An Overview of the Stochastic Dual Dynamic Programming Method

Alex Dunyak

April 30, 2020

1 Introduction

In 1991, M.V.F. Pereira and L.M.V.G. Pinto published *Multi-stage stochastic optimization applied to energy planning* [1]. This introduced the Stochastic Dual Dynamic Programming method to solve sequential linear programming problems of the form

$$\min_{\substack{A_1x_1=b_1\\x_1\geq 0}} c_1^T x_1 + \mathbb{E}\left[\min_{\substack{B_2x_1+A_2x_2=b_2\\x_2\geq 0}} c_2^T x_2 + \mathbb{E}\left[\cdots + \mathbb{E}\left[\min_{\substack{B_TX_{T-1}+A_Tx_T=b_T\\x_T\geq 0}} c_T^T x_T\right]\right]\right]. \tag{1}$$

Here, the vectors c_t and b_t, and matrices A_t and B_t are modeled as random variables, with c_1, A_1, and b_1 being deterministic. The variable x_t is continuous and real-valued. For notation, we will define the history of the process $\xi_{1:t}$ to be the set of tuples

$$\xi_{1:t} = \{(c_1, b_1, A_1, B_1), (c_2, b_2, A_2, B_2), \cdots, (c_t, b_t, A_t, B_t)\}, \tag{2}$$

The typical way to solve a problem of this form is to directly use dynamic programming. This is a classical result in optimal control, but it has many downsides. In particular, unless the system has underlying structure to exploit, we must discretize the state space to apply dynamic programming. This is obviously impractical for systems with large state spaces. As Pereira and Pinto note, a system with ten states discretized into four values per state will have 4^{10} values, and applying dynamic programming naively will necessitate iterating over all of these possible states to find the future cost. One alternative approach, as demonstrated by Gal [2], is to sample the states and fit a parametric function over them describing the future cost. For example, Gal suggests that if the value function is approximately linear for K states, it is sufficient to sample $K+1$ linearly independent states at each time. With enough data and the right parameterized functions, such a scheme may give a very close approximation of the value function.

Building on this approach, Pinto and Pereira posited that the solution can be represented exactly as a piece-wise linear function. This realization is at the

core of the SDDP method. In this paper, we will explain Pereira and Pinto's method, and show some of the developments that were made in the 20 years proceeding the original paper by reviewing Alexander Shapiro's 2011 paper, *Analysis of stochastic dual dynamic programming method* [3].

2 The Stochastic Dual Dynamic Programming Algorithm

First, we will introduce the problem posed by Pereira and Pinto [1] for two-stages, then extend it into multiple stages. Finally, we will show the extension Shapiro makes allowing for the use of continuum valued noise.

Suppose that we have a two-stage linear programming problem:

$$\begin{aligned} \min_{x} \quad & c_1^T x_1 + p_1 c_2^T x_{21} + p_2 c_2^T x_{22} + \cdots + p_m c_2^T x_{2m} \\ \text{s.t.} \quad & A_1 x_1 \geq b_1, \\ & B_2 x_1 + A_2 x_{21} \geq b_{21}, \\ & B_2 x_1 + A_2 x_{22} \geq b_{22}, \\ & \vdots \\ & B_2 x_1 + A_2 x_{2m} \geq b_{2m}. \end{aligned} \tag{3}$$

This can be interpreted as a stochastic linear programming problem by requiring $\sum_{j=1}^{m} p_j = 1$. We can make the dynamic programming structure clear by noting that if there are m possible stochastic outcomes, there are m second-stage subproblems. Subproblem j could then be written

$$\begin{aligned} \alpha_{1j}(x_1) = \min \quad & c_2^T x_{2j}, \\ \text{s.t.} \quad & A_2 x_{2j} \leq b_{2j} - B_2 x_1. \end{aligned} \tag{4}$$

We then denote the expected value of all subproblems as

$$\bar{\alpha}_1(x_1) = \sum_{j=1}^{m} p_j \alpha_{1j}(x_1), \tag{5}$$

and the whole problem can be described as

$$\begin{aligned} \min \quad & c_1^T x_1 + \bar{\alpha}_1(x_1) \\ \text{.s.t.} \quad & A_1 x_1 = b_1. \end{aligned} \tag{6}$$

