Towards multi-timescale energy provisioning using Stochastic Dual Dynamic Programming

Rodrigo Porteiro
Universidad ORT Uruguay
rporteiro@uni.ort.edu.uy

Andres Ferragut
Universidad ORT Uruguay
ferragut@ort.edu.uy

Fernando Paganini
Universidad ORT Uruguay
paganini@ort.edu.uy

***Abstract*—The operation of the electric grid in systems with a large hydroelectric component is often cast as a dynamic programming problem, in which state variables are reservoir levels. To avoid the curse of dimensionality in discretizing such states, the SDDP technique has been successfully applied. However, new demands are being placed on the optimization, with the penetration of renewable sources of faster variability, and the possible incorporation of shorter term energy storage.**

In this paper, we present preliminary work on extending the SDDP framework to such two time-scale problems. We apply the method to a stylized model of the Uruguayan system, relying on new open source implementations of SDDP to carry out the computations. Our results indicate that the method remains tractable despite the increased problem dimension.

I. Introduction

The economic dispatch of electrical generation to cover customer demand is a classical concern of vertically integrated utilities, or more recently of system operators in charge of energy pools. The problem has too many facets to be summarized here; see e.g. [1], [2] for extensive treatments.

In systems with large hydroelectric dams, dispatch decisions become coupled over very long time-scales (weeks, months) since they depend on the availability of water in reservoirs, itself dependent on slow patterns of rainfall. The natural setting is then of Dynamic Programming [3], where the state vector includes (at least) a variable for each reservoir level. Computing the value function for a discretization of these state variables suffers from the curse of dimensionality. In [4], Pereira and Pinto introduced Stochastic Dual Dynamic Programming (SDDP). an approximation strategy which has led to successful industrial dispatch tools [5]. More recently, this method has received wider academic attention [6], [7], and new open source implementations have become available [8]. The main elements of SDDP are summarized in Section II.

Dispatching tools are always under pressure to incorporate more features and details. In particular we highlight the changes brought about by the large scale incorporation of renewable sources like wind and solar power, which are non-dispatchable and present variability at very short time-scales. Their direct incorporation in dynamic programming would imply reducing the time step to the order of an hour, coexisting with the long horizons of the hydro component, a very high dimensional problem. Moreover, if short term storage devices (e.g. batteries) are introduced in the grid to mitigate this variability, their extra state variables worsen the curse of dimensionality. Reaching an adequate compromise between model fidelity and computational cost is a non-trivial task.

In this paper we carry out a preliminary investigation of this two-time scale problem, within the SDDP framework. As described in Section III, we will maintain state variables only for the slower time step (in this case one day), and handle the hourly balance with additional linear constraints within each stage. This implies a larger subproblem for each step of SDDP, but within the realm of linear programming it remains tractable. In Section IV we apply this decomposition to the Uruguayan system. We use a Markov chain model of a discrete "hydrologic state" to model inflows, and fit its parameters to historical data from Uruguay, as is standard local practice [9]. This state is added to reservoir levels for dynamic programming. For the faster time-scale, for the moment we are using empirical traces for demand and renewable sources, the medium term goal is a more detailed accounting of their stochasticity. In Section V we present our experimental results with this method. Conclusions are given in Section VI.

II. Stochastic dual dynamic programming

Consider first the simplest model of operation of a hydrothermal system over a horizon T, with time steps indexed by $k = 0, \dots T-1$. It includes a state vector $x_k \in \mathbb{R}^n$ representing the current storage levels at n reservoirs; water inflows are modeled as disturbances $w_k \in \mathbb{R}^p$ which are typically random and may have complex evolution patterns. The control vector $u_k \in \mathbb{R}^m$ models the actions that the system operator may take: the amount of generation drawn from hydro plants, the amount of spillage, and economic dispatch decisions from the alternative thermoelectric generators in order to provision the demand at each stage. We postpone other aspects of the problem such as non-dispatchable renewables, or energy exchanges with neighboring systems.

