_t, u_t) \\ & u_t \in U_t(x_{t-1}, \xi_t) \\ & \beta_t^l \geq Q_{t+1}(x_t, \xi_{t+1}^l), \quad \forall l = 1, ..., L \\ & \delta_t^l \geq \beta_t^l - z_t, \quad \forall l = 1, ..., L \\ & \delta_t^l \geq 0, \quad \forall l = 1, ..., L \end{aligned} \tag{4}$$

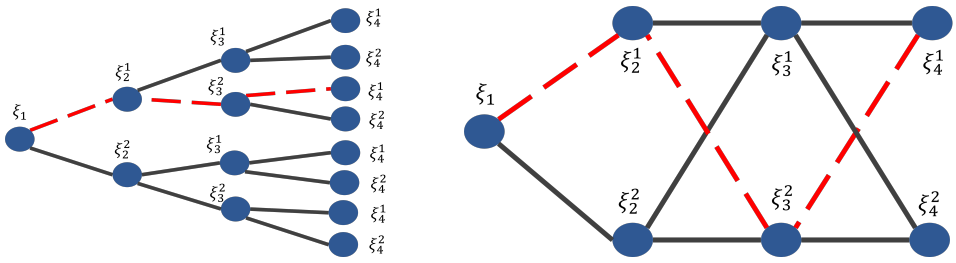
Note that there is one explicit future cost function for each possible scenario of the next stage $\{\xi_{t+1}^l\}_{l=1}^L$.

## 2.1 Stochastic dual dynamic programming

Multi-stage stochastic programming consists of a sequence of decision processes, each one depending on the preceding and on the realization of random variables. In theory, the random process can be continuous so that infinitely many outcomes are possible in each stage. However, for the application of the SDDP algorithm, it is common to consider a discretized version of the problem, in which all possible realizations of

a random process can be presented in the form of a scenario tree. This discretization step is the first application of Monte Carlo sampling required by SDDP practitioners.

To apply the SDDP algorithm, we make an additional requirement: random variables must be *stagewise independent*. That is, the uncertainty $\xi_t$ is independent of all the previous stages 1 to $t-1$. The root node is the deterministic first-stage variable $\xi_1$ and for each node at stage $t$, there are the same possible realizations of the random process in the next stage, $t+1$: $\{\xi_{t+1}^l\}_{l=1}^L$ In this particular case, the scenario tree degenerates into the so-called *recombining scenario tree*, which is not an actual tree, but a graph that compactly represents a scenario tree with stagewise independency. Figure 1 presents a stagewise independent scenario tree and the corresponding recombining scenario tree, a path from the first to the last stage is highlighted in red as an example.




(a) Original tree (b) Recombining tree

Figure 1: Equivalence between a stagewise independent scenario tree and a recombining scenario tree. The dashed red trajectory is equivalent in both representations.

The $Q$ functions are not known, in fact, they are what is being approximated. In SDDP, these future cost functions are approximated as the maximum of cutting planes. We name these approximations as $\tilde{Q}(x)$. In more detail, we define:

$$\tilde{Q}(x) = \max\{\pi_i^\top (x - \bar{x}_i) + q_i | i \in \mathcal{C}\}, \tag{5}$$

where the triplets $(\pi_i, \bar{x}_i, q_i)$ are the cut coefficients and $\mathcal{C}$ is the set of indices of cuts. At each iteration of the SDDP algorithm, more cuts will be added, improving the approximation. By construction $\tilde{Q}(x) \leq Q(x), \forall x$, hence an approximation from below. In the first appearances of SDDP in the literature $\rho[Q]$ was approximated as a single function in the so-called single-cut version. Here, we apply the multicut representation, in which $\rho[Q]$ is first explicitly expanded as

a combination of $L$ functions, one of each of the possible uncertainty realizations in the next stage, as described above, and then, each of these functions is approximated individually by cutting planes. Applying the multicut approximation to (4) leads to:

$$
\begin{aligned}
\tilde{Q}_t(x_{t-1}, \xi_t) = & & (6)\\
\min_{x_t, u_t, \delta_t^l, z_t, \beta_t^l} \quad & C_t(x_{t-1}, \xi_t, u_t) + \tfrac{1-\lambda}{L} \textstyle\sum_{l=1}^{L} \beta_t^l + \\
& \qquad \lambda z_t + \tfrac{\lambda}{(1-\alpha)L} \textstyle\sum_{l=1}^{L} \delta_t^l \\
\text{s.t.} \quad x_t &= D_t(z_t, \xi_t, u_t) \\
u_t &\in U_t(z_t, \xi_t) \\
z_t &= x_{t-1} \qquad : \ \pi_{t-1} \\
\beta_t^l &\geq \pi_{t,l,i}^{\top}(x_t - \bar{x}_{t,l,i}) - q_{t,l,i}, \forall i \in \mathcal{C}_t^l, l = 1, ..., L \\
\delta_t^l &\geq \beta_t^l - z_t, \quad \forall l = 1, ..., L \\
\delta_t^l &\geq 0, \quad \forall l = 1, ..., L
\end{aligned}
$$

Where we added the equation $z_t = x_{t-1}$ that simplifies the construction of the cut, which will only need the dual of the constraint, $\pi_{t-1}$, the objective value $q_{t-1}$ and the input state $x_{t-1}$. We denote the immediate cost, the evaluation of $C_t$ in the optimal solution, as $c_t$.

This formulation leads to larger optimization problems, but more detailed information is passed along. Also, the multicut approach is expected to be especially effective if the number of scenarios is significantly smaller than the number of constraints in the subproblem [2]. Moreover, the explicit representation of the FCF of each node in the next stage will be instrumental to the next developments.

We can summarize a standard version of the SDDP algorithm as follows. Advanced versions of the algorithm, including Markov representations of uncertainties, infinite horizons, hazard-decision formulations, non-homogeneous trees, and so on, are possible, see [4, 14].

We start with the *forward pass*. Following its name, the forward pass starts from the first node, with initial state $x_0$ and (deterministic) uncertainty realization $\xi_1$, and problems are solved in the forward direction of time. For each stage, $t$, only one node problem is solved given its past, the current realization of the random variable, $\xi_t^l$, and an uncertain view of the future. This uncertain view must ensure causality and non-anticipativity, thus, the future is not known and the decision must be taken assuming that all nodes in the next stage are equally probable. Hence, problem (4) is solved considering the future cost functions for each $l \in 1, ..., L$. After such a problem is solved, we must sample the next realization of the uncertainty $\xi_{t+1}^l$ so that the next problem can be solved until the process ends at stage $T$. A uniform sample is performed in the traditional SDDP. The sampling

step is the second application of Monte Carlo and it will be the key to our proposed method.

The average cost of multiple of these solution paths is used to estimate upper bounds. The solution to the problem in the first stage leads to a deterministic lower bound, as opposed to the estimated statistical upper bound. Many stopping rules have been proposed after the first one by [9]. In this work, any rule based on the deterministic lower bound and the estimated statistical upper bound can be considered.

After one or multiple scenario path is solved, we move to the *backward pass*. A scenario path, $s$, solution is fully characterized by the set of tuples $\{(x_{t-1}^s, \xi_t^s)\}_{t\in\{1,...,T\}}$ of optimized states and sampled uncertainties. As the name hints, the backward pass will solve problems in the reverse direction of time. Starting at stage $T$, all nodes associated with this stage will be solved, but all are conditioned to the input coming from the solutions path: $(x_{T-1}^s, \xi_T^s)$. The solutions of each node $l \in L$ will lead to a cut, $i$, containing $(\pi_{t,l,i}, q_{t,l,i})$, that will be stored in a cut pool indexed by $\mathcal{C}_t^l$ and associated with that node. The stagewise independency combined with the cut construction (a function of state and uncertainty) allows cut sharing so that the cuts generated by the current path are valid for all other possible paths. The process continues up to the second stage, which generates cuts for the first stage.

We summarize the SDDP algorithm in Algorithm 1. We present a batched version of the algorithm. Instead of creating a single solutions path, $s$, per iteration, we create $S$ scenario paths. This is important because we measure iterations as batches of scenarios. Also, this highlights the actual implementation used that solves each scenario path in parallel. We also fix a maximum number of iterations, $K$, so that the algorithm stops if convergence is too slow. For further details, see [9].