As each subproblem is simply a linear program, it is convex and we can find its dual. The dual of subproblem j would be

$$\begin{aligned} \max \quad & \pi(b_{2j} - B_2 x_1), \\ \text{s.t.} \quad & A_2^T \pi \leq c_{2j}. \end{aligned} \tag{7}$$

A classical result of convex optimization is linear program duality, so if both the primal and dual subproblem have feasible solutions, they both obtain their optimal value, and the optimal solution exists as a vertex on the constraint set. Pereira and Pinto note that by labeling the constraint vertices as $\Pi = \{\pi_1, \pi_2, \cdots, \pi_v\}$, the dual could equivalently be written as

$$\begin{aligned} \alpha_{1j}(x_1) = \min_{\alpha} \quad & \alpha \\ \text{s.t.} \quad & \alpha \geq \pi^i(b_{2j} - B_2 x_1) \text{ for } i = 1, \cdots, v. \end{aligned} \tag{8}$$

Next, suppose we have n trial values of x_1, labeled $\{\hat{x}_{11}, \cdots, \hat{x}_{1,n}\}$. By solving (8) for each of the n trial values, we find the associated multipliers $\Pi_n = \{\pi_1, \pi_2, \cdots, \pi_n\}$. We can then use these multipliers to find a piecewise linear approximation of the cost of each subproblem j (for any input x_1) by solving

$$\begin{aligned} \hat{\alpha}_{1j}(x_1) = \min_{\alpha} \quad & \alpha_j \\ \text{s.t.} \quad & \alpha_j \geq \pi^i(b_{2j} - B_2 x_1) \text{ for } i = 1, \cdots, n. \end{aligned} \tag{9}$$

The function $\hat{\alpha}_{1j}(x_1)$ is a lower bound of (7), as noted by Pereira and Pinto, as it has fewer constraints. This then finds the lower bound of the entire system by substituting (8) into (3) for each subproblem, which becomes

$$\begin{aligned} \underline{z} = \min_{x_1, \alpha} \quad & c_1^T x_1 + \sum_{j=1}^{m} p_j \alpha_{1j} \\ \text{s.t.} \quad & A_1 x_1 \geq b_1, \\ & \alpha_{1j} - \pi^i(b_{2j} - B_2 x_1) \geq 0 \text{ for } i = 1, \cdots n, \ j = 1, \cdots, m. \end{aligned} \tag{10}$$

This gives the original problem (3) an error bound. Given trial decision $\hat{x}_1$, we can find the lower bound using ((10)) $\underline{z}$, and evaluate the approximate value function of each subproblem directly on $\hat{x}_1$ to find an upper bound $\bar{z}$

$$\begin{aligned} \bar{z} = \min_{x_1, \alpha} \quad & c_1^T x_1 + \sum_{j=1}^{m} p_j \hat{\alpha}_{1j}(\hat{x}_1) \\ \text{s.t.} \quad & A_1 x_1 \geq b_1. \end{aligned} \tag{11}$$

The gap between $\bar{z}$ and $\underline{z}$ tells us how for the trial decision $\hat{x}_1$ is from the true optimal decision.

Extending this to multiple stages is relatively straightforward. It consists of a forward pass of the algorithm using the approximation of $\bar{\alpha}(x_t)$ of the problem with a trial decision $\hat{x}_t$, and a backwards pass to find the multiplier associated with each time step to improve the future cost approximation. Then, the upper and lower bounds are calculated with the same method as the two-step program. In the same manner, the upper bounds and lower bounds are steered towards the actual value of the multi-stage problem.

