Published online ahead of print September 15, 2011

INFORMS Journal on Computing
Articles in Advance, pp. 1–18
ISSN 1091-9856 | EISSN 1526-5528

http://dx.doi.org/10.1287/ijoc.1110.0470
© 2011 INFORMS

SMART: A Stochastic Multiscale Model for the Analysis of Energy Resources, Technology, and Policy

Warren B. Powell, Abraham George, Hugo Simão, Warren Scott
Department of Operations Research and Financial Engineering, Princeton University, Princeton, New Jersey 08544
{powell@princeton.edu, abegeorge@gmail.com, hpsimao@princeton.edu, wscott@princeton.edu}

Alan Lamont, Jeffrey Stewart
Science and Technology Directorate, Lawrence Livermore National Laboratory, Livermore, California 94550 {e504786@mail.llnl.gov, stewart28@llnl.gov}

We address the problem of modeling energy resource allocation, including dispatch, storage, and the long-term investments in new technologies, capturing different sources of uncertainty such as energy from wind, demands, prices, and rainfall. We also wish to model long-term investment decisions in the presence of uncertainty. Accurately modeling the value of all investments, such as wind turbines and solar panels, requires handling fine-grained temporal variability and uncertainty in wind and solar in the presence of storage. We propose a modeling and algorithmic strategy based on the framework of approximate dynamic programming (ADP) that can model these problems at hourly time increments over an entire year or several decades. We demonstrate the methodology using both spatially aggregate and disaggregate representations of energy supply and demand. This paper describes the initial proof of concept experiments for an ADP-based model called SMART; we describe the modeling and algorithmic strategy and provide comparisons against a deterministic benchmark as well as initial experiments on stochastic data sets.

Key words: artificial intelligence; simulation; statistical analysis; analysis of algorithms; queues
History: Accepted by David Woodruff, Area Editor for Heuristic Search and Learning; received August 2009; revised August 2010, April 2011; accepted April 2011. Published online in *Articles in Advance*.

1. Introduction

Over the next few decades, the United States needs to transition to an energy system with much lower carbon emissions and reduced dependence on unreliable supplies. This process requires significant investment in research and development (R&D) and new infrastructure. The investments have to be made in the presence of multiple sources of uncertainty that could influence the future energy system, such as the mix of primary energies (nuclear, renewable, carbon sequestration); the balance of energy carriers, including electricity, hydrogen, and alcohol and synthetic fuels; and end-use strategies (level of demand, demand response). A major component will be renewable energy that depends on intermittent sources such as wind and solar. The volatility of these sources can be mitigated with different types of storage such as batteries and pumped hydro.

The dynamics of our system are driven by exogenous factors (wind, solar, climate, technology, prices, supplies, and demands) and market decisions (how much energy should be produced by each source, how much of this energy should be used for each type of demand, how much energy should be stored, and how much capacity should be added or retired each year in response to changing conditions). To model investment decisions and government policies, the model has to extend over a multidecade horizon. To accurately represent the economics of energy technologies, our model has to capture the hourly fluctuations of wind, daily solar cycles, seasonal climate changes, and annual changes in energy technologies. Finally, it is widely understood that a key enabling technology for intermittent energy is storage, which complicates the design of algorithms by coupling all the time periods together.

Past models have represented market investment decisions by assuming that they can be reasonably approximated as optimal solutions to deterministic linear programs. These models have been effective at capturing fundamental differences in supply and demand, where the most famous example is the groundbreaking policy conclusions from the PIES model (Hogan 1975). In recent years, there has been growing interest in the use of models to guide investment and policy decisions in new technologies such as wind and solar and to guide government policies for carbon taxes and R&D to meet goals such as 20% energy from renewables by 2030. It has long been recognized, however, that models that represent intermittent sources such as wind and solar overstate their value if they use averages (wind turbines are dramatically more productive if wind blows at a constant speed). The problem with intermittent sources is that they are not dispatchable (directly controllable in real time), requiring investments in other energy technologies to make up for low periods, even if the average production is very high (see Lamont 2008 for a thorough discussion of these issues). Energy policy researchers have found that using time increments longer than an hour begin to introduce significant errors. Storage can overcome the limitations of intermittent sources, but representing storage imposes additional demands on a model.

This paper extends a deterministic energy planning model called META$*$Net (see Lamont 1997, 2008), developed at Lawrence Livermore National Laboratory, which has been used in two forms: a model that captures hourly variations over a year and a model that extends for several decades in increments of one to five years. META$*$Net is an aggregate model, which captures approximately 13 different sources of energy to satisfy 9 types of demand, spanning different types of electric power as well as different energy forms for industrial and residential purposes. Aggregate planning models such as these stand between simple models that analyze the economics of different types of energy generation in isolation and detailed network models used in operational planning for utilities and grid operators.

Separate from the need to handle fine-grained time scales is the need to capture different types of uncertainty. This ranges from uncertainty about wind (wind operators often have to commit to a certain level of production a day in advance) to uncertainties about policies (imposition of a carbon tax), technology (whether will we be able to sequester carbon), prices (what the price of oil will be in five years), and climate (how global warming will affect snow accumulation and rainfall). These uncertainties can affect the risks incurred in an investment, significantly reducing the appeal of technologies that look attractive if the future unfolds as expected.

Our goal is to develop a model that is stochastic (handling different sources of uncertainty) and multiscale (handling different levels of spatial and temporal granularity). We also need to incorporate storage, which has the effect of linking different time periods together. Finally, we are interested in near-optimal policies, which we achieve using approximate dynamic programming (ADP). This dimension provides a sharp contrast with other papers that may use a stochastic *model* but do not account for uncertainty when making a decision. We use deterministic benchmarks to demonstrate the quality of our solutions, but we are limited to qualitative evaluations to assess our ability to properly reflect uncertainty in our policies.

These goals represent what we feel we are able to achieve in this paper, but there are other goals to which we aspire. We are able to handle fine-grained sources of uncertainty (e.g., wind, rainfall, prices, demand), but we would like to handle coarse-grained forms of uncertainty, which include changes in tax policy or breakthroughs in technology. We can handle spatially disaggregate supply and demand patterns, but we cannot model individual power plants; this requires the introduction of integer variables that govern when a plant is in use or not.

This paper describes an early implementation of *SMART* (Stochastic Multiscale model for the Analysis of energy Resources, Technology, and policy). SMART uses a control-theoretic framework to step forward through time in hourly increments over a multidecade horizon, which makes modeling energy storage possible. Approximate dynamic programming is used to produce policies that properly model the flow of information when making decisions. Although designed to handle stochastic problems, SMART can be applied to deterministic problems, allowing us to compare the performance of the model to a deterministic linear programming model when such a model can be solved. We show that SMART closely matches the optimal solution from a deterministic linear program, which provides an important benchmark for our ADP algorithm. We also show that it exhibits robust behaviors when applied to a stochastic problem. To the best of our knowledge, this is the first stochastic multiscale model that scales to handle hundreds of thousands of time periods while producing solutions that closely match the results of optimal solutions for deterministic formulations.

SMART can be used for both long-term policy studies and economic analyses of portfolios of energy production and storage technologies. It can be used to simulate energy investment decisions—under the assumption that a cost-minimizing model mimics market behavior—or as a normative model to determine how to reach specific energy targets. It can be used to quantify risk and to study the impact of different energy investment strategies. It can also provide accurate estimates of the value of intermittent energy sources and the value of storage using realistic assumptions about our ability to forecast energy supply.

The rest of this paper is organized as follows. Section 2 reviews the literature for models on energy policy, and it briefly touches on the literature for energy storage and wind. Section 3 provides a sketch of our modeling and algorithmic strategy. An outline of the mathematical model is described in §4, but the full model is contained in the Online Supplement (available at http://joc.pubs.informs.org/ecompanion.html), which contains a number of important equations to which the paper refers. Our model captures both the physical dynamics and the flow of information and decisions, but it does not provide a method for making decisions. Section 5 provides two optimization formulations to drive the process of making decisions: a deterministic linear programming model (which we later use as a benchmark) and a stochastic optimization model. Section 6 describes an algorithmic strategy for solving the stochastic optimization problem based on approximate dynamic programming. In §7, we describe benchmarking experiments that compare the solution obtained using approximate dynamic programming to the optimal solution produced by a linear programming algorithm using a deterministic model. Experiments on stochastic applications are also reported to demonstrate specific behaviors that we would expect from a nonanticipative decision rule. Section 8 concludes the paper.

2. Literature Review

There are several bodies of literature that are relevant to our research: deterministic energy investment models, stochastic optimization models in energy, models for energy storage, and studies of the value of energy from wind.

Following the original PIES model (Hogan 1975), a series of deterministic optimization models have been developed to model energy investment decisions over multidecade horizons. These include the National Energy Modeling System (NEMS) (2003) that takes as input demands and costs of various energy resources from a family of models. The MARKet ALlocation (MARKAL) model is a deterministic, optimization-based system that integrates the supply and end-use sectors of an economy (see Fishbone and Abilock 1981, Loulou et al. 2004). MARKAL minimizes the discounted total cost of the system, which includes investment, maintenance, and operations costs of the technologies and the purchase costs of fuels, and it also accounts for revenue generated. META$*$Net (see Lamont 1997, 2008) combines the techniques used in NEMS as well as in MARKAL in that it performs market equilibration in a linear programming-based model.

The early stochastic energy investment models can be divided into two categories. We refer to the first category as partially stochastic models, such as the stochastic version of MARKAL (Johnson et al. 2006) developed by the Environmental Protection Agency. In this version of MARKAL, a "stochastic wrapper" is used, wherein random parameters are sampled from probability distributions, after which a deterministic optimization problem is solved. This process is repeated a number of times, allowing the outputs to be analyzed statistically. This means that for each run on a sample realization, decisions are allowed to see events in the future. Kanudia and Loulou (1998) present a version of MARKAL that explicitly incorporates multiple scenarios and includes as many replications of the MARKAL variables as the number of scenarios considered. The optimization is done by minimizing total expected costs over all the scenarios but where the solution for each scenario is independent of the other scenarios. Again, this strategy allows a decision to see outcomes in the future for a particular scenario.

Another example of a partially stochastic model is the stochastic version of MESSAGE (Model for Energy Supply Strategy Alternatives and their General Environmental impact) from the International Institute for Applied Systems Analysis. The conventional MESSAGE model (Messner and Strubegger 1995) uses point estimates to forecast technology parameters. The stochastic MESSAGE model (Messner et al. 1996) aims to reduce the risk of underestimating the total investment cost as a stochastic optimization problem, using a sufficiently large number of sample realizations of the investment costs.

The second category of stochastic models uses the framework of stochastic programming to handle the modeling of random information and decisions (see Wallace and Fleten 2003 for an excellent introduction to these models). Stochastic programming models use the notion of scenario trees to represent uncertain information. The challenge with this framework is that the size of a scenario tree grows exponentially with the number of time periods, requiring the use of aggregation strategies such as large time steps and small numbers of scenarios per time period. See Birge and Louveaux (1997) for an in-depth treatment of this approach, and see Høyland and Wallace (2001), Gröwe-Kuska et al. (2003), and Kaut and Wallace (2003) for research on generating scenario trees. A competing strategy uses dual information to generate cuts to approximate the impact of decisions made now on the future. This was originally introduced as the L-shaped method (Van Slyke and Wets 1969) and was later adapted (for two-stage problems) using a Monte Carlo-based framework as stochastic decomposition in Higle and Sen (1991) (see Higle and Sen 1996 for a more in-depth treatment).