## 2.2 Risk adjusted scenario sampling

The unique feature of the multicut formulations is that it uses a different variable $\beta^l$ for each scenario of the next stage. Therefore the linear programming formulation of the CVaR provides us with enough information to know which scenarios are actually considered when computing the `CVaR`.

We know from the literature [12] that at the optimal of equation (6), the decision variable $z_t$ will be equal to the Value-at-Risk of the variables $\beta_t^l$ for $l \in 1, ..., L$. Furthermore, with the exception of degenerate cases, the decision variables $\delta_t^l$ are non-zero only if the

**Algorithm 1** Multicut Batched Traditional SDDP

**for** $k = 1, ..., K$ **do**
    (Forward Step)
    **for** $t = 1, ..., T$ **do**
        **for** $s = 1, ..., S$ **do**
            Sample $\xi_t^s$ from $\{\xi_t^l\}_{l=1}^L$
            Solve (4) to obtain the state $x_t^s$ and immediate cost $c_t^s$
        **end for**
    **end for**
    (Convergence Check)
    Estimate statistical upper bound from $\{c_t^s\}_{s=1,...,S,t=1,...,T}$
    Store deterministic lower bound $c_1$
    **if** Stopping condition is met **then**
        Stop
    **end if**
    (Backward Step)
    **for** $t = T, ..., 2$ **do**
        **for** $s = 1, ..., S$ **do**
            **for** $l = 1, ..., L$ **do**
                Solve (4) to obtain the cut $i$ represented by $\pi_{t,l,i}, \bar{x}_{t,l,i}, q_{t,l,i}$
                Improve the representation of (5)
            **end for**
        **end for**
    **end for**
**end for**

scenario $l$ is used to compute the `CVaR` value, they are equal to how much $\beta_t^l$ is above the `VaR`. Since the $\beta_t^l$ represent the cost-to-go for each opening, we may view their weight on the objective as representing the contribution of each scenario for calculating the cost.

To make the weights explicit, we start ordering the scenarios by the value of $\beta_t^l$ such that

$$\beta_t^{(1)} \leq ... \leq \beta_t^{(l)} \leq ... \leq \beta_t^{(L)},$$

and call $\nu$ the last opening such that $\delta_t^{(l)} = 0$ (that is, the cost-to-go is above the Value-at-Risk). From the previous discussion, we know that $\beta_t^\nu = z_t$. Then, by ordering the scenario we get:

$$\begin{aligned}
\rho_{\xi_{t+1}}[Q_{t+1}(\cdot)] &= \frac{1-\lambda}{L}\sum_{l=1}^{L}\beta_t^l + \lambda z_t + \frac{\lambda}{(1-\alpha)L}\sum_{l=1}^{L}\delta_t^l \\
&= \frac{1-\lambda}{L}\sum_{l=1}^{L}\beta_t^{(l)} + \lambda z_t + \frac{\lambda}{(1-\alpha)L}\sum_{l=\nu}^{L}(\beta_t^{(l)} - z_t) \\
&= \frac{1-\lambda}{L}\sum_{l=1}^{L}\beta_t^{(l)} + \lambda\beta_t^\nu + \frac{\lambda}{(1-\alpha)L}\sum_{l=\nu+1}^{L}(\beta_t^{(l)} - \beta_t^\nu) \\
&= \sum_{l=1}^{\nu-1}\frac{1-\lambda}{L}\beta_t^{(l)} + \left(\frac{1-\lambda}{L} + \lambda - \frac{\lambda(L-\nu)}{(1-\alpha)L}\right)\beta_t^\nu \\
&\qquad + \sum_{l=\nu+1}^{L}\left(\frac{1-\lambda}{L} + \frac{\lambda}{(1-\alpha)L}\right)\beta_t^{(l)}
\end{aligned}$$

From the above, we note that computing the risk measure is the same as taking a linear combination of the cost-to-go at all openings with weights

$$w_t^l = \begin{cases} \frac{1-\lambda}{L} + \frac{\lambda}{(1-\alpha)L}, & \beta_t^l > z_t \\ \frac{1-\lambda}{L} + \lambda - \frac{\lambda(L-\nu)}{(1-\alpha)L}, & \beta_t^l = z_t \\ \frac{1-\lambda}{L}, & \beta_t^l < z_t. \end{cases}$$

Notice also that, for any stage $t$, $\sum_{l=1}^{L} w_t^l = 1$. Thus, the weights represent a probability distribution on the openings such that calculating the risk measure is the same as calculating the average with respect to the $w_t^l$.