The running costs per stage are modeled through a function $g_k(x_k, u_k, w_k)$ (e.g. the cost of fuel in thermal plants). The objective is to obtain a *policy* $\pi = \{\mu_k\}_{k=0,\dots,T-1}$ such that the following optimum is attained:

$$\min_{\pi} E_w \left[\sum_{k=0}^{T-1} g_k(x_k, \mu_k(x_k, w_k), w_k) \right]. \tag{1}$$

The above problem is coupled by the dynamic constraints of the system, namely:

$$x_{k+1} = f_k(x_k, \mu_k(x_k, w_k), w_k),$$

where $f_k(x_k, u_k, w_k)$ is a function that maps the current state to the future state based on the actions performed and the disturbances realized. In particular it will include the water balance equations at reservoirs. Note that we allow the policy rule μ_k to depend on the current state *and* disturbance, i.e. the inflows measured at stage k [10].

Dynamic Programming [3] allows the decoupling of the optimization problem over the time axis, by computing recursively the cost-to-go or value function:

$$V_k(x_k) = \min_{\pi} E_w \left[\sum_{j=k}^{T-1} g_j(x_j, \mu_k(x_j, w_j), w_j) \right]. \quad (2)$$

Since this is a function of the real-valued state, it is customary to compute it for a certain discretization. The curse of dimensionality [10] is that the number of such evaluations grows exponentially with the state dimension. In our context, this restricts the scope of classical dynamic programming techniques to a few lakes and coarse quantization patterns.

A powerful alternative with proven convergence guarantees is Stochastic Dual Dynamic Programming, first proposed in [4], but with roots going back to the work of Kelley on cutting-plane methods [11] and Bender's decomposition techniques [12]. SDDP applies to the case where the costs g_k are convex functions and the dynamical constraints f_k are affine; in this case the value function (2) is convex, which ensures that any supporting hyperplane of the value function constitutes a global lower bound on the cost-to-go.

Based on this observation, the SDDP algorithm performs a series of iterations or *passes* where at each step a new supporting hyperplane for V_k is added. Therefore, on pass l of the algorithm, we have a global lower bound V_k^l of the cost to go as the maximum over all supporting hyperplanes, i.e. a piecewise linear convex function. The computation of a new supporting hyperplane is achieved by solving:

$$\hat{\beta}_k^{(l+1)}(w) = \min_{x,u} \left\{ g_k(x,u,w) + V_{k+1}^{(l+1)}(f_k(x,u,w)) \right\},$$
$$s.t. \quad x = x_k^{(l)} \quad [\hat{\lambda}_k^{(l+1)}(w)].$$

i.e. one step of the Bellman iteration with input $V_{k+1}^{(l+1)}$, the current value estimate. The added constraint generates an objective value $\hat{\beta}_k^{(l+1)}(w)$ and a dual variable $\hat{\lambda}_k^{(l+1)}(w)$ for each noise realization. Averaging over the noise yields

$$\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w)],$$
$$\lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w)].$$

It follows from the optimization that

$$\beta_k^{(l+1)} + (\lambda_k^{(l+1)})^T (x_k - x_k^{(l)})$$

is a supporting hyperplane for V_k at $x = x_k^{(l)}$. This adds a cut that refines the cost-to-go estimation on each pass. Implementation details will be discussed when applying the method in Section V. Convergence properties and a detailed implementation are given in [7].

A. Adding memory to the hydrologic uncertainty

In the preceding discussion we assumed that the noise realizations are independent and identically distributed. However, water inflows are typically correlated across stages. A useful tool to model this time dependence is to consider a Markov state h_k that summarizes the current hydrological environment. The dynamics of this state are governed by a non-homogeneous Markov chain with transition probabilities:

$$P_{hh'}^{(k)} = P(h_{k+1} = h' \mid h_k = h).$$

Transition probabilities may be estimated from historical data. In [6] a model with two states (dry and wet) is introduced; local practice in Uruguay (e.g., [9]) is to use a 5-level model of the hydrologic state as described in Sec. IV-B. Given that the hydrologic state is $h_k = h$, the possible inflows are drawn accordingly with some conditional distribution $w_k \mid h$.

Incorporating the Markov transitions in dynamic programming leads to enlarging the state to $\tilde{x}_k = (x_k, h_k)$ and a generalized Bellman iteration, decomposed as:

$$\hat{V}_k((x,h),w) = \min_u \left\{ g_k(x,u,w) + \sum_{h'} P_{hh'}^{(k)} V_{k+1}(x',h') \right\}$$
$$V_k(x,h) = E\left[\hat{V}_k((x,h),w) \mid h_k = h \right],$$

where $x' = f_k(x,u,w)$ represents the next state. The first equation computes an estimation of the cost-to-go for each noise realization (note that w should be coherent with h) and the second equation corresponds to averaging over noise realizations, which is done using the current conditional distribution for the current hydro state h.

