Tutorial:
Stochastic Optimization in Energy

ISO New England

August 18, 2014

Warren B. Powell
CASTLE Labs
Princeton University
http://www.castlelab.princeton.edu

Mission statement

- Main goals:
  - First, we seek to develop a modeling framework that unifies the competing approaches to stochastic optimization, providing a roadmap through the "jungle of stochastic optimization."
  - Second, I would like to build bridges between this framework and applications in energy systems, highlighting the richness of energy. This means learning some new words and a little math.
  - Third, we wish to formalize current industry practice and highlight its strengths.
  - Finally, I would like to bring out some unrecognized weaknesses in the approach known as "stochastic programming" when applied to the stochastic unit commitment problem in the context of handling renewables.

Mission statement

- During this tutorial, we would like you to become conversant with the following terms:
  - "Stochastic"
  - Online/offline optimization
  - Policy
  - Lags
  - Transition function
  - Lookahead model/policy
  - Multistage vs. sequential vs. multiperiod
  - Base model versus lookahead model
  - Scenario vs. sample path

Mission statement

- We would also like to help you understand what people are saying when they talk about
  - Stochastic programming
  - Dynamic programming (or approximate dynamic programming)
  - Robust optimization

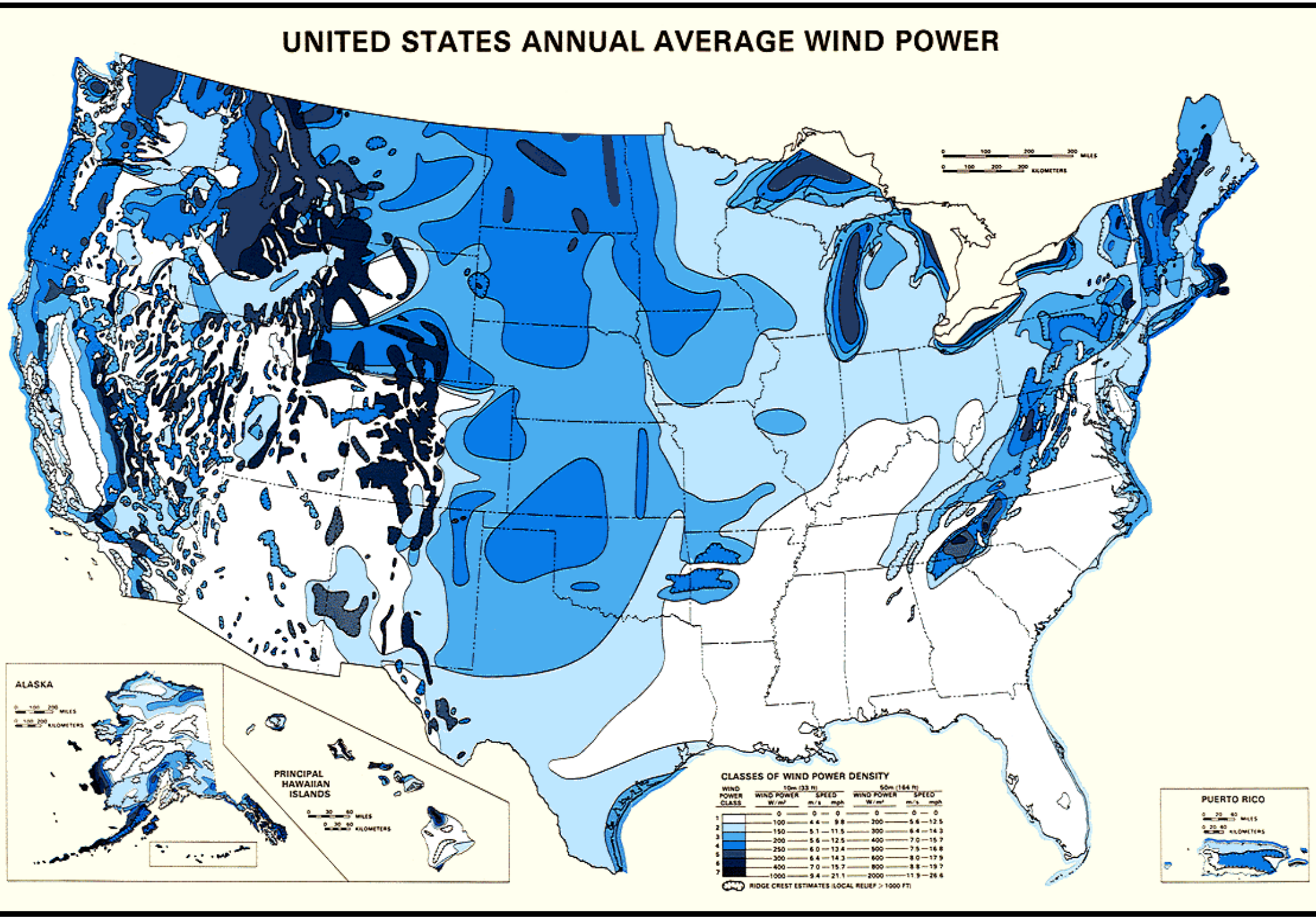
- Each of these represent a particular community that studies “stochastic optimization,” but in the setting of sequential problems, each is actually a class of policy (or two)

Lecture outline

- The uncertainty of renewables

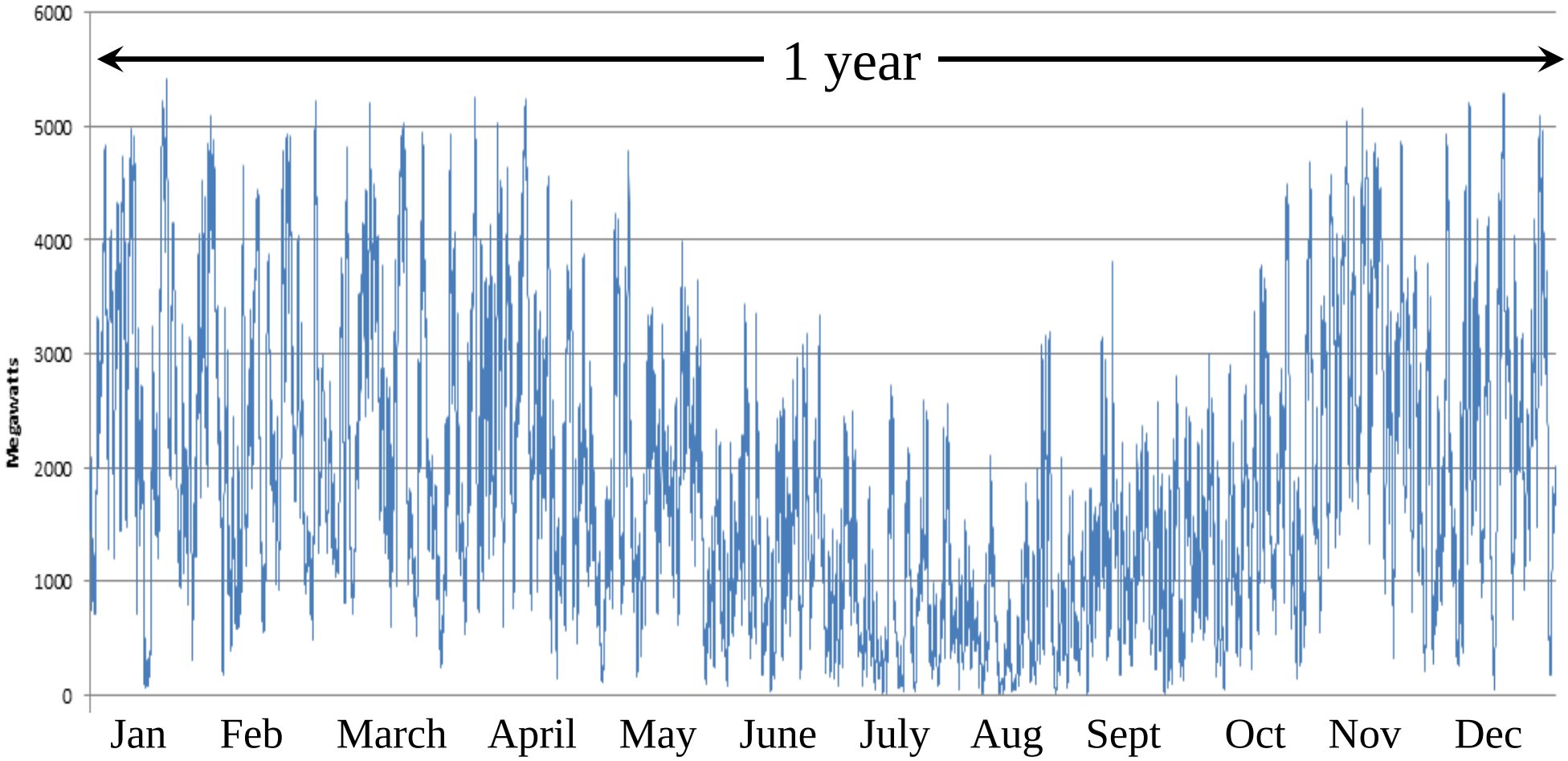
Energy and uncertainty

- There are many decisions in energy that require dealing with uncertainty:
  - » Stochastic unit commitment – Handling generator or transmission failures, weather variations, price spikes, and the growing impact of renewables
  - » Signing electricity contracts – Requires betting on fuel prices and transmission investments over coming years.
  - » Equipment replacement – Predicting failures in the presence of lightning strikes and power surges.
  - » Planning long term generation investment in the face of uncertainty in the evolution of technology and the price of commodities.
- In recent years, the growing presence of wind and solar is putting more emphasis on handling uncertainty correctly.


UNITED STATES ANNUAL AVERAGE WIND POWER
ALASKA
PRINCIPAL HAWAIIAN ISLANDS
PUERTO RICO
CLASSES OF WIND POWER DENSITY
WIND POWER CLASS
10m (33 ft)
50m (164 ft)
WIND POWER W/m²
SPEED m/s mph
RIDGE CREST ESTIMATES (LOCAL RELIEF > 1000 FT)

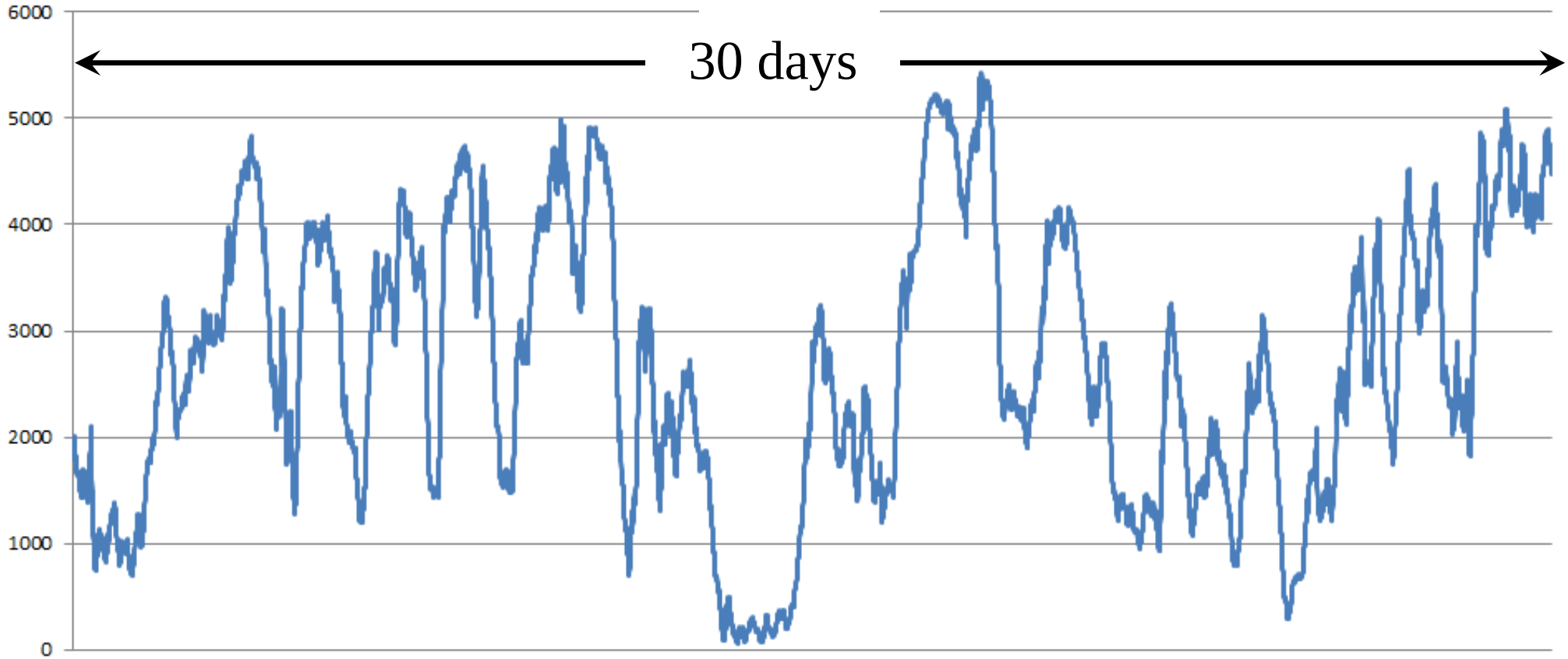
Energy from wind

- Wind power from all PJM wind farms

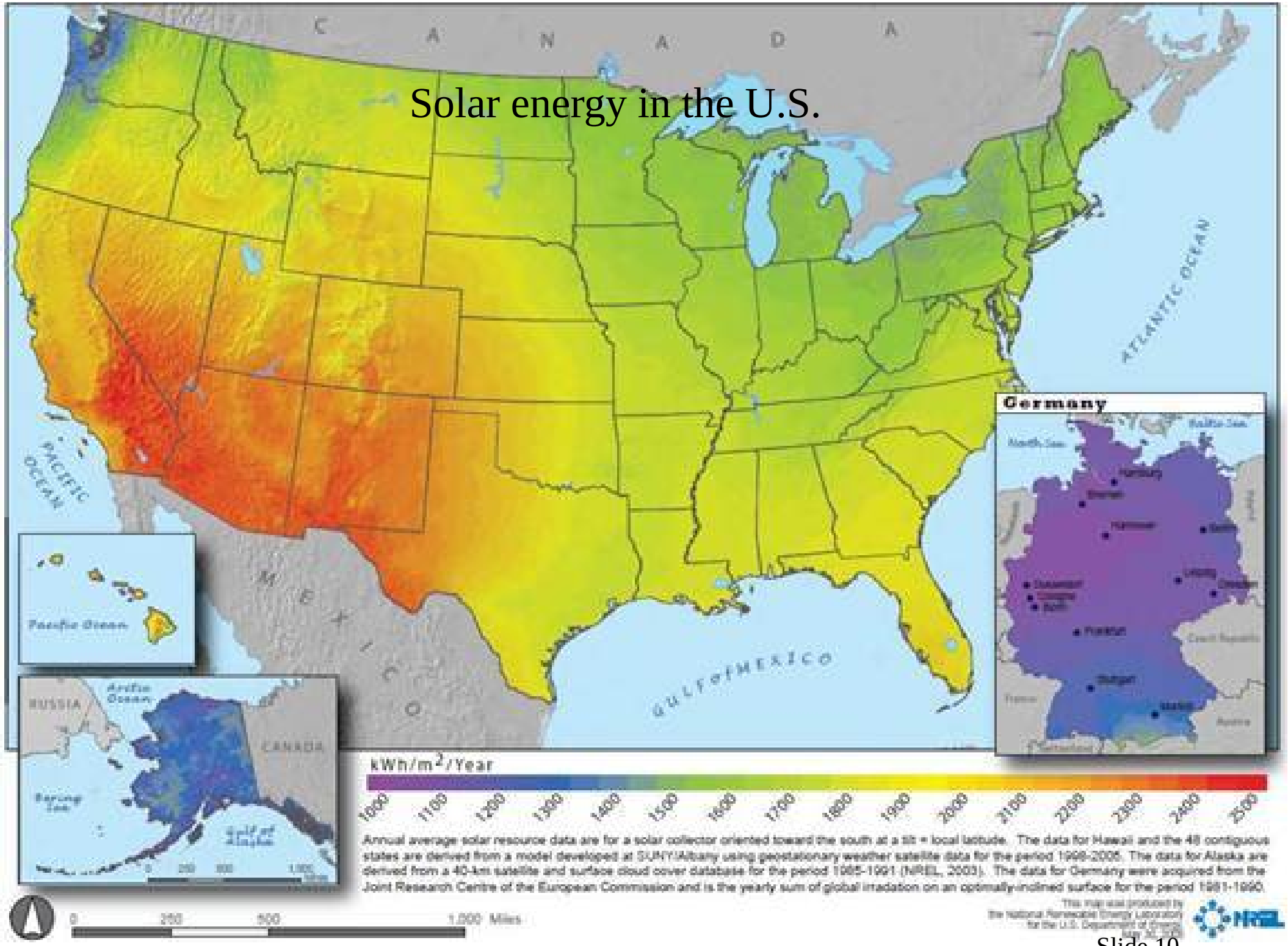


Energy from wind

- Wind from all PJM wind farms



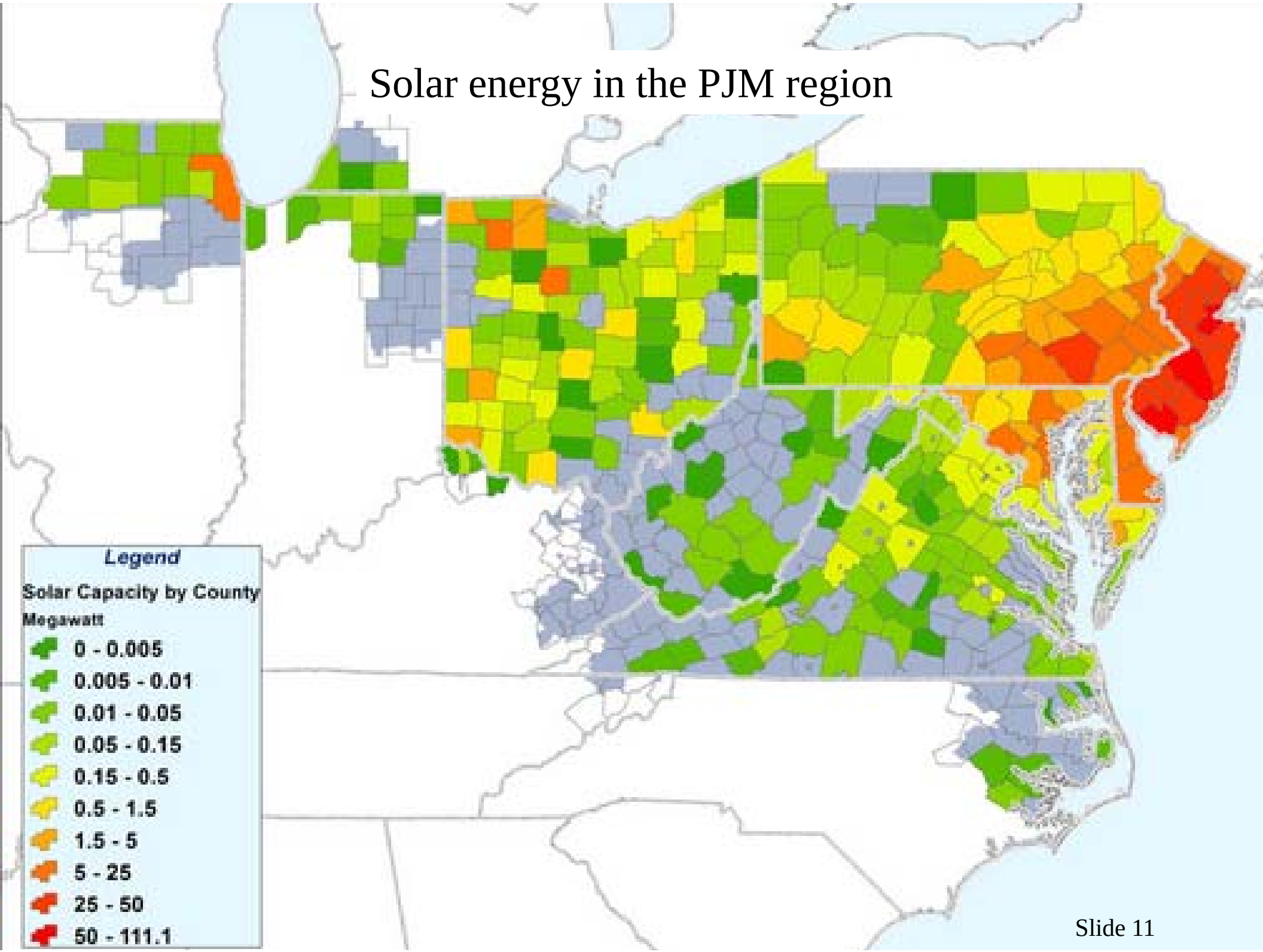
Solar energy in the U.S.