Given the above-defined weights, it is only necessary to modify line 5 of Algorithm 1 to obtain a version of SDDP that is able to compute the upper bounds of the CVaR-based risk-averse multistage stochastic program. Clearly, there is no performance degradation on

the computational cost of each iteration and the changes required are minimal: obtain the weights and change the sampling method.

Our implementation extends SDDP by using the optimal decision variables $\beta_t^l, \delta_t^l, z_t$ calculated at the backward step to discover the weights $w_t^l$ of choosing opening $l$ at stage $t$. This is then applied on the forward step to sample the scenarios according to this new probability.

This method can be interpreted as an endogenous importance sampling scheme to compute the weight distributions used in the forward pass to obtain scenario paths. As the SDDP progressively improves the approximation of the future cost function, the weights obtained at each iteration improve until they arrive at a distribution that leads to the upper bound of the problem.

In the special case of pure CVaR as a risk measure ($\lambda = 1$), the weights on non-CVaR contributing scenarios go to zero, which might lead to issues in exploring the scenario tree. This issue might affect the convergence. In this case and in similar cases where $\lambda$ is close to zero, one possible alternative is to proceed as follows: in even iteration ($k$ is even), use the risk-adjusted sampling approach proposed here, in odd iteration ($k$ is odd), use standard sampling, but do not compute upper bound nor check convergence. Of course, other alternating schemes might be used to improve the exploration of the scenario tree.

We finish with an illustration of the proposed method (with $0 < \lambda < 1$). Figure 2 presents a depiction of the forward pass right after solving the subproblem of the first stage (marked in yellow). At this point, we know which scenarios contribute to the CVaR (line in red) and which do not (lines in green). Consequently, we can compute the weight that will be used to sample the scenario in stage 2, which is depicted in the plot of weights as a function of the sample that has more mass in the sample associated with the scenario active for CVaR.

# 3 Numerical experiments

In this section, we compare the variants of SDDP using traditional sampling, which leads to a naive upper bound known to be incorrect, and with our modified sampling method based on multicut formulation, which will lead to better upper bounds. We apply the methods to the hydrothermal dispatch model that will be detailed next. We consider 3 case studies: first, a simple problem in which we can evaluate the entire tree, second, a classical model for the Brazilian interconnected power system found in [13], and, third, a case with real data from the Colombian system. Data for these cases are summarized in Table 1. In all cases we used the `CVaR` parameters $\alpha = 0.5$ and $\lambda = 0.5$.

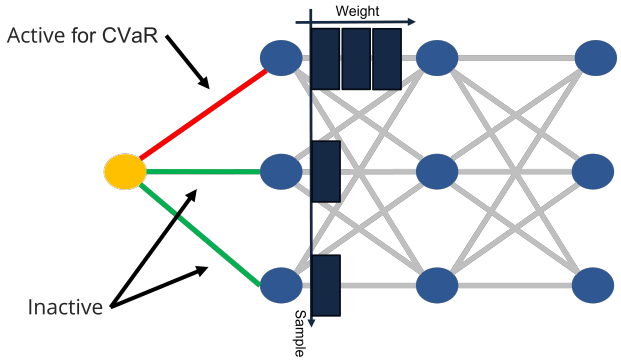



Figure 2: Illustration of sampling method for upper bound estimation of risk-averse SDDP.

## 3.1 Hydrothermal dispatch model

In the following model, we have the sets: $J^L$ is the set of lags in an autoregressive model, $J^G$ is the set of thermal plants, $J^R$ is the set of renewable plants, $J^H$ is the set of hydro plants, $J_n^F$ is the set of other nodes or areas connected to node $n$, $J_j^U$ is the set of hydro plants upstream of plant $j$. An additional $n$ as a subscript in $J^G$, $J^R$, and $J^H$ restricts the generators to the ones the node $n$. The coefficients of the model are defined as: $\tilde{D}_n(\xi)$ is the demand of subsystem (node) $n$ and scenario $\xi$, $F_{n,m}$ is the maximum energy flow between nodes $n$ and $m$, $C_j$ is the operating cost of thermal $j$, $G_j$ is the maximum generation of thermal $j$, $\tilde{R}_j(\xi)$ is the maximum generation of renewable $k$ at scenario $\xi$ $V_j$ is the maximum storage of hydro $j$, $U_j$ is the maximum flow through the turbine of hydro $j$, $p_j$ is the production coefficient of hydro $j$, $\phi_{j,k}$ is the inflow auto-regressive coefficient of hydro $j$ and lag $k$, $\tilde{\varepsilon}_i^t(\xi)$ is the inflow noise coefficient of hydro $j$ and scenario $\xi$.