One limitation of models based on stochastic programming is that the scenarios (outcomes of random information) cannot depend on the sequence of decisions. For example, if decisions lead to energy shortages, prices may spike. Stochastic programming requires that scenarios be generated in advance, and this limits their ability to capture the interaction between decisions and exogenous events. However, stochastic programming is able to capture complex, history-dependent processes such as the timing of a carbon tax or breakthroughs in technology.

The last class of stochastic energy investment models can be viewed as pure simulation models. Perhaps the most visible example of this class is the Stochastic Energy Deployment Systems (SEDS) (see http://seds.nrel.gov/), which uses classical Monte Carlo simulation to step forward through time. Let ω^n represent the sample realization of all random variables for the nth iteration of the model, and let $c_{te}(\omega^n)$ be the cost of energy technology $e \in \mathscr{E}$ in year t while following sample path ω^n. SEDS makes additional investments, given by x^n_{te}, according to the expression

$$x^n_{te} = \beta^D_t \frac{\exp -\beta^s c_{te}(\omega^n)}{\sum_{e' \in \mathscr{E}} \exp -\beta^s c_{te'}(\omega^n)}, \tag{1}$$

where β^D_t is a set of parameters that (over time) allows investments to be scaled to demand, and β^s is a scaling coefficient that controls the degree to which investments are directed toward lower-cost technologies. There is no attempt to formally optimize energy investments. A major goal of the model is to provide fast estimates of the effects of policy changes in a fairly transparent way. The model works in yearly (or even multiyear) increments, so there is no attempt to capture fine-grained variations in wind and solar energy or demand.

There is a separate literature on the use of storage, largely divided between the management of water reservoirs and other forms of energy storage. The literature on water reservoir management is fairly extensive (see, e.g., Foufoula-Georgiou and Kitanidis 1988, Lamond and Sobel 1995, Archibald et al. 1997, and Cervellera et al. 2005). Algorithmic research addressing the problem of optimizing flow controls under uncertainty has progressed along two lines. The first uses dynamic programming, typically resorting to some sort of numerical or approximate strategy to handle the curse of dimensionality that arises with multiple reservoirs (see Nandalal and Bogardi 2007 and the references cited there). A major breakthrough in this approach involves approximating the impact of decisions made now on the future by using cuts generated from the dual of the linear program in the next time period. Pereira and Pinto (1991) were the first to use this strategy in an energy setting under the name of stochastic dual dynamic programming, modeling a large, multireservoir hydroelectric system. A separate modeling and algorithmic strategy within the stochastic programming community uses the concept of scenario trees, where sample paths of exogenous information are generated, retaining the entire history at each node in the tree. Jacobs et al. (1995) and De Ladurantaye et al. (2007) give nice illustrations of this strategy in a hydroelectric setting. Heuristic approaches to reservoir management are described in Johnson et al. (1991) and Edmunds and Bard (1990a, b).

A separate literature has evolved around the modeling of storage devices to handle either structural or random variations between our ability to generate power and the demand for power. Brunetto and Tina (2007) use a deterministic model to solve the problem of day-ahead electricity markets, completely ignoring the high level of uncertainty that arises in this setting. Castronuovo and Lopes (2004) describe wind as a stochastic process but then represent only the variability of wind, not its uncertainty. Korpaas et al. (2004) and Paatero and Lund (2005) also use deterministic models to study storage for wind. Garcia-Gonzalez et al. (2008) and Brown et al. (2008) use the idea of stochastic wind scenarios, but allow decisions to "see" the wind in the future within a particular scenario (violating what are known as nonanticipativity conditions).

There is a body of literature that is working to understand the impact of large amounts of wind on the power grid. Sioshansi and Short (2009) use a two-day deterministic forecast of wind in a rolling horizon model to simulate decisions. Sioshansi (2010) uses a deterministic unit commitment model (requiring a deterministic forecast of wind) to study the impact of real-time pricing on the use of energy from wind. Holttinen et al. (2011) undertake a careful study of the effect of large quantities of wind (also see the summary in Holttinen et al. 2009). However, no formal mathematical models are given (this includes the doctoral dissertation on which this work is based—Holttinen 2004), and it does not appear that the study uses a policy that explicitly accounts for the uncertainty in wind. Separate from the issue of handling uncertainty is the challenge of capturing hourly variations in long-term models. A modeling strategy that avoids capturing every hour over a year (8,760 time periods) involves simulating only a sample of different time periods (capturing, for example, different times of the year and different times of the day), but such an approach makes it impossible to accurately model storage (whether it is hydro, compressed air, or batteries), which requires the ability to step forward across all time periods to calculate storage levels.

3. Strategy for a Stochastic Multiscale Model

The goal of our research is to develop a model that allows us to make short-term decisions about electric dispatch, energy resource allocation, and storage over the short term (one year) as well as long-term investment decisions in the presence of different sources of uncertainty. We are interested in questions such as the impact of increasing the fraction of energy generated from wind and solar and in obtaining an accurate estimate of the marginal value of wind and solar in the presence of storage. At the same time, we are not looking to address decisions such as where to locate a wind farm, or whether a particular coal or natural gas facility should be used at a particular point in time. These details are important for operational problems, such as unit commitment, or specific investments, such as additions to the grid or energy facilities at specific locations. Our model does not represent individual energy generators because the resulting model would have to include thousands of integer variables per time period. Even without these details, we face a difficult algorithmic challenge.

We anticipate running our model over two time frames. For studies demonstrating the effect of the impact of storage on the marginal value of wind and solar, it makes sense to run the model in hourly increments over a year. For studies on the impact of tax subsidies, commodity prices, and breakthroughs in technology, we will want to extend the horizon over several decades while retaining the ability to model variations in wind, solar, and demand in hourly increments. For problems where we are only interested in modeling a single year, we may be interested in a spatially disaggregate model, whereas for long-term investment decisions, a spatially aggregate model may be sufficient.

We use the modeling and algorithmic framework of approximate dynamic programming where dispatch, storage, and investment decisions at hour h in year t are represented by the vector x_{th}, which is determined by a decision function (or *policy*, in the language of dynamic programming) $X^{\pi}(S_{th})$, which depends on the system state variable S_{th}. The term S_{th} is designed to include only the information available at time (t, h), and as a result, decisions are not allowed to anticipate events in the future.

Our strategy involves stepping forward through time as a simulation model would, but it repeats the process while learning from each iteration. Once a year (corresponding to $h = 0$), we make investment decisions for new energy capacity that will arrive in a future year. Then, for the remainder of the year, we step forward to solve hourly dispatch problems and thereby determine how much energy to produce from each source to satisfy each type of demand. Energy demands comprise residential, commercial, and industrial demands for electricity and natural gas (residential uses include cooking and heating), as well as freight transportation and aircraft demands for electricity, hydrogen, and hydrocarbon fuels. Primary energy resources include coal, petroleum, natural gas, wind, solar, temperature gradients (geothermal, ocean, thermal), nuclear, biomass, and tides. These undergo conversion processes in plants to generate electricity, hydrocarbon fuels, or hydrogen, each of which is then channeled to the appropriate sources of demand. Whereas most of the conversion plants that handle carbon-based energy sources release CO_2 to the atmosphere, some technologies are able to sequester CO_2. The dispatch problem also includes decisions of how much to store and how much to convert to different types of energy (for example, natural gas can be used to generate electricity, which in turn can be used to convert biomass to fuel or to compress hydrogen).

We demonstrate SMART using two networks. The first is a spatially aggregate model, a portion of which is illustrated in Figure 1, that represents 13 different types of energy supplies and 9 types of energy demand. If there is a shortage of energy supplied from these resources, there also exists the option of importing electricity. Each node in the network represents an aggregated version of a facility in a typical energy network. Resource nodes provide the supply of raw resources (such as coal, natural gas, wind, solar, biomass, and nuclear) at a certain cost. Conversion nodes take as inputs the raw resource and some type of power needed to generate more power in some form such as electricity, natural gas, and transportation fuel. The flows out of each of these nodes are constrained by the capacity available at that node, which is controlled by investment decisions. Demand nodes capture the demand for one of the various types of power. Market nodes accumulate energy from different sources to be distributed to the various types of demand (these can be sectors or locations). The network also captures interactions such as the need to produce more electricity if it is needed to produce hydrogen or ethanol. We may also add other nodes in this illustration to capture costs involved in the transportation of the energy resource (for example, from the source to the conversion plant) and the distribution of the final output (for example, electricity through transmission lines).

The second network model, used in a study of intermittent energy by Tagher (2010), captures the generation of electricity from coal, nuclear, natural gas, and wind from each of the 48 contiguous states to satisfy different demand sectors in each of those states plus the District of Columbia. Unlike our aggregate network model, this model only considers electricity. The model did not have an explicit representation of the grid, but state-to-state distances were used to compute transportation costs and transmission losses, using individual links for state-to-state transmission. This representation of power flows, which captures