For completeness, Pereira and Pinto assume that the vectors b_{tj} are independent random variables, and that they are discretized into m scenarios. The full algorithm can then be written as:

1. Define the trial decisions $\hat{x}_{ti}$, for $i = 1, \cdots n$ and $t = 1, \cdots T$. Define a terminating condition ϵ.
2. For $t = T, T-1, \cdots, 2$ (backwards)

   - For each trial decision $\hat{x}_{ti}$, $i = 1, \cdots n$:
     - For each scenario $b_{tj}, j = 1, \cdots m$:
       * Solve the following approximate problem for $t, \hat{x}_{ti}$, and b_{tj}:

$$\begin{aligned} \min_{x_t} \quad & c_t^T x_t + \hat{\alpha}(x_t) \\ \text{s.t.} \quad & A_t x_t \geq b_{tj} - B_t \hat{x}_{(t-1)i} \end{aligned} \tag{12}$$

       Store the result as $\hat{x}_{ti}$.

     Find the expected vertex value $\bar{\pi}_{t-1ij} = \sum_{j=1}^{m} p_{tj} \pi_{t-1ij}$, and construct a supporting hyperplane of the future cost function at stage $t-1$, $\bar{\alpha}_{t-1}(x_{t-1})$.
3. Find the upper $\bar{z}$ and lower $\underline{z}$ bounds for each trial decision using equations (11) and (10) respectively.
4. If $\|\bar{z} - \underline{z}\| < \epsilon$, then break. Otherwise, update $\hat{x}$ and go to step 2.

A close reading of this algorithm reveals a concerning feature: although it was created to avoid the combinatorial explosion associated with state discretization, the algorithm discretizes the stochasticity of the system. This is still infeasible computationally for systems with a long running time and many possible scenarios. Pereira and Pinto addressed this problem for a finite number of scenarios, and Shapiro extended it into continuous probability distribuitions.

2.1 Reducing the required number of computations

Fundamentally, both [1] and [3] address the problem of lowering the number of computations for this variety of system in the same manner, with Shapiro explicitly citing Pereira and Pinto's approach. Pereira and Pinto simply modified their algorithm to sample a scenario b_{tj} according to the distribution of b_t. This lead to a modification of the termination criteria, however. The real lower bound can be obtained from the first stage approximation as before, but the upper bound needs to be estimated. By taking the n trial decisions, running the Monte Carlo algorithm on each of them, and averaging, the upper bound can be approximated:

$$\bar{z} = c_1 x_1 + \frac{1}{n} \sum_{i=1}^{n} z_i. \tag{13}$$

This still leads to uncertainty around the upper bound which can be measured by the sample standard deviation (given n Monte Carlo upper bound samples):

$$\sigma_z = \sqrt{\frac{1}{n^2}\sum_{i=1}^{n}(\bar{z} - z_i)^2}. \tag{14}$$

From this, we can define any confidence interval we want. Pereira and Pinto go on to say that this could be used as an additional convergence criteria, by stopping the algorithm when the lower bound is within the confidence interval of the upper bound. However, as Shapiro notes, this in itself does not indicate that the problem has been solved to an acceptable degree. For high variance systems, the confidence interval could be very large. He suggests instead terminating when the lower bound is within ϵ of the top of the upper bound's confidence interval.

2.2 Extending to continuous distributions

One issue with Pereira and Pinto's approach is that it only considers a finite number of possible events. This is an issue, as many systems assume that noise is distributed continuously, such as Gaussian noise. Shapiro extends the Monte Carlo approach to deal with these situations. In doing so, he must re-analyze the error bounds given to be sure that the algorithm will give approximate answers for the original problem.

Instead of attempting to solve the problem analytically for continuous probability densities, Shapiro samples the distribution to create what he calls the sampled average approximation problem, and then he uses the SDDP algorithm on the sampled program. Formally, he expands the problem to allow for random matrices as well, so we will use Shapiro's notation for the rest of this subsection. Suppose we have a multi-stage stochastic linear programming problem, with the first stage

$$\min_{\substack{A_t x_t = b_t - B_t x_{t-1} \\ x_t \geq 0}} c_t^T x_t + \mathcal{Q}_{t+1}(x_t) \tag{15}$$

where $x \in \mathbb{R}^n$, A is an $n \times n$ matrix, $B_1 = 0$ almost surely, b_t is a vector, $\mathcal{Q}_{t+1}(x) = \mathbb{E}[Q_{t+1}(x, \xi)]$, and $Q_{t+1}(x, \xi)$ is the optimal value of the next stage with the data realization ξ. As before, the future cost function $Q(x, \xi)$ is convex in x for all time steps. We assume that ξ is distributed according to the continuous distribution P.