Table 1: Study parameters for numeric experiments.

| Parameter | Full Tree | Brazil | Colombia |
|---|---|---|---|
| Hydro Reservoirs | 2 | 4 | 38 |
| Hydro Plants | 2 | 4 | 136 |
| Thermal Plants | 3 | 95 | 202 |
| Renewable Plants | 0 | 0 | 12 |
| Stages | 10 or 7 | 120 | 60 |
| Max autoregressive size | 0 | 0 | 6 |

Finally, the decision variables of the optimization problem are: $g_j$, the generation of thermal $j$; $r_j$, the generation of renewable $j$; $f_{n,m}$, the flow of energy leaving node $n$ and reaching node $m$; $u_j$, the turbine flow at hydro $j$; $z_j$, the spill flow at hydro $j$; $v_j^t$, the storage at hydro $j$, at the beginning of stage $t$, and at the end of stage $t-1$; $a_j^t$, the inflow at hydro $j$, stage $t$. $a^{[t]}$ stands for the vector of all inflows before stage $t$.

$$Q_t(\{v_j^t, a_j^{[t-1]}\}_{j\in J^H}, \xi_t) =$$
$$\min_{a,b} \sum_{j\in J^G} C_j g_j + \rho_{\xi^{t+1}} \left[ Q_{t+1}(\{v_j^t, a_j^{[t-1]}\}_{j\in J^H}, \xi_{t+1}) \right] \tag{7}$$
$$s.t.$$
$$\sum_{j\in J_n^G} g_j + \sum_{j\in J_n^H} p_j u_j + \sum_{j\in J_n^R} r_j + \sum_{(n,m)\in J_n^F} f_{n,m} - f_{m,n} = \tilde{D}_n(\xi_t) \tag{8}$$
$$v_j^{t+1} = v_j^t - u_j - z_j + \sum_{n\in J_j^U} (u_n + z_n) + a_j^t, \quad \forall j \in J^H \tag{9}$$
$$0 \le v_j \le V_j, \quad 0 \le u_j \le U_j, \quad 0 \le z_j, \quad \forall j \in J^H \tag{10}$$
$$0 \le g_j \le G_j, \quad j \in J^G \tag{11}$$
$$0 \le r_j \le \tilde{R}_j(\xi_t), \quad \forall j \in J^R \tag{12}$$
$$0 \le f_{n,m} \le F_{n,m}, \quad \forall (n,m) \in J^F \tag{13}$$
$$a_j^t = \sum_{l=1}^{L} \phi_{j,l} a_j^{t-l} + \tilde{\varepsilon}_j(\xi_t), \quad \forall j \in J^H \tag{14}$$

In the above model, (7) is the objective function defined as the sum of the immediate cost (thermal generation cost) plus the future cost. (8) is the energy balance equation stating that the sum of energy generated in a node, plus the incoming energy minus the outgoing energy, must match the demand, we assume there is enough thermal energy to match the demand in all stages and scenarios. (9) is the water mass balance stating that the reservoir level at the end of stage $t$ must match the initial content minus the outflow (spilled and through turbine) plus the outflow of upstream plants plus the lateral inflow. (10)-(13) define physical bounds on variables. (14) is the autoregressive equation of the inflow process that allows representing some non-stagewise independent processes in SDDP.

## 3.2 Full tree simple example

We first consider, as a simple example, a problem small enough that we can compute its true optimal value by traversing the full decision

tree. Our purpose is to view it as a sanity check to see that our upper bounds are not only above the lower bounds but indeed above the true value.