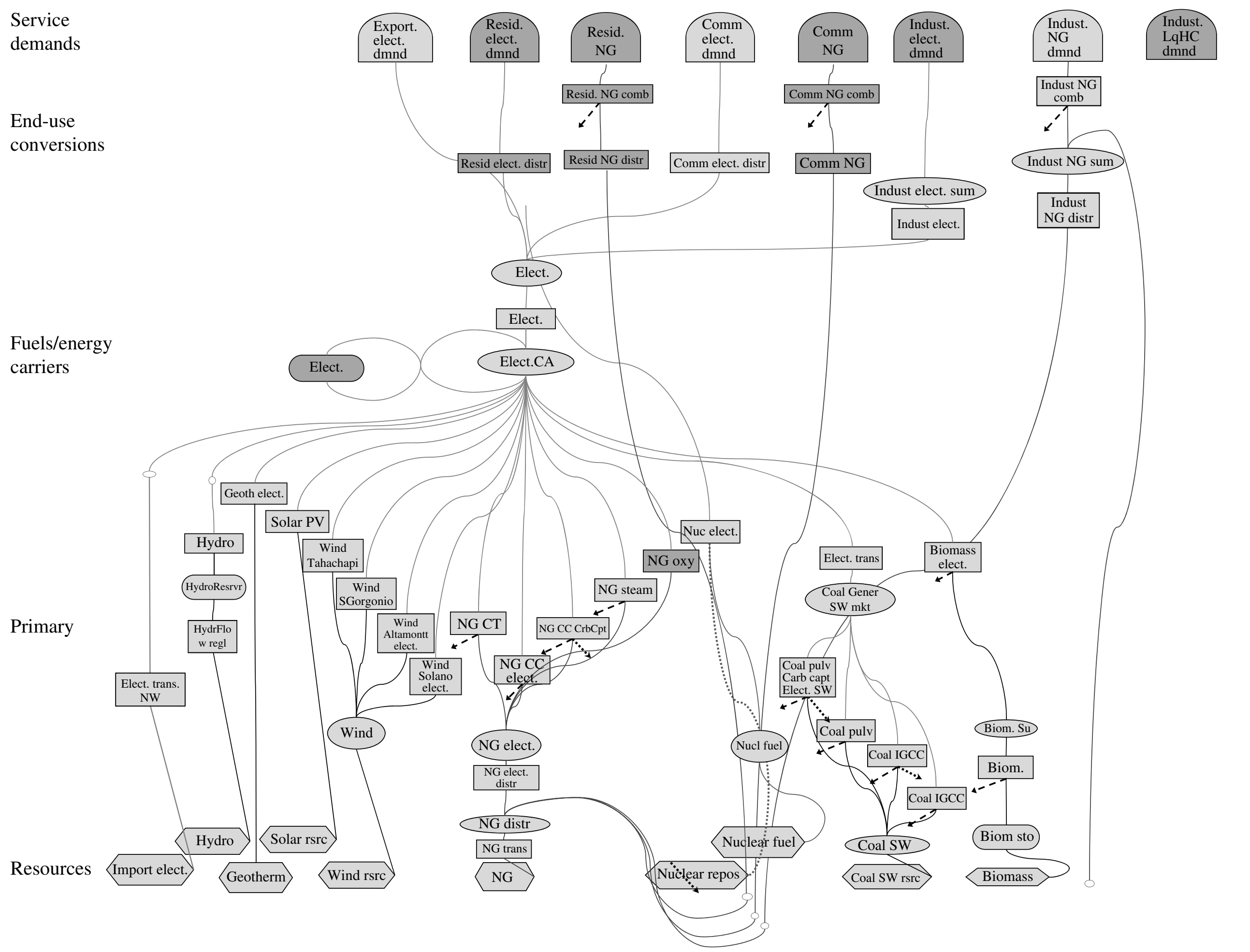

Figure 1 Illustration of the Energy Distribution Network

point-to-point movements without explicitly modeling the grid, is a common approximation made in policy studies. The linear program for a single time period had approximately 40,000 rows and columns.

We do not model individual energy generators nor do we model the grid itself. This level of detail is required for operational models and for certain types of regional planning models that require the identification of bottlenecks in the grid. Including this detail in a model that spans 20 years in hourly increments is a goal of the energy modeling community, but we are not ready to claim that we can handle a model with this granularity. We include transmission losses at an aggregate level, and we can model limits on interregional flows (as the CAISO model does). The model can be used to compare the economics of wind versus shale gas, coal, and nuclear under different taxation policies and targets for renewables.

Our modeling and algorithmic strategy involves simulating our way through the time periods, solving small optimization problems to handle dispatch and storage decisions (each hour) and investment decisions (each year). As we step forward through time, we use Monte Carlo simulation to sample realizations from any random variables. Our challenge is designing a decision function $X^\pi(S_{th})$ that produces economically rational decisions without violating restrictions on the availability of information.

We begin by providing an outline of the model of the problem in §4 (the complete model is given in §A.1 of the Online Supplement), followed by both deterministic and stochastic optimization formulations in §5. The mathematical model covers both the aggregate and disaggregate networks.

4. Outline of the Energy Model

We let $t \in \mathcal{T}$ index years, and we let $0 \le h < 8{,}760$ index the hours within a year. Our model consists of the following five fundamental components:

S_{th} = The state of the system at time (t, h).

x_{th} = The vector of decisions at time (t, h).

$W_{th} =$ The vector of random variables representing all the new information that just became available in the hour prior to (t, h).

$S^M(S_{th}, x_{th}, W_{t,h+1}) =$ The transition function (or system model) that governs the evolution from state S_{th} to $S_{t,h+1}$ (or from $S_{t,H}$ to $S_{t+1,0}$), given decision x_{th} and new information $W_{t,h+1}$.

$C_{th} =$ The contribution function capturing rewards earned during the hour preceding (t, h).

The state variable consists of the current status of our energy investments R_{th}, energy held in different forms of storage y_{th}, energy demands D_{th}, rainfall levels p_{th}, and a generic "state-of-the-world" vector ρ_{th} that captures information about technology, climate, government policy, and the availability of intermittent energy such as wind and solar.

We let the attributes of energy resources or storage be represented by the vector a. In our aggregate model, a would represent the type of technology, its location, and age. For the disaggregate model, we add the attribute "state" to capture geography. We let $\mathcal{A}$ be the set of all possible attribute vectors, and we let $\mathcal{A}^{\text{Conv}}$ and $\mathcal{A}^{\text{Stor}}$ represent the sets of attribute vectors that correspond to conversion and storage resources, respectively. We let R_{tha} be the level of investment in a technology with attribute $a \in \mathcal{A}$ (for example, this could be the megawatt-hours of generating capacity of a wind farm), and we let $R_{th} = (R_{tha})_{a\in\mathcal{A}}$ be the vector of energy investments. The vector $y_{th} = (y_{tha})_{a\in\mathcal{A}^{\text{Stor}}}$ captures the amount of energy in each type of storage (note that our numerical work includes only hydrostorage). It is easy to see that the disaggregate model, with the inclusion of a single spatial attribute, dramatically increases the dimensionality of the vector R_{th}.

The decision variable x_{th} consists of the vector of investments in energy conversion capacity x_{th}^{cap} and dispatch decisions x_{th}^{disp}. We assume that we make decisions on energy investments only once a year at hour $h = 0$, which means we are only interested in x_{t0}^{cap}, with $x_{th}^{\text{cap}} = 0$ for $h > 0$. Energy dispatch decisions govern the flow of energy from each source node (e.g., a coal mine or a source of natural gas), through conversion nodes (coal, nuclear, and oil plants), through various transmission and storage nodes, and finally to demand nodes. A portion of the aggregate network is shown in Figure 1. This is the network at a point in time (a particular hour). These networks are linked over time solely through the deposits to and withdrawals from storage devices. We let a be the attributes of a node in the dispatch network, which would represent moving energy over a link in the network. We let $x_{th,aa'}$ be the amount of energy moved from node a to node a'. We let $\theta_{aa'}$ capture transmission and storage losses when moving from a to a'; θ can also handle changes in units (e.g., from barrels of oil to megawatt-hours of energy).

Our decisions are governed by physical constraints. For example, there are conservation of flow constraints in the dispatch network as well as upper bounds on the amount of energy that can be stored in a particular device or location at a point in time. The flow conservation constraints are given by

$$\sum_{a'\in\overleftarrow{\mathcal{A}}(a)} \theta_{a'a} x_{th,a'a}^{\text{disp}} - \sum_{a''\in\overrightarrow{\mathcal{A}}(a)} x_{th,aa''}^{\text{disp}} = 0 \quad \text{for nonstorage locations,} \tag{2}$$

$$y_{tha} + \sum_{a'\in\overleftarrow{\mathcal{A}}(a)} \theta_{a'a} x_{th,a'a}^{\text{disp}} - \sum_{a''\in\overrightarrow{\mathcal{A}}(a)} x_{th,aa''}^{\text{disp}} \geq 0 \quad \text{for storage locations.} \tag{3}$$

We let $\mathcal{X}_{th}$ be the feasible region at time (t, h), and we require that $x_{th} \in \mathcal{X}_{th}$.

Once we have made a decision, the system then evolves over time, with new information arriving that also changes the state of the system. We view new information as a random variable that represents some sort of exogenous change to one of our state variables. Thus, $\hat{R}_{th}$ can be exogenous changes to our energy investments (e.g., Hurricane Katrina taking drilling rigs out of commission), $\hat{D}_{th}$ can be random changes in demand for different types of electricity, $\hat{p}_{th}$ can be the random change in the amount of rainfall, and $\hat{\rho}_{th}$ can be a change in technology, climate, or policy (e.g., the government just imposed a carbon tax). These changes may take place on hourly time scales but will often occur on longer time scales (we model technology as changing only once per year). We let W_{th} capture the vector of all these sources of random information that occurred in the hour preceding (t, h).

We later need some notation for describing a set of sample realizations. Let Ω be the set of all possible realizations of the sequence $(W_{11}, \ldots, W_{1H}, W_{21}, \ldots, W_{2H}, \ldots, W_{TH})$, where T is the number of years and H is the number of time periods in a year. We let $W_{th}(\omega)$ be a sample realization of the random variables in W_{th} when we are following sample path ω. If we are following sample path ω^n for iteration n, we will index variables such as the state variable using S_{th}^n to represent the actual state at time (t, h) while following sample path ω^n.

All of the physics of the problem are captured by the transition function (also known as the system model, plant model, or model) using

$$S_{t,h+1} = S^M(S_{th}, x_{th}, W_{t,h+1}), \quad h = 0, 1, \ldots, H-1,$$
$$S_{t+1,0} = S^M(S_{tH}, x_{tH}, W_{t+1,0}).$$

For example, changes in R_{th} and ρ_{th} would be written as

$$R_{t,h+1} = R_{th} + x^{\text{cap}}_{th} + \hat{R}_{t,h+1}, \tag{4}$$

$$\rho_{t,h+1} = \rho_{th} + \hat{\rho}_{t,h+1}. \tag{5}$$

Our model assumes that capacity investment decisions made at the beginning of one year arrive at the beginning of the next year. In practice, an investment may take several years before it comes online. Even assuming a one-year delay requires that we make these decisions under uncertainty. If we wish to capture multiyear delays, we have an instance of a lagged asset acquisition problem. This issue has been addressed in an ADP setting in Godfrey and Powell (2002b), Topaloglu and Powell (2006), and Nascimento and Powell (2009).

We note that it is possible to capture the dependence of exogenous information on the state of the system. For example, if we overinvest in ethanol capacity or wind turbines, we would expect that prices might drop. We did not explicitly do this in our numerical work, but the extension is a minor one. We may also have processes (e.g., prices, wind, demand) with autocorrelation. When this happens, we have to capture at least some of the history of these processes in the state variable. It is easy to incorporate these issues in the transition functions above, but it can complicate the challenge of designing good policies.

We measure our performance by adding up costs over time. Total capacity acquisition costs, dispatch costs, fuel costs, and operating costs are given by

$C^{\text{cap}}_{th}(S_{th}, x^{\text{cap}}_{th}) =$	The incremental capital costs in hour h of year t, and
$C^{\text{disp}}_{th}(S_{th}, x^{\text{disp}}_{th}) =$	The total costs resulting from the dispatch of energy in hour h of year t.

We may express the total cost function in hour h of year t using

$$C_{th}(S_{th}, x_{th}) = C^{\text{cap}}_{th}(S_{th}, x^{\text{cap}}_{th}) + C^{\text{disp}}_{th}(S_{th}, x^{\text{disp}}_{th}).$$

We have not addressed the problem of how to make decisions. In the next section, we present two models, a deterministic linear programming model and a stochastic model, that lead to different methods for making a decision.

5. Optimization Formulations

Our modeling framework left unanswered the mechanism for actually making decisions. Section 5.1 describes a deterministic linear programming formulation that can be solved by using commercial packages. Section 5.2 provides a stochastic optimization formulation for which exact solution algorithms are not available. The remainder of this paper then focuses on designing an approximation strategy to solve the stochastic optimization model.