As before, the SDDP technique can be applied by substituting a current piecewise linear estimate of the cost to go function and solving:

$$\hat{\beta}_k^{(l+1)}(w) = \min_{x,u} \left\{ g_k(x,u,w) + \sum_{h'} P_{hh'}^{(k)} V_{k+1}^{(l+1)}(x',h') \right\},$$
$$s.t. \quad x = x_k^{(l)} \quad [\hat{\lambda}_k^{(l+1)}(w)]. \quad (3)$$

A suitable lower bound for the enriched cost to go $V_k(x,h)$ is then obtained by averaging as before, but using the conditional distribution given h:

$$\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w_k) \mid h_k = h],$$
$$\lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w_k) \mid h_k = h].$$

The above coefficients again define a supporting hyperplane for the cost-to-go. In practice, this conditional distribution is not known but empirically estimated from historical data: the conditioning part then becomes simply averaging over scenarios that belong to the current hydrologic state h (e.g. wet realizations only), instead of the full set.

III. Multi-timescale decomposition

The above discussion applies in general to dynamic programming formulations of the stochastic scheduling problem, in particular applied to hydro-thermal systems. However, a large share of the energy dispatch in current systems comes from renewable energy sources such as wind and solar, varying in a *faster* timescale than the classical dispatch of hydro storage. It is thus necessary to include in the general hydro-thermal scheduling this refined timescale to model fast variations and short-term storage (batteries). We now explain how to introduce this multi-timescale decomposition in the SDDP framework in a simple model to convey the main ideas, and then describe our implementation for a real system.

To model this fast timescale, consider the state x_k at stage k as composed of the current level of the reservoirs and storage facilities in the system. The basic dynamic equation for a single reservoir is:

$$x_{k+1} = x_k - u_k - s_k + w_k, \tag{4}$$

where u_k is the amount of water drawn from the reservoir, s_k is the amount of spillage and w_k is the (random) water inflow.

We propose to decompose u_k into a finer timescale, that is:

$$u_k = \sum_{i=0}^{t-1} u_{ik},$$

where u_{ik} denotes the water used for generation at each step of the detailed timescale (e.g hours instead of days or weeks).

If d_{ik} is the demand at the finer timescale and e_{ik} is the energy available from renewables, the power balance equations are given for each k by:

$$d_{ik} = e_{ik} + \nu u_{ik} + p_{ik} \quad i = 0, \dots, t-1. \tag{5}$$

Here ν is an efficiency coefficient of the hydroelectric generator (for simplicity assumed independent of the reservoir level) and p_{ik} is the thermal generation. Assuming the only running cost is the price c_k of the latter, the cost function at stage k is:

$$g_k(x_k, (u_{ik}, s_k, p_{ik})) = \sum_{i=0}^{t-1} c_k p_{ik}.$$

While this function does not depend explicitly on the storage level and the power drawn from reservoirs and spillage, it does so indirectly through constraint (5) since demand must be met.

We now enrich this model with a short-term storage battery. Let b_{ik} denote the amount of stored energy at step i of stage k. We denote by $q^d_{ik} \in [0, \bar{Q}^d]$ the energy drawn from the battery for demand supply at a discharge efficiency η_d, and by $q^c_{ik} \in [0, \bar{Q}^c]$ the energy injected into storage at an efficiency η_c. The balance equations now become:

$$d_{ik} = e_{ik} + \nu u_{ik} + p_{ik} + q^d_{ik} - q^c_{ik} \quad i = 0, \dots, t-1. \tag{6}$$

The battery dynamics can now be included as a constraint in the fast timescale taking the efficiencies into account:

$$b_{i+1,k} = b_{ik} + \eta_c q^c_{ik} - \frac{1}{\eta_d} q^d_{ik}, \quad i = 0, \dots, t-1, \tag{7}$$

with the identification $b_{0k} = b_k$, the storage level at the beginning of the stage, and $b_{t,k} = b_{k+1}$, i.e. the final storage in the short timescale becomes the initial storage for the next stage. We should also add the battery capacity constraints

$$\underline{B} \leq b_{i,k} \leq \overline{B} \quad \text{for each } i, k.$$

Since all the storage dynamics are affine, we are still within the domain of applicability of the SDDP technique.