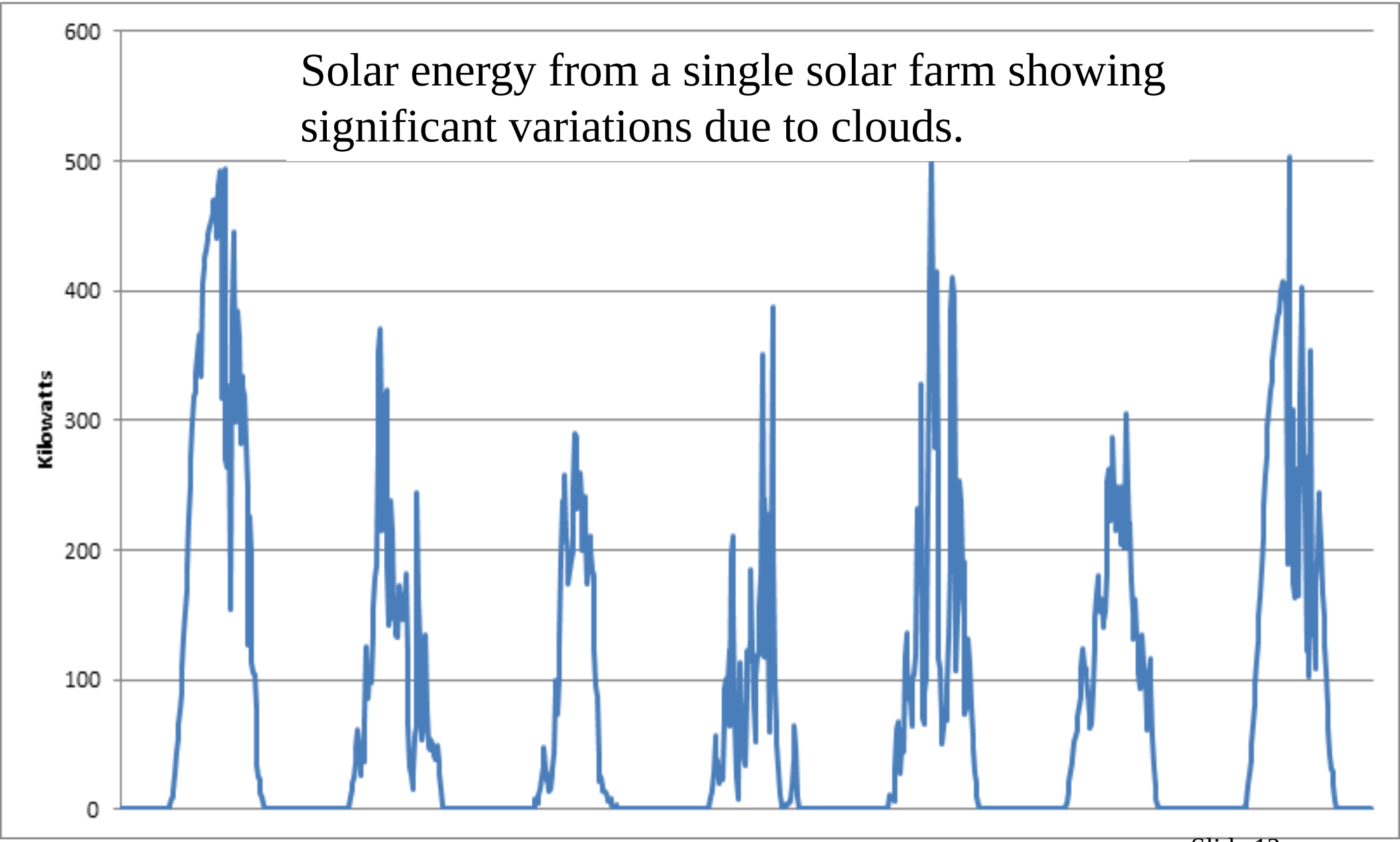
Annual average solar resource data are for a solar collector oriented toward the south at a tilt = local latitude. The data for Hawaii and the 48 contiguous states are derived from a model developed at SUNY/Albany using geostationary weather satellite data for the period 1998-2005. The data for Alaska are derived from a 40-km satellite and surface cloud cover database for the period 1985-1991 (NREL, 2003). The data for Germany were acquired from the Joint Research Centre of the European Commission and is the yearly sum of global irradiation on an optimally-inclined surface for the period 1981-1990.

This map was produced by the National Renewable Energy Laboratory for the U.S. Department of Energy

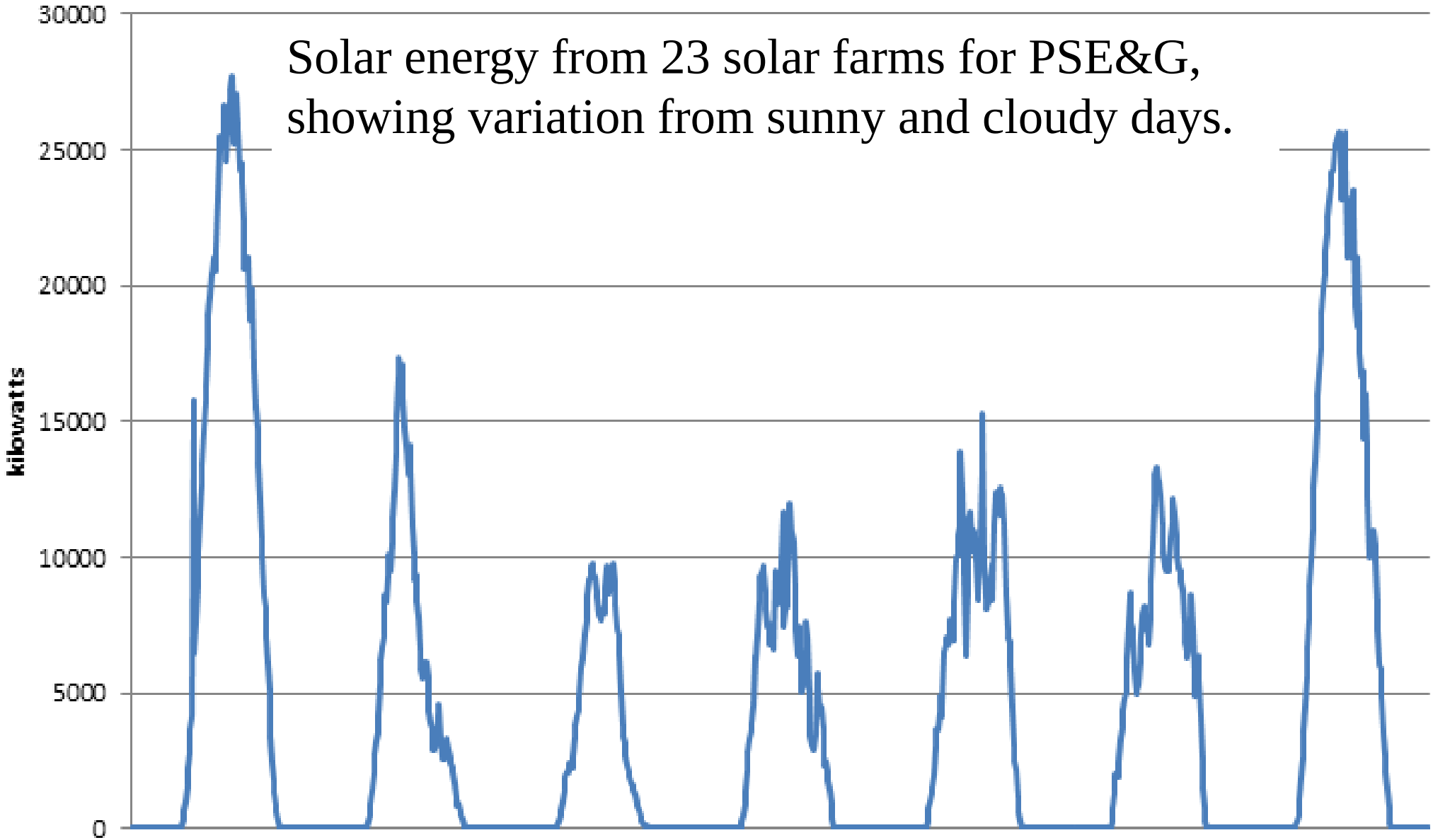
Solar energy in the PJM region



Solar from PSE&G solar farms



Solar from PSE&G solar farms

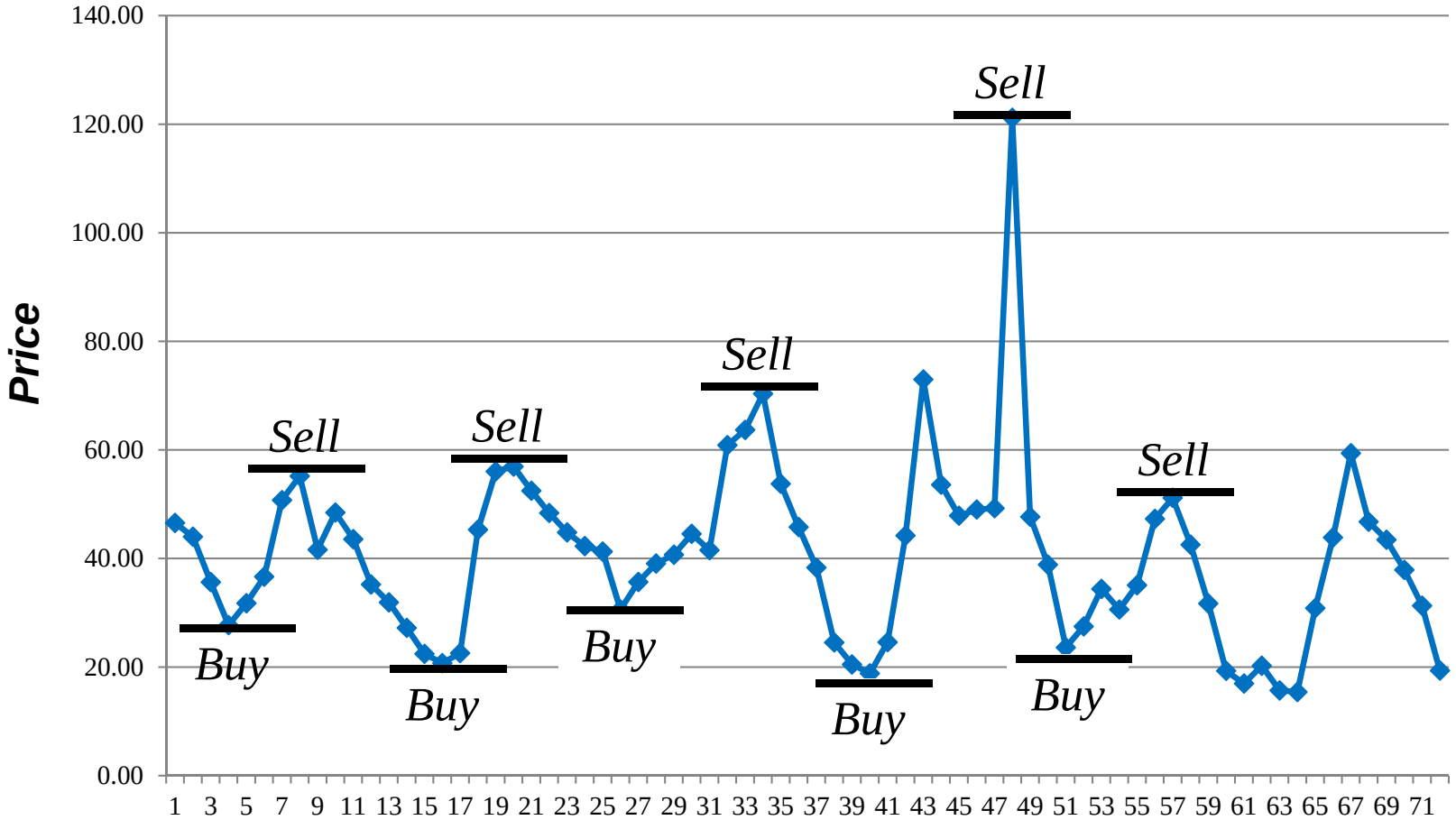


Energy uncertainty

- Handling uncertainty can be subtle
  - » We need to understand how to plan in the face of uncertainty, so that we can understand how much energy we can absorb from renewables.
  - » The problem is that it is *very* easy to cheat when running these simulations.
  - » The next slide describes a study performed by a battery manufacturer promoting the use of their battery for "battery arbitrage" where you store energy when prices are low, and then sell when they are high. You will see in the next slide why you can't actually do this.
  - » We will then show that our "cheating" policy dramatically overstates the value of the battery.
  - » It will seem as if this is a silly mistake, but it is easy to make, and this mistake is being made in all "stochastic unit commitment models" that are being proposed today (see if you can catch when this is happening).

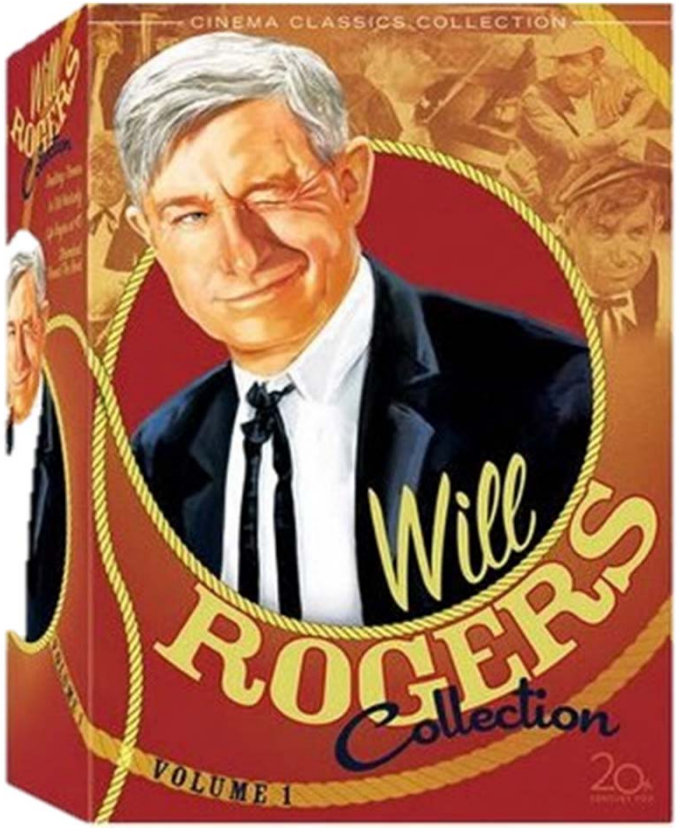
Valuing a battery

- Challenge: find a policy for charging and discharging the battery so we can estimate the value of the battery.
  - » Strategy posed by the battery manufacturer: “Buy low, sell high.” The problem is that you do not know the highest or lowest price until the end of the day. You can only run this policy on a historical dataset.



Valuing a battery

Buy low, sell high was first proposed by the famous financial engineer, Will Rogers, who said:



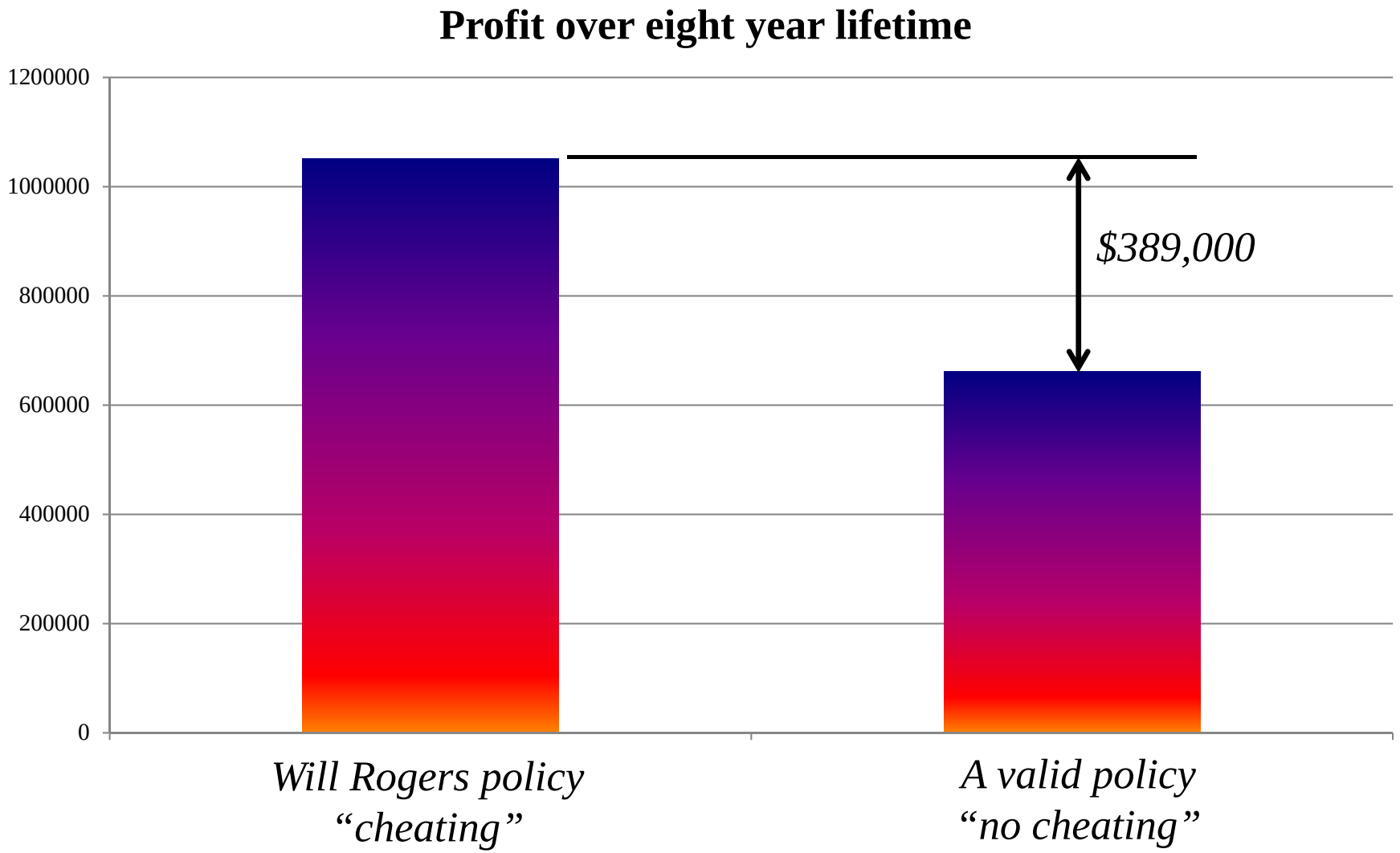
Don't gamble; take all your savings and buy some good stock and hold it till it goes up, then sell it. If it don't go up, don't buy it.

Will Rogers

It is not enough to model the *variability* of a process. You have to model the *uncertainty* – the flow of information.

Valuing a battery

- The value of perfect information



Our "no cheating" policy (known as an "admissible policy") is not optimal, but it is the best within a particular class of policies.

The impact of renewables

- Some authors claim renewables can cover entire load! Here, the problem is an oversimplified model (rather than cheating)

Cost-minimized combinations of wind power, solar power and electrochemical storage, powering the grid up to 99.9% of the time

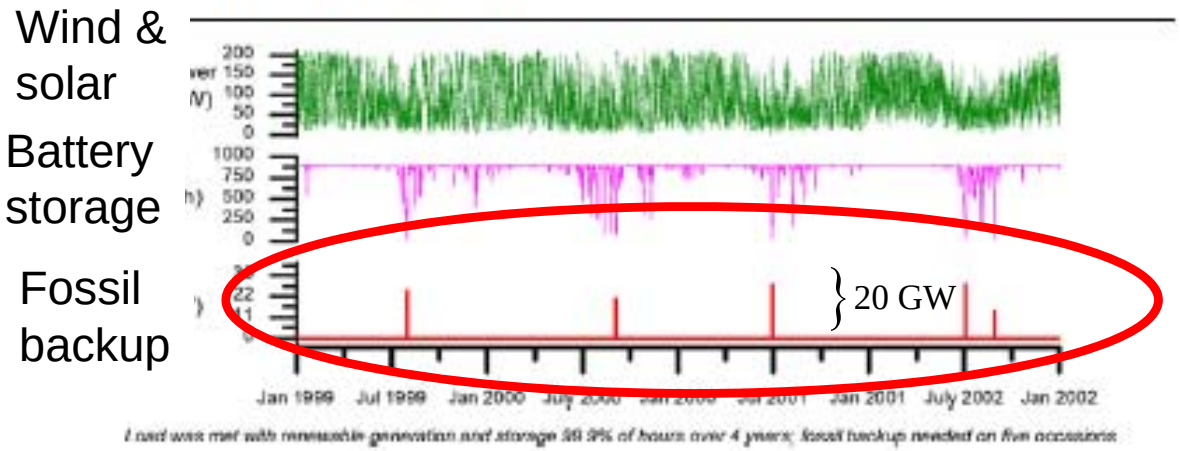
Cory Budischak [a,b,*], DeAnna Sewell [c], Heather Thomson [c], Leon Mach [d], Dana E. Veron [c], Willett Kempton [a,c,e]

[a] Department of Electrical and Computer Engineering, University of Delaware, Newark, DE 19716, USA
[b] Department of Energy Management, Delaware Technical Community College, Newark, DE 19713, USA
[c] Center for Carbon-Free Power Integration, School of Marine Science and Policy, College of Earth Ocean and Environment, University of Delaware, Newark, DE 19716, USA
[d] Energy and Environmental Policy Program, College of Engineering, University of Delaware, Newark, DE 19716, USA
[e] Center for Electric Technology, DTU Elektro, Danmarks Tekniske Universitet, Kgs. Lyngby, Denmark

HIGHLIGHTS

- We modeled wind, solar, and storage to meet demand for 1/5 of the USA electric grid.
- 28 billion combinations of wind, solar and storage were run, seeking least-cost.
- Least-cost combinations have excess generation (3× load), thus require less storage.
- 99.9% of hours of load can be met by renewables with only 9–72 h of storage.
- At 2030 technology costs, 90% of load hours are met at electric costs below today's.

GRAPHICAL ABSTRACT



Load was met with renewable generation and storage 99.9% of hours over 4 years; fossil backup needed on five occasions.

News flash – Oct 29, 2013

- PJM-funded study performed by GE using an older model (MARS) that was not designed to handle uncertainty.

Independent grid operator study confirms wind power's economic, environmental value

29 October 2013 by Michael Goggin 

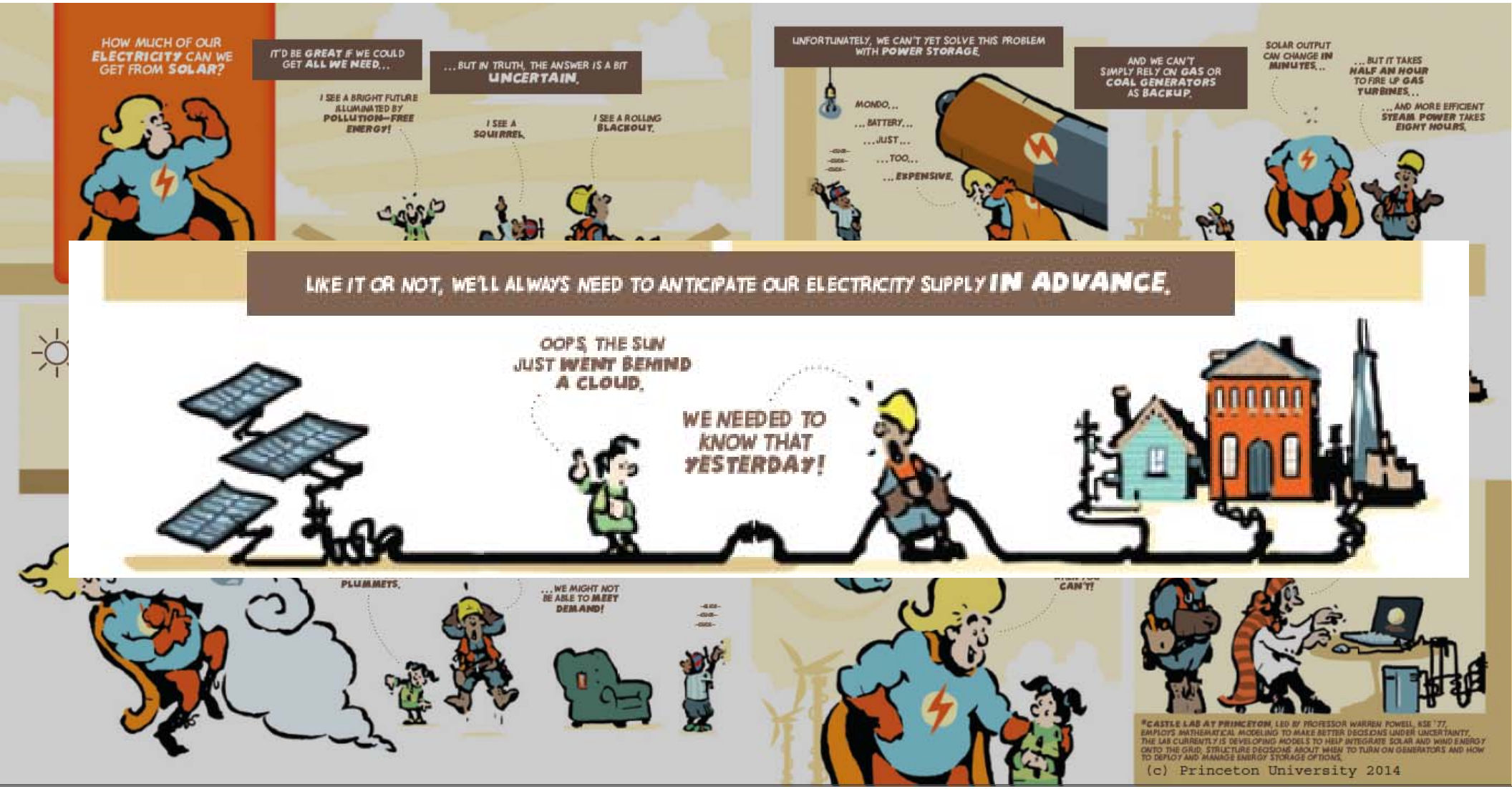
New research released by an independent grid operator confirms that wind energy is drastically decreasing both the price of electricity and emissions of harmful pollutants. The study was led by PJM, the independent grid operator for all or parts of 13 Mid-Atlantic and Great Lakes states (Delaware, Illinois, Indiana, Kentucky, Maryland, Michigan, New Jersey, North Carolina, Ohio, Pennsylvania, Tennessee, Virginia, and West Virginia) and DC. The results are posted here.