We begin by considering a case with 10 stages, 2 scenarios per stage and, consequently, 512 possible scenario paths in the tree, since the decision tree has exactly $\text{scenario}^{\text{stages}-1}$ possible paths. We also compare the methods in the same case but but altering the stages, scenarios and paths in the tree to, respectively, 7, 3 and 729. We show the results in table 2.

As one can see, the traditional sampling estimates a naive upper bound that is consistently below the real cost calculated using the deterministic equivalent. Meanwhile, the multicut-based sampling estimates approximately the same cost as the full tree deterministic equivalent. In the 2 scenarios cases, it is slightly above and in the 3 scenarios case it is the same value. This may happen because of round-off errors.

Table 2: Total cost calculated via different methods.

| Method | 2 scenarios | 3 scenarios |
|---|---|---|
| Deterministic Equivalent | 685.4 | 462.4 |
| Traditional Naive Upper Bound | 636.4 | 444.7 |
| Multicut-based Upper Bound | 685.8 | 462.4 |

## 3.3 Brazil

We compare the methods in the case study of [13] for the Brazilian interconnected power system operation planning. The convergence charts for each method are shown in figure 3.

We see how the multicut-based sampling estimates for each iteration an upper bound above the lower bound while the naive upper bound (known to be incorrect) computed with the traditional sampling does not. It is also very interesting to notice that there is a meaningful increase in the calculated lower bound. We attribute the improvement in the lower bound to an exploration of the scenario tree that is better aligned with the nested CVaR objective.

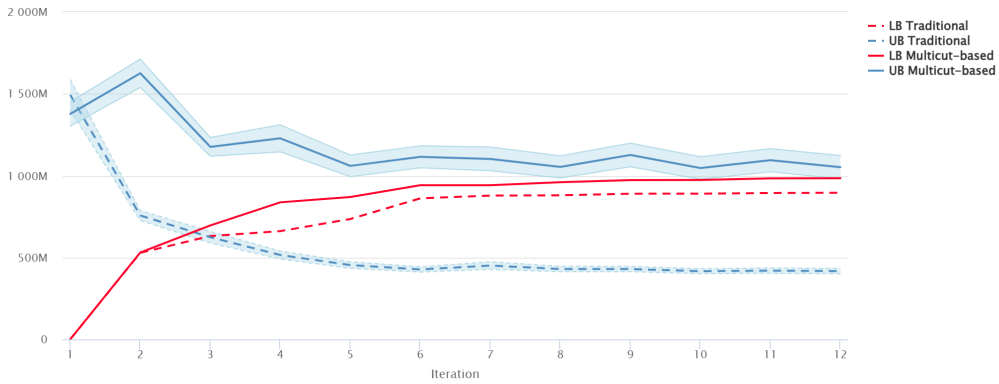



Figure 3: Convergence chart for the Brazilian system.

## 3.4 Colombia

Finally, we compare both sampling schemes for a case study with real data from Colombia, including renewable plants and an autoregressive model for inflows. It uses an interconnected network, and there is a non-trivial hydro topology, as opposed to the other case studies in which hydro plants are in parallel. The parameters are summarized in table 1.

Here we see the same results as before, the multicut-based sampling estimates much better upper bounds and also shows an increase in the value of the lower bound, in comparison to the traditional sampling method. We note here that the upper bound in the first iteration is below the lower bound of the second iteration. We attribute this to the fact that, in the first iteration of SDDP, little is known about the future cost functions, hence, the weights extracted from the optimal solution of the CVaR linear programming formulation might be far from the optimal ones.

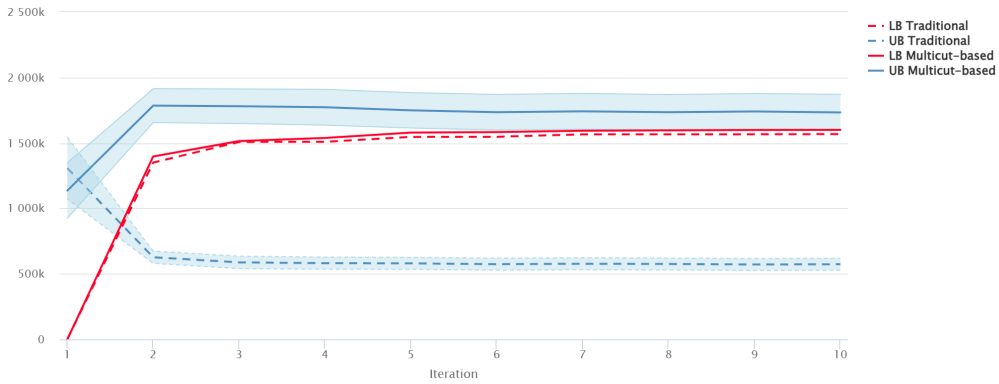



Figure 4: Convergence chart for the Colombian system.