Shapiro relaxes the requirement on a finite number of scenarios by sampling N (independent) values of ξ to create the sample average approximation (SAA) problem with the assumption that ξ is stagewise independent. We call the sampled probability distribution P_N, which assigns probability $1/N$ to each

sample of $\{\xi_1, \xi_2, \cdots, \xi_N\}$:

$$\min_{\substack{Ax=b \\ x\geq 0}} c_t^T x_t + \tilde{\mathcal{Q}}_{t+1}(x_t), \tag{16}$$

$$\tilde{\mathcal{Q}}(x) := \mathbb{E}_{P_N}[Q(x,\xi)] = \frac{1}{N}\sum_{i=1}^{N} Q(x,\xi_i),$$

$$A_t x_t = b_t - B_t \bar{x}_{t-1}.$$

Shapiro cites his previous work [4] to give a required number of samples per stage. In particular, under most circumstances to arrive within $\epsilon > 0$ of the true value of the full problem, the number of samples must be on the order of ϵ^{-2}. He proves that expanded results using elementary large deviations theory applied to the multi-stage stochastic programs we see here. This sample requirement is relatively common for Monte Carlo estimators [5].

2.3 Convergence

This is one area that Shapiro went significantly beyond Pereira and Pinto, as the original paper [1] did not explicitly consider convergence of the algorithm. The generous interpretation is that convergence is relatively trivial; the forward problem is solvable and the backward problem is solvable in a finite number of steps, then the algorithm would eventually converge because there are a finite number of scenarios to evaluate. This is the approach taken by [6], which formally extends convergence proofs from two-stage problems to multi-stage problems. However, this does not directly apply to the infinite scenario case presented by Shapiro.

Shapiro notes that the backward step of the SDDP algorithm is actually Kelley's cutting plane algorithm [7], which is a classical geometric method of solving linear programs. Kelley's algorithm is well-studied and is known to terminate in a finite number of steps [8].

Then, Shapiro defines one natural assumption on the system that is necessary to ensure convergence:

Assumption 1: *All stages of the problem (15) have finite optimal values for all realizations of the data ξ used in the sample averaging approximation problem (16).*

Recalling that the future cost Q and future cost approximation $\tilde{\mathcal{Q}}$ are convex and piecewise linear, this implies that the problem has a finite optimal value and an optimal solution. This leads Shapiro to the following proposition:

Proposition:*Suppose that in the forward steps of the SDDP algorithm the subsampling procedure is used, the above assumption holds. Assume also that in the backward steps, the average value over the cutting planes found by applying the cutting plane algorithm to the linear programs defined by the trial decisions. Then, with probability 1, after a sufficiently large number of forward and backward algorithms, the forward step procedure finds an optimal policy for the SAA problem.*

The proof of this proposition is essentially based on the fact that by sampling the distribution a finite number of times to make the sampled average approximation problem, the number of possible scenarios is reduced to a finite quantity. Then, by repeating the algorithm indefinitely, the forward step of the algorithm attempts every possible scenario with probability 1, hinging on the fact that the scenarios are sampled independently from each other.

3 Risk-Aversion

After extending the SDDP algorithm to systems with continuous probability distributions, Shapiro turns his attention to the question of risk-aversion in multi-stage stochastic optimization. The issue is that while the SDDP algorithm minimizes the average cost, some realizations of ξ may be much worse than the average performance. To handle this, he considers the following constraint,

$$\Pr\{Q(x,\xi) \leq \eta\} \geq 1-\alpha. \tag{17}$$

For simplicity of notation, he writes the constraint in terms of Value-at-Risk of a random variable Z,

$$\mathrm{V@R}_\alpha[Z] := \inf\{u : \Pr(Z \leq u) \geq 1-\alpha\}. \tag{18}$$

The issue with value-at-risk is that the function may not be convex, regardless of the convexity of Z. Instead of value at risk, we define conditional value-at-risk, as suggested by [9]:

$$\mathrm{CV@R}_\alpha[Z] := \inf_{u\in\mathbb{R}}\{u+\alpha^{-1}\mathbb{E}[Z-u]\} = \mathrm{V@R}_\alpha[Z]+\alpha^{-1}\mathbb{E}[Z-\mathrm{V@R}_\alpha(Z)]_+. \tag{19}$$

The minimum is attained when $u = \mathrm{V@R}_\alpha[Z]$, leading to the above equality. One useful property of $\mathrm{CV@R}_\alpha[Z]$ is that it preserves the convexity of Z, so $\mathrm{CV@R}_\alpha[Q(x,\xi)]$ would also be convex.