Highlights of the study:

- **Wind energy produces massive reductions in electricity production costs and wholesale prices.** Obtaining 20% of PJM's electricity from wind energy reduces the cost of producing electricity by $9 billion annually (about 25% of overall production costs of $37 billion), while 30% wind reduces production costs by $13 billion (about 35%) each year. Wholesale electricity prices

Dealing with uncertainty

- This cartoon highlights the issue of uncertainty:



Available at http://energysystems.princeton.edu

Dealing with uncertainty

- Observations:
  - Uncertainty requires dealing with the flow of *information*. You cannot see it or touch it.
  - It is very easy to cheat when you are working with historical information (as in Will Rogers' "buy low, sell high" policy).
  - Mathematical models which handle decisions and information ("stochastic optimization) are, well, pretty arcane.
  - The point of this tutorial is to remind ourselves that we deal with uncertainty in our everyday lives. We hope to build a bridge between the various academic communities that work in this area. In particular, we will build on the insights that already exist within the energy community.

Lecture outline

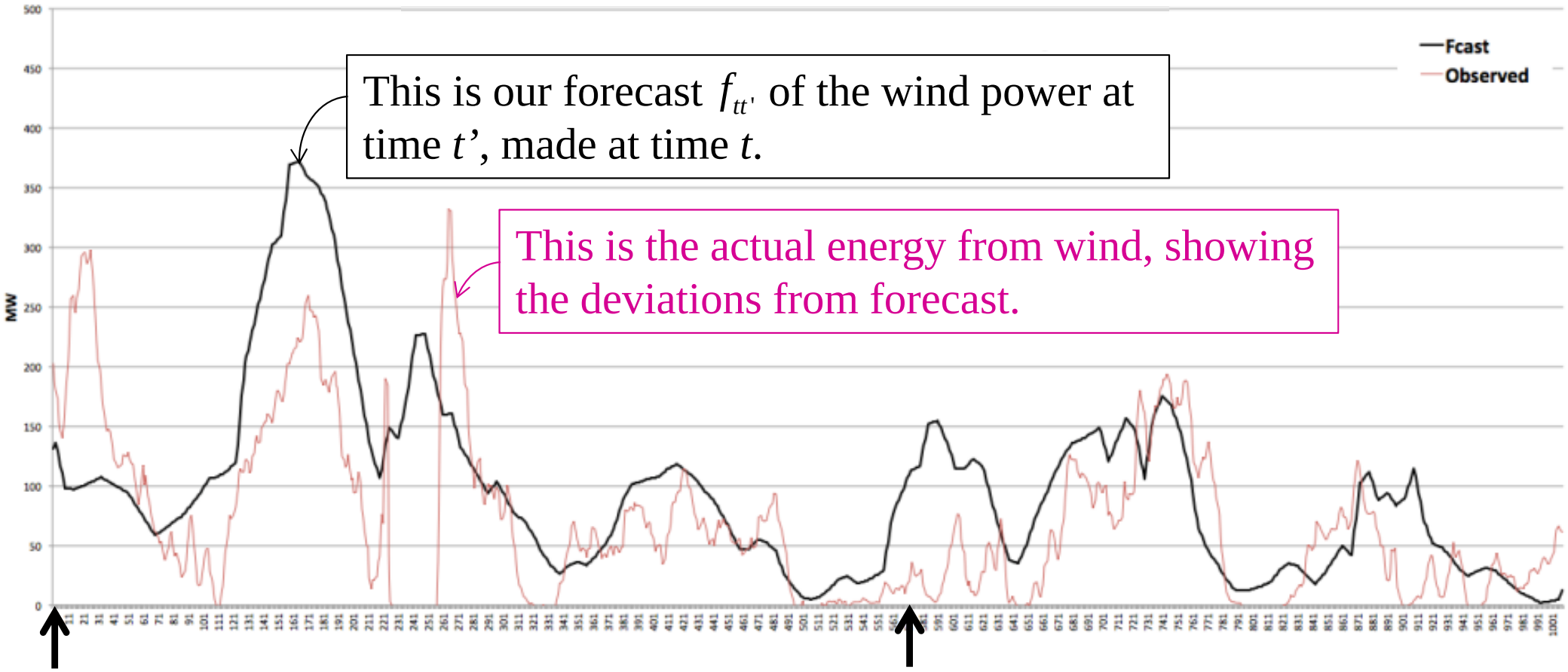
- Stochastic – What does it mean, when is it important?

Stochastic – what is it?

- What is stochastic?
  - » Stochastic means random or uncertain
  - » Stochastic does not mean that something varies.
  - » Stochastic means that it varies in a way that is different than the forecast. More precisely, it varies in a way that is different than what we expected:
    - The pattern of the sun is not stochastic, even though it varies.
    - Solar energy is usually stochastic, but only because we have trouble forecasting cloud cover.
    - We may be able to predict a storm…
    - … but our predictions are not perfect. The storm might arrive earlier or later, winds might be heavier or lighter.

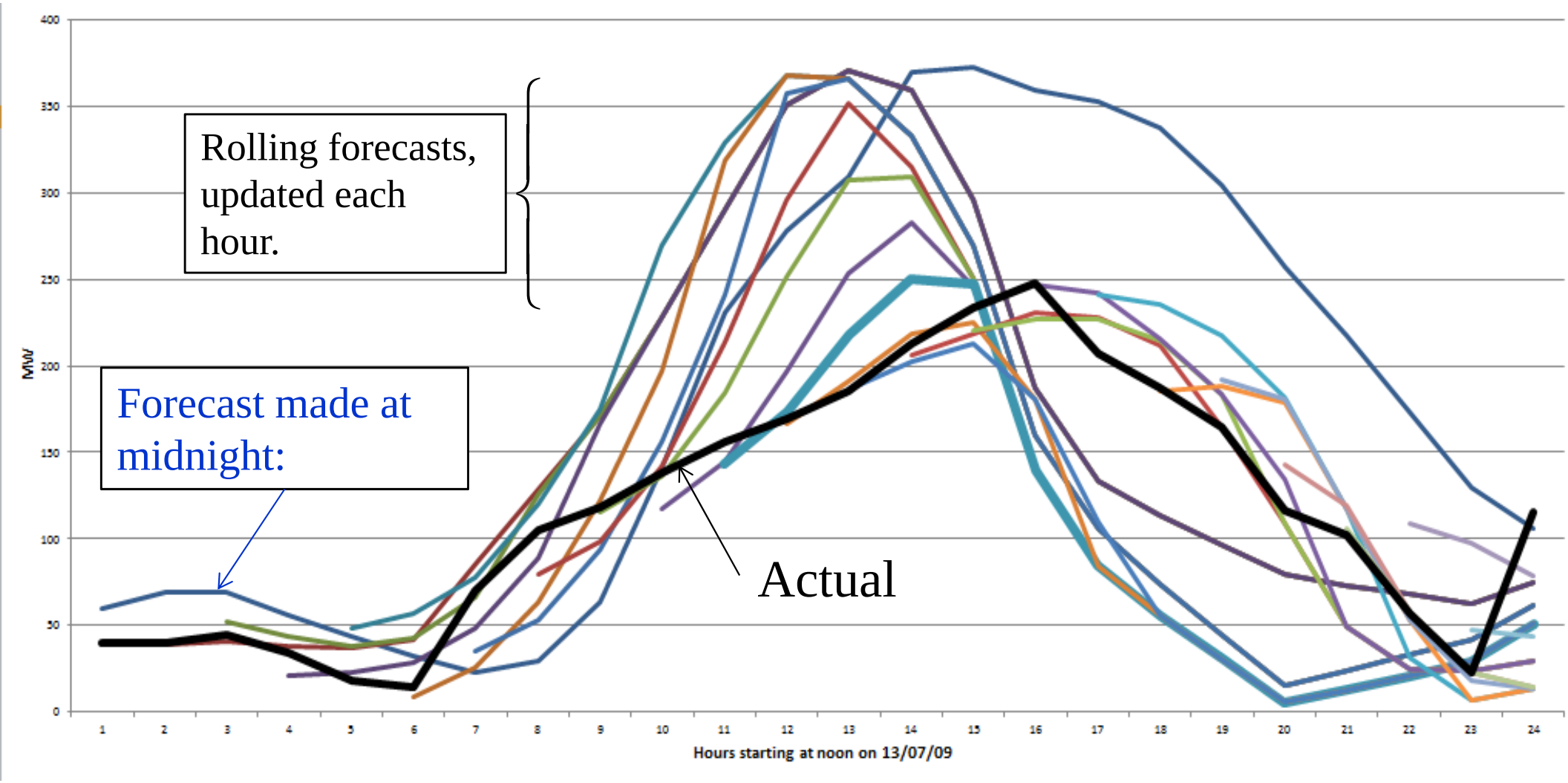
Stochastic – what is it?

- Illustration of forecasted wind power and actual
  - » The forecast (black line) is deterministic (at time t, when the forecast was made). The actuals are stochastic.



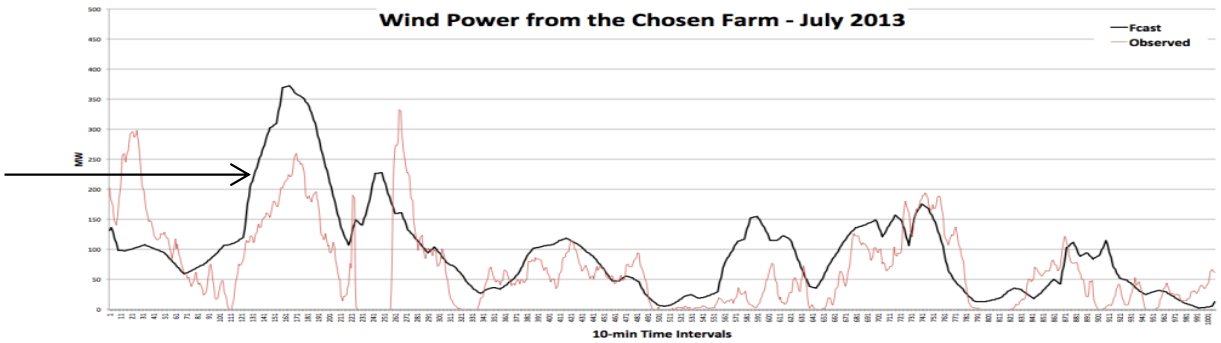
Stochastic – what is it?

- Forecasts evolve over time as new information arrives:



Stochastic – what is it?

- Two types of uncertainty arise in forecasting:
  - » At time *t*, we have forecasts for different times into the future:

$f_{tt'} =$ 

    - The forecast is an imperfect estimate of the actual load at time t':

$$L_{t'} = f_{t,t'} + \varepsilon^{L}_{t'}$$

The actual load at time *t'* is "stochastic" at time *t*.

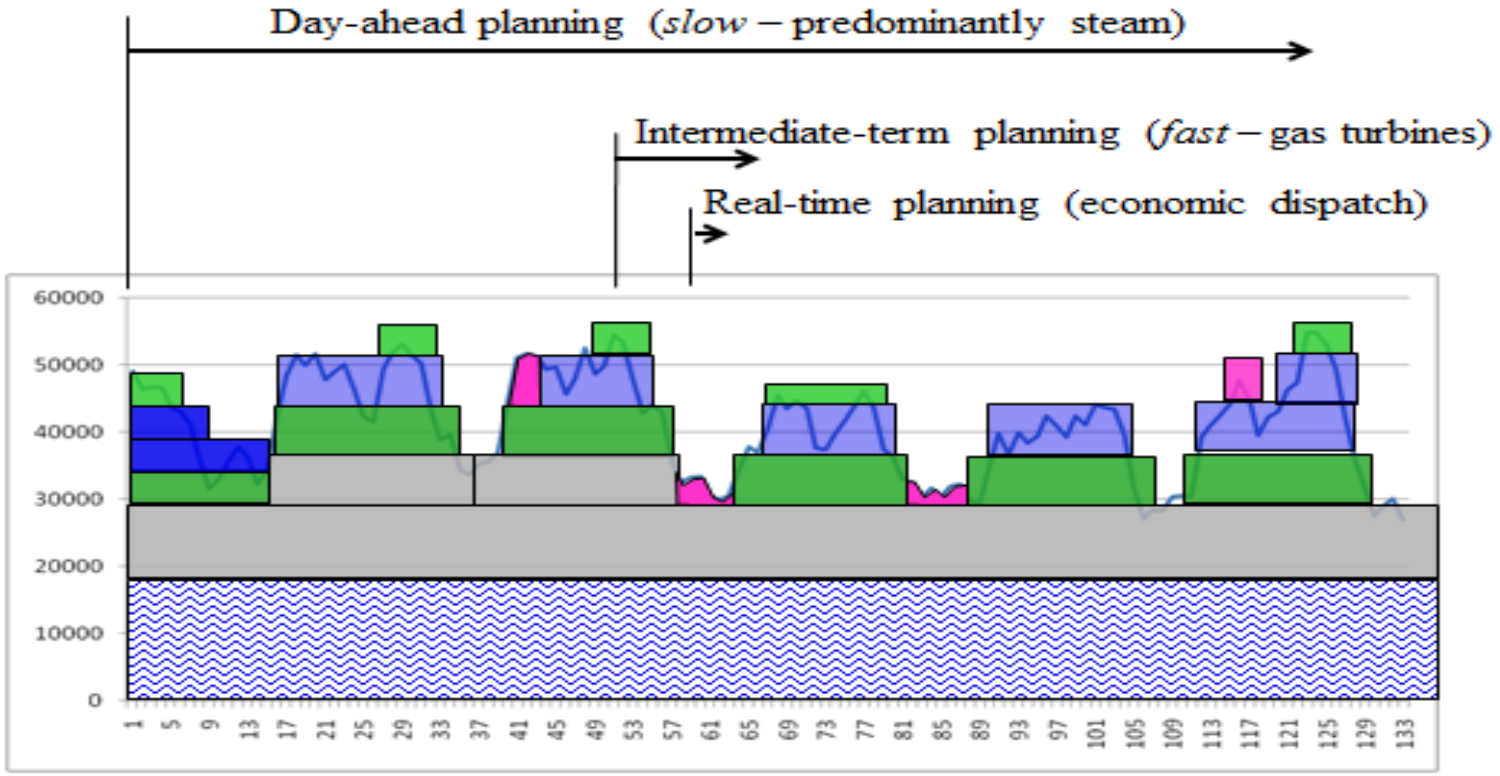
  - » As new information arrives, the forecasts themselves change from time t to t+1:

$$f_{t+1,t'} = f_{t,t'} + \varepsilon^{f}_{t+1,t'}$$

This change in the forecast is "stochastic" at time *t*.

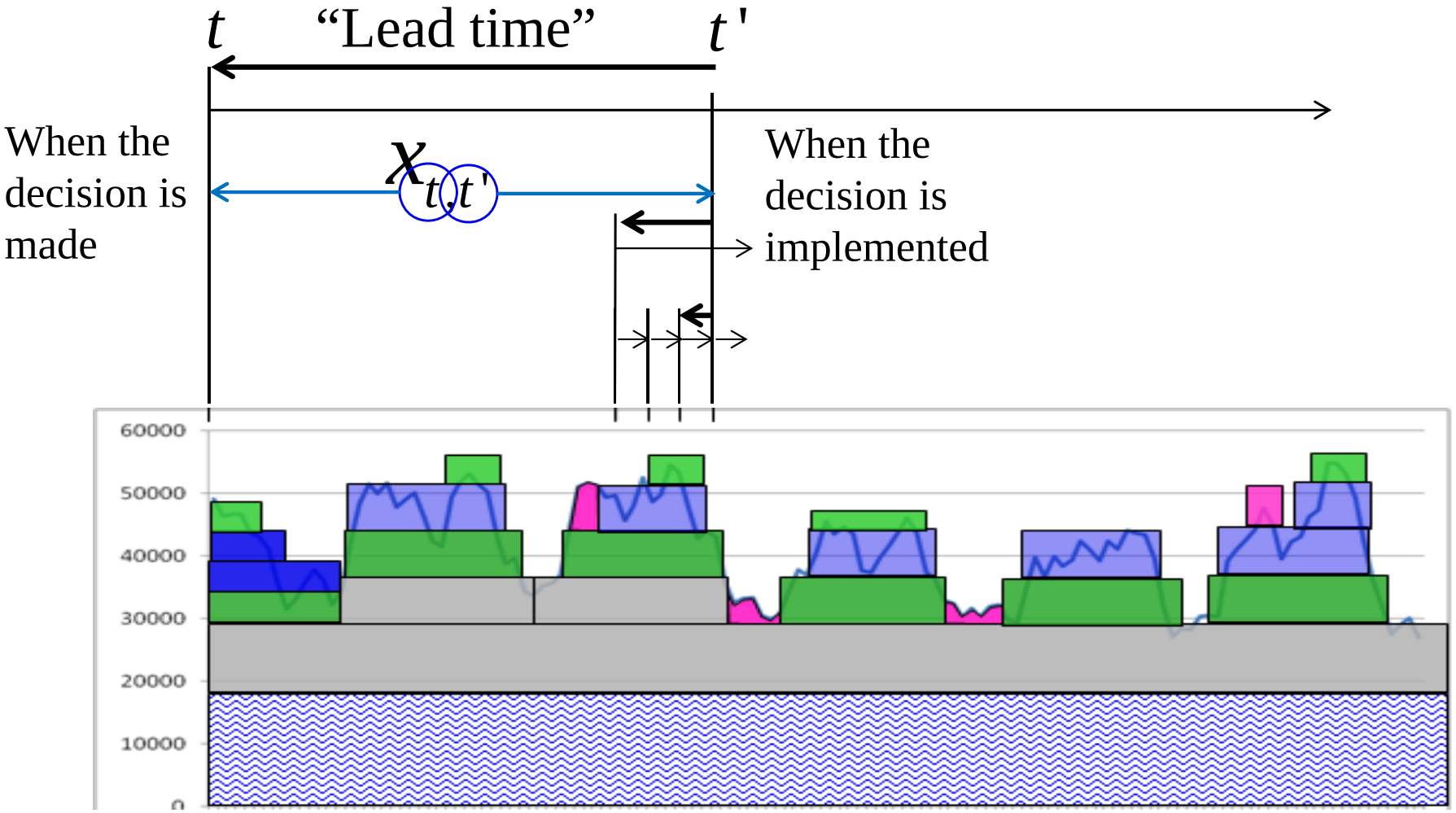
Stochastic – when is it important

- In energy, we often have to make a decision now to do something in the future.
  - » Scheduling energy generation requires planning over different horizons.



Stochastic – when is it important

- Nesting of decisions – We need to keep track of when a decision is made (t) and when it will be implemented (t')



Stochastic – when is it important

- Nesting of decisions – We need to keep track of when a decision is made (*t*) and when it will be implemented (*t'*)
  - The difference between *t'* and *t* is called the *lead time* (also called the *lag*).
  - The time at which a decisions made (*t*), which might be noon of one day, determines what information was available.
  - When the decision is implemented in the future (say, a generator coming on at 3pm tomorrow), we do not know the state of the world at 3pm tomorrow.
  - This is when "stochastic" *might* be important.
  - For unit commitment, it *is* important, but this is nothing new. Uncertainty has *always* been a part of unit commitment.

Lecture outline

- Modeling stochastic, dynamic problems

Deterministic modeling

- For deterministic problems, we speak the language of mathematical programming:
  - A linear program over time:

$$\min \sum_{t=0}^{T} c_t x_t$$

$$A_t x_t - B_{t-1} x_{t-1} = b_t$$

$$D_t x_t \leq u_t$$

$$x_t \geq 0$$

  - This framework is probably Dantzig's biggest contribution – it is a language spoken around the world!
  - But what happens when we insert uncertainty? At that point we fragment into a jungle of different communities that work in stochastic optimization.


Stochastic programming
Robust optimization
Approximate dynamic programming
Model predictive control
Optimal control
Online learning
Reinforcement learning
Markov decision processes
Simulation optimization

Modeling

- We lack a standard language for modeling sequential, stochastic decision problems.
  - » In the slides that follow, we propose to model problems along five fundamental dimensions:
    - State variables
    - Decision variables
    - Exogenous information process
    - Transition function
    - Objective function
  - » This framework is widely followed in the control theory community, and almost completely ignored in operations research and computer science.
  - » We do not even agree on what we are looking for! [Hint: instead of *decisions*, we are looking for *policies*.]

Modeling dynamic problems

- The system state:

$S_t = (R_t, I_t, K_t)$ = System state, where:

R_t = Resource state (physical state)

Location/status of truck/train/plane

Inventory of trailers

I_t = Information state

Prices

Weather

K_t = Knowledge state ("belief state")

Belief about traffic delays

Belief about the status of equipment

The state S_t is a minimally dimensioned function of history that is necessary and sufficient to model the system from time *t* onward.

Modeling dynamic problems

- Decisions:

Computer science

a_t = Discrete action

Control theory

u_t = Low-dimensional continuous vector

Operations research

x_t = Usually a discrete or continuous but high-dimensional vector of decisions.

At this point, we do not specify *how* to make a decision. Instead, we define the function $A^{\pi}(s)$ (or $X^{\pi}(s)$ or $U^{\pi}(s)$), where π specifies the class of policy, and any tunable parameters (which we represent using θ).