5.1. A Deterministic Linear Programming Model

If we assume that all the exogenous information is deterministic (and known in advance), we can formulate the decision problem as a single (albeit very large) linear program. The objective function is given by

$$\min_x \sum_{t\in\mathcal{T}} \sum_{h\in\mathcal{H}} C_{th}(S_{th}, x_{th}).$$

This has to be solved subject to two sets of constraints. The first set governs decisions made at a point in time, which is given by Equations (14)–(19) in §A.1 of the Online Supplement. The second set of equations are the transition equations that link activities over time (basically the transition function), which are captured by Equations (20)–(24) in §A.1 of the Online Supplement. It is significant that the linear programming model views all these constraints together over all points in time. This is why a linear programming model (even with deterministic formulations) quickly becomes too large to solve. For example, a 20-year, spatially aggregate model with hourly dispatch decisions produced a constraint matrix with more than 34 million rows and 30 million columns.

5.2. A Stochastic Optimization Model

The linear programming model suffers from two important limitations. First, it is unable to handle longer planning horizons while retaining the ability to model intermittent supply and demand on an hourly level (along with storage). Second, it cannot handle the different forms of uncertainty that arise in energy policy analysis. For this reason, we pose the problem of controlling the stochastic version of the problem in terms of finding a policy (or decision function) to make decisions about dispatch, storage, and investments over time. We represent the decision function using

$X^{\pi}_{th}(S_{th}) =$	A function that returns a decision $x_{th} \in \mathcal{X}_{th}$, given the state S_{th} under the policy $\pi \in \Pi$.

The set Π refers to the set of potential decision functions, or policies, whose meaning becomes clearer below. At this point, we refer abstractly to the space of all possible decision functions.

The problem being stochastic, our goal is to find the optimal policy from among the set Π of policies by solving

$$\min_{\pi\in\Pi} \mathbb{E}\left\{\sum_{t\in\mathcal{T}} \sum_{h\in\mathcal{H}} C_{th}(S_{th}, X^{\pi}_{th}(S_{th}))\right\}, \tag{6}$$

where $S_{t,h+1} = S^M(S_{th}, x_{th}, W_{t,h+1})$ (for $h = 0, \ldots, H-1$) and $S_{t+1,0} = S^M(S_{tH}, x_{tH}, W_{t+1,0})$. The challenge, of course, is designing the policy $X^{\pi}_{th}(S_{th})$.

We note that by construction of our decision function (in particular, its dependence on the state variable), it is not allowed to depend on events in the future. This seemingly essential property is violated by the stochastic models that are allowed to make a decision as a function of a "scenario" that might specify, for example, the entire sample path of prices or energy from wind.

6. Algorithmic Strategy

There are a number of strategies that have been used to solve stochastic optimization problems as stated in Equation (6). These include various types of myopic policies (such as the logit investment rule in SEDS); rolling horizon policies (which use a point forecast of the future to make decisions now); simulation optimization, where we optimize the parameters of a myopic policy (Swisher et al. 2000); stochastic programming (Higle and Sen 1996, Birge and Louveaux 1997, Wallace and Fleten 2003); and dynamic programming. We adopt the framework of dynamic programming in §6.1, which introduces a number of computational hurdles. These are addressed in §6.2 using approximate dynamic programming.

6.1. Dynamic Programming Formulation

Our strategy to solve the energy model involves formulating the model as a dynamic program, solving decision problems for each hour $h \in \mathcal{H}$ in each year $t \in \mathcal{T}$. We first define the value $V_{th}(S_{th})$ of a state S_{th} as the sum of the contributions that we expect to make, starting in state S_{th}, if we act optimally (make optimal decisions) until the end of the time horizon. Bellman's equation enables us to recursively compute the optimal value functions associated with each state,

$$V_{th}(S_{th}) = \max_{x_{th} \in \mathcal{X}_{th}^{\text{disp}, n}} \left(-C_{th}(S_{th}, x_{th}) + \mathbb{E}(V_{t,h+1}(S_{t,h+1})) \mid S_{th}\right), \tag{7}$$

where $S_{t,h+1} = S^M(S_{th}, x_{th}, W_{t,h+1})$.

As pointed out earlier, for a given year t, we assume that capacity additions are made only at the beginning ($h = 0$) of the year with new capacity arriving in some future year; the capacity resource state remains unchanged during the remaining hours. The value function $V_{th}(S_{th})$ in Equation (7) captures the information about the future state of the system while solving the problem for the current time, including the value of water left behind in the reservoir after the decisions in the current time period are implemented.

In the remainder of the paper, we consider only a single form of storage—namely, the reservoir that stores water collected from precipitation. This means that the set $\mathcal{A}^{\text{Stor}}$ consists of a single aggregate reservoir, but it is straightforward to adapt our model to multiple reservoirs, as well as multiple types of storage. The water from the reservoir may be used to satisfy the energy demands now or stored to be used to satisfy demands later in time. Hydro energy is typically among the cheaper types of energy and tends to be used before more expensive energy types (such as coal or nuclear) if we do not provide for storage. Ideally, our model of reservoir management should mimic real-life behavior where water would be allowed to collect and be stored in the reservoir during periods of heavy precipitation to be used later during dry periods.

Solving the dynamic program in Equation (7) is generally hard even for problems with small state spaces. The traditional dynamic programming approach for solving Equation (7) suffers from three curses of dimensionality arising from the state space ($\{S_{th}\}_{h\in\mathcal{H},\, t\in\mathcal{T}}$), the exogenous information ($\{W_{th}\}_{h\in\mathcal{H},\, t\in\mathcal{T}}$), and the action space ($\{\mathcal{X}_{th}^{\text{disp}}\}_{h\in\mathcal{H},\, t\in\mathcal{T}}$) (see Powell 2007, Chapter 5). We now describe the techniques we use to overcome these.

We first define the postdecision state variable, $S_{th}^x = S^{M,x}(S_{th}, x_{th})$. The postdecision state for hour h is the state of the system immediately after making the decisions at hour h but before the occurrence of the random events ($W_{t,h+1}$) between hours h and $h+1$. In our analysis, we focus on two important components of the state variable, namely, the capacity resource state R_{th} and the reservoir level y_{th}. The postdecision transitions of these components can be written as

$$R_{tha}^x = R_{tha} + x_{tha}^{\text{cap}}, \quad a \in \mathcal{A}^{\text{Conv}},$$

$$y_{tha}^x = y_{tha} + \sum_{a' \in \overleftarrow{\mathcal{A}}(a)} \theta_{a'a} x_{th,a'a}^{\text{disp}}, \quad a \in \mathcal{A}^{\text{Stor}}.$$

We then construct the value function around the postdecision state $S_{th}^x = (R_{th}^x, y_{th}^x)$ using

$$V_{th}^x(S_{th}^x) = \mathbb{E}[V_{t,h+1}(S_{t,h+1}) \mid S_{th}^x]. \tag{8}$$

This allows us to avoid having to compute an expectation within the optimization formulation of Equation (7). The modified dynamic program can be written as

$$V_{th}(S_{th}) = \max_{x_{th} \in \mathcal{X}_{th}^{\text{disp}}} (-C(S_{th}, x_{th}) + V_{th}^x(S_{th}^x)).$$

This leaves the problem of computing $V_{th}^x(S_{th}^x)$. For this, we turn to the techniques of approximate dynamic programming. It is important to realize that our strategy represents a significant departure from classical modeling and algorithmic strategies used in the approximate dynamic programming literature (notably Bertsekas and Tsitsiklis 1996 and Sutton and Barto 1998), which approximate the value function around the predecision state. This strategy still leaves us with the challenge of computing or approximating the expectation. We avoid this by directly approximating the value function around the postdecision state.

6.2. An Approximate Dynamic Programming Algorithm

Our solution strategy involves choosing an appropriate functional form for the value functions. Our idea is very similar to that used in Pereira and Pinto (1991) for water reservoir management, with the significant difference that we are using separable, piecewise linear approximations of the value of resources in the future, where resources can be the amount of energy in storage as well as energy investments. This contrasts with the use in Pereira and Pinto (1991) of multidimensional cuts based on Benders decomposition, which carries with it provable convergence properties (examples of convergence proofs are given in Higle and Sen 1991 and Ruszczyński 1993). We have compared both of these strategies and found that the separable approximations produced much faster convergence than approximations based on Benders cuts (see Powell et al. 2004 for an analysis in the context of two-stage problems) and very accurate solutions for complex multistage problems that arise in transportation (see Powell and Topaloglu 2003, Topaloglu and Powell 2006). However, both strategies use the same basic idea of forming piecewise linear approximations based on dual information.

Starting with initial value function approximations, we step forward in time, solving the decision problem at each time using the value function approximation of the future states. We repeat this procedure over several iterations, updating the value function approximations after each iteration. This iterative updating enables us to form an approximation of the expectation in Equation (8) (actually, we only need to approximate the slopes). Although this algorithmic strategy is not generally optimal, there are convergence proofs for special versions of the problem (discussed in §6.4 below), and in §7 we benchmark against the linear programming formulation for a deterministic version of the problem that demonstrates very good agreement.

Using some policy π, we may write the objective function for time t as follows:

$$\begin{aligned} F^{\pi}_{th}(S_{th}) &= -C_{th}(S_{th}, X^{\pi}_{th}) - \sum_{h'>h} C_{th'}(S_{th'}, X^{\pi}_{th'}) \\ &\quad - \sum_{t'>t}\sum_{h\in\mathcal{H}} C_{t'h}(S_{t'h}, X^{\pi}_{t'h}) \\ &= -C_{th}(S_{th}, X^{\pi}_{th}) + V^{x}_{th}(S^{x}_{th}), \end{aligned} \tag{9}$$

where

$$V^{x}_{th}(S^{x}_{th}) = -\sum_{h'>h} C_{th'}(S_{th'}, X^{\pi}_{th'}) - \sum_{t'>t}\sum_{h\in\mathcal{H}} C_{t'h}(S_{t'h}, X^{\pi}_{t'h})$$

is the same as the postdecision value function in Equation (8). Because this function is unknown, we use an approximation that we denote as $\bar{V}^{x}_{th}(S^{x}_{th})$. We now introduce the approximation that the value function is separable in the postdecision capacity resource state R^{x}_{th} and reservoir level y^{x}_{th}. We express this approximation as the sum of components denoting the value of the capacity resource state and the value of water remaining in the reservoir:

$$\begin{aligned} \bar{V}^{x}_{th}(S^{x}_{th}) &= \bar{V}^{\mathrm{cap}}_{th}(R^{x}_{th}) + \bar{V}^{\mathrm{hydro}}_{th}(y^{x}_{th}) \\ &= \sum_{a\in\mathcal{A}^{\mathrm{Conv}}} \bar{V}^{\mathrm{cap}}_{tha}(R^{x}_{tha}) + \bar{V}^{\mathrm{hydro}}_{th}(y^{x}_{th}). \end{aligned}$$

It is easy to show that $V(S)$ is concave in R and y, and we approximate $V(R)$ and $V(y)$ using separable, piecewise linear functions (using a very fine discretization, because the quantities are continuous). It is important to recognize that we are only concerned with the derivative of $\bar{V}^{x}_{th}(S^{x}_{th})$ rather than the actual value.