Once again, while $\mathrm{CV@R}_\alpha[Q(x,\xi)] \leq \eta$ could be be used as a constraint, Shapiro suggests moving it to the objective function as a modified term:

$$\rho_\lambda(Z) := (1-\lambda)\mathbb{E}[Z] + \lambda\mathrm{CV@R}_\alpha[Z], \tag{20}$$

where $\lambda \in [0,1]$ is viewed as a compromise between optimizing for average performance ($\lambda = 0$) and optimizing for risk mitigation. The modified objective is then

$$\min_{x \text{ feasible}} c^T x + \rho_\lambda[Q(x,\xi)], \tag{21}$$

which expands to

$$\min_{x \text{ feasible}, u\in\mathbb{R}} c^T x + \lambda u + \mathbb{E}\{(1-\lambda)Q(x,\xi) + \lambda\alpha^{-1}[Q(x,\xi)-u]_+\}. \tag{22}$$

We can then formulate this into a two-stage stochastic linear program:

$$\min_{A_1 x = b_1, u\in\mathbb{R}} c_1^T x + \lambda u + \mathbb{E}[V(x,u,\xi)], \tag{23}$$

with $V(\cdot,\cdot,\cdot)$ defined as the optimal solution of the linear program

$$V(x,u,\xi) = \min_{y} (1-\lambda)c_2^T y + \lambda\alpha^{-1}[c_2^T y - u], \tag{24}$$
$$\text{s.t. } B_1 x + A_2 y = b_2, \quad y \geq 0.$$

As this is simply a two-stage stochastic linear program, the analysis for the risk-neutral case can be (almost) directly applied. When this program is extended into a multi-stage problem, the expectations need to be replaced by the conditional risk operators

$$\rho_{t|\xi_{1:t-1}}[Z] := (1-\lambda_t)\mathbb{E}[Z|\xi_{1:t-1}] + \lambda_t \text{CV@R}_{\alpha_t}[Z|\xi_{1:t-1}], \tag{25}$$

giving the whole problem the nested formulation

$$\min_{\substack{A_1x_1=b_1\\x_1\geq 0}} c_1^T x_1 + \rho_{2|\xi_1}\left[\min_{\substack{A_2x_2+B_2x_1=b_2\\x_2\geq 0}} c_2^T x_2 + \rho_{3|\xi_{1,2}}\left[\cdots + \rho_{T|\xi_{1:T-1}}\left[\min_{\substack{A_Tx_T+B_Tx_{T-1}=b_T\\x_T\geq 0}} c_T^T x_t\right]\right]\right]. \tag{26}$$

Following this, he preceeds in a very similar manner as the risk-neutral analysis. He defines a backwards dynamic programming approach to find

$$Q_t(x_{t-1},\xi_t) = \inf_{x_t}\{c_t^T x_t + \mathcal{Q}_{t+1}(x_t) : B_t x_{t-1} + A_t x_t = b_t, x_t \geq 0\}, \tag{27}$$

where

$$\mathcal{Q}_{t+1}(x_t) := \rho_{t+1|\xi_{1:t}}[Q_{t+1}(x_t,\xi_{t+1})], \quad \mathcal{Q}_{T+1}(x_T) = 0. \tag{28}$$

Using the definition of CV@R_α (19), he refines this equation slightly to make a version that is more amenable to computation.