Modeling dynamic problems

- Exogenous information:

W_t = New information = $\left(\hat{R}_t, \hat{D}_t, \hat{p}_t, \hat{E}_t\right)$

$\hat{R}_t$ = Rainfall, generator failures, cloud cover, storms

$\hat{D}_t$ = Variations in customer loads

$\hat{p}_t$ = Changes in prices

$\hat{E}_t$ = Information about the environment (temperature, ...)

Note: Any variable indexed by *t* is known at time *t*. This convention, which is not standard in control theory, dramatically simplifies the modeling of information.

Throughout, we let the Greek letter ω represent a sample realization of the sequence $W_1, W_2, ..., W_T$.

Modeling dynamic problems

- The transition function

$$S_{t+1} = S^M(S_t, \ x_t, \ W_{t+1})$$

$$R_{t+1} = R_t + x_t + \hat{R}_{t+1}$$ Water in the reservoir

$$p_{t+1} = p_t + \hat{p}_{t+1}$$ Spot prices

$$D_{t+1} = D_t + \hat{D}_{t+1}$$ Electrical loads

Also known as the:

- "System model"
- "State transition model"
- "Plant model"
- "Plant equation"
- "Transition law"
- "Transfer function"
- "Transformation function"
- "Law of motion"
- "Model"

For many applications, these equations are unknown. This is known as "model-free" dynamic programming.

Stochastic optimization models

- The objective function

$$\min_{\pi} \mathbb{E}^{\pi} \left\{ \sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^{\pi}(S_t)\right) \right\}$$

Expectation over all random outcomes

Finding the best policy

Cost function

State variable

Decision function (policy)

Given a *system model* (transition function)

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}(\omega)\right)$$

We refer to this as the *base model* to avoid confusions with *lookahead models* we will introduce later.

Objective functions

- There are different objectives that we can:
  - » Expectations

$$\min_x \mathbb{E}F(x,W)$$

  - » Risk measures

$$\min_x \mathbb{E}F(x,W) + \theta\mathbb{E}\left[F(x,W) - f_\alpha\right]_+^2$$

$$\min_x \rho\left(F(x,W)\right) \qquad \rho \in \text{Convex/coherent risk measures}$$

  - » Worst case ("robust optimization")

$$\min_x \max_w F(x,w)$$

"Robust optimization" is attracting more interest as a policy for solving sequential problems using *expectations*!

Designing a policy

- In practice, we cannot compute the expectation in:

$$\min_\pi \mathbb{E}^\pi \left\{ \sum_{t=0}^{T} \gamma^t C\left(S_t, X^\pi(S_t)\right) \right\}$$

- Instead, we might do one long simulation…

$$\min_\pi \hat{F}^\pi = \sum_{t=0}^{T} \gamma^t C\left(S_t(\omega), X_t^\pi(S_t(\omega))\right)$$

- …or we might average across several simulations:

$$\min_\pi \bar{F}^\pi = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=0}^{T} \gamma^t C\left(S_t(\omega^n), X_t^\pi(S_t(\omega^n))\right)$$

Stochastic optimization models

- With deterministic problems, we want to find the best *decision:*

$$\min_{x_0,\ldots,x_T} \sum_{t=0}^{T} c_t x_t$$

- With stochastic problems, we want to find the best *function* (policy) for making a decision:

$$\min_{\pi} \mathbb{E}^{\pi} \left\{ \sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^{\pi}(S_t)\right) \right\}$$

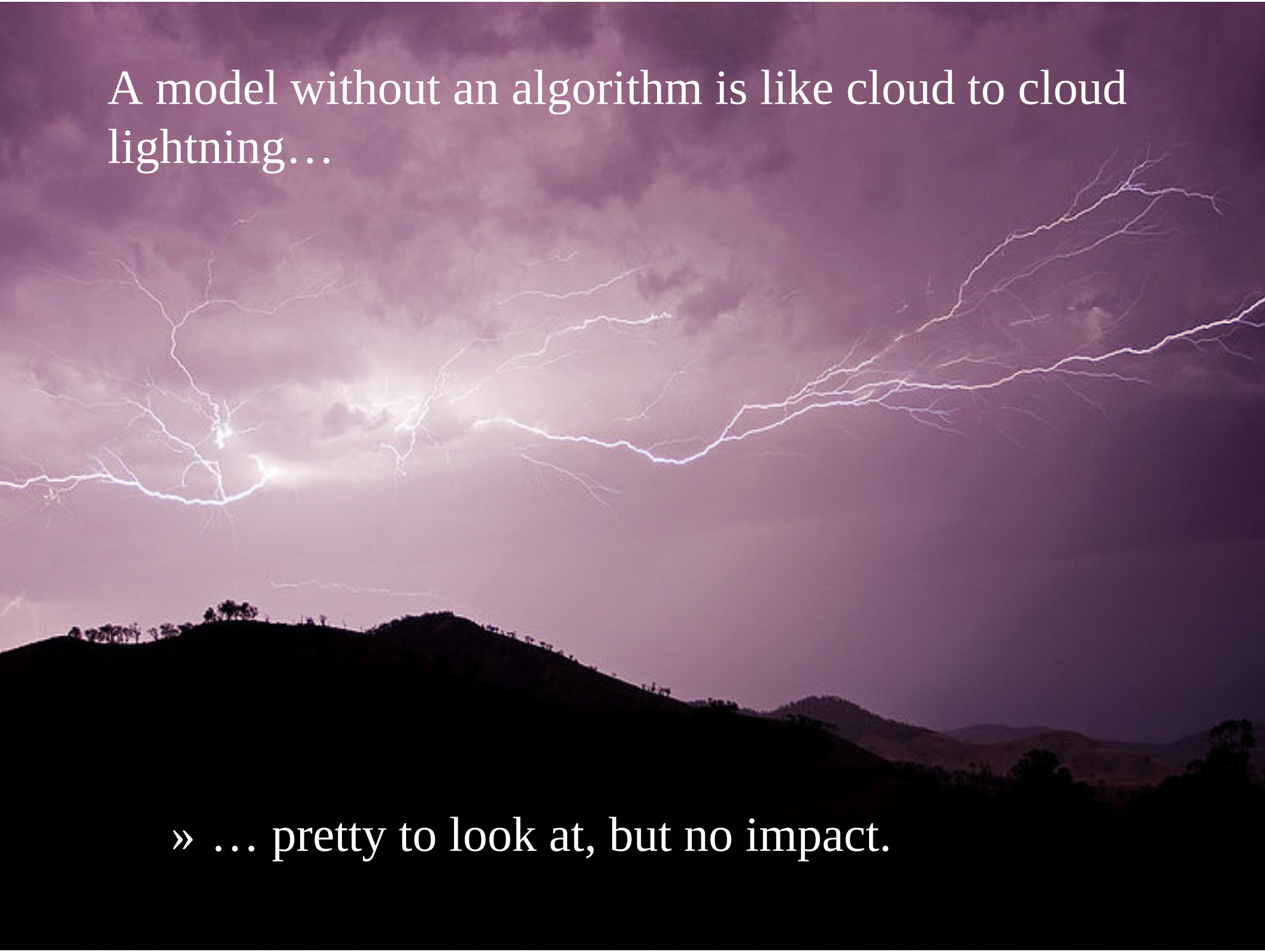
  - … which is sometimes written

$$\min_{x_0,\ldots,x_T} \mathbb{E}^{\pi} \left\{ \sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^{\pi}(S_t)\right) \right\}$$

where x_t is $\mathcal{F}_t$ – measurable (which means we have the information to compute it at time t).

Stochastic optimization models

- Notes:
  - The problem with writing the problem in this way is that there is no obvious path to computation.
  - Formulations that require “minimizing over policies” come across as a lot of hand waving. They are popular with mathematicians, but seem meaningless to people interested in doing computation.
  - The last formulation is the hardest to understand – we are optimizing over a set of decisions X_t that depend on the information available at time *t*, *which is not known now!* That means that these decisions are random variables! What in the world does this mean???!!!
  - It turns out, this is *exactly* the formulation favored by the stochastic programming community solving the stochastic unit commitment problem!
  - Mathematically pretty to look at, but no impact, just like cloud-to-cloud lightning…


A model without an algorithm is like cloud to cloud lightning…
» … pretty to look at, but no impact.

SMART-ISO: Calibration

- Any dynamic model consists of two fundamental equations:
  - The decisions (determined by a *policy*)

$$x_t = X^{\pi}(S_t)$$

  - The dynamics (captured by the *transition function*)

$$S_{t+1} = S^{M}\left(S_t, x_t, W_{t+1}\right)$$

  - The transition function captures all the physics of the problem (some communities call this the “model”)
  - Given a policy, we can simulate a policy by stepping forward through time, replacing all the dynamics of the system.
  - Now how do we find the best policy?

Modeling

- We have to start by describing what we mean by a policy.
  - » Definition:

    A policy is a mapping from a state to an action. … any mapping.

- There appear to be four fundamental classes of policies:
  - » Policy function approximations (PFAs)
  - » Cost function approximations (CFAs)
  - » Value function approximations (VFAs)
  - » Lookahead policies

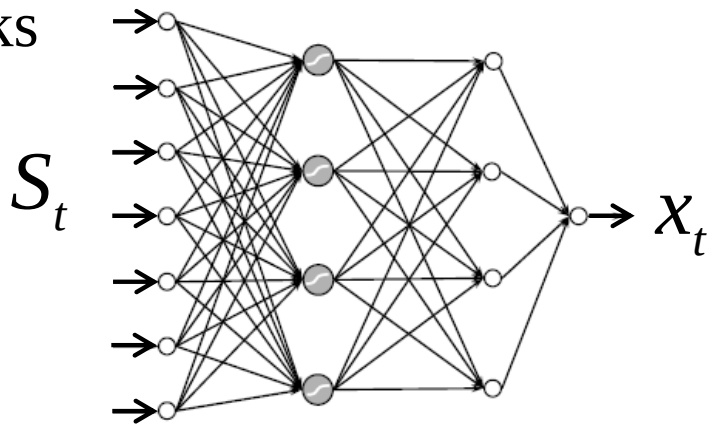
Four classes of policies

1) Policy function approximations (PFAs)

- Lookup table
  - When the chess board is in a certain state, make a particular move (used in early stages of the game).
- Parameterized functions
  - Recharge the battery when the price is below θ^{charge} and discharge when the price is above $\theta^{\text{discharge}}$
- Regression models

$$X^{PFA}(S_t \mid \theta) = \theta_0 + \theta_1 S_t + \theta_2 \left(S_t\right)^2$$

- Neural networks



Four classes of policies

2) Cost function approximations (CFAs)

» A simple myopic policy might ignore the future:

$$X^M(S_t) = \arg\max_{x_t} C(S_t, x_t)$$

» We may use a ***cost function approximation*** to encourage good behaviors over time:

$$X^{CFA}(S_t \mid \theta) = \arg\max_{x_t} \bar{C}^{\pi}(S_t, x_t \mid \theta)$$

» For example, we might add a correction term

$$X^{CFA}(S_t \mid \theta) = \arg\max_{x_t} \underbrace{C(S_t, x_t) + \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t, x_t)}_{\bar{C}^{\pi}(S_t, x_t \mid \theta)}$$

» Below, we are going to use CFAs to create modified objective functions to make deterministic approximations robust. We claim this is standard industry practice.

Four classes of policies

3) Value function approximations (VFAs)

» Using the pre-decision state

$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \gamma \mathbb{E}\left\{ \bar{V}_{t+1}(S_{t+1}) \,|\, S_t \right\} \right)$$

» Or the post-decision state:

Value of downstream state given the action we take now.

$$X_t^{VFA}(S_t) = \arg\max_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^x \left(S_t^x(S_t, x_t) \right) \right)$$

» This is what most people associate with "approximate dynamic programming" or "reinforcement learning".

» The value function captures the value of being in a downstream state as a result of the action *x* taken now.

Four classes of policies

4) Lookahead policies

Plan over the next T periods, but implement only the action it tells you to do now.

- » Deterministic forecast

$$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

- » Stochastic programming (e.g. two-stage)

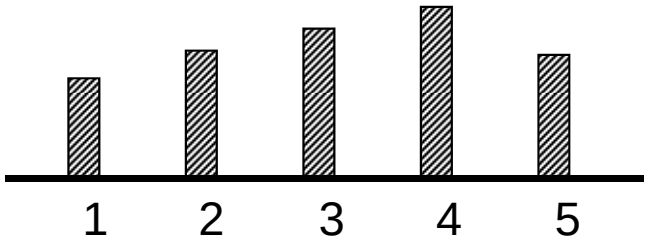
$$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \left(\tilde{x}_{t,t+1}(\omega), \ldots, \tilde{x}_{t,t+T}(\omega)\right), \omega \in \tilde{\Omega}_t} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\omega \in \tilde{\Omega}_t} p(\omega) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\omega), \tilde{x}_{tt'}(\omega))$$

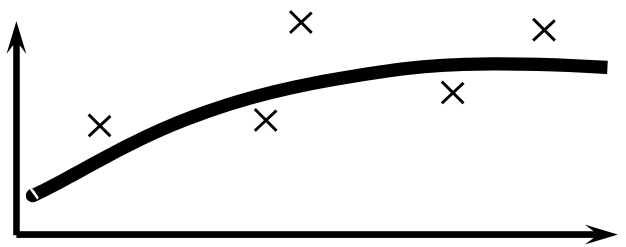
Scenario trees

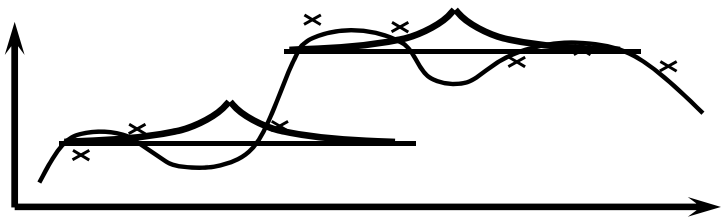
- » Rolling/receding horizon procedures
- » Model predictive control
- » Rollout heuristics
- » Decision trees, Monte Carlo tree search

Four classes of policies

- Three of the policies require approximating functions (PFAs, CFAs and VFAs) – there are three broad ways of classifying functions:
  - » Lookup table
    - Given a discrete state, return a discrete action or value
  - » Parametric models
    - Linear models (“basis functions”)
    - Nonlinear models
  - » Nonparametric models
    - Kernel regression
    - Neural networks







Four classes of policies

- (Slightly arrogant) claim:
  - These are the fundamental building blocks of *all* policies.
  - This means that we can use these policies to describe what the ISOs are already doing, which we claim is a good starting point.
- Many variations can be built using hybrids
  - Lookahead plus value function for terminal reward
  - Myopic or lookahead policies with policy function approximation
  - Modified lookahead policy (lookahead with CFA)

Searching for policies

- We start by formulating our (approximate) objective function (which has to be simulated):

$$\min_{\pi} \hat{F}^{\pi} = \sum_{t=0}^{T} \gamma^t C\left(S_t(\omega), X_t^{\pi}(S_t(\omega))\right)$$

$$S_{t+1} = S^M\left(S_t, X_t^{\pi}(S_t), W_{t+1}\right)$$

- Finding the best policy means:

  - » Search over different classes of policies:
    - PFAs, CFAs, VFAs and lookaheads.
    - Hybrids (VFA+lookahead, CFA+PFA, …)

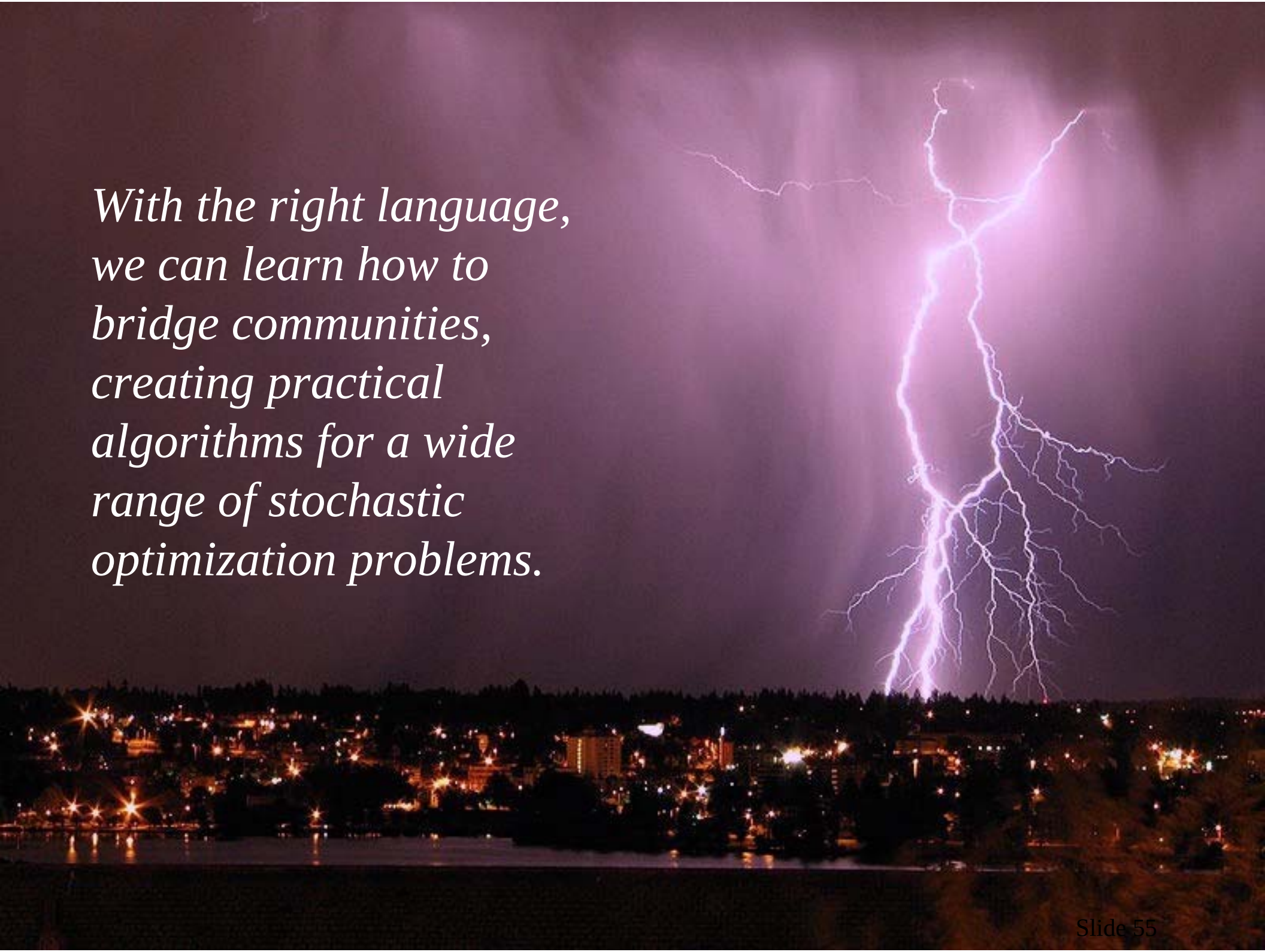
  - » Search over tunable parameters within each class.

Searching for policies

- There are tunable parameters for *every* class of policy:
  - » PFAs – These are parametric functions characterized by parameters such as:
    - $\theta = (s, S)$ parameters for an inventory policy
    - $X^{PFA}(S_t) = \theta_0 + \theta_1 S_t + \theta_2 S_t^2$
  - » CFAs – A parameterized cost function
    - Bonus and penalties to encourage certain behaviors
    - Constraints to ensure buffer stocks or schedule slack
  - » VFAs – These *may* be parameterized approximations of the value of being in a state:
    - $\bar{V}(S_t) = \sum_{f \in F} \theta_f \phi_f(S_t)$
  - » Lookaheads – Choices of planning horizon, number of stages, number of scenarios, …. All can be tuned.

Searching for policies

- Parameter tuning can be done offline or online:
  - Offline (in the lab)
    - Stochastic search
    - Simulation optimization
    - Global optimization
    - Black box optimization
    - Ranking and selection (for off-line learning)
    - Knowledge gradient (offline)
  - Online (in the field)
    - Bandit problems
      - Gittins indices
      - Upper confidence bounding
    - Knowledge gradient (online)


With the right language, we can learn how to bridge communities, creating practical algorithms for a wide range of stochastic optimization problems.

Comparison

- The next slide might be the most important in the presentation…
- … it shows the elements of a deterministic optimization problem, and the corresponding elements of a stochastic optimization problem.
- The communities that use lookahead policies (stochastic programming, robust optimization) almost uniformly overlook the objective function as shown.
- Also, contrast “decisions” for deterministic problems and “policies” for stochastic problems.