The hourly dispatch problem involves solving the optimization problem

$$x^{\mathrm{disp},n}_{th} = \arg\max_{x_{th}\in\mathcal{X}^{\mathrm{disp},n}_{th}} \left(-C^{\mathrm{disp}}_{th}(S^{n}_{th}, x_{th}) + \bar{V}^{\mathrm{hydro}}_{th}(y^{x}_{th})\right), \tag{10}$$

where S^{n}_{th} is the specific state that we are in at hour h and year t, while following sample path ω^{n}. The variable y^{x}_{th} is a function of both S^{n}_{th} and x^{disp}_{th}. Equation (10) defines our policy π, and the space of policies Π is the set of all possible value function approximations. The dispatch problem at time t is illustrated in Figure 2, where the value function approximation (the value of water stored in the reservoir) is given by a piecewise linear function. It is important to emphasize that this is a very small linear program (for a spatially aggregate model), which scales fairly easily to spatially disaggregate applications (see Powell et al. 2002, Powell and Topaloglu 2005, Topaloglu and Powell 2006, and Simão et al. 2009 for numerous transportation applications). This is important because we have to solve 8,760 of these for each year of planning.

The annual capacity acquisition problem, solved at the beginning of each year, is given by

$$x^{\mathrm{cap},n}_{t0} = \arg\max_{x^{\mathrm{cap}}_{t0}} \left(-C^{\mathrm{cap}}_{t0}(S^{n}_{t0}, x_{t0}) + \bar{V}^{\mathrm{cap}}_{t0}(R^{x}_{t0})\right) \tag{11}$$

subject to

$$R^{x}_{t0} = R_{t0} + x^{\mathrm{cap},n}_{t0}, \tag{12}$$

$$x^{\mathrm{cap},n}_{t0} \geq 0. \tag{13}$$

This problem is illustrated in Figure 3. Here, we use a value function approximation for each type of energy resource. The only decision of how much to purchase is handled by the piecewise linear value function giving the value of resources in the future.

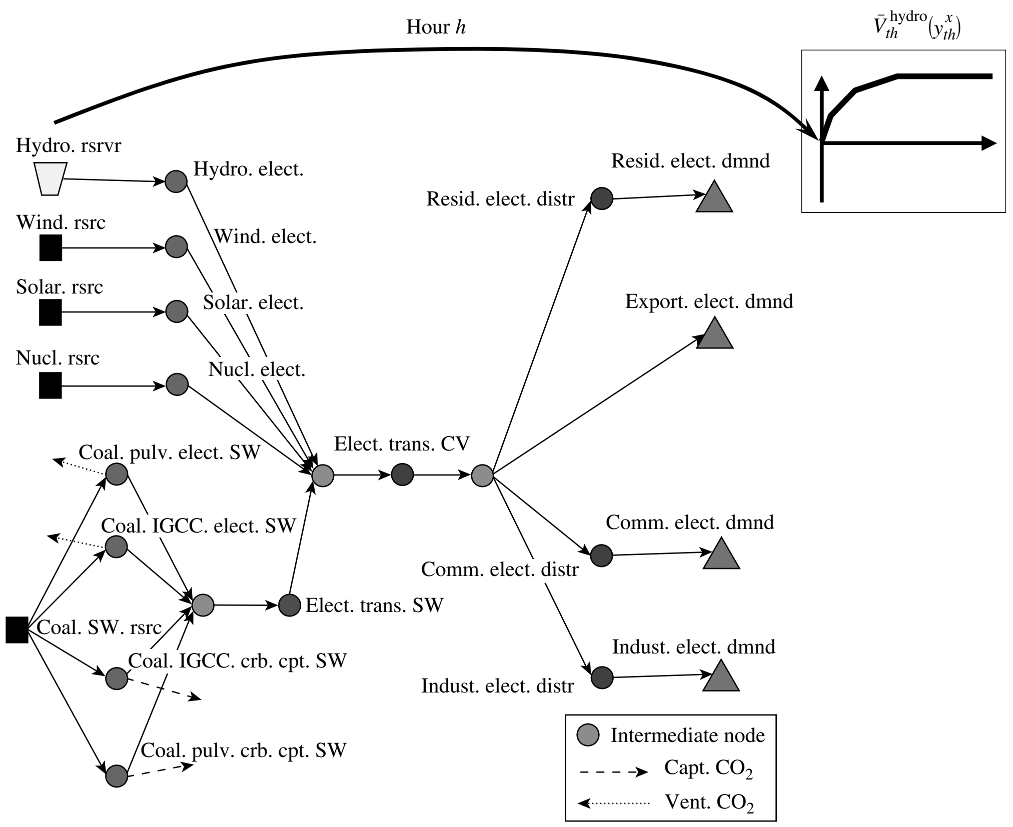

Figure 2 **The Hourly Optimization Problem with Value Function Approximations for the Reservoir Level**

Using ADP, we learn the value function approximations using iterative updating operations. Let n denote the iteration counter while following sample path ω^n. For resource allocation problems, we have found that it is effective to compute the marginal value of an additional unit of a resource (this can be energy resources R_{th} or the energy in storage y_{th}). Let

$\hat{v}_{tha}^{\text{cap},n}$ = A sample estimate of the marginal value of increasing R_{tha} by one unit computed while following sample path ω^n,

$\hat{v}_{tha}^{\text{hydro},n}$ = A sample estimate of the marginal value of increasing y_{tha} by one unit computed while following sample path ω^n.

$\hat{v}_{tha}^{\text{hydro},n}$ is computed relatively simply by finding the marginal impact of the value of additional energy in storage for a particular hour. Computing $\hat{v}_{tha}^{\text{cap},n}$ requires calculating the incremental value of additional energy investment over an entire year. This requires finding the incremental value of, say, additional wind turbine generating capacity for each hour over the year, taking into account the presence of storage. This calculation allows us to capture the impact of hourly, daily, and seasonal variability, as well as the marginal impact of additional storage.

In SMART, we used the updating equations given in Powell (2007, §11.3). Note that the value $\hat{v}_{th}^{\text{hydro},n}$ or $\hat{v}_{th}^{\text{cap},n}$ found at time t and iteration n is used to update the value function approximation around the postdecision state variable in the *previous* time period

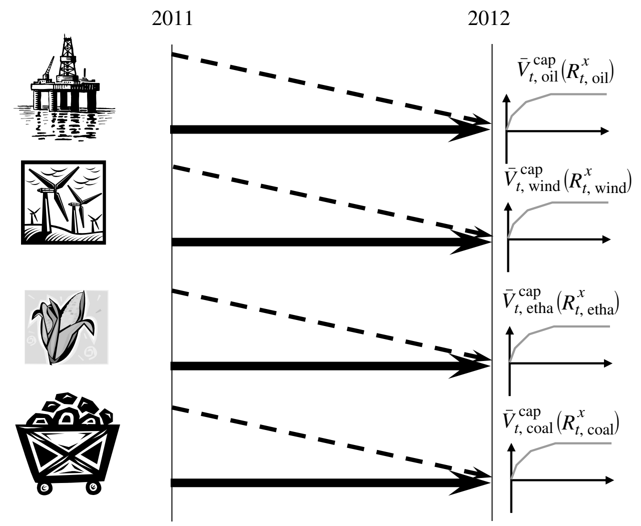
2011
2012
$\bar{V}_{t,\text{oil}}^{\text{cap}}(R_{t,\text{oil}}^{x})$
$\bar{V}_{t,\text{wind}}^{\text{cap}}(R_{t,\text{wind}}^{x})$
$\bar{V}_{t,\text{etha}}^{\text{cap}}(R_{t,\text{etha}}^{x})$
$\bar{V}_{t,\text{coal}}^{\text{cap}}(R_{t,\text{coal}}^{x})$

Figure 3 **The Annual Capacity Acquisition Problem with Value Function Approximations for the Future Capacity Resource State**

(see Powell 2007, §4.4). We represent general updating operations for the value functions as follows:

$$\bar{V}^{\text{hydro},\,n}_{t,\,h-1}(y^x_{t,\,h-1}) \leftarrow U^V(\bar{V}^{\text{hydro},\,n-1}_{t,\,h-1}, y^{x,\,n}_{t,\,h-1}, \hat{v}^{\text{hydro},\,n}_{th}),$$
$$\bar{V}^{\text{cap},\,n}_{t,\,h-1}(R^x_{t,\,h-1}) \leftarrow U^V(\bar{V}^{\text{cap},\,n-1}_{t,\,h-1}, R^{x,\,n}_{t,\,h-1}, \hat{v}^{\text{cap},\,n}_{th}).$$

The details of the calculation of the derivatives $\hat{v}^{\text{hydro},\,n}_{th}$ and $\hat{v}^{\text{cap},\,n}_{th}$ are somewhat involved and are given in §§A.2 and A.3, respectively, of the Online Supplement.

A complete summary of the approximate dynamic programming algorithm is given in Figure 7 in the Online Supplement (moved because of space constraints).

6.3. Discussion

The essential feature of the algorithm is that it steps forward through time, using a Monte Carlo sample of any random variable, and solves sequences of very small linear programs. In this sense, the model scales well to handle more complex information processes, or a spatial representation of the problem (for example, dividing the country into 10 to perhaps 100 regions). If the model is being used over a 30-year horizon, in hourly increments, we would have to model more than 250,000 time periods. This is quite manageable for an aggregate model (as described in our experimental work), but CPU times would start to become an issue with a sufficiently fine-grained spatial model.

Simulating a policy forward in time makes it very easy to handle considerable complexity in the transition function. We can handle complexities such as exogenous information such as prices, which depend on decisions about the supply of energy generating capacity. We can also handle autocorrelations in wind, price, and demand processes. However, adding dimensions to the state variable can potentially complicate the process of making decisions. Ideally, we would like to calculate a value function that has all the dimensions of the state variable, but at the same time, we have to recognize that these additional dimensions may have little or no effect on the quality of a decision. These issues represent important areas of research that are beyond the scope of this paper.

We have illustrated the logic using value function approximations that ignore other aspects of the state variable such as the state of climate (whether global warming is progressing faster than expected), technology (whether there was a significant change in the performance of batteries or the cost of extracting and sequestering carbon), markets (whether the price of oil is over \$150), and government policy (whether there is a carbon tax).

We can handle these dimensions, in theory, by simply replacing the separable piecewise linear value functions $\bar{V}^{\text{cap}}_{th}(R^x_{th})$ and $\bar{V}^{\text{hydro}}_{th}(y^x_{th})$ with functions that also depend on the state variable ρ_{th}, as in $\bar{V}^{\text{cap}}_{th}(R^x_{th} \mid \rho_{th})$ and $\bar{V}^{\text{hydro}}_{th}(y^x_{th} \mid \rho_{th})$, respectively. The algorithm would be virtually unchanged (see the storage problem described in Nascimento and Powell 2009 for an illustration). The vector ρ_{th} evolves exogenously over time and is already being computed. All that is needed is indexing the value functions by this information.