This more detailed modeling comes at a computational cost: instead of having a single control variable $u_k \in \mathbb{R}$ for the water usage, we expand it into $(u_{ik}, i = 1, \dots, t) \in \mathbb{R}^t$. The same applies to the thermal power and battery injection variables. In Section V we investigate the impact of this enlarged problem size on computation times.

IV. Application example: the Uruguayan system

In order to evaluate the performance of SDDP algorithm with the extensions proposed in Sec. III, a model of an electrical system inspired in the Uruguayan hydro-power operation was developed and solved using the library [8]. In our model, we take the time horizon to be one year with a daily stage step ($T = 365$, $k = 0, \dots, T-1$). Power generation is provided by four interconnected hydroelectric dams and a single thermal power plant representing the aggregate thermal generation. The hydroelectric dams included correspond to the 4 Uruguayan plants: Bonete, Baygorria, Palmar in the *Rio Negro* and Salto Grande in the *Rio Uruguay*.

To model renewables, we also include a single wind power plant and a single solar power plant, again representing the total renewable aggregates. Finally we include also control variables to model energy export and import from nearby systems, with its associated costs and utilities.

In each daily stage, we divide the control variables over twenty-four detailed timesteps representing hourly behavior ($t = 24, i = 0, \dots, t-1$), and the detailed balance equations (5) are considered for each hour.

To feed our model, we take the hourly demand (d_{ik}), and the hourly solar and wind power generation (e_{ik}) corresponding to the year 2017 [13], with demand scaled according to its annual growth to represent the current situation.

The randomness in our model comes from the weekly inflow realizations (w_k) for the three largest reservoirs (Bonete, Palmar and Salto Grande). A Markov model is estimated from historical inflow data as discussed below in Sec. IV-B.

A. Detailed model

For each hydroelectric plant in the system we define the hourly control variables for the amount of turbined water u^j_{ik}, and a daily control variable for spillage s^j_k. Here $j = 1, \dots, 4$ represents the four dams: Bonete, Baygorria, Palmar and Salto respectively. We denote by $u^j_k = \sum_{i=0}^{t-1} u^j_{ik}$ the total water turbined for the stage k on dam j, and by y^j_k the total outflow $y^j_k = u^j_k + s^j_k$.

The state in our model is the current water volume in each reservoir, i.e. $x_k = (x^j_k, j = 1, \dots, 4)$. Recall that each reservoir evolves according to eq. (4). However, since reservoirs on the same river are interconnected, the state dynamics can be represented in matrix form as:

$$x_{k+1} = Ax_k + By_k + w_k. \tag{8}$$

Here A is the 4×4 identity matrix, $y_k = (y_k^j, j = 1, \dots, 4)$ is the vector of outflows, and the matrix B captures the river network, which in our case is given by:

$$B = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \tag{9}$$

which captures the fact that outflows from dams in the same river flow into the next one downstream. Finally, the vector $w_k = (w_k^j, j = 1, \dots, 4)$ are the random water inflows whose dynamics are explained below.

The remaining control variables in our model are the thermal power generation p_{ik} at each step and the energy exchanges with nearby systems (exports and imports) p_{ik}^{exp} and p_{ik}^{imp}. The power balance equations (5) in this setting thus become:

$$d_{ik} + p_{ik}^{exp} = e_{ik} + \sum_j \nu_j u_{ik}^j + p_{ik} + p_{ik}^{imp}, \tag{10}$$

for $i = 0, \dots, t-1$, $k = 0, \dots, T-1$. The sum over j represents the contribution of hydro plants and ν_j is the efficiency coefficient of each one. Equations (10) operate as constraints on the stage optimization problem (3).

For the stage cost, we take:

$$g_k = \sum_{i=1}^{t} c_k p_{ik} - c_k^{exp} p_{ik}^{exp} + c_k^{imp} p_{ik}^{imp}, \tag{11}$$

where $c_k, c_k^{exp}, c_k^{imp} > 0$ represent the thermal generation cost and the exchange prices with neighbor systems at stage k.

Finally, all control variables are bounded by the corresponding generation limits, and the state variables are bounded by the corresponding maximum reservoir levels.

B. Markov Model Estimation

We now describe our model for the hydrologic state of the system. As explained above, a method to incorporate time correlation between water inflows amenable to treatment by dynamic programming, is to use a Markov chain to model the time evolution of the hydrologic state [6]. We use here a 5-level Markov chain with state $h_k \in \{1, \dots, 5\}$, representing *very dry*, *dry*, *normal*, *wet* and *very wet* conditions.