Comparing modeling frameworks

- Deterministic
  - » Objective function

$$\min_{x_0,\ldots,x_T} \sum_{t=0}^{T} c_t x_t$$

  - » Decision variables:

$$(x_0,\ldots,x_T)$$

  - » Constraints:
    - at time t

$$\left.\begin{aligned} A_t x_t &= R_t \\ x_t &\geq 0 \end{aligned}\right\} \mathcal{X}_t$$

    - Transition function

$$R_{t+1} = b_{t+1} + B_t x_t$$

- Stochastic
  - » Objective function

$$\min_{\pi} \mathbb{E}^{\pi} \left\{ \sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^{\pi}(S_t)\right) \right\}$$

  - » Policy

$$\pi : S \mapsto X$$

  - » Constraints at time t

$$x_t = X_t^{\pi}(S_t) \in \mathcal{X}_t$$

  - » Transition function

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}\right)$$

  - » Exogenous information

$$(W_1, W_2, \ldots, W_T)$$

Lecture outline

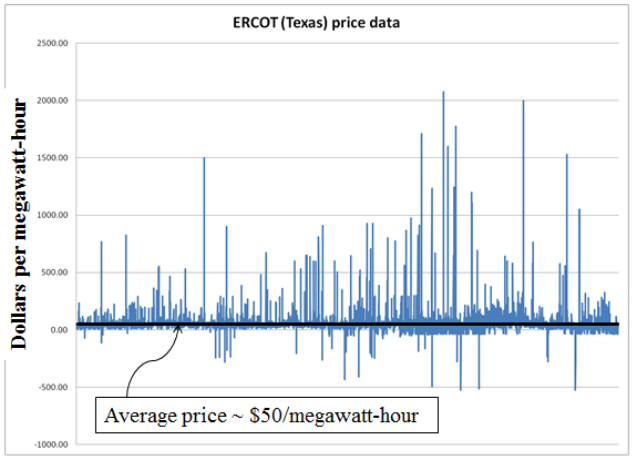
- The four classes of policies
  - Policy function approximations (PFAs)
  - Robust cost function approximations (CFAs)
  - Value function approximations (VFAs)
  - Lookahead policies

Lecture outline

- The four classes of policies
  - Policy function approximations (PFAs)
  - Robust cost function approximations (CFAs)
  - Value function approximations (VFAs)
  - Lookahead policies

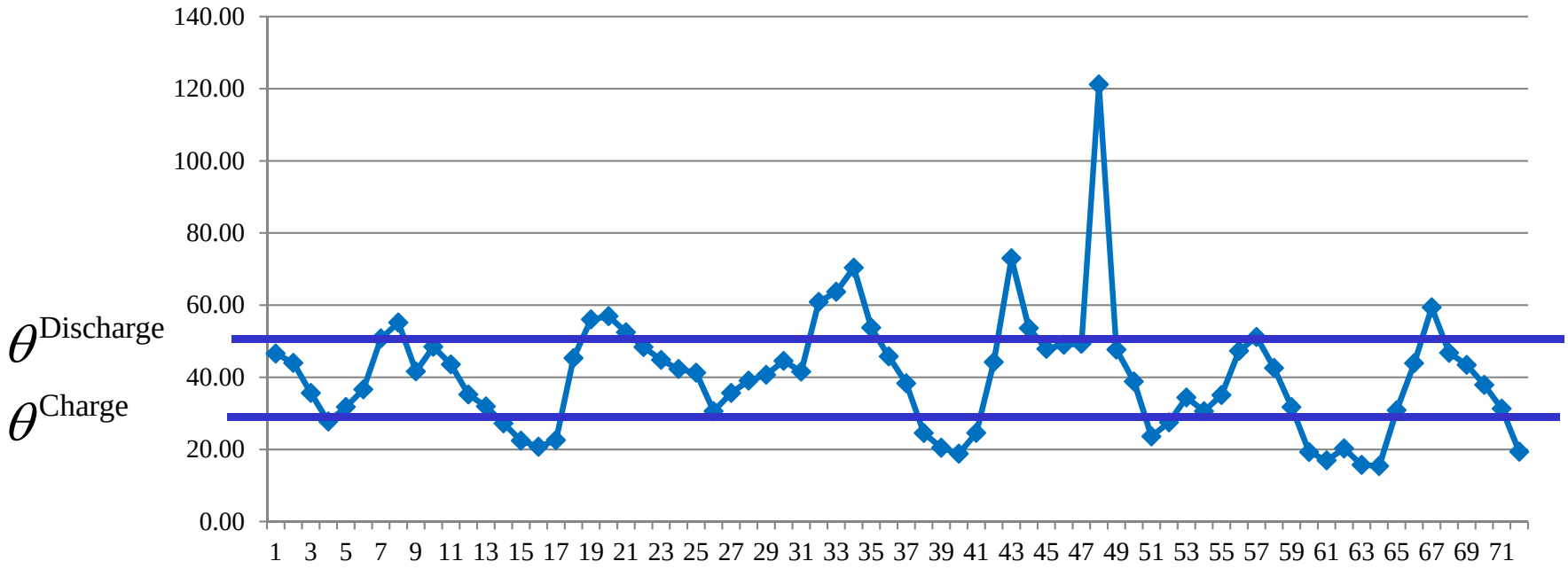
Policy function approximations

- Battery arbitrage – When to charge, when to discharge, given volatile LMPs



Policy function approximations

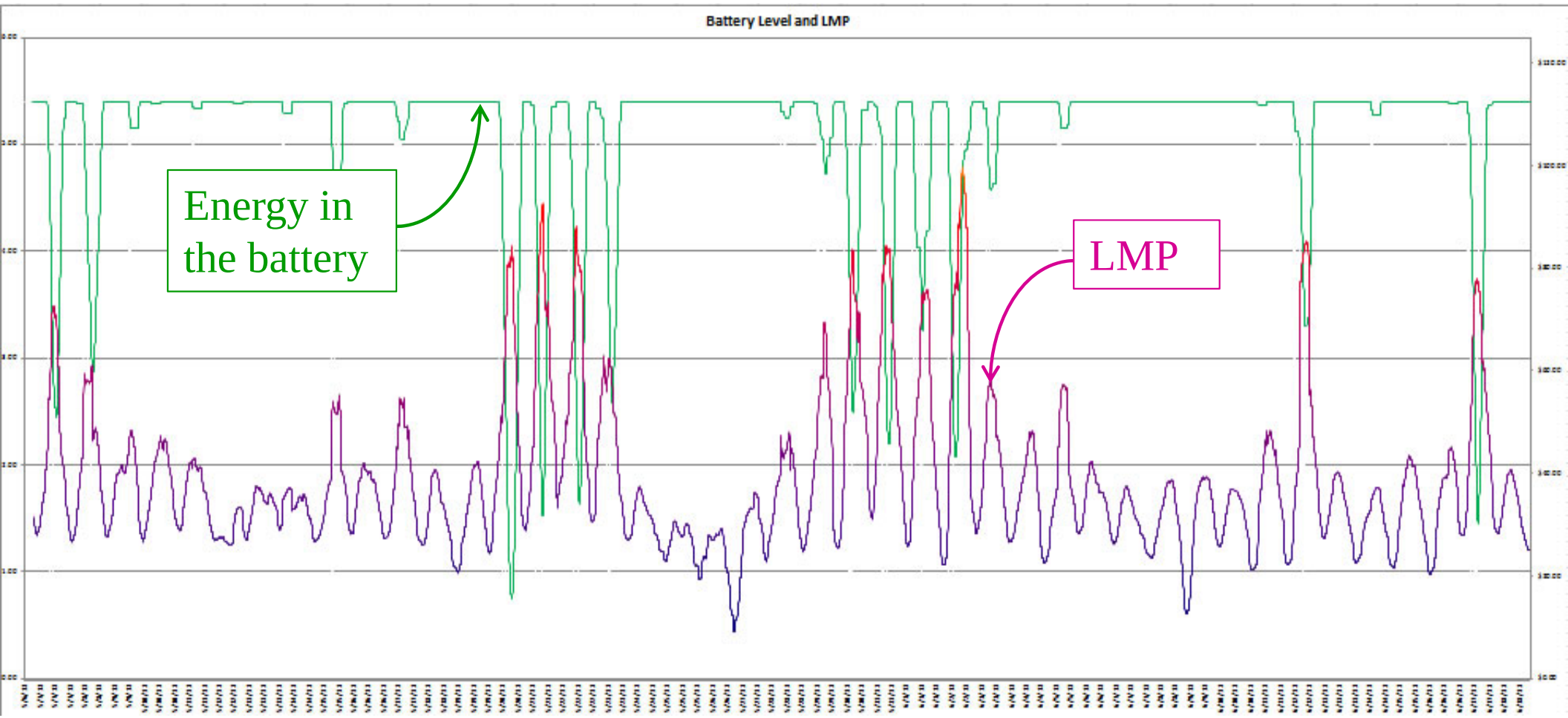
- Grid operators require that batteries bid charge and discharge prices, an hour in advance.



- We have to search for the best values for the policy parameters θ^{Charge} and $\theta^{Discharge}$.

Policy function approximations

- The charge/discharge policy holds energy in the battery until prices peak, and then sells quickly. Charging is done with the prices drop below some point.



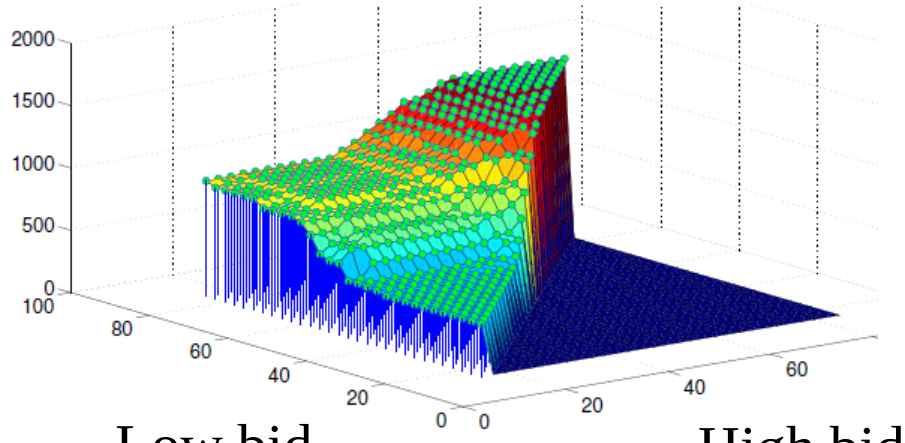
Policy function approximations

- Our policy function might be the parametric model (this is nonlinear in the parameters):

$$X^{\pi}(S_t \mid \theta) = \begin{cases} +1 & \text{if } p_t < \theta^{\text{charge}} \\ 0 & \text{if } \theta^{\text{charge}} < p_t < \theta^{\text{discharge}} \\ -1 & \text{if } p_t > \theta^{\text{charge}} \end{cases}$$

- ... or perhaps a lookup table:

$$X^{\pi}(S_t \mid \theta) =$$

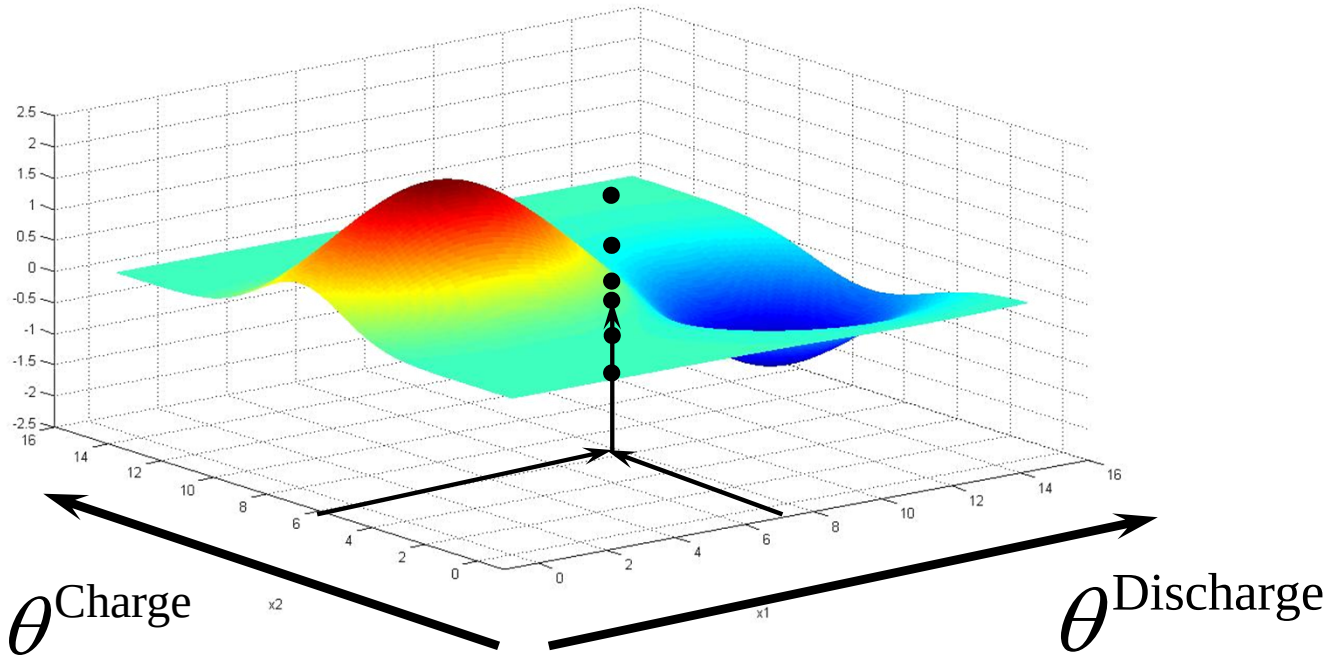


Policy function approximations

- Finding the best policy
  - » We need to maximize

$$\max_\theta F(\theta) = \mathbb{E}\sum_{t=0}^{T} \gamma^t C\left(S_t, X_t^\pi(S_t \mid \theta)\right)$$

  - » We cannot compute the expectation, so we run simulations:



Co-optimizing solar-storage

SMART-Solar: Co-optimization of solar and storage

Regular Charge @ $	56
Regular Discharge @ $	71
Cold Charge @ $	95
Cold Discharge @ $	115
Hot Charge @ $	100
Hot Discharge @ $	118
Grid Charge @ $	60
Grid Discharge @ $	120
Temp (F)	
Hot	75
Cold	20
Inverter loss	20%
Total battery size (MWh)	8
Rate (MW)	4
Upper boundary	90%
Lower boundary	10%
Interconnection approval (MW)	8.5
Battery size for ES (MWh)	6
Battery in use	Yes
Policy/CPLEX	Policy

User gets to tune the buy/sell parameters. Edit the parameters, and all costs are updated.

This is a form of "policy search" using "offline learning" (that is, learning in a computer simulator).

Editing the parameter changes the revenue from energy shifting:

ROI 15yr	59.10%
ROI 20yr	82.00%
Revenue from ES (Annual, Yr 1)	$80,772.68
Revenue from FR (Annual, Yr 1)	$1,012,252.39
Revenue from Solar (Annual, Yr 1)	$346,068.10
Revenue from SREC (Annual, Yr 1)	$543,422.98

Help

All bold values can be changed

Co-optimizing solar-storage

SMART-Solar: Co-optimization of solar and storage

Regular Charge @ $	56
Regular Discharge @ $	71
Cold Charge @ $	95
Cold Discharge @ $	115
Hot Charge @ $	100
Hot Discharge @ $	118
Grid Charge @ $	60
Grid Discharge @ $	120
Temp (F)	
Hot	75
Cold	20
Inverter loss	20%
Total battery size (MWh)	8
Rate (MW)	4
Upper boundary	90%
Lower boundary	10%
Interconnection approval (MW)	8.5
Battery size for ES (MWh)	6
Battery in use	Yes
Policy/CPLEX	Policy

We can have a single set of buy-sell parameters, but here we use three sets, for hot, cold and in-between.

Of course, this makes the problem harder. We might even want to make the parameters depend on other information, such as a weather forecast.

This starts to get *much* harder. But conceptually, still easy to understand.

5.58
(67,078.29)
29,776.26
45,079.40
78,315.69
(33,117.84)
85,043.93
74,578.03
22,756.86
-6.14%
29.96%
59.10%
82.00%
$80,772.68
012,252.39
346,068.10
543,422.98

elp

All bold values can be changed

Policy function approximations

- A number of fields work on this problem under different names:
  - Stochastic search
  - Stochastic programming (“two stage”)
  - Simulation optimization
  - Black box optimization
  - Global optimization
  - Control theory (“open loop control”)
  - Sequential design of experiments
  - Bandit problems (for on-line learning)
  - Ranking and selection (for off-line learning)
  - Optimal learning

Lecture outline

- The four classes of policies
  - Policy function approximations (PFAs)
  - Robust cost function approximations (CFAs)
  - Value function approximations (VFAs)
  - Lookahead policies

Robust cost function approximations

- Imagine that we want to pick the generators bidding the lowest cost c_g for generator $g \in G$, so that we produce enough power to cover the load L_t

$$X_t^{\pi}(S_t) = \arg\min_{x_t} \sum_{g \in G} c_g x_{tg}$$

subject to

$$\sum_{g \in G} x_{tg} \geq L_t$$

  - This would be called a myopic policy – minimize current costs, and ignore the impact on the future.
  - This can work well (it can even be optimal), but usually not.

Robust cost function approximations

- We might be worried that our actual load will be more than what we forecast. We can handle this by inserting a buffer

$$X_t^{\pi}(S_t \mid \theta) = \arg\min_{x_t \in \mathcal{X}_t} \sum_{g \in G} c_g x_{tg}$$

subject to

$$\sum_{g \in G} x_{tg} \geq L_t + \theta$$

» Now we have to figure out a good value of θ.

Robust cost function approximations

- We can tune our parameter θ just as we tuned our policy function approximation. We need to solve

$$\max_\theta F(\theta) = \mathbb{E}\sum_{t=0}^{T} c_t X_t^\pi(S_t \mid \theta)$$

As before, we need to replace the expectation with a simulation:

$$\max_\theta \bar{F}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T} c_t X_t^\pi(S_t(\omega^n) \mid \theta)$$

Now use the same methods we used when search for the best PFA.

Robust cost function approximations

- We can write a robust CFA using generic notation:

$$X_t^\pi(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t^\pi(\theta)} \bar{C}^\pi(S_t, x_t \mid \theta)$$

  - where $\bar{C}^\pi(S_t, x_t \mid \theta)$ represents any sort of approximation of the cost function. In addition, we can modify the constraints, which we can represent using $\bar{X}_t^\pi(S_t \mid \theta)$.

- One example of a CFA uses a correction term:

$$X_t^\pi(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t^\pi(\theta)} C(S_t, x_t) + \sum_{f \in F} \theta_f \phi_f(S_t)$$

  - This can work (but only for special cases).

Robust cost function approximations

- Notes:
  - Robust cost function approximations are generally used when a deterministic model is “pretty good” but produces solutions that are vulnerable to extreme outcomes in the future.
  - Robust CFAs are often confused as “deterministic models” because, well, they are deterministic models, just as *all* policies are deterministic functions. But they are modified deterministic models, and this makes all the difference.
  - The modifications (such as buffer reserves) have to be tuned in a stochastic base model, which is why they are robust policies.

Lecture outline

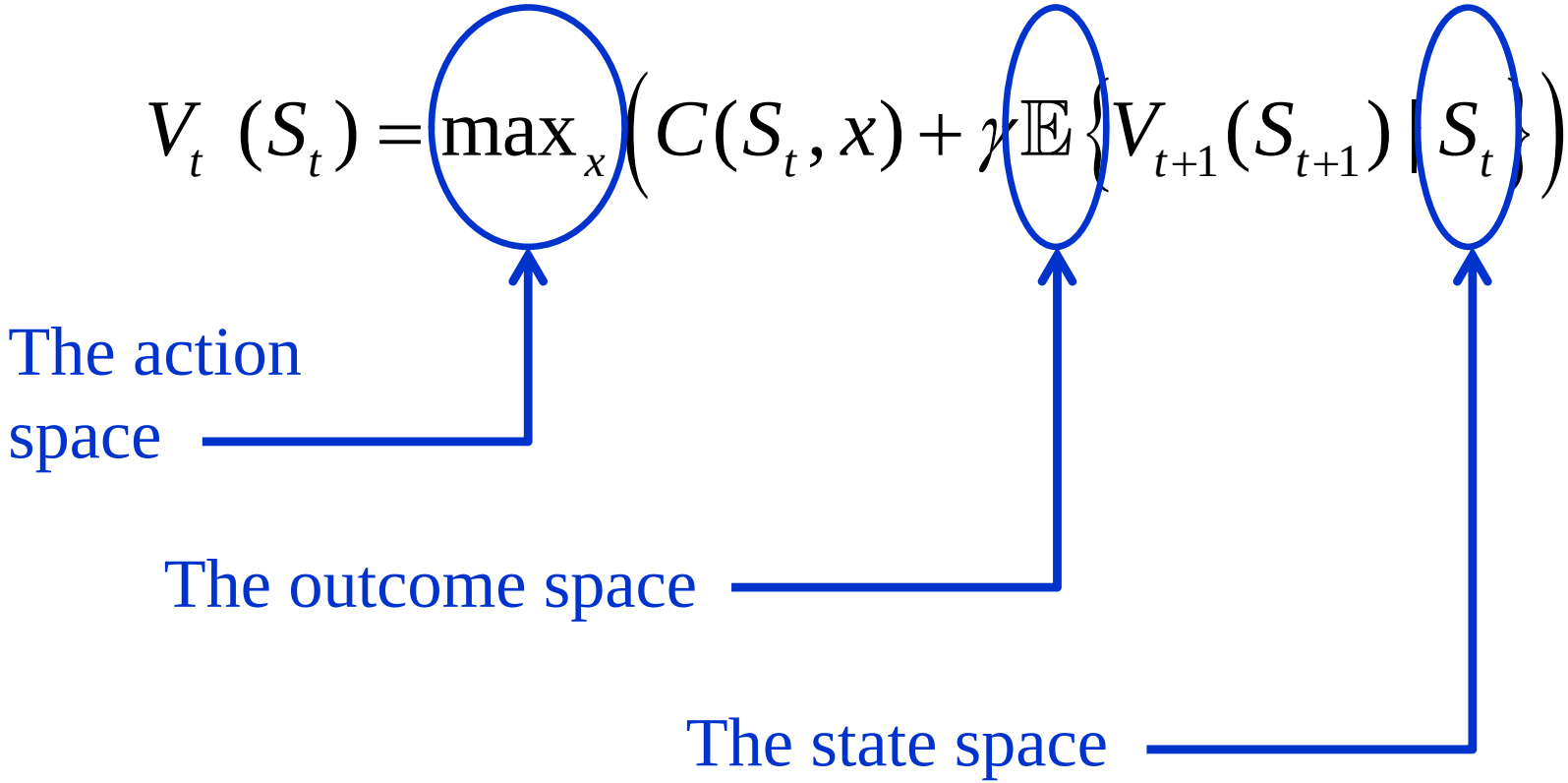
- The four classes of policies
  - Policy function approximations (PFAs)
  - Robust cost function approximations (CFAs)
  - Value function approximations (VFAs)
  - Lookahead policies

Dynamic programming

- History
  - In the 1950's, Richard Bellman introduced the "principle of optimality" (often called "Bellman's principle of optimality".
  - It comes from recognizing a simple relationship that applies to all sequential, stochastic optimization problems.
  - Even Bellman quickly realized that his method suffered from the "curse of dimensionality," but later this method would become the foundation for solving truly large scale problems using a field known as "approximate dynamic programming."