Of course, this is much easier to write than to implement. The number of possible values of ρ_{th} is extremely large, and as a result, finding a value function approximation $\bar{V}^{\text{cap}}_{th}(R^x_{th} \mid \rho_{th})$ for each ρ_{th} is computationally impossible when ρ_{th} has more than two or three dimensions. There is a wide range of statistical strategies for handling this problem (see, for example, Hastie et al. 2009) that represent promising avenues for research. There is also a growing base of literature in approximate dynamic programming that draws on techniques such as kernel regression (see Ormoneit and Sen 2002, Fan and Gijbels 1996) that offer considerable promise.

6.4. Convergence Analysis

Our solution strategy is not optimal, but we offer both theoretical arguments and, in §7, empirical evidence to support the claim that the quality of the solution is extremely good.

First, the algorithm for optimizing storage over a single year is an instance of a provably convergent algorithm, as given in Nascimento and Powell (2011) (building on the same proof technique as that given in Nascimento and Powell 2009). This proof technique applies even if we index the value functions on the exogenous state vector ρ_{th}. Central to this convergence proof is that the storage (which is the only means by which one subproblem interacts with another) is a scalar (the amount stored in a reservoir). Just the same, optimizing the storage in water reservoirs is a particularly difficult problem, because we need to decide whether to store additional water in the rainy months of February and March to satisfy needs later in the summer, thousands of hours into the future.

Second, the use of separable, piecewise linear approximations has been proven to be optimal for two-stage problems if the second stage is separable and near optimal even when the second stage is nonseparable (see Powell et al. 2004). Separable approximations are not, in general, optimal for nonseparable, multistage resource allocation problems. However, we can obtain provably convergent algorithms for multistage problems. If we were to introduce other forms of storage, the algorithm would only be an approximation because of our use of separable functions. For practical uses, however, separable approximations have been found to scale very well to high-dimensional problems with fast convergence.

Theoretical convergence proofs, although reassuring, are no guarantee of empirical performance. Our research has been preceded by extensive experimental research with algorithms of this type for transportation applications (see Godfrey and Powell 2002a, Topaloglu and Powell 2006). In the next section, we report on experimental comparisons of the approximate dynamic programming algorithm on deterministic and stochastic versions of the problem.

7. Experiments

In this section, we design experiments that focus on the ability of the ADP algorithm in §6.2 to learn near-optimal policies in terms of how much to store and what investment decisions to make. We begin by comparing ADP to linear programming (LP) for a deterministic version of the problem using an aggregate national model, which is small enough that we can solve the linear program. We also demonstrate that the algorithm can be run on a spatially disaggregate model and show that CPU times scale sublinearly as the size of the network grows. We then demonstrate the algorithm on a stochastic problem. Our problem, with more than 175,000 time periods, is far beyond the capabilities of any known algorithm for stochastic optimization, and as a result, we do not have an optimal benchmark for this problem; we do show, however, that SMART produces robust behaviors.

We begin in §7.1 by describing the performance indices that we use to evaluate our algorithmic strategy. Section 7.2 reports on CPU times as we increase the number of time periods and as we make the transition from a spatially aggregate to a disaggregate network. In §7.3, we describe our experiments in a purely deterministic setting, where we run tests over a single-year planning horizon with deterministic precipitation. In §7.4, we discuss experiments with stochastic precipitation scenarios. In §7.5, we focus on experiments on capacity acquisition in a multidecade setting.

7.1. Performance Measures

In this section, we design performance measures to determine how closely the ADP algorithm matches the optimal solutions generated by an LP solver. We use three types of policies to construct our performance measures: ADP (the solution produced using approximate dynamic programming), OPT (the optimal solution of the deterministic problem produced by solving the linear program), and MYOPIC, which is the result of a myopic policy (the same as ADP with value function approximations set to zero). The myopic policy is used only to compute one of the performance metrics. Below, we let strategy s refer to ADP, OPT, or MYOPIC.

We define the following terms that will be used for computing the statistics that we measure.

F^s = The objective function value as computed using strategy s. For example, F^{MYOPIC} denotes the myopic solution, and F^{OPT} represents the optimal objective function value.

x^s_{tha} = The total usage corresponding to resource node with attribute vector $a \in \mathcal{A}^{\text{Res}}$ ($a^{\text{Type}} \in$ {coal, solar, nuclear, ...}) in hour h of year t according to strategy s
$= \sum_{a' \in \vec{\mathcal{A}}(a)} x^s_{th,\,aa'}$.

y^s_{th} = The reservoir level in hour h of year t obtained by following strategy s.

c_a = Cost of adding a unit of capacity of type a.

x_{ta} = Amount of capacity type a added in year t using ADP.

x^*_{ta} = The optimal amount of capacity type a to be added in year t.

z^*_a = The optimal amount of capacity type a that is used in the starting year.

We use these to compute the following measures.

General performance indices

E^s_1 = The error in the objective function (using strategy s) from the optimal objective function as a percentage of the optimal
$= F^s - F^{\text{OPT}})/F^{\text{OPT}} \times 100\%$.

E^s_2 = The error in the objective function (using strategy s) as a percentage of the error in the objective function for a myopic policy
$= E^s_1/E^{\text{MYOPIC}}_1 \times 100\%$.

Performance indices for hydro storage

$$R^s_1 = \frac{\sum_{t,h,a} |x^s_{tha} - x^{\text{OPT}}_{thi}|}{\sum_{t,h,a} |x^{\text{OPT}}_{tha}|},$$

$$R^s_2 = \frac{\sum_{t,h} (y^s_{th} - y^{\text{OPT}}_{th})^2}{\sum_{t,h} (y^{\text{MYOPIC}}_{th} - y^{\text{OPT}}_{th})^2}.$$

Indices for capacity matching

$$R^{\text{cap}}_1 = \frac{\sum_a c_a \sum_t |x_{ta} - x^*_{ta}|}{T\sum_a c_a z^*_a} \times 100\%,$$

$$R^{\text{cap}}_2 = \frac{\sum_a c_a |\sum_t x_{ta} - \sum_t x^*_{ta}|}{T\sum_a c_a z^*_a} \times 100\%.$$

E^s_2 is a good indicator of the improvement in the solution quality obtained by using strategy s compared to a myopic solution. R^s_1 measures the deviation of the solution from the optimal as a fraction of the optimal solution. R^s_2 is a measure of how well strategy s improves over a myopic solution of the storage problem.

7.2. Execution Times and Scalability

A major feature of approximate dynamic programming is that it can scale to handle important dimensions such as fine-grained temporal and spatial modeling, as well as uncertainty. In this section, we summarize CPU times as we make the transition to handle problems with more than 175,000 time periods and as we move from a spatially aggregate to a disaggregate model. All runs were performed on a 2.5 GHz, Intel Xeon processor with access to 64 GB of memory. The linear program was solved using CPLEX Version 12. The model is coded in Java.

We first ran a series of tests to demonstrate the ability of the ADP algorithm to handle a large number of time periods, which we compare to solving a single large deterministic model. We emphasize that the energy modeling community feels that it is very important to model wind at hourly increments, because longer increments have the effect of averaging out the peaks and valleys that plague this important energy source. Figure 4 shows the computational time in hours to solve the linear program as a function of the number of years. With a horizon of four years (35,000 time periods), the linear program using the aggregate network required 20 hours, growing superlinearly with the horizon.

To compare ADP to the linear programming solution, we first performed a series of comparisons of the ADP objective function and solution (capacity investment and the amount held in storage). We found, using the aggregate model and a deterministic data set, that we could always obtain solutions within 1% (and often within a fraction of a percent) using 300 iterations. Using this setting, Figure 4 shows that ADP can solve the four-year horizon in under an hour.

We next turned to our spatially disaggregate model. This model represents energy generation and consumption at the level of the 48 states in the continental United States, plus demand in the District

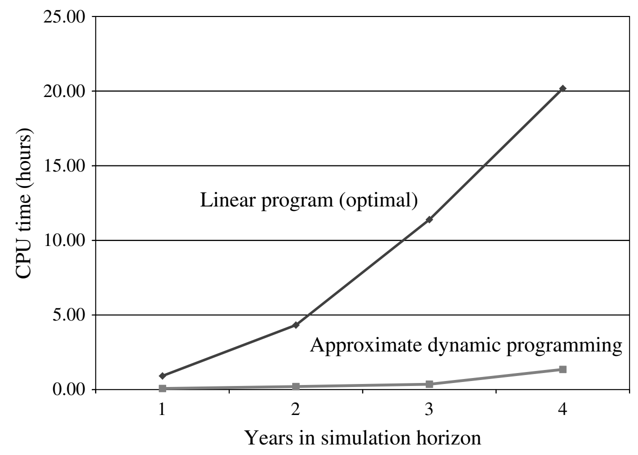

Figure 4 **Time to Solve the Energy Model as a Function of the Number of Years in the Simulation Model**

Note. With the increasing number of years in the model, run times increase dramatically in order to obtain an optimal solution, whereas ADP produces near-optimal results in a fraction of that time.

Table 1 **Size of the Linear Programs for the Aggregate and Disaggregate Networks (for a Single Time Period), CPU Times for First Iteration Over a Year, for 20 Iterations, Average Over Last 19 Iterations, and the CPU Time per Year Normalized by the Number of Rows in the Linear Program**

	Aggregate data set	Disaggregate data set
Rows	198	40,229
Columns	174	39,705
CPU secs/first iteration	11.73	1,412.2
CPU secs/20 iterations	180.7	22,119.8
CPU secs/iteration	8.893	1,089.9
CPU secs/row/iteration	0.045	0.027

of Columbia. It covers four types of energy supply (coal, nuclear, natural gas, and wind) and four types of energy demand (residential, commercial, industrial, and transportation). The network makes it possible to consider different types of technologies (e.g., investments in coal with carbon capture and sequestration). The resulting network has approximately 40,000 rows and columns.

Table 1 shows the size of the linear programs for the aggregate and disaggregate data sets. It then shows the CPU time (in seconds) for simulating a single year (8,760 subproblems), for 20 iterations (of a year), the CPU seconds per iteration (as an average over the last 19 iterations), and the CPU seconds per iteration divided by the number of rows in the linear program. The last statistic provides an indication of how the CPU times grow with the size of the linear program. We note that the model builds the full linear program and solves it from scratch only once each year; for the remaining 8,759 subproblems, the only changes that occur are in the costs and right-hand sides of the constraints, making it easy to use the previous solution.

The table shows that the CPU time per row, per year of simulation, actually drops as the problem size grows, from 0.045 CPU seconds/row (based on an average over the 20 year run) for the small aggregate data set to 0.027 CPU seconds/row for the large disaggregate data set.

Because ADP scales linearly with the number of iterations and the number of time periods, we can quickly estimate run times as we increase the number of iterations. Three hundred iterations of the small network over a single year (which provided near-optimal solutions when we compared the small network to the optimal solution) can be run in under an hour, making it possible to perform studies of wind penetration with different assumptions on storage. If the same run is extended over 20 years (to capture investment decisions), the run time grows to 15 hours, which is certainly manageable as an overnight run. Three hundred iterations of the large network over an entire year, although hopelessly intractable as a single deterministic model (350 million rows and columns), can be run in approximately 90 hours using ADP. This compares well to reports of run times of 30 days for some strategic planning models that solve daily-unit commitment models over a year, but at this point, it certainly seems to be an opportunity for parallel computation, an avenue we intend to pursue.