To fit the Markov chain model we use the historical data available since 1909, which records the average inflow for each of the three larger reservoirs (Bonete, Palmar and Salto). h_k will be treated as a non-homogeneous Markov chain (reflecting seasonal variations), so it is necessary to estimate transition matrices for each stage k. The historical data is available on a weekly timescale; to adapt it to a daily step, we will estimate a transitions for each week of the year and assign these transition probabilities to the first day of each week, keeping the state constant afterwards during that week.

To keep our hydrologic state simple we first reduce the historical series of three inflow values to a single scalar value W_l for each week l; W_l is defined as a weighted average of the 3 inflows in proportion to the corresponding dam power. Next, we quantize W_l to 5 levels, from very dry to very wet: this is done from quantiles of the distribution of annual values of W_l for each fixed week of the year. By repeating this procedure for all weeks of the year, a weekly historical series is obtained, where each week has a discrete state value h_l. To estimate the transition matrix associated to week l, we proceed as usual by counting transitions between states: the entry $P_{hh'}^{(l)}$ of that matrix is calculated by counting in historical series how many transitions exist between state h in week l and state h' in week $l+1$ and then dividing that value by the total visits to state h in week l. Finally, the stage transition matrix $P_{hh'}^{(k)}$ is taken as $P_{hh'}^{(l)}$ when k corresponds to the beginning of week l and the identity matrix otherwise.

This Markov chain is incorporated into our dynamic model by considering the enlarged state $\tilde{x}_k = (x_k, h_k)$, where x_k follows (8) as before, and h_k follows the Markov transitions. In order to sample inflow realizations w_k^j for each reservoir in the dynamics (8), given that the current hydrologic state is h_k, we use the empirical distribution of the historical water inflows w_k^j for week k and reservoir j which are tagged with state h_k according to the previous thresholds. These are then adjusted to represent daily inflows. Additionally, recall that one of the reservoirs has negligible inflows, so in our model we take $w_k^3 = 0$ for all k.

C. Incorporating short-term storage

One of the main benefits of the SDDP technique is that incorporating extra states, while enlarging the computational complexity, does not produce an exponential increase in computation times. Moreover, our multi-timescale formulation can also model short-term storage as discussed in Sec. III, while keeping the long term dynamics of the water inflows and dams. This has become important to model the interaction between short and long term storage in the presence of renewable sources, which are inherently non-dispatchable.

To explore this scalability when the state space dimension increases, we now enrich our model with a short term storage battery. The state of our system is now $\tilde{x}_k = ((x_k, b_k), h_k)$, where b_k represents the energy stored in the battery at the beginning of the stage (day). The reservoir dynamics are the same as before for x_k (eq. (8)), as well as the Markov dynamics for the hydrologic state.

As for the control variables, apart from u_{ik}^j and s_k^j for the dams, the thermal generation p_{ik} and the energy exchanges $p_{ik}^{exp}, p_{ik}^{imp}$ with other systems, we have the amount of energy that is injected or retrieved from the battery at each detailed timestep, q_{ik}^c and q_{ik}^d respectively.

When faced with a demand d_{ik} and renewable generation e_{ik}, the system with combined dams and energy storage must satisfy the following modification of eq. (10):

$$d_{ik} + p_{ik}^{exp} + q_{ik}^c = e_{ik} + \sum_j \nu_j u_{ik}^j + p_{ik} + p_{ik}^{imp} + q_{ik}^d, \tag{12}$$

for $i = 0, \dots, t-1$, $k = 0, \dots, T-1$. The hourly battery dynamics is captured by (7) which we reproduce here for easy reference:

$$b_{i+1,k} = b_{ik} + \eta_c q^c_{ik} - \frac{1}{\eta_d} q^d_{ik}, \quad i = 0, \dots, t-1,$$

with the identifications $b_{0k} = b_k$ and $b_{tk} = b_{k+1}$. This completes the definition of our model, in the next section we discuss the optimization results.

V. Implementation and Simulations

We now summarize the results obtained by applying the SDDP technique to the stylized model of the Uruguayan system we developed. Our main focus throughout this Section is on *performance* considerations, and not explicitly cost considerations. We aim to quantify the complexity and computation time required to incorporate the detailed timescale, as well as a larger number of state variables. The costs reported below are just an indication and do not reflect real operation costs.