# 4 Conclusions

In this work, we presented a simple strategy to estimate the upper bound of a risk-averse multistage stochastic program solved with SDDP. The methodology relies on the multicut reformulation to obtain weights used to sample scenarios of the next stage in the forward step of SDDP. Such weights appear endogenously in the optimal solution of the CVaR linear formulation. This methodology can be applied in the traditional convergence check of SDDP first proposed by [9] and by other sample-based convergence criteria.

The simplicity of the methodology allows it to be easily implemented in existing SDDP code without major work. Moreover, since the only change in the algorithm is the sampling in the forward pass, there is no performance degradation in each iteration. Therefore, the methodology is computationally efficient.

We empirically demonstrated the effectiveness of this methodology by solving the full tree deterministic equivalent of a small example case and obtaining the same solution with the modified SDDP. Moreover, we solved large hydrothermal dispatch problems: a standard academic version of the Brazilian system and a realistic representation of the Colombian system. The solution of the latter two cases showed that the upper bounds behave much better than the naive approach and that the lower bounds were consistently improved. We attribute the improved lower bounds to a better exploration of the scenario tree.

One possible drawback is that the methodology might be optimistic. The optimal resampling weights are only known when the Bellman functions are well approximated. Otherwise, the sampling weights will be optimistic.

Finally, we highlight that there is a possible variant of this scheme

for the single-cut case. The single-cut version of SDDP will not include the CVaR explicit linear formulation and hence, will not be able to obtain the weights directly from the solutions of the current linear program. However, there is some information that could be cached while computing the single cuts, which is the weight used to average the cuts coming from each scenario. If we cache this information, then after solving a subproblem combine the average of these weights according to the active cuts in the optimal solution, we can obtain weights to be used in the forward pass. The main drawback of this version is that old cuts might have very poor weights associated with them, as they were obtained early in the algorithm. Moreover, there is a need to cache more information associated to cuts.

# References


[1] Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, Jul 1999.

[2] John R Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[3] Bernardo Freitas Paulo da Costa and Vincent Leclère. Dual SDDP for risk-averse multistage stochastic programs, 2021.

[4] Oscar Dowson. The policy graph decomposition of multistage stochastic programming problems. *Networks*, 76(1):3–23, 2020.

[5] Csaba I. Fábián. Handling CVaR objectives and constraints in two-stage stochastic models. *European Journal of Operational Research*, 191(3):888–911, Dec 2008.

[6] Vincent Guigues, Alexander Shapiro, and Yi Cheng. Risk-averse stochastic optimal control: an efficiently computable statistical upper bound, 2021.

[7] Vincent Leclère, Pierre Carpentier, Jean-Philippe Chancelier, Arnaud Lenoir, and François Pacaud. Exact converging bounds for stochastic dual dynamic programming via fenchel duality. *SIAM Journal on Optimization*, 30(2):1223–1250, Jan 2020.

[8] Rui Peng Liu and Alexander Shapiro. Risk neutral reformulation approach to risk averse stochastic programming. *European Journal of Operational Research*, 286(1):21–31, Oct 2020.

[9] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(1-3):359–375, May 1991.

[10] A.B. Philpott and V.L. de Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470–483, Apr 2012.

[11] Andy Philpott, Vitor de Matos, and Erlon Finardi. On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970, Aug 2013.

[12] R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *The Journal of Risk*, 2(3):21–41, 2000.

[13] Alexander Shapiro, Wajdi Tekaya, Joari Paulo da Costa, and Murilo Pereira Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.

[14] Alexandre Street, Davi Valladao, André Lawson, and Alexandre Velloso. Assessing the cost of the hazard-decision simplification in multistage stochastic hydrothermal scheduling. *Applied Energy*, 280:115939, 2020.