7.3. Experiments on Hydro Storage

The first set of experiments focuses on testing how well the hydro-usage behavior of the solution from the ADP algorithm matches the optimal solution from the LP model using our aggregate dispatch model, under the assumption of deterministic precipitation. We focus on hydro storage because this is the most difficult form of storage from an algorithmic perspective. The reason for this complexity is that we may put water in storage in March so that it can be used in August, thousands of hours in the future, which is a far more complex problem than storing energy in a battery at night so that it can be used during the day. We only solve the problem over a single year, because the reservoir empties out at least once each year.

We conducted our experiments in a variety of deterministic settings. The demands were (1) constant, (2) sinusoidally varying, or (3) varying with no apparent pattern. Another setting that could be varied was whether or not wind (varying by the hour) was included as a competing source of energy. We tested our algorithm for a one-year horizon with storage value functions computed for every 10-hour increment. We then repeated the test for one-hour increments. For some problems, this means that the decision of how much water to draw from the reservoir in March had to reflect usage behavior in July, thousands of hours into the future.

Figure 5 shows the reservoir level over the course of a one-year period for a deterministic problem produced by solving the linear program, which gives an optimal solution (Figure 5, panels (a) and (c)), and from the ADP algorithm (Figure 5, panels (b) and (d)). These were run for constant demand (Figure 5, panels (a) and (b)) and a sinusoidal demand (Figure 5, panels (c) and (d)). Note that the heaviest precipitation occurs during late winter and spring. The model used a slight penalty for holding water in the reservoir so that the optimal solution was to build up reserves as late as possible (this choice was made purely for algorithmic testing purposes). With this

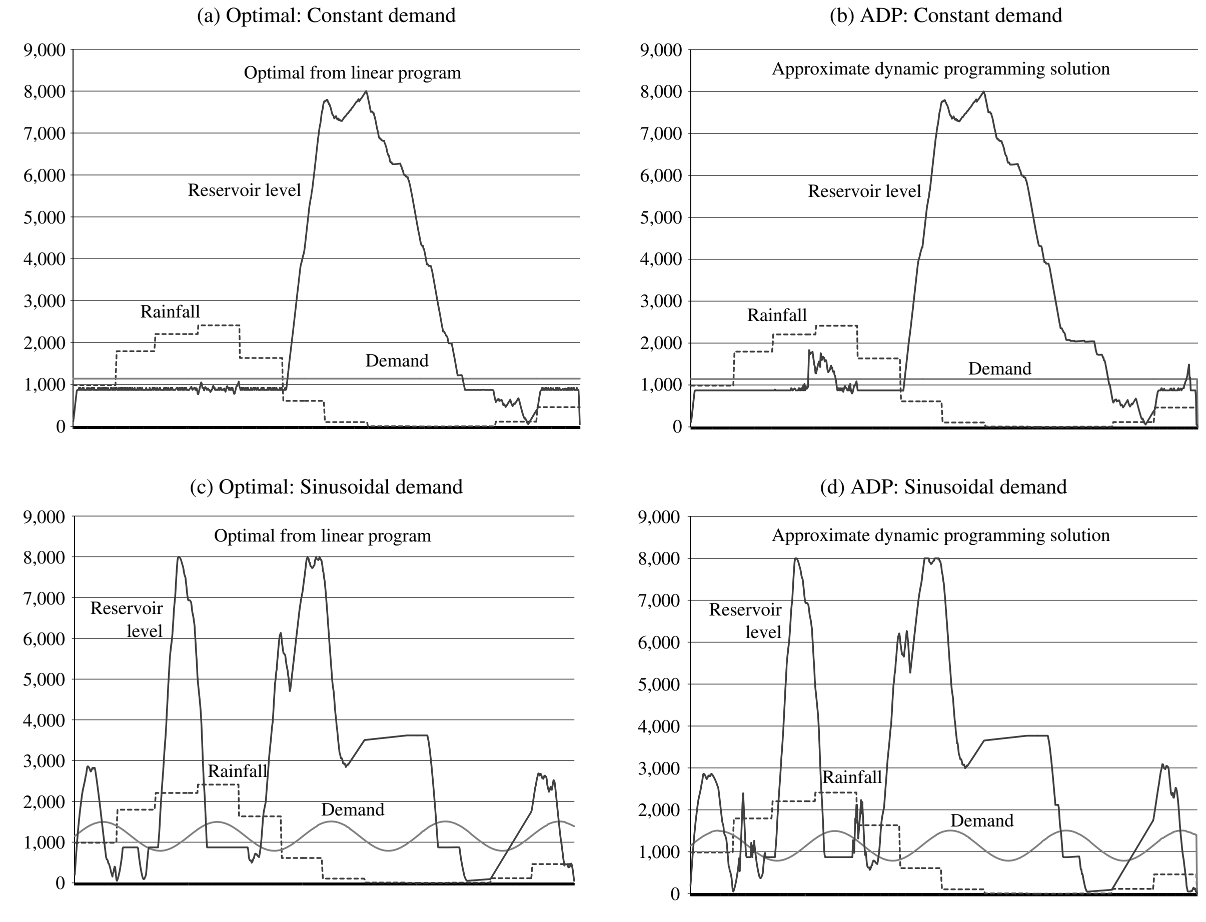

Figure 5 **Optimal Reservoir Levels (a) and (c) and Reservoir Levels from the ADP Algorithm (b) and (d) for Constant and Sinusoidal Demand**

Table 2 **Performance Statistics of the ADP Algorithm for Different Problem Instances of a One-Year Energy Model with the Dispatch Problem Solved Every *H* Hours**

Data set	Demand type	H (hrs/time period)	Wind	E_1^{ADP} (%)	E_2^{ADP} (%)	R_1^{ADP}	R_2^{ADP}
1	Constant	10	0	0.0008	2.26	0.02	0.006
2	Constant	10	✓	0.0003	0.63	0.04	0.007
3	Sinusoidal	10	0	0.0004	0.02	0.04	0.084
4	Sinusoidal	10	✓	0.0613	2.67	0.04	0.007
5	Fluctuating	10	0	0.0004	0.04	0.03	0.011
6	Fluctuating	10	✓	0.0000	0.01	0.03	0.005
7	Constant	1	0	0.0001	0.06	0.06	0.001
8	Constant	1	✓	0.0001	0.08	0.07	0.001
9	Sinusoidal	1	0	0.0012	0.12	0.03	0.002
10	Sinusoidal	1	✓	0.0020	0.20	0.03	0.001
11	Fluctuating	1	0	0.0000	0.01	0.02	0.001
12	Fluctuating	1	✓	0.0001	0.02	0.01	0.011

Note. A zero under wind means no wind, and a check mark means wind was included.

setup, the algorithm realized that it was necessary to begin building up reservoirs in the March–April time frame to meet needs that would occur thousands of hours into the future. The results show an extremely close match between the optimal solution and that produced by the ADP algorithm.

In Table 2, we summarize the performance statistics of the ADP algorithm. We observe that the ADP algorithm is able to produce solutions that are consistently within a very small margin of error of the optimal solutions.

7.4. Hydro Storage with Stochastic Precipitation

We now consider the case where the precipitation is stochastic. We generate 50 different precipitation scenarios for the year, and we use these to train the value function approximations for approximate dynamic programming. In Figure 6, we show the storage level produced by ADP for a single sample path, along with the storage level for each sample path when optimized knowing the entire sample path. The ADP solution has the intuitive behavior of building up storage earlier in the season to prepare for potentially lower rainfalls later in the season. By contrast, when we are allowed to optimize given the entire rainfall trajectory, the reservoir levels are held at lower levels until later in the season, reflecting the benefits of advance information. We believe that this illustrates the central weakness of models that sample uncertain information over the entire horizon, and we then find the optimal solution given advance information.

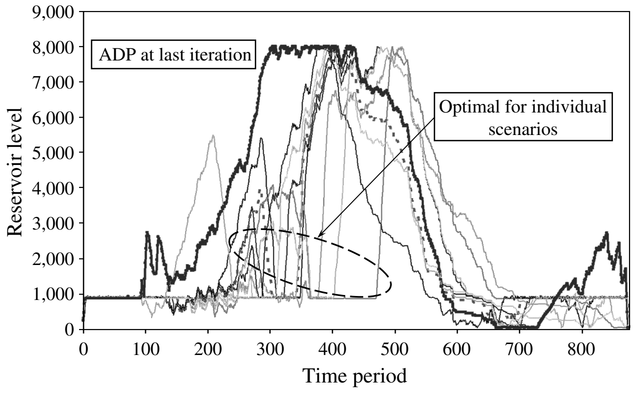

Figure 6 **Reservoir Level Produced by ADP (Last Iteration) for Stochastic Precipitation, and Optimal Reservoir Levels Produced by LP When Optimized Over Individual Scenarios**

7.5. Experiments on Capacity Acquisition

In the final set of experiments, we test the ability of the ADP algorithm to optimize capacity acquisition decisions over a multidecade horizon under deterministic conditions. Given an initial set of capacities at the various conversion nodes, we allow the average demand to increase over the years. Eventually, the current level of capacities will be insufficient to cover the growing demands for energy. The ADP algorithm was found to produce a solution within 2.5% of the optimal solution after 100 iterations, 0.70% after 200 iterations, and 0.24% after 500 iterations.

In Table 3, we summarize the performance statistics after 500 iterations of the ADP algorithm for a number of problem scenarios. Here again, the ADP algorithm is seen to approach within 1% of the optimal solution quality for a variety of deterministic data sets.

Table 3 **Performance Statistics of the ADP Algorithm for Different Problem Instances of a Multiyear Energy Model**

Data set	Demand type	Wind	Years	E_1^{ADP} (%)	E_2^{ADP} (%)	R_1^{cap}	R_2^{cap}
1	Constant	0	5	0.03	0.85	0.008	0.008
2	Constant	0	20	0.52	1.37	0.031	0.067
3	Constant	✓	20	0.24	0.72	0.027	0.061
4	Fluctuating	0	20	0.34	0.92	0.029	0.061
5	Fluctuating	✓	20	0.30	0.91	0.026	0.056

8. Conclusions

We have presented a stochastic, multiscale energy resource planning model that handles competing energy resources and demands, a wide range of forms of uncertainty, and the ability to model hourly variations over a multidecade horizon. The computational research shows that approximate dynamic programming can produce solutions that closely match the optimal solution of a deterministic linear programming model while also providing solutions for problems that are far larger than what can be solved as a linear program.

The ADP model uses a proper model of the flow of information and decisions, and as a result, it avoids the shortcuts that have characterized many of the efforts at introducing uncertainty in optimization-based models. The experimental work shows that ADP produces more robust behaviors, such as storing rainfall earlier in the season, than any of the solutions from a deterministic model acting on a specific scenario. It also exhibits nonanticipative behavior when faced with a major source of uncertainty, such as the introduction of a carbon tax. Although our algorithm is supported by some important theoretical results, we do not have a provably optimal algorithm for stochastic data sets that can serve as a comparable benchmark.