All our model variants were implemented in Julia [14], a new fast open source mathematical modeling language. In particular, to model our problem we use the framework for SDDP implemented by the package SDDP.jl [8].

In the following we discuss three modeling variants: the first one (M1) constitutes a base model for benchmarking, incorporating only the slower (daily) time-scale and not including short-term storage, so its state includes just the reservoir levels, and energy balance is enforced on a daily scale. Note that this however assumes in particular that the renewable input can be fully utilized, an assumption that may not be valid due to the inability to dispatch these sources.

In the second variant (M2) we incorporate the faster (hourly) timescale to our base model, but we do not include short-term storage. This second variant includes all the power constraints of the short timescale, and in particular reflects that the renewable sources cannot be dispatched at will. These two variants without storage share a state space of four continuous states (the reservoirs) and a discrete Markov hydrologic state. However the model M2 incorporates a larger space of control variables within the detailed timescale. This enables us to evaluate the impact in execution time of incorporating the detailed dispatch without enlarging the state space.

The third variant (M3) we consider adds a short term storage (battery) to the model M2. In this case, a state variable is added to keep track of the storage level at the slow timescale. So in this case we are also enlarging the state space with respect to M2. Here the main idea is to compare against classical Stochastic Dynamic Programming (SDP) techniques (e.g. [9]). In SDP adding a state variable typically requires quantization of this extra state. The SDP recursion also requires the enumeration of all combinations of initial state values for all states. If the added variable is quantized in l levels, then computational effort increased at least by a factor of l. In SDDP the impact of adding a state variable does not produce such an explosion in computational effort with the increased dimensionality, because states are not treated via quantization but rather using cuts for a global approximation of the future cost function. The real impact in execution time is determined by the total number of iterations required by SDDP to reach convergence, due to the richer state space which must be explored. However, this increment in the number of iterations is still less expensive than the quantization approach. Our case studies just described aim to validate this behavior.

TABLE I
Demand and renewable power statistics and generation costs

Averages and total demand and resources	
Average demand	1334 MWh
Average renewable energy	570.4 MWh
Annual demand	11654 GWh
Annual renewable energy	4984 GWh
Costs	
Thermal Cost	200 USD/MWh
Export	15 USD/MWh
Import	360 USD/Mwh

TABLE II
Short term storage parameters for the detailed model (M3)

Storage Parameters	
$\underline{B}$	0 MWh
$\overline{B}$	500 MWh
$\bar{Q}^d$	250 MW
$\bar{Q}^c$	250 MW
$\eta_c = \eta_d$	0.98

A. Model input data

All our models are fed with the following input parameters: the overall demand, as well as solar and wind power generations are the taken from the corresponding ones of the year 2017 [13]. The generation costs, as well as the price of energy exchanges are chosen to be similar to the average costs of the current Uruguayan system in order to obtain a reasonable approximation of current costs. However, we emphasize that these costs should not be expected to reflect the operation cost of the real system. In Table I we present summarize some of these inputs.

Additionally, for the model M3 we use the parameters for the short term storage summarized in Table II.

As a convergence criterion for the SDDP algorithm, we choose the *Bound Stalling* stopping rule in all three models. Under this rule, the SDDP algorithm will terminate after the cost lower bound has failed to improve for a given number of iterations (set to 5 in our case). After convergence, the obtained policy is fed to a Monte Carlo simulation for different random inflow data, performing 1000 repetitions. The simulated average operation cost is then calculated by averaging over all repetitions. All the computations presented below were carried out on a 3.40 GHz Intel i7, 32GB RAM computer.

TABLE III
EXPERIMENTAL RESULTS

Model	Iterations	Execution Time (s)	Cost Bound (MUSD)	Simulated Avg. Cost (MUSD)
M1	29	318	52.25	54.91
M2	71	1611	79.33	85.79
M3	99	3127	73.73	78.58

B. Experimental results

The results for each of the three variants described are presented in Table III. The iterations column shows how many steps the SDDP algorithm takes to attain its bound, and the time required to converge is reported in the execution time column, Finally, we report the attained cost lower bound as well as the average cost obtained by simulation using the corresponding policy.

As a guideline to interpret the results, recall from Table I that the annual energy that must be injected into the system, assuming all renewable energy can be used, is about $6670 GWh$. From the obtained cost, we can infer that the policy is covering this energy with approximately 96% hydro power, and 4% thermal generation, aligned with the current situation in Uruguay.