Then, he N samples the distribution of ξ to make the SAA problem for risk-sensitive optimization, once again turning a problem with infinite possiblities into a finite (but large) number of scenarios. Finally he defines the forward and backward steps of the SDDP algorithm for the modified problem, which are very similar to the risk-neutral case. In particular, for the backward steps, he finds a cutting plane using subdifferentials to create a piecewise linear approximation of the true future cost $\tilde{\mathcal{Q}}_t(\cdot)$, and he uses this approximation to find a lower bound.

To find an upper bound for a given time t, once again several samples of ξ are generated. Then, for a trial decision $\hat{x}_t$, the following equation is evaluated

$$\bar{x}_t = c_t^T \hat{x}_t + \tilde{\mathcal{Q}}_{t+1}(\hat{x}_t). \tag{29}$$

Once the lower bound and upper bound confidence interval are found, the same termination conditions can be used as in the risk-neutral case.

4 Applications

The original application of the SDDP algorithm was for planning the operation of a hydrothermal generating system [1]. Energy generation and storage is a

difficult problem because the usage of currently stored water to create hydro-electric power may cause issues in the future if there is a low inflow. The exact model for hydroelectric generation was given by their previous paper [10], which formulated the problem as a sequence of linear programs with the addition of multiple inquality constraints. The motivation for applying the SDDP algorithm to this problem was to reduce the curse of dimensionality, as discretizing each state into twenty values with only five reservoirs results in ten trillion possible states to evaluate. From here, they demonstrated that in only five iterations of the SDDP algorithm, the optimality gap of the system nearly vanished.

Many future works draw inspiration from the SDDP algorithm, and many of these even directly apply their research to energy generation. For example, Lohndorf et al. [11] combine approaches used in approximate dynamic programming with the SDDP algorithm to once again optimize a hydro-electric generator. The general idea of their paper is to use SDDP and approximate DP to construct an approximate polyhedron around their pos-decision variable to estimate the value function. Similarly, Scott and Read [12] apply dual dynamic programming techniques to an electricity market described by Cournot duopolies.

5 Impact

Pereira and Pinto's original paper has seen significant impact. A quick Google Scholar search shows that it has been cited by over one thousand articles, and is widely cited as an approach to address optimization problems with a very large number of scenarios. Many general optimization books and articles (such as [13],[14]) cite their paper and algorithm as one source of inspiration for stochastic optimization. In particular, this approach has seen use in the energy management sector, with some usage in financial engineering.

Somewhat surprisingly, the SDDP algorithm does not seem to have made a significant impact outside of optimization in the context of energy sectors, and even more specifically for the management of resevoirs in power systems. This may be because the form of problem solved (as in, sequential linear programs) may be a bit too specfic for general use.

Shapiro's paper, less than ten years old, has around three hundred citations. As expected, since the algorithm introduces a risk-averse SDDP derivation, the citations are more generally to do with risk management than with the energy sector. As the work is in general more theoretically focused than Pereira and Pinto's paper, more of the citing works are mathematical programming oriented papers, with less focus on the energy sector specifically.

6 Limitations of the method

There are a few general limitations to the SDDP approach. While it avoids the curse of dimensionality in terms of state associated with dynamic programming,

for a T stage problem taking N samples per stage, the method necessarily solves approximately $T \cdot N$ linear programming problems. This is made worse by the fact that the backwards step is a variant of Kelley's cutting plane algorithm, which is known to have a worst-case scenario whose computational difficulty increases exponentially in the number of states [15]. For this to be relaxed, a new algorithm to solve the backward steps more efficiently would need to be found. Currently, there is not an algorithm that could easily be modified to suit this problem. As with many optimization algorithms, this places a limit on how tractable the algorithm is for systems with large states or long run-times.

Another computational difficulty mentioned by Shapiro is that, while he does prove the sampled algorithm will converge in an indefinite amount of time when applied to the sample averaging approximation problem, the lower bound may increase very slowly. Even though the lower bound seems to converge, the actual problem may not be solved to optimality.

To be clear, a number of these computational issues can be mitigated by a careful implementation of the algorithm [16] (as is expected, because many worst-case scenarios are pathological and require strange constructions), but it is still a concern.