We show that the model can scale to handle spatially disaggregate problems, using a data set that models each state in the continental United States. We show that the model scales sublinearly with respect to the number of rows in the linear program for each time period, using the results of two networks with 200 and 40,000 rows.

The major unresolved algorithmic issue involves the handling of coarse-grained uncertainty, such as the imposition of a carbon tax or a breakthrough in technology. ADP has no difficulty simulating different scenarios describing these sources of uncertainty, but estimating a value function approximation that takes these issues into account in a proper way represents a new area of research. However, using suitably designed approximation strategies, ADP can produce good solutions that are nonanticipative (do not peek into the future) and that properly capture the dynamics in a realistic way.

References

Archibald, T. W., K. I. M. McKinnon, L. C. Thomas. 1997. An aggregate stochastic dynamic programming model of multireservoir systems. *Water Resources Res.* **33**(2) 333–340.

Bertsekas, D. P., J. N. Tsitsiklis. 1996. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA.

Birge, J. R., F. Louveaux. 1997. *Introduction to Stochastic Programming*. Springer-Verlag, New York.

Brown, P. D., J. A. Peas-Lopes, M. A. Matos. 2008. Optimization of pumped storage capacity in an isolated power system with large renewable penetration. *IEEE Trans. Power Systems* **23**(2) 523–531.

Brunetto, C., G. Tina. 2007. Optimal hydrogen storage sizing for wind power plants in day ahead electricity market. *IET Renewable Power Generations* **1**(4) 220–226.

Castronuovo, E. D., J. A. P. Lopes. 2004. On the optimization of the daily operation of a wind-hydro power plant. *IEEE Trans. Power Systems* **19**(3) 1599–1606.

Cervellera, C., V. C. P. Chen, A. Wen. 2005. Optimization of a large-scale water reservoir network by stochastic dynamic programming with efficient state space discretization. *Eur. J. Oper. Res.* **171**(3) 1139–1151.

De Ladurantaye, D., M. Gendreau, J.-Y. Potvin. 2007. Optimizing profits from hydroelectricity production. *Comput. Oper. Res.* **36**(2) 499–529.

Edmunds, T. A., J. F. Bard. 1990a. A decomposition technique for discrete time optimal control problems with and application to water resources management. *Math. Comput. Modelling* **13**(1) 61–78.

Edmunds, T. A., J. F. Bard. 1990b. Time-axis decomposition of large-scale optimal control problems. *J. Optim. Theory Appl.* **67**(2) 259–277.

Fan, J., I. Gijbels. 1996. *Local Polynomial Modelling and Its Applications*. Chapman & Hall, London.

Fishbone, L. G., H. Abilock. 1981. MARKAL—A linear programming model for energy systems analysis: Technical description of the BNL version. *Internat. J. Energy Res.* **5**(4) 353–375.

Foufoula-Georgiou, E., P. K. Kitanidis. 1988. Gradient dynamic programming for stochastic optimal control of multidimensional water resources systems. *Water Resources Res.* **24**(8) 1345–1359.

Garcia-Gonzalez, J., de la Muela, L. M. Santos, A. M. Gonzalez. 2008. Stochastic joint optimization of wind generation and pumped-storage units in an electricity market. *Power Systems, IEEE Trans.* **23**(2) 460–468.

Godfrey, G. A., W. B. Powell. 2002a. An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times. *Transportation Sci.* **36**(1) 21–39.

Godfrey, G. A., W. B. Powell. 2002b. An adaptive dynamic programming algorithm for dynamic fleet management, II: Multiperiod travel times. *Transportation Sci.* **36**(1) 40–54.

Gröwe-Kuska, N., H. Heitsch, W. Römisch. 2003. Scenario reduction and scenario tree construction for power management problems. A. Borghetti, C. A. Nucci, M. Paolone, eds. *IEEE Bologna Power Tech Conf. Proc.*, Vol. 3. IEEE Press, Los Alamitos, CA.

Hastie, T., R. Tibshirani, J. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, New York.

Higle, J. L., S. Sen. 1991. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Math. Oper. Res.* **16**(3) 650–669.

Higle, J. L., S. Sen. 1996. *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Hogan, W. W. 1975. Energy policy models for project independence. *Comput. Oper. Res.* **2**(3–4) 251–271.

Holttinen, H. 2004. The impact of large scale wind. Ph.D. thesis, VTT Technical Research Center, Espoo, Finland.

Holttinen, H., P. Meibom, A. Orths, B. Lange, M. O'Malley, J. O. Tande, A. Estanqueiro et al. 2011. Impacts of large amounts of wind power on design and operation of power systems, results of IEA collaboration. *Wind Energy* **14**(2) 179–192.

Holttinen, H., P. Meibom, A. Orths, F. Van Hulle, B. Lange, M. O. Malley, J. Pierik et al. 2009. Design and operation of power systems with large amounts of wind power: State-of-the-art report. VTT Working Paper 82, VTT Technical Research Center, Espoo, Finland.

Høyland, K., S. W. Wallace. 2001. Generating scenario trees for multistage decision problems. *Management Sci.* **47**(2) 295–307.

Jacobs, J., G. Freeman, J. Grygier, D. Morton, G. Schultz, K. Staschus, J. Stedinger. 1995. SOCRATES—A system for scheduling hydroelectric generation under uncertainty. *Ann. Oper. Res.* **59**(1) 99–133.

Johnson, S. A., J. R. Stedinger, K. Staschus. 1991. Heuristic operating policies for reservoir system simulation. *Water Resources Res.* **27**(5) 673–685.

Johnson, T. L., J. F. DeCarolis, C. L. Shay, D. H. Loughlin, C. L. Gage, S. Vijay. 2006. MARKAL scenario analyses of technology options for the electric sector. Technical report, U.S. Environmental Protection Agency, Washington, DC.

Kanudia, A., R. Loulou. 1998. Robust responses to climate change via stochastic MARKAL: The case of Québec. *Eur. J. Oper. Res.* **106**(1) 15–30.

Kaut, M., S. W. Wallace. 2003. Evaluation of scenario-generation methods for stochastic programming. *Pacific J. Optim.* **3**(2) 257–271.

Korpaas, M., A. T. Holen, R. Hildrum. 2004. Operation and sizing of energy storage for wind power plants in a market system. *Internat. J. Electrical Power Energy Systems* **25**(8) 599–606.

Lamond, B. F., M. J. Sobel. 1995. Exact and approximate solutions of affine reservoir models. *Oper. Res.* **43**(5) 771–780.

Lamont, A. 1997. User's guide to the META∗Net economic modeling system version 2.0. Technical report, Lawrence Livermore National Laboratory, Livermore, CA.

Lamont, A. D. 2008. Assessing the long-term system value of intermittent electric generation technologies. *Energy Econom.* **30**(3) 1208–1231.

Loulou, R., G. Goldstein, K. Noble. 2004. Documentation for the MARKAL family of models. Technical report, Energy Technology Systems Analysis Programme, International Energy Agency, Paris.

Messner, S., M. Strubegger. 1995. User's guide for MESSAGE III. Technical report, International Institute for Applied Systems Analysis, Laxenburg, Austria.

Messner, S., A. Golodnikov, A. Gritsevskii. 1996. A stochastic version of the dynamic linear programming model MESSAGE III. *Energy* **21**(9) 775–784.

Nandalal, K. D. W., J. J. Bogardi. 2007. *Dynamic Programming Based Operation of Reservoirs*. Cambridge University Press, Cambridge, UK.

Nascimento, J. M., W. B. Powell. 2009. An optimal approximate dynamic programming algorithm for the lagged asset acquisition problem. *Math. Oper. Res.* **34**(1) 210–237.

Nascimento, J. M., W. B. Powell. 2011. An optimal approximate dynamic programming algorithm for the energy dispatch problem with grid-level storage. Technical report, Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ.

NEMS. 2003. The National Energy Modeling System: An overview 2003. Technical report, Energy Information Administration, Office of Integrated Analysis and Forecasting, U.S. Department of Energy, Washington, DC.

Ormoneit, D., S. Sen. 2002. Kernel-based reinforcement learning. *Machine Learning* **49**(2–3) 161–178.

Paatero, J. V., P. D. Lund. 2005. Effect of energy storage on variations in wind power. *Wind Engrg.* **8**(4) 421–441.

Pereira, M. V. F., L. M. V. G. Pinto. 1991. Multistage stochastic optimization applied to energy planning. *Math. Programming* **52**(1–3) 359–375.

Powell, W. B. 2007. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, Hoboken, NJ.

Powell, W. B., H. Topaloglu. 2003. Stochastic programming in transportation and logistics. A. Ruszczyński, A. Shapiro, eds. *Stochastic Programming*. Handbooks in Operations Research and Management Science, Vol. 10. Elsevier, Amsterdam, 555–636.

Powell, W. B., H. Topaloglu. 2005. Fleet management. S. W. Wallace, W. T. Ziemba, eds. *Applications of Stochastic Programming*. Society for Industrial and Applied Mathematics/MPS, Philadelphia, 185–216.

Powell, W. B., A. Ruszczyński, H. Topaloglu. 2004. Learning algorithms for separable approximations of discrete stochastic optimization problems. *Math. Oper. Res.* **29**(4) 814–836.

Powell, W. B., J. A. Shapiro, H. P. Simão. 2002. An adaptive dynamic programming algorithm for the heterogeneous resource allocation problem. *Transportation Sci.* **36**(2) 231–249.

Ruszczyński, A. 1993. Parallel decomposition of multistage stochastic programming problems. *Math. Programming* **58**(1–3) 201–228.

Simão, H. P., J. Day, A. P. George, T. Gifford, J. Nienow, W. B. Powell. 2009. An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Sci.* **43**(2) 178–197.

Sioshansi, R. 2010. Evaluating the impacts of real-time pricing on the cost and value of wind generation. *IEEE Trans. Power Systems* **25**(2) 741–748.

Sioshansi, R., W. Short. 2009. Evaluating the impacts of real-time pricing on the usage of wind generation. *IEEE Trans. Power Systems* **24**(2) 516–524.

Sutton, R. S., A. G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Swisher, J. R., P. D. Hyden, S. H. Jacobson, L. W. Schruben. 2000. A survey of simulation optimization techniques and procedures. *Proc. Winter Simulation Conf., Orlando, FL*, IEEE Press, Los Alamitos, CA, 119–128.

Tagher, N. P. 2010. Powering America: Optimizing electricity generation for the United States until 2030. Ph.D. thesis, Princeton University, Princeton, NJ.

Topaloglu, H., W. B. Powell. 2006. Dynamic programming approximations for stochastic, time-staged integer multicommodity-flow problems. *INFORMS J. Comput.* **18**(1) 31–42.

Van Slyke, R. M., R. Wets. 1969. *L*-Shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.* **17**(4) 638–663.

Wallace, S. W., S.-E. Fleten. 2003. Stochastic programming models in energy. A. Ruszczynski, A. Shapiro, eds. *Stochastic Programming*. Handbooks in Operations Research and Management Science, Vol. 10. Elsevier, Amsterdam, 637–677.