Dynamic programming

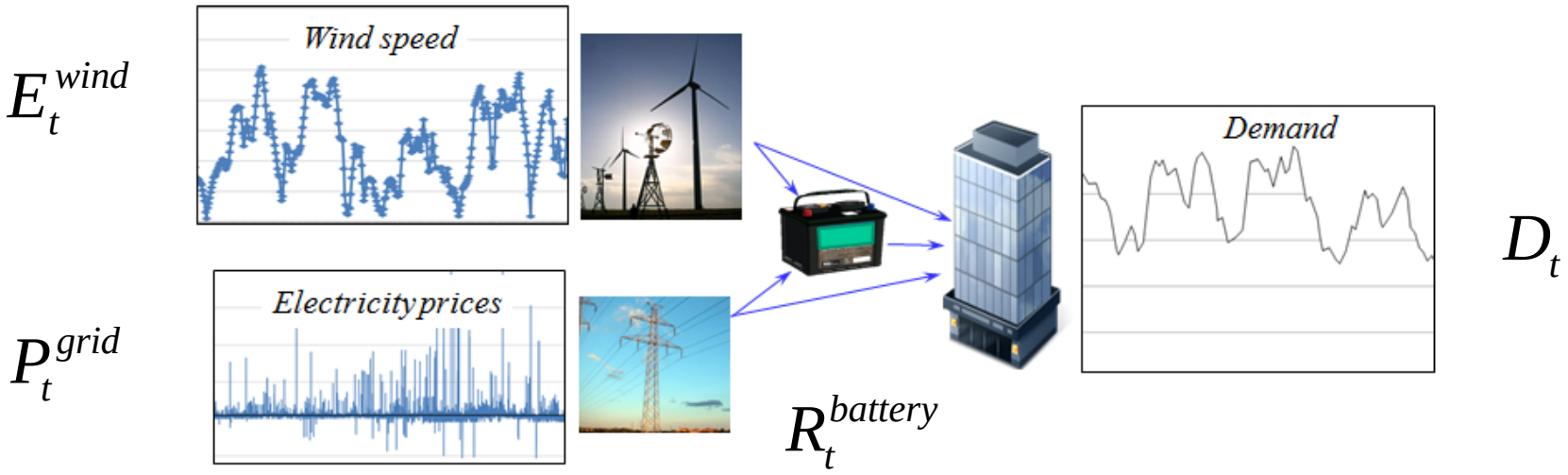
- Classical backward dynamic programming



» The three curses of dimensionality

Dynamic programming

- Energy storage with stochastic prices, supplies and demands.



$$E_{t+1}^{wind} = E_t^{wind} + \hat{E}_{t+1}^{wind}$$
$$P_{t+1}^{grid} = P_t^{grid} + \hat{P}_{t+1}^{grid}$$
$$D_{t+1}^{load} = D_t^{load} + \hat{D}_{t+1}^{load}$$
$$R_{t+1}^{battery} = R_t^{battery} + x_t$$

W_{t+1} = Exogenous inputs

S_t = State variable

x_t = Controllable inputs

Dynamic programming

- Bellman's optimality equation

$$V_t(S_t) = \min_{x_t \in \mathcal{X}} \left(C(S_t, x_t) + \gamma \mathbb{E}\left\{ V_{t+1}(S_{t+1}(S_t, x_t, W_{t+1})) \mid S_t \right\} \right)$$

$$\begin{bmatrix} E_t^{wind} \\ P_t^{grid} \\ D_t^{load} \\ R_t^{battery} \end{bmatrix}$$

$$\begin{bmatrix} x_t^{wind-battery} \\ x_t^{wind-load} \\ x_t^{grid-battery} \\ x_t^{grid-load} \\ x_t^{battery-load} \end{bmatrix}$$

$$\begin{bmatrix} \hat{E}_{t+1}^{wind} \\ \hat{P}_{t+1}^{grid} \\ \hat{D}_{t+1}^{load} \end{bmatrix}$$

» This illustrates the "three curses of dimensionality" – the state variable, the decision variable, and the exogenous information.

Dynamic programming

- Backward dynamic programming in one dimension

Step 0: Initialize $V_{T+1}(R_{T+1}) = 0$ for $R_{T+1} = 0,1,...,100$

Step 1: Step backward $t = T, T-1, T-2, ...$

Step 2: Loop over $R_t = 0,1,...,100$

Step 3: Loop over all decisions $0 \le x_t \le R_t$

Step 4: Take the expectation over all rainfall levels (also discretized):

$$\text{Compute } Q(R_t, x_t) = C(R_t, x_t) + \sum_{w=0}^{100} V_{t+1}(\min\left\{R^{\max}, R_t - x + w\right\}) P^W(w)$$

End step 4;

End Step 3;

Find $V_t^*(R_t) = \min_{x_t} Q(R_t, x_t)$

Store $X_t^{\pi^*}(R_t) = \arg\min_{x_t} Q(R_t, x_t)$. (This is our policy)

End Step 2;

End Step 1;

Dynamic programming

- Observations:
  - This is a beautiful procedure if you have a one (or two) dimensional state variable.
  - An undergraduate can program this in an afternoon using Matlab.
  - But, take a look at what happens when we have multiple dimensions.
  - Note that we can have multiple dimensions in:
    - The state space
    - The decision space
    - The outcome space
  - This is known as the “three curses of dimensionality”

Dynamic programming

- Dynamic programming in multiple dimensions

Step 0: Initialize $V_{T+1}(S_{T+1}) = 0$ for all states.

Step 1: Step backward $t = T, T-1, T-2, \ldots$

Step 2: Loop over $S_t = (R_t, D_t, p_t, E_t)$ (four loops)

Step 3: Loop over all decisions x_t (a problem if x_t is a vector)

Step 4: Take the expectation over each random dimension $(\hat{D}_t, \hat{p}_t, \hat{E}_t)$

Compute $Q(S_t, x_t) = C(S_t, x_t) +$

$$\sum_{w_1=0}^{100}\sum_{w_2=0}^{100}\sum_{w_3=0}^{100} V_{t+1}\left(S^M\left(S_t, x_t, W_{t+1} = (w_1, w_2, w_3)\right)\right) P^W(w_1, w_2, w_3)$$

End step 4;

End Step 3;

Find $V_t^*(S_t) = \min_{x_t} Q(S_t, x_t)$

Store $X_t^{\pi^*}(S_t) = \arg\min_{x_t} Q(S_t, x_t)$. (This is our policy)

End Step 2;

End Step 1;

Approximate dynamic programming

- The curse of dimensionality:
  - If our state variable (and action variable, and the random variable) are all one-dimensional, this will run very quickly.
  - Two dimensions is OK, but run times of seconds to a minute becomes several minutes.
  - Three dimensions might take a hour to several hours.
  - Four dimensions can take weeks.

Approximate dynamic programming

- Various strategies then evolved under the umbrella of “approximate dynamic programming” or “reinforcement learning”:
  - » Approximate value iteration
    - Mimics backward dynamic programming
  - » Approximate policy iteration
    - Mimics policy iteration
  - » Policy search
    - Based on the field of stochastic search

Approximate dynamic programming

Step 1: Start with a pre-decision state S_t^n

Step 2: Solve the deterministic optimization using an approximate value function:

$$\hat{v}_t^n = \min_x \left(C(S_t^n, x_t) + \bar{V}_t^{n-1}(S^{M,x}(S_t^n, x_t)) \right)$$

to obtain x_t^n.

Deterministic optimization

Step 3: Update the value function approximation

$$\bar{V}_{t-1}^n(S_{t-1}^{x,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n$$

Recursive statistics

Step 4: Obtain Monte Carlo sample of $W_t(\omega^n)$ and compute the next pre-decision state:

$$S_{t+1}^n = S^M(S_t^n, x_t^n, W_{t+1}(\omega^n))$$

Simulation

Step 5: Return to step 1.

Approximate value iteration

Step 1: Start with a pre-decision state S_t^n

Step 2: Solve the deterministic optimization using an approximate value function:

$$\hat{v}_t^n = \min_x (C(S_t^n, x_t) + \sum_f \theta_f^{n-1} \phi_f(S^M(S^m, x)))$$

to obtain x_t^n.

Deterministic optimization

An approximate value function

Step 3: Update the value function approximation

$$\bar{V}_{t-1}^n(S_{t-1}^{x,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n$$

Recursive statistics

Step 4: Obtain Monte Carlo sample of $W_t(\omega^n)$ and compute the next pre-decision state:

$$S_{t+1}^n = S^M(S_t^n, x_t^n, W_{t+1}(\omega^n))$$

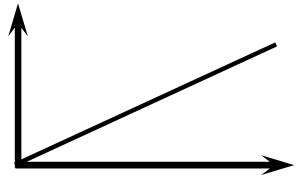
Simulation

Step 5: Return to step 1.

Approximating value functions

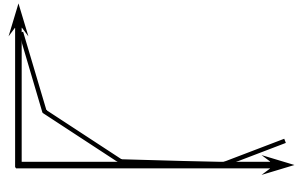
- Approximations for resource allocation problems
  - » Linear (in the resource state):

$$\bar{V}_t(R_t^x) = \sum_{a \in \mathcal{A}} \bar{v}_{ta} \cdot R_{ta}^x$$



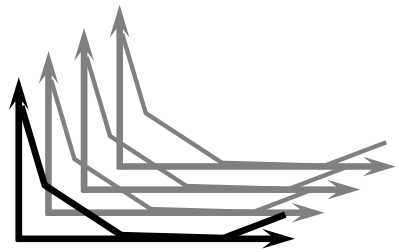
  - » Piecewise linear, separable:

$$\bar{V}_t(R_t^x) = \sum_{a \in \mathcal{A}} \bar{V}_{ta}(R_{ta}^x)$$



  - » Indexed PWL separable:

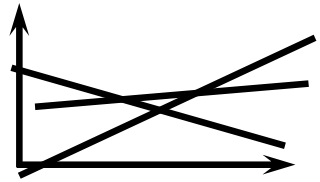
$$\bar{V}_t(R_t^x) = \sum_{a \in \mathcal{A}} \bar{V}_{ta}\left(R_{ta}^x \mid \text{"state of the world"}\right)$$



  - » Benders cuts

$$\min cx + z$$

$$z \geq a_i + b_i x$$

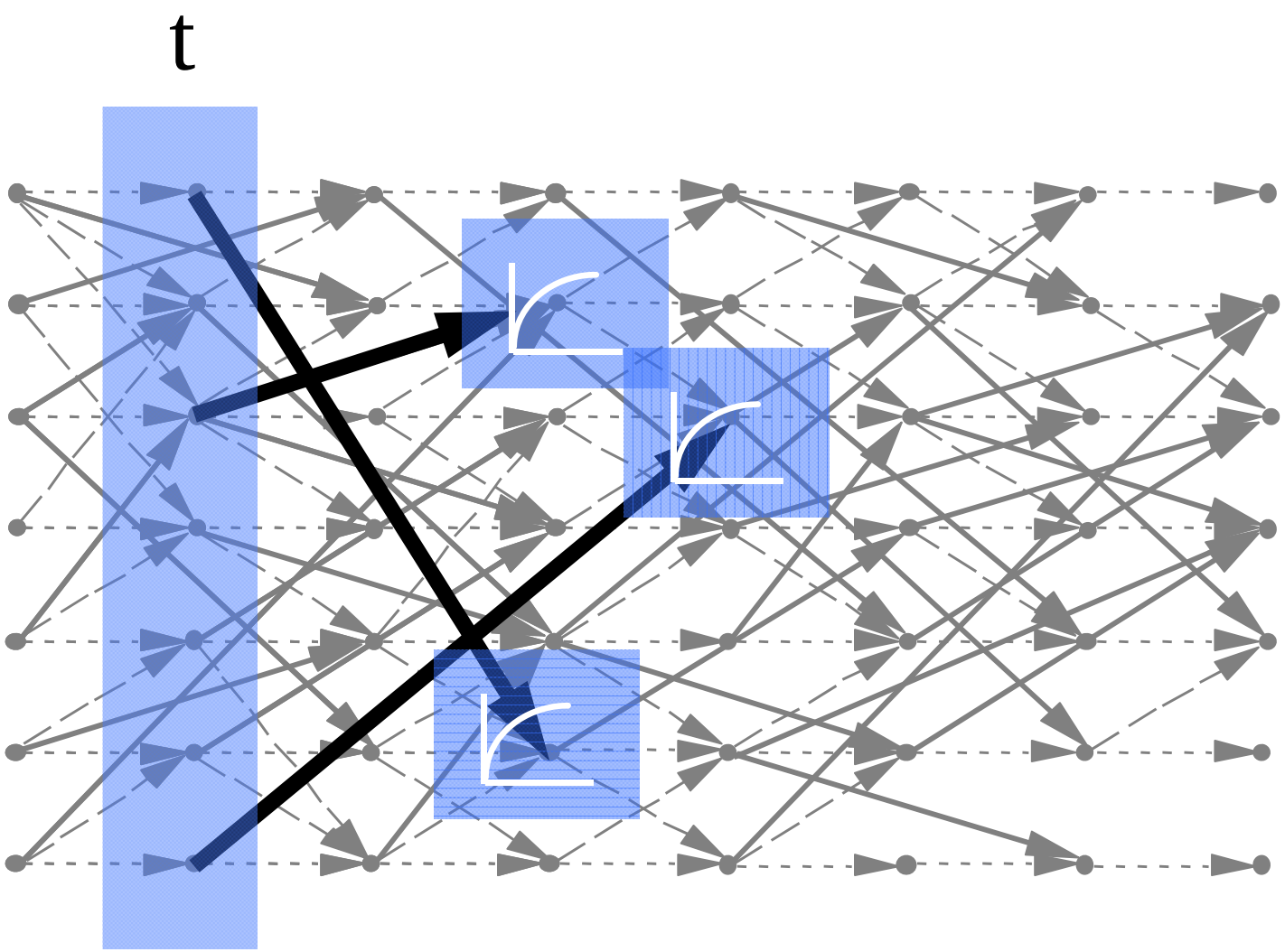


Approximate dynamic programming

- We can fit these value functions as follows:
  - Start with some initial approximation (perhaps zero)
  - Simulate the policy (that is, step through time making decisions using our current value function approximation)
  - Update the VFAs by collecting appropriate information.
  - Do this iteratively. If we can exploit problem structure, we might get a pretty good solution (but, there are pitfalls – this takes experience).

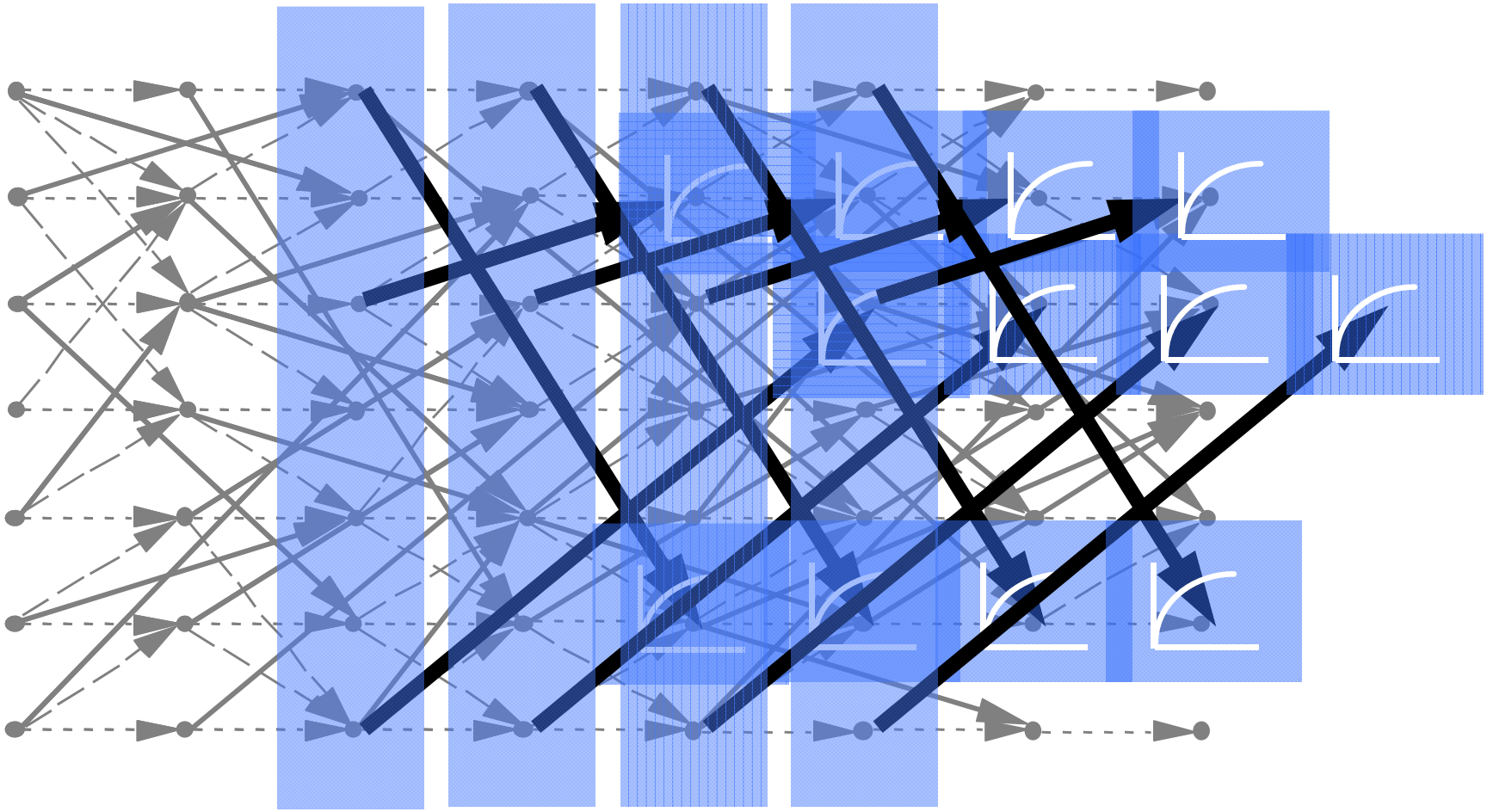
Approximate dynamic programming

- We learn the value functions as we simulate the policy



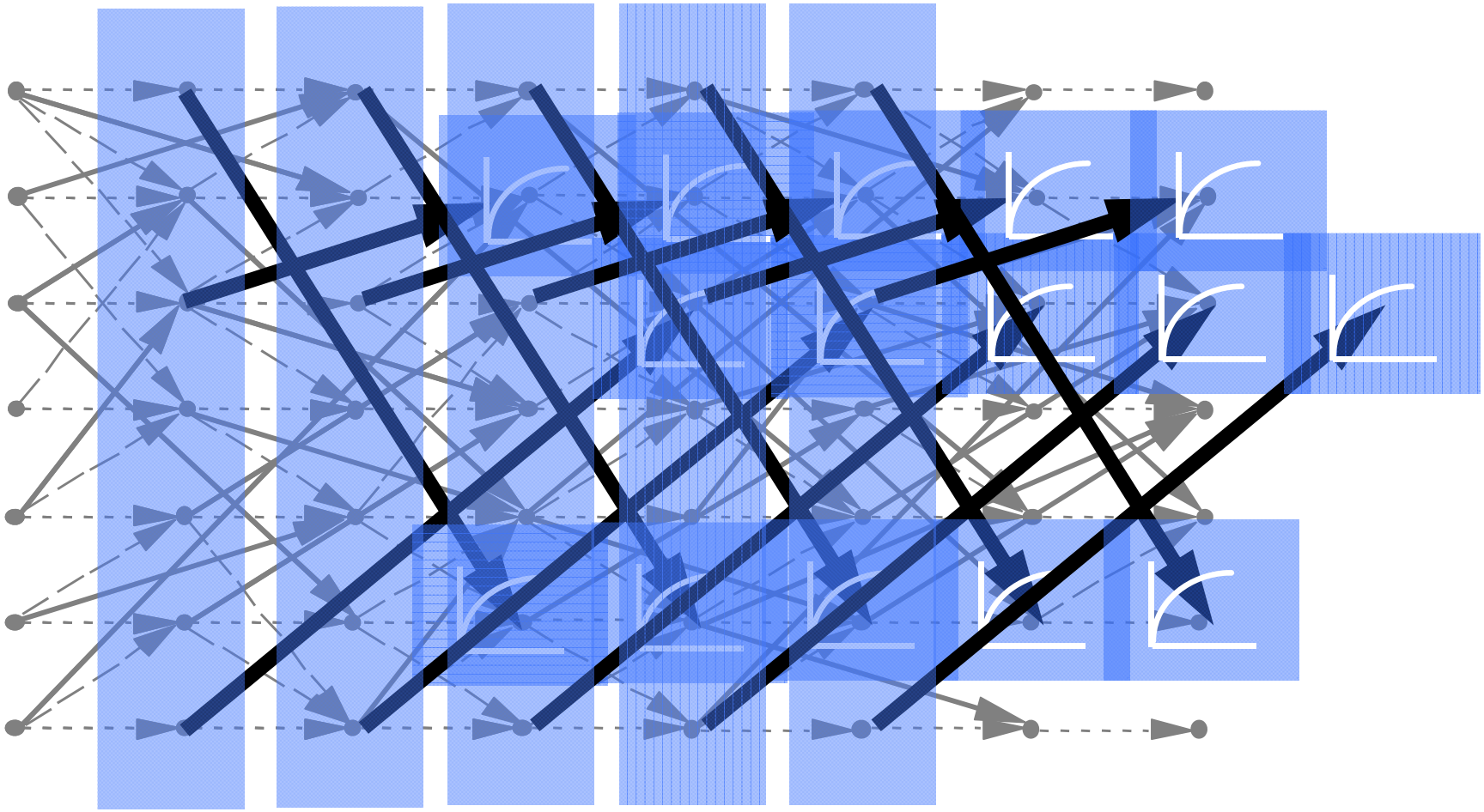
Approximate dynamic programming

- We learn the value functions as we simulate the policy



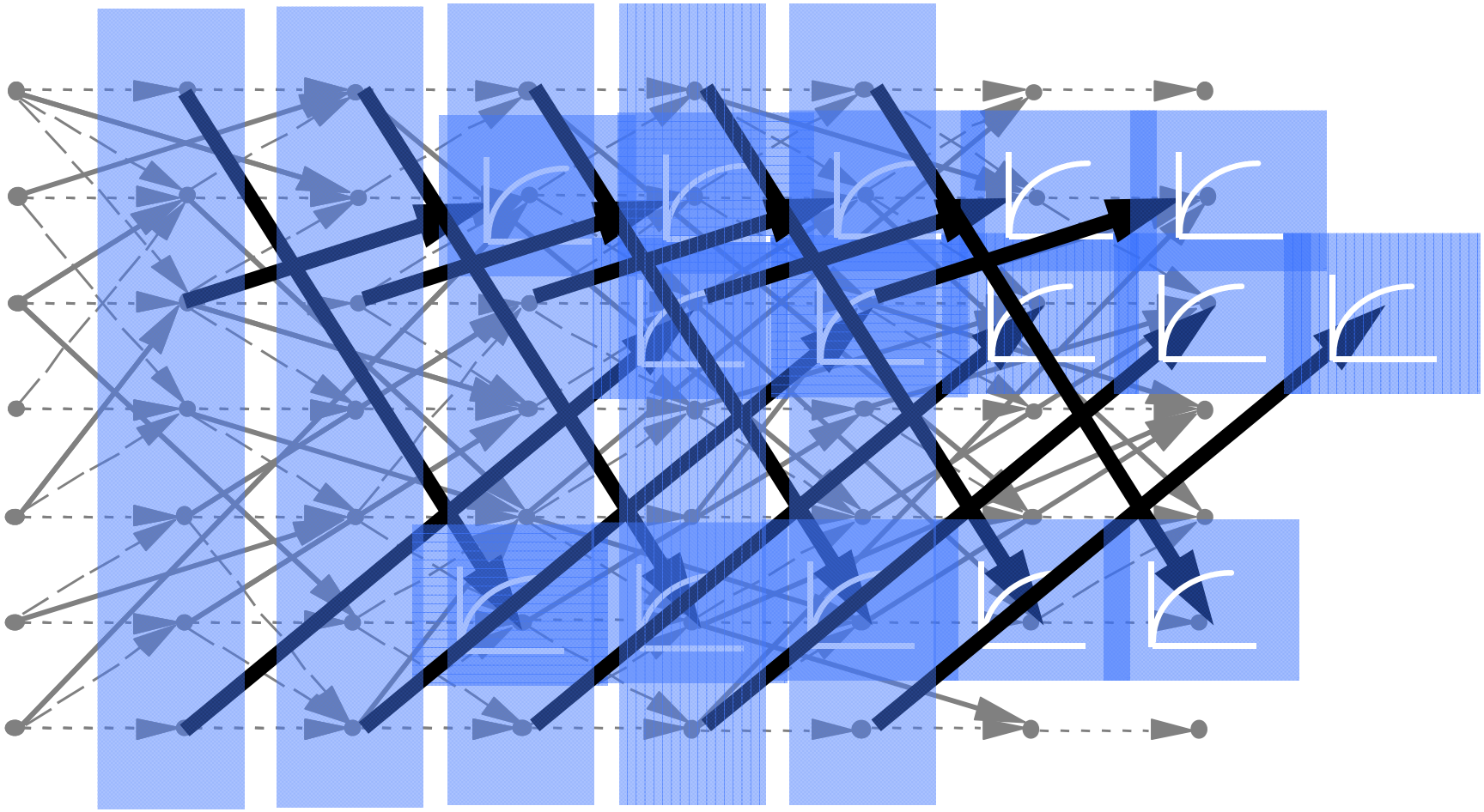
Approximate dynamic programming

- We learn the value functions as we simulate the policy



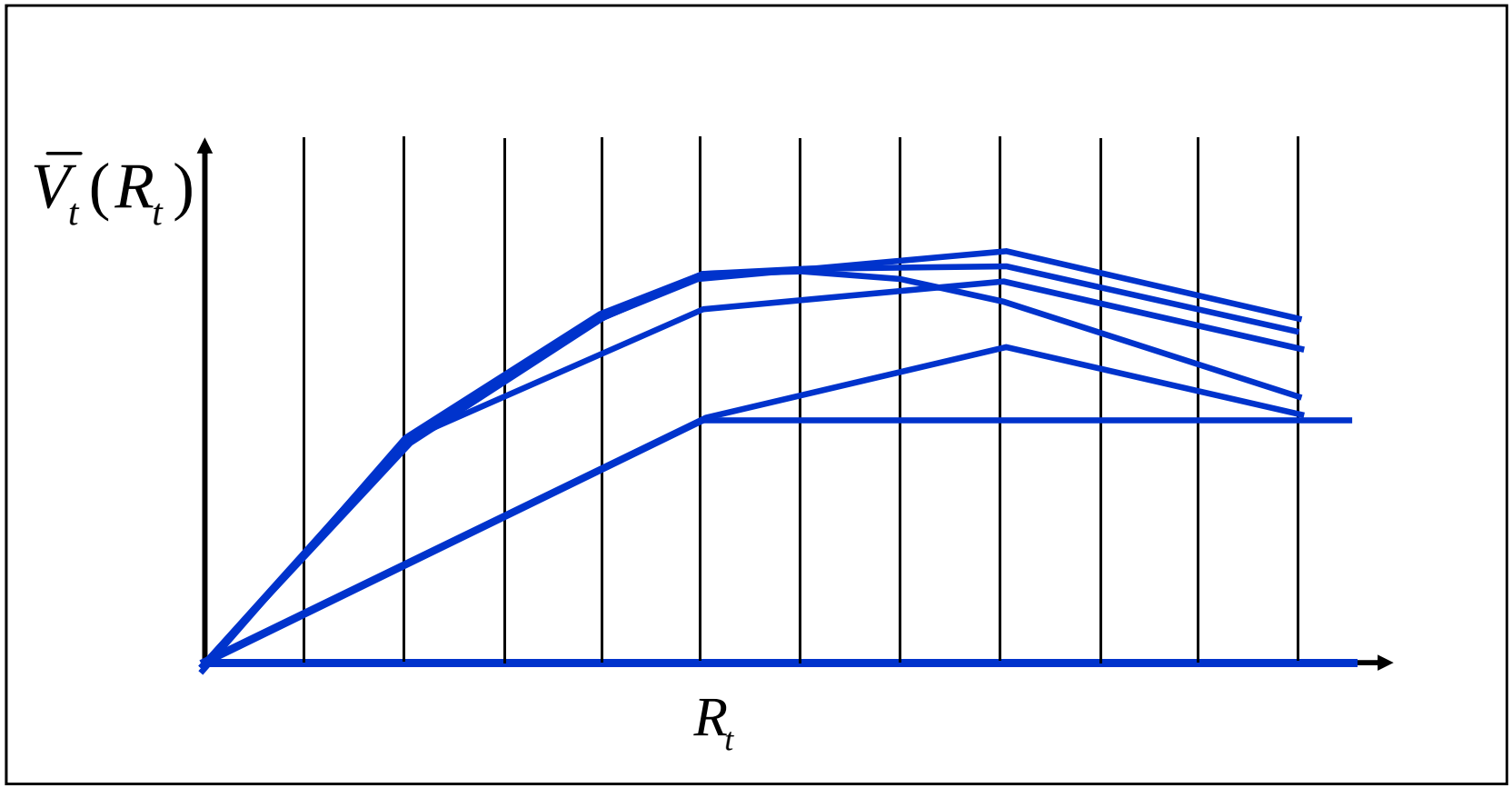
Approximate dynamic programming

- We learn the value functions as we simulate the policy



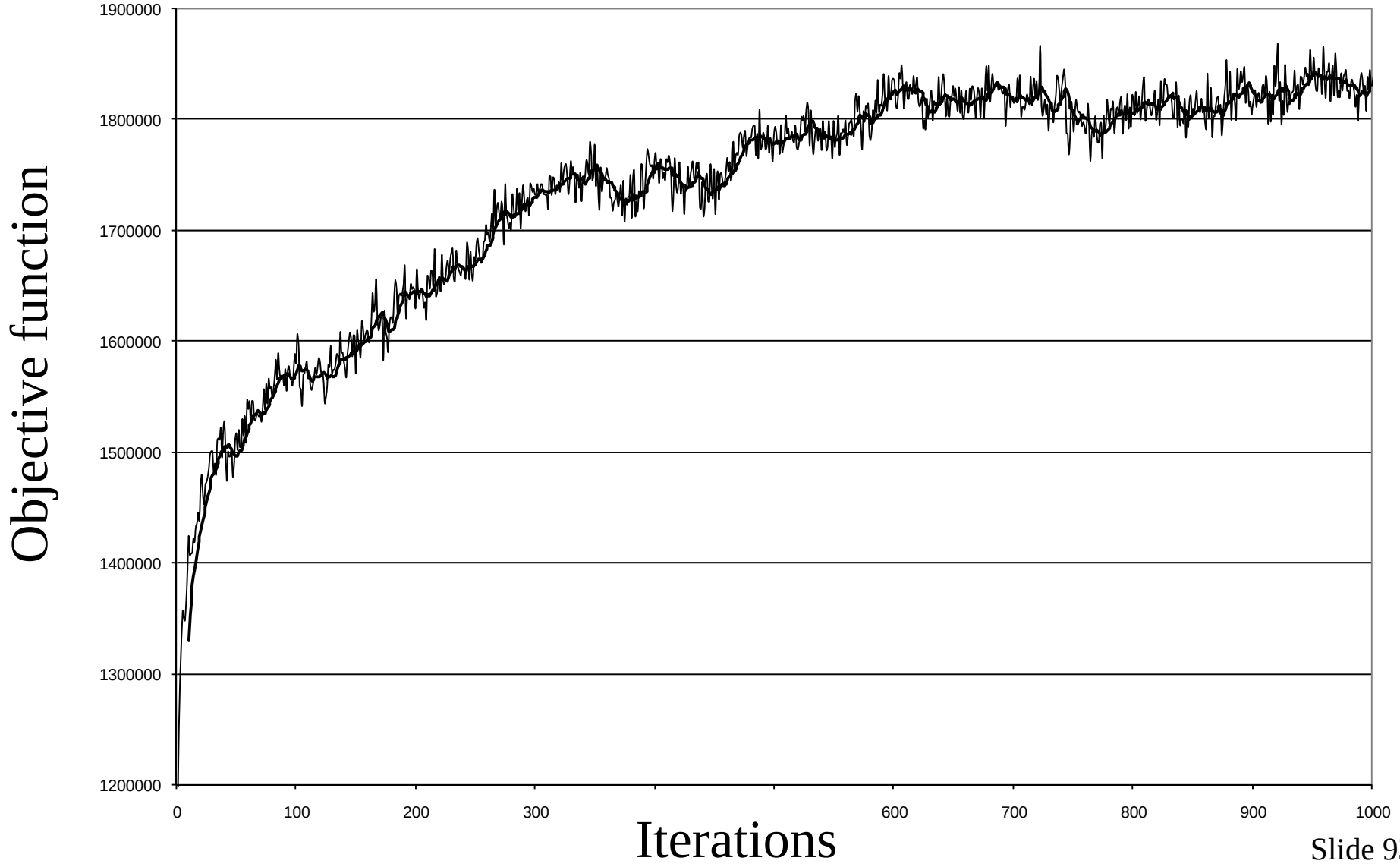
Exploiting concavity

- Derivatives are used to estimate a piecewise linear approximation



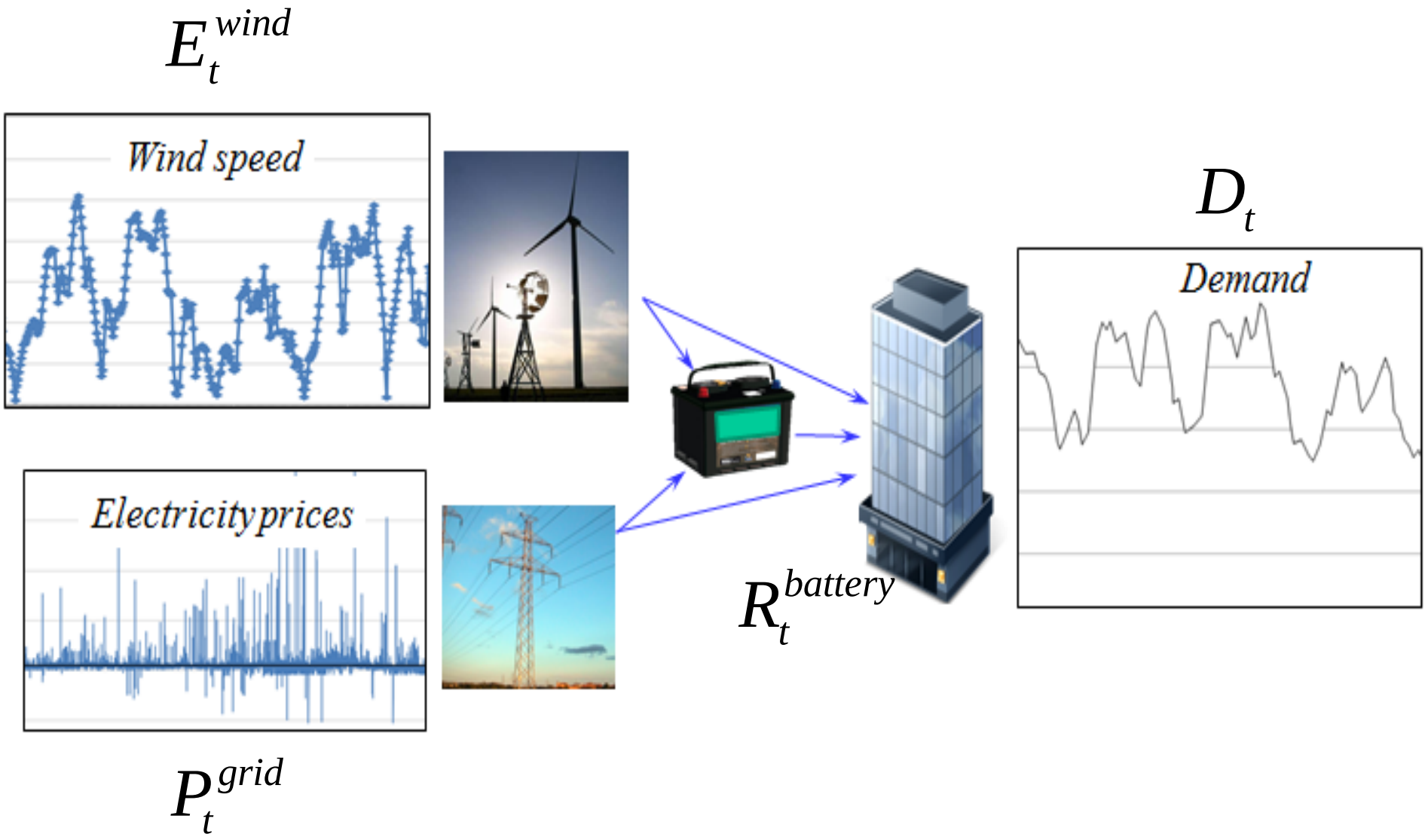
Approximate dynamic programming

- With luck, your objective function improves



Dynamic programming

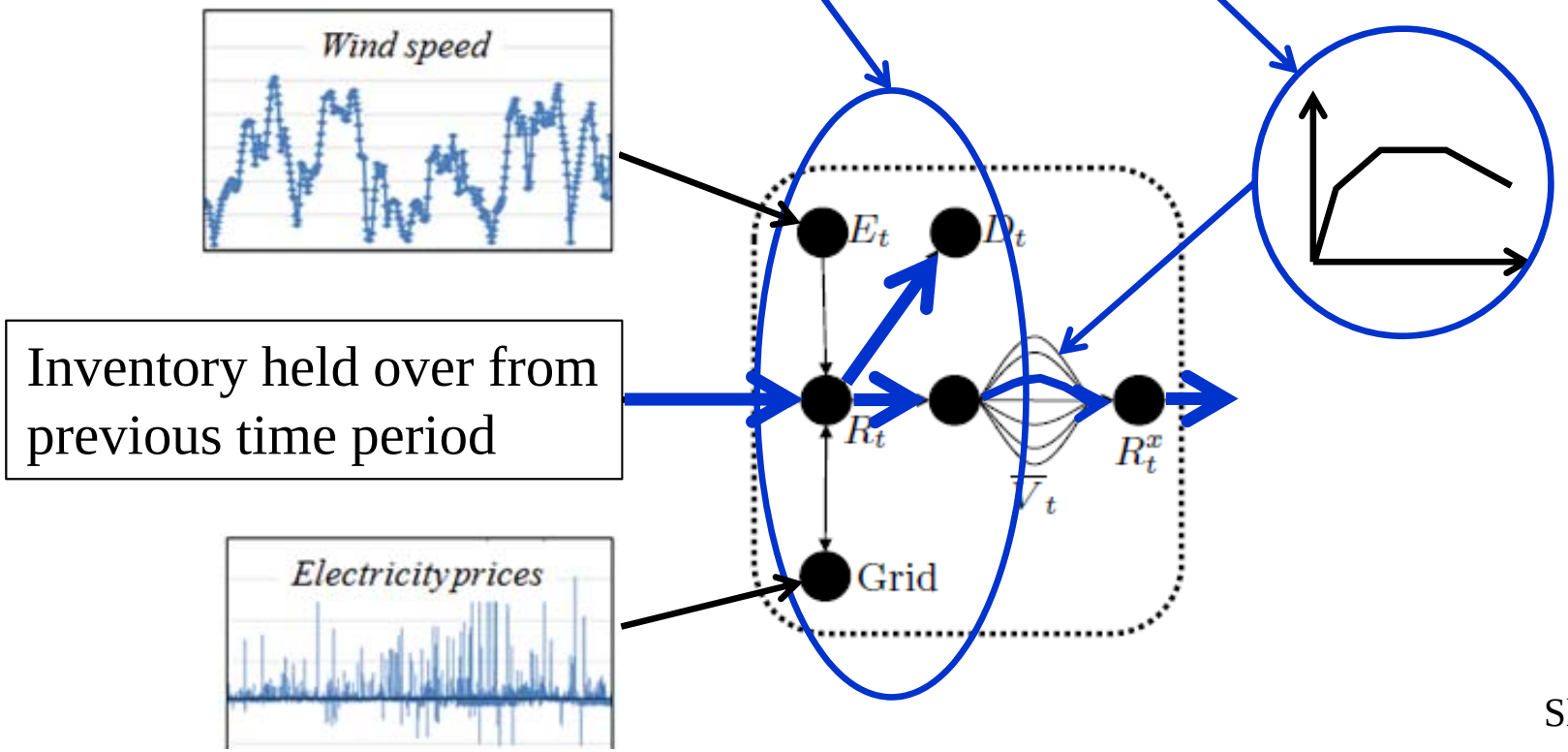
- Optimizing energy flows with a single battery



Approximate dynamic programming

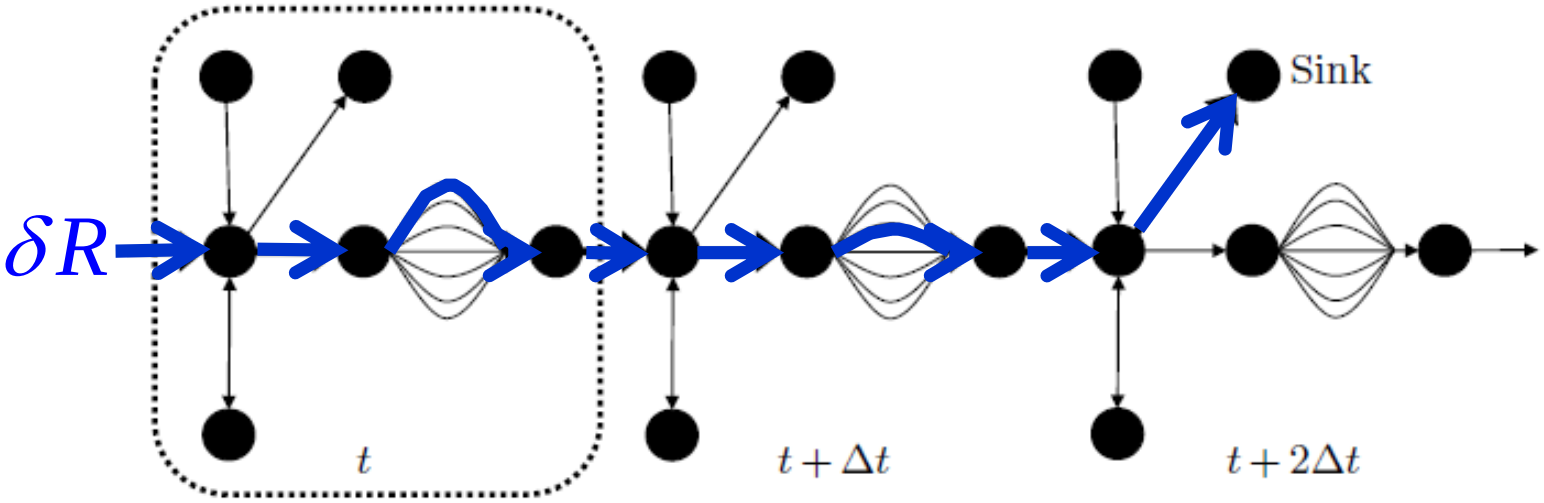
- Bellman's optimality equation
  - » We optimize energy flows using a piecewise linear, concave approximation of the value of energy in the battery:

$$V_t(S_t) = \min_{x_t \in \mathcal{X}} \left(C(S_t, x_t) + \gamma \bar{V}_t^x \left(S_t^x (S_t, x_t) \right) \right)$$



Approximate dynamic programming

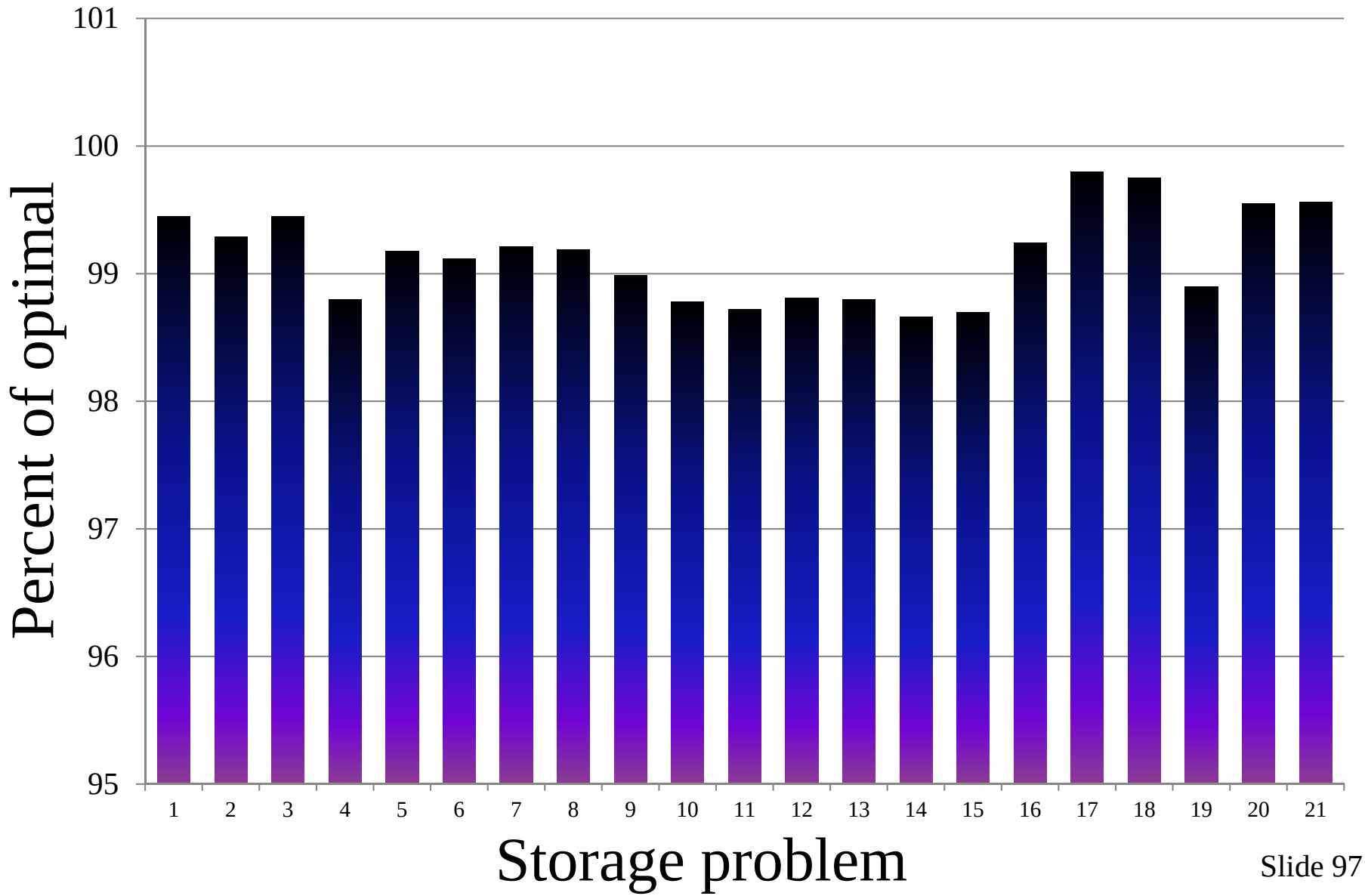
- We update the piecewise linear value functions by finding the marginal value of additional energy in our battery:



  - » The cost along the marginal path is the derivative of the simulation with respect to the flow perturbation.

Approximate dynamic programming

- The algorithm is near-optimal across 21 benchmark problems:



Approximate dynamic programming

- It is possible to apply this idea to grid scale storage
  - » We just have to estimate the marginal value of energy in *each* battery on the grid (avoids curse of dimensionality when there is more than one battery)
  - » The methods can account for the ability to ramp generators within ramping constraints.
  - » It also can handle congestion on the grid.
  - » We spend a lot of time evaluating algorithms. One strategy we have used is to use ADP (which can handle uncertainty) and compare it to the optimal solution we get on a deterministic problem (found using linear programming)

- Imagine 25 large storage devices spread around the PJM grid:





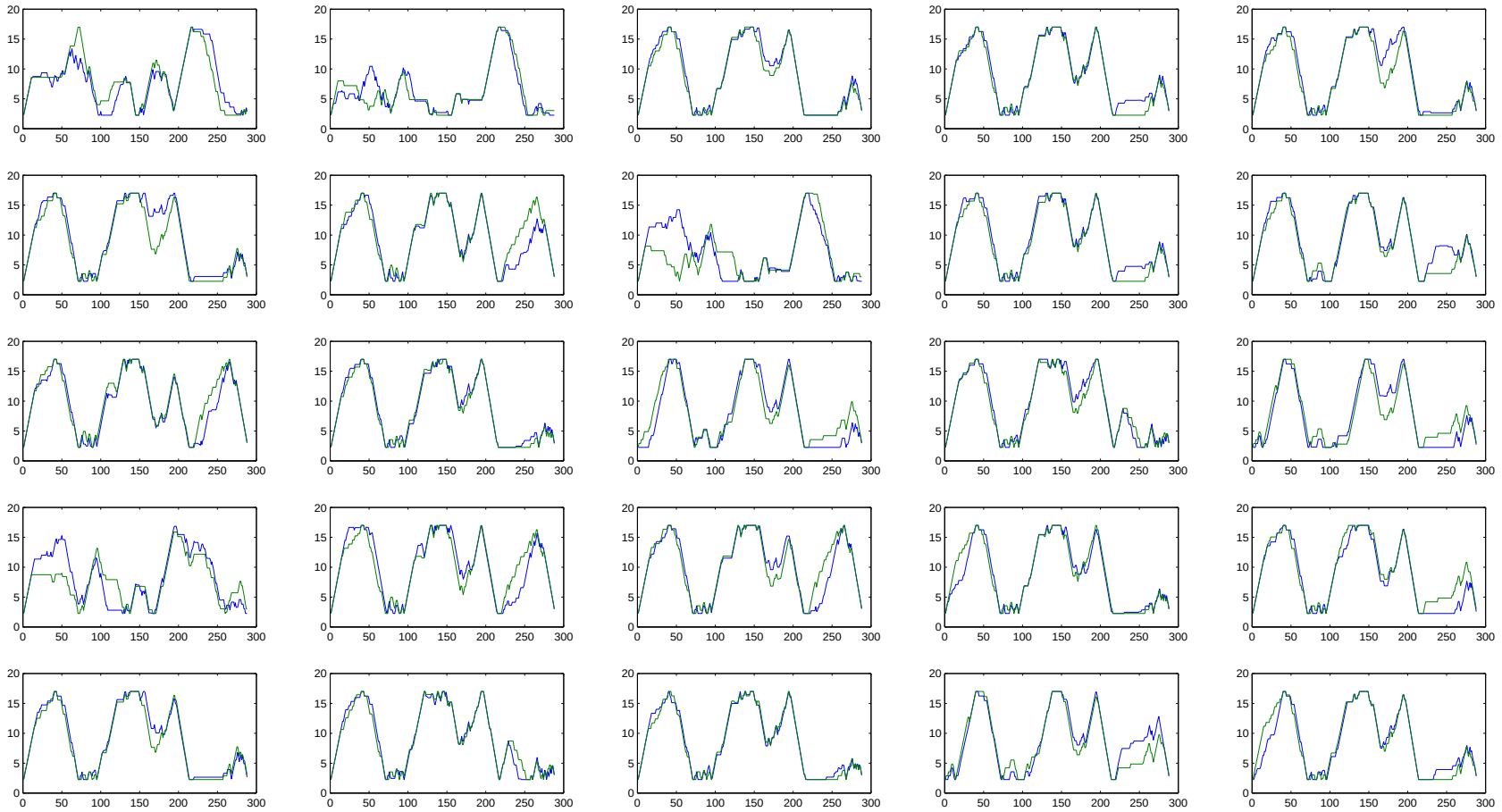







Grid level storage

- ADP (blue) vs. LP optimal (black)
  - » Note the very close match.



Approximate dynamic programming

- Observations
  - Dynamic programming (or approximate dynamic programming) is a policy that uses a value function to approximate the impact of a decision now on the future.
  - It works well on problems where the value functions are easy to estimate.
  - A popular class of applications involves “storage” as in the control of batteries, or the management of hydro reservoirs.
  - It would not be useful for the stochastic unit commitment problem…
  - … and it struggles with forecasts.

Lecture outline

- The four classes of policies
  - Policy function approximations (PFAs)
  - Robust cost function approximations (CFAs)
  - Value function approximations (VFAs)
  - Lookahead policies

Lookahead policies

- Up to now, we have considered policies that require coming up with some sort of function approximation:
  - PFA – approximating the function relating state to action
  - CFA – We need to approximate our cost function in some way
  - VFA – We have to approximate the value of being in a state.
- The challenge:
  - Approximating functions is not too hard when you can exploit structure (e.g. you know the shape).
  - When all else fails, you need to use a lookahead policy.

Lookahead policies

- The optimal policy requires solving

$$X^*(S_t) = \arg\min_{x_t} C(S_t, x_t) + \mathbb{E}\left\{\min_{\pi\in\Pi} \sum_{t'=t+1}^{T} C(S_{t'}, X^{\pi}_{t'}(S_{t'})) \mid S_t\right\}$$

Expectation that we cannot compute

Minimization that we cannot compute

  - » Calculating the downstream cost is generally impossible, so we have to use some sort of approximation.

Lookahead policies

- We use a series of approximations:
  - Stage aggregation – Replacing multistage problems with two-stage approximations.
  - Outcome aggregation/sampling – Simplifying the exogenous information process (“scenarios”)
  - Discretization – Of time, states and decisions (PJM steps forward in 5 minute time steps, but uses hourly time steps in their lookahead model)
  - Horizon truncation – Replacing a longer horizon problem with a shorter horizon (PJM might optimize out 2 days instead of 7)
  - Dimensionality reduction – We may ignore some variables (such as forecasts) in the lookahead model that we capture in the base model (these become *latent* (hidden) variables in the lookahead model).

Lookahead policies

- Notes:
  - In a complex problem such as stochastic unit commitment, we have to use all five classes of approximations.
  - The stochastic programming community (including people working on stochastic unit commitment), tend to focus on the generation of scenarios.
  - The two-stage approximation is rarely debated, but it is actually a significant approximation.
  - Horizon truncation is also an issue. Longer horizons can fix this, but run times can increase faster than linearly (and possibly much faster).

Four classes of policies

1) Policy function approximations (PFAs)
   - » Lookup tables, rules, parametric functions
2) Cost function approximation (CFAs)
   - » $X^{CFA}(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$
3) Policies based on value function approximations (VFAs)
   - » $X_t^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) Lookahead policies
   - » ***Deterministic lookahead:***

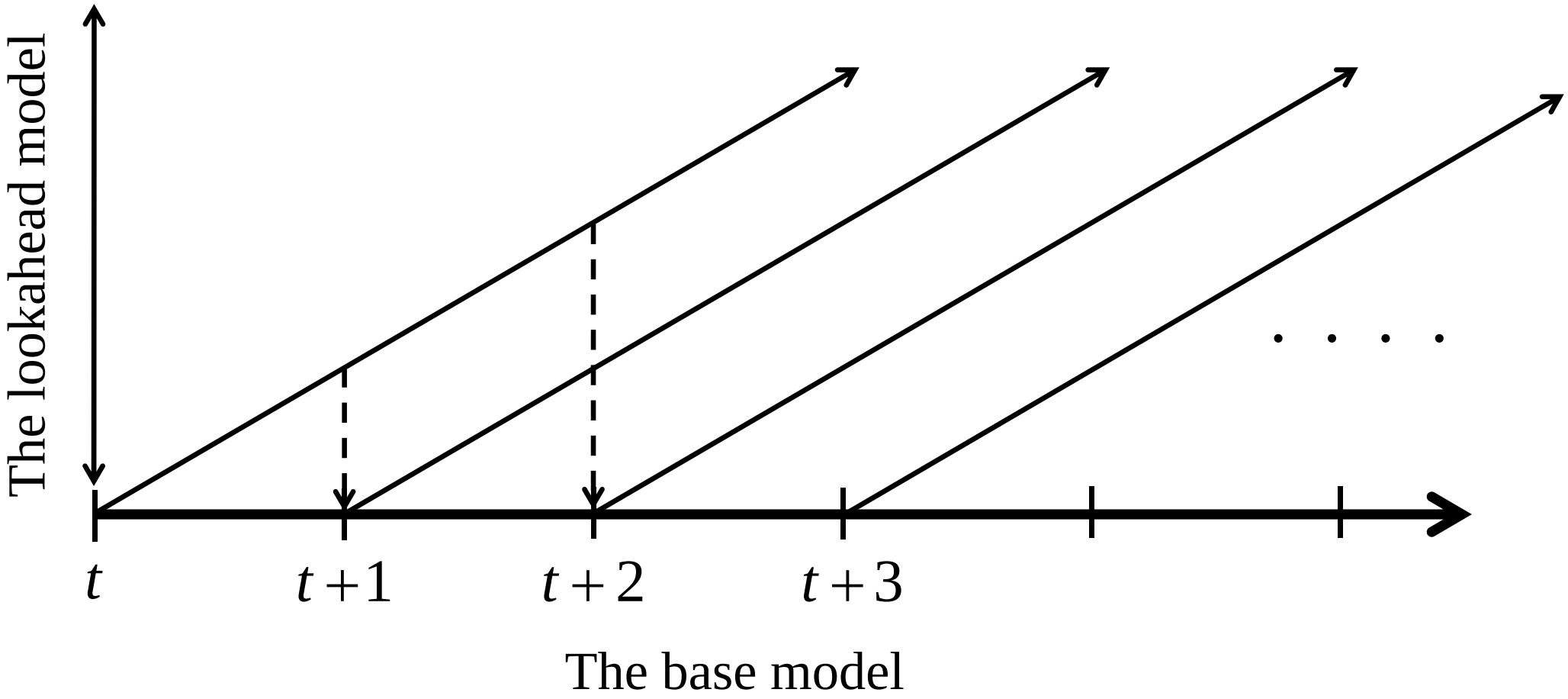
$$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » ***Stochastic lookahead (e.g. stochastic trees)***

$$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

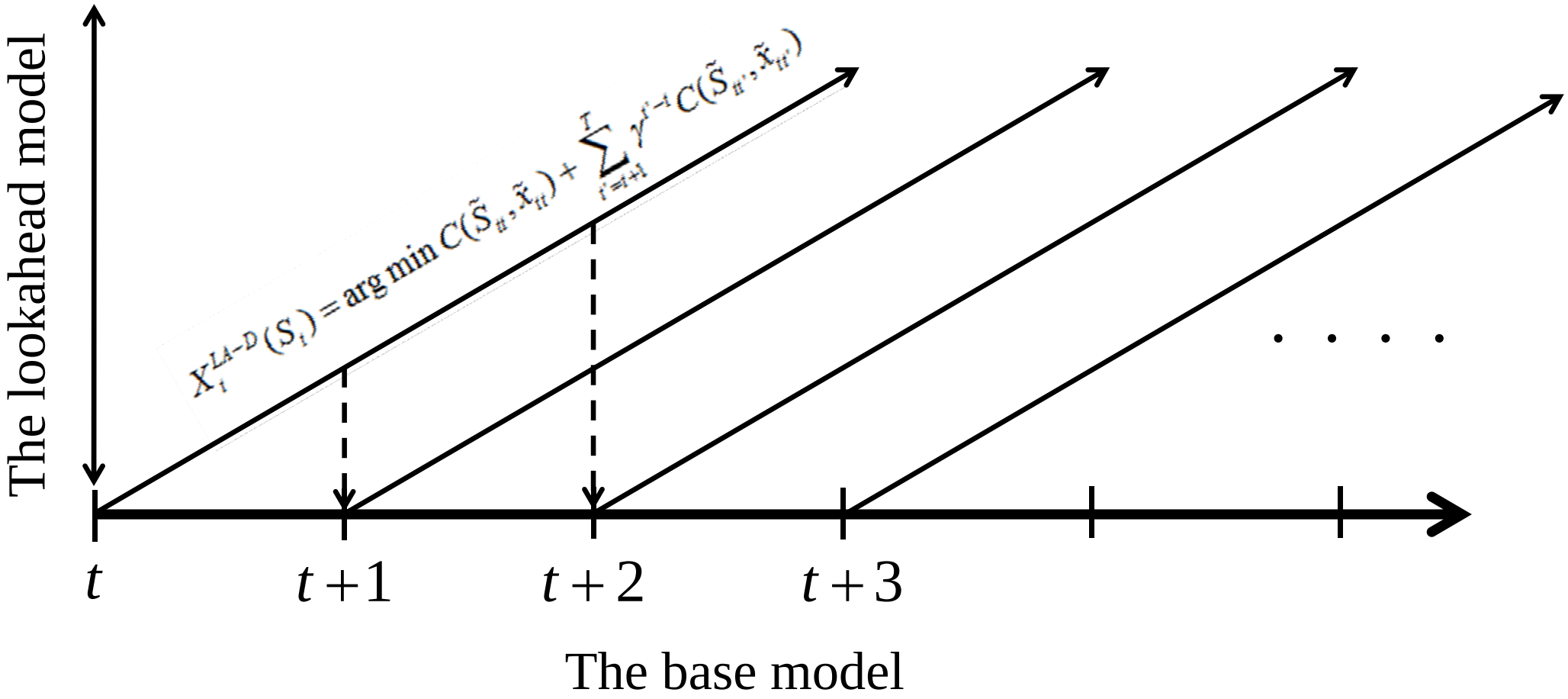
Lookahead policies

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



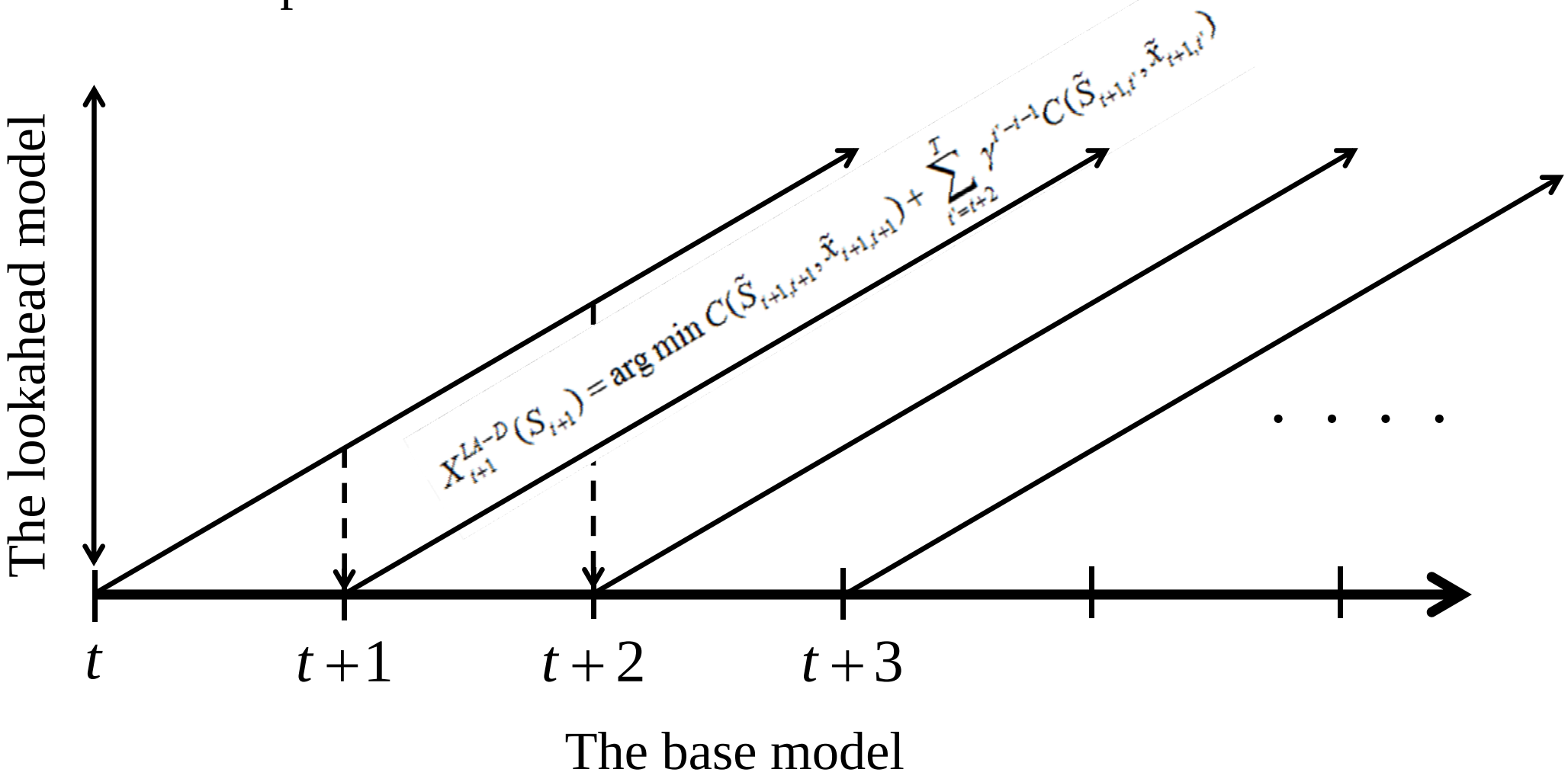
Lookahead policies

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



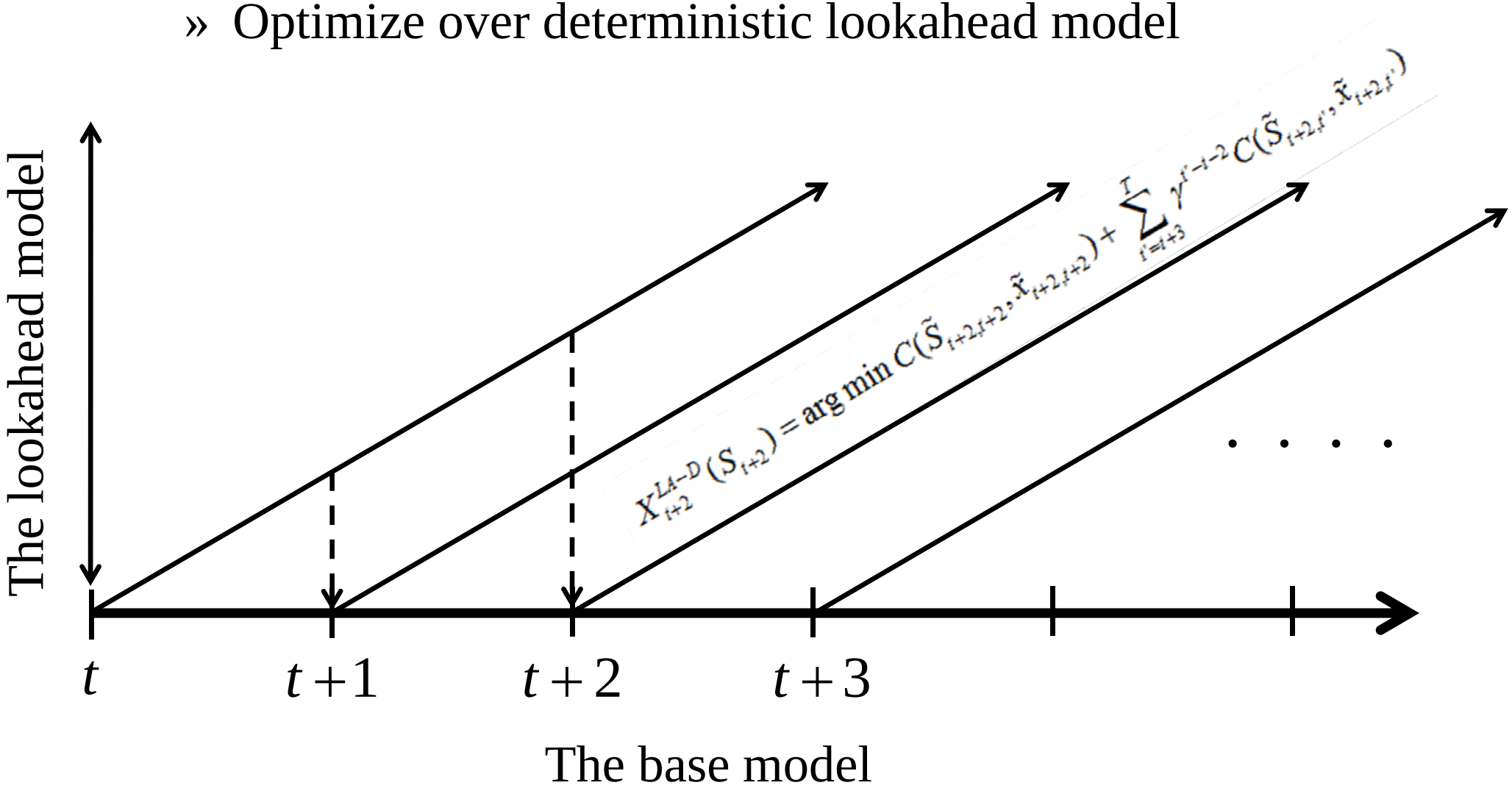
Lookahead policies

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



Lookahead policies

- Lookahead policies peek into the future
  - » Optimize over deterministic lookahead model



Lookahead policies

- Notes:
  - Deterministic lookahead policies are perhaps the most widely used heuristic.
  - A major feature is that they easily handle forecasts (something we largely ignored with PFAs and VFAs).
  - In practice, we suspect it is quite rare that anyone uses a pure deterministic lookahead, but more on this later.
  - Just because the future is stochastic (uncertain) does not mean you have to use a stochastic lookahead model (deterministic approximations can be quite good, and are sometimes optimal).

Four classes of policies

1) Policy function approximations (PFAs)
   - » Lookup tables, rules, parametric functions
2) Cost function approximation (CFAs)
   - » $X^{CFA}(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$
3) Policies based on value function approximations (VFAs)
   - » $X_t^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) Lookahead policies
   - » ***Deterministic lookahead:***