One possible theoretical improvement is to remove the requirement on stage-wise independence, as for many systems the noise is not independent. In some situations can be addressed (i.e. in linear-Gaussian systems, colored noise can be turned into white noise by expanding the states of the system), but this is not always the case. Unlike the computational issues above, this seems like a straightforward area for direct improvement of the algorithm.

Another potentially useful development would be to extend the dual dynamic programming approach to general nonlinear convex models. However, this would almost certainly exacerbate the computational problems of the algorithm, and it would introduce many new theoretical difficulties with establishing error bounds.

7 Conclusion

The stochastic dual dynamic programming algorithm is an interesting approach to solving sequential linear programming problems. In this paper, we discussed the original implementation of Pereira and Pinto's algorithm, going into detail on the two-stage problem, then we discussed how Shapiro's paper showcased theoretical improvements on the original implementation, as well as a derivation of the algorithm for risk-sensitive applications. From this, we learned how results from classical convex optimization such as duality can be applied to optimal control, even when it is not clear how the two fields are more than tangentially related. In time, hopefully extensions of this method to more general models will be derived, further allowing the two fields to intermingle.

References

[1] M. V. F. Pereira and L. M. V. G. Pinto, "Multi-stage stochastic optimization applied to energy planning," *Mathematical Programming*, vol. 52, pp. 359–375, May 1991.

[2] S. Gal, "The Parameter Iteration Method in Dynamic Programming," *Management Science*, vol. 35, no. 6, pp. 675–684, 1989. Publisher: INFORMS.

[3] A. Shapiro, "Analysis of stochastic dual dynamic programming method," *European Journal of Operational Research*, vol. 209, pp. 63–72, Feb. 2011.

[4] A. Shapiro, "On complexity of multistage stochastic programs," *Operations Research Letters*, vol. 34, pp. 1–8, Jan. 2006.

[5] A. Shapiro, "5. Statistical Inference," in *Lectures on Stochastic Programming*, MOS-SIAM Series on Optimization, pp. 155–252, Society for Industrial and Applied Mathematics, Jan. 2009.

[6] A. Philpott and Z. Guan, "On the convergence of stochastic dual dynamic programming and related methods," *Operations Research Letters*, vol. 36, pp. 450–455, July 2008.

[7] J. E. Kelley, "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. Publisher: Society for Industrial and Applied Mathematics.

[8] A. Ruszczyński, "Decomposition Methods," in *Handbooks in Operations Research and Management Science*, vol. 10 of *Stochastic Programming*, pp. 141–211, Elsevier, Jan. 2003.

[9] R. T. Rockafellar and S. Uryasev, "Optimization of conditional value-at-risk," *Journal of Risk*, Mar. 2000.

[10] M. V. F. Pereira and L. M. V. G. Pinto, "Stochastic Optimization of a Multireservoir Hydroelectric System: A Decomposition Approach," *Water Resources Research*, vol. 21, no. 6, pp. 779–792, 1985. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/WR021i006p00779.

[11] N. Löhndorf, D. Wozabal, and S. Minner, "Optimizing Trading Decisions for Hydro Storage Systems Using Approximate Dual Dynamic Programming," *Operations Research*, vol. 61, pp. 810–823, Aug. 2013.

[12] T. J. Scott and E. G. Read, "Modelling Hydro Reservoir Operation in a Deregulated Electricity Market," *International Transactions in Operational Research*, vol. 3, no. 3-4, pp. 243–253, 1996. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1475-3995.1996.tb00050.x.

[13] N. V. Sahinidis, "Optimization under uncertainty: state-of-the-art and opportunities," *Computers & Chemical Engineering*, vol. 28, pp. 971–983, June 2004.

[14] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, Oct. 2007. Google-Books-ID: WWWDkd65TdYC.

[15] Y. Nesterov, "Nonsmooth Convex Optimization," in *Introductory Lectures on Convex Optimization: A Basic Course* (Y. Nesterov, ed.), Applied Optimization, pp. 111–170, Boston, MA: Springer US, 2004.

[16] V. L. de Matos, A. B. Philpott, and E. C. Finardi, "Improving the performance of Stochastic Dual Dynamic Programming," *Journal of Computational and Applied Mathematics*, vol. 290, pp. 196–208, Dec. 2015.