Comparing the three models, the first one obtains a lower cost, which is expected since it only considers daily energy balance and this implies that the entire renewable energy available can be exploited. In mathematical terms, the model M1 is a relaxation of M2, and hence it will always attain a lower cost. The model M2 takes into account the detailed timescale, and thus the fact that demand and renewable generation may not be aligned during the day. The increase in cost is due to the fact that energy shifting now cannot be achieved, and therefore extra thermal energy must be introduced into the system. Some of this cost is reclaimed by the model M3, which introduces the short term storage battery, now allowing the system to better exploit the renewable power.

More importantly, we discuss the convergence time and performance of our implementation. The benchmark model M1 converges in only 29 iterations and 318 seconds, despite including all four reservoirs in the state, as well as the Markovian description of inflow evolution. This illustrates the power of the SDDP approach, which enables us to model the complete system with a daily timescale and obtain fast results. The second model incorporates the detailed timestep, multiplying by 24 the number of control variables. However this does not lead to a large increase in computation time. The true power of the SDDP approach becomes evident in the third model, where a new state and its associated control variables are introduced, but the computation time is less than doubled. The achieved performance shows that SDDP constitutes a promising approach to model the system using the multi-timescale decomposition we developed.

VI. Conclusion

In this paper we proposed an adaptation of the stochastic dual dynamic programming technique to solve the optimal provisioning problem in a combined hydro-thermal system. The main contribution is to incorporate into the SDDP framework a detailed timescale, enabling us to model short term effects, which is needed to capture the variability of the renewable sources and to better gauge the impact of incorporating short term energy storage into the system. The results obtained show promising computation times, particularly when the state space is enlarged, which opens the door to incorporating further details into the model.

Several lines of future work remain open: an important one is to exploit the fact that the SDDP technique is amenable to parallel computation, a feature that was not exploited in our simulations. This will drastically improve the computation times involved. A second important line of work is to include network considerations into the mix, by considering the network constraints, which is typically left out due to computational complexity, but with these techniques seems now in reach.

Acknowledgment

This work was partially supported by ANII-Uruguay under grant FSE_1_2016_1_131605.

References

[1] C. Harris, *Electricity markets: pricing, structures and economics*. John Wiley & Sons, 2011, vol. 565.
[2] A. J. Conejo, M. Carrión, J. M. Morales *et al.*, *Decision making under uncertainty in electricity markets*. Springer, 2010, vol. 1.
[3] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 2005, vol. 1, no. 3.
[4] M. V. Pereira and L. M. Pinto, "Multi-stage stochastic optimization applied to energy planning," *Mathematical programming*, vol. 52, no. 1-3, pp. 359–375, 1991.
[5] "PSR," https://www.psr-inc.com/softwares-en/, 2018, [Online; accessed Aug-2018].
[6] A. B. Philpott and V. L. De Matos, "Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion," *European Journal of Operational Research*, vol. 218, no. 2, pp. 470–483, 2012.
[7] P. Girardeau, V. Leclere, and A. B. Philpott, "On the convergence of decomposition methods for multistage stochastic convex programs," *Mathematics of Operations Research*, vol. 40, no. 1, pp. 130–145, 2014.
[8] O. Dowson and L. Kapelevich, "SDDP.jl: a Julia package for Stochastic Dual Dynamic Programming," *Optimization Online*, 2017. [Online]. Available: http://www.optimization-online.org/DB_HTML/2017/12/6388.html
[9] G. Casaravilla, R. Chaer, and P. Alfaro, "Simsee: Simulador de sistemas de energía eléctrica," Proyecto PDT 47/12. Technical Report 7, Universidad de la Repúlica (Uruguay). Facultad de Ingeniería. Instituto de Ingeniería Elétrica, Number 7-Dec, Tech. Rep., 2008.
[10] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007, vol. 703.
[11] J. E. Kelley, Jr, "The cutting-plane method for solving convex programs," *Journal of the society for Industrial and Applied Mathematics*, vol. 8, no. 4, pp. 703–712, 1960.
[12] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische mathematik*, vol. 4, no. 1, pp. 238–252, 1962.
[13] "Administración del mercado eléctrico," http://adme.com.uy, 2018, [Online; accessed Aug-2018].
[14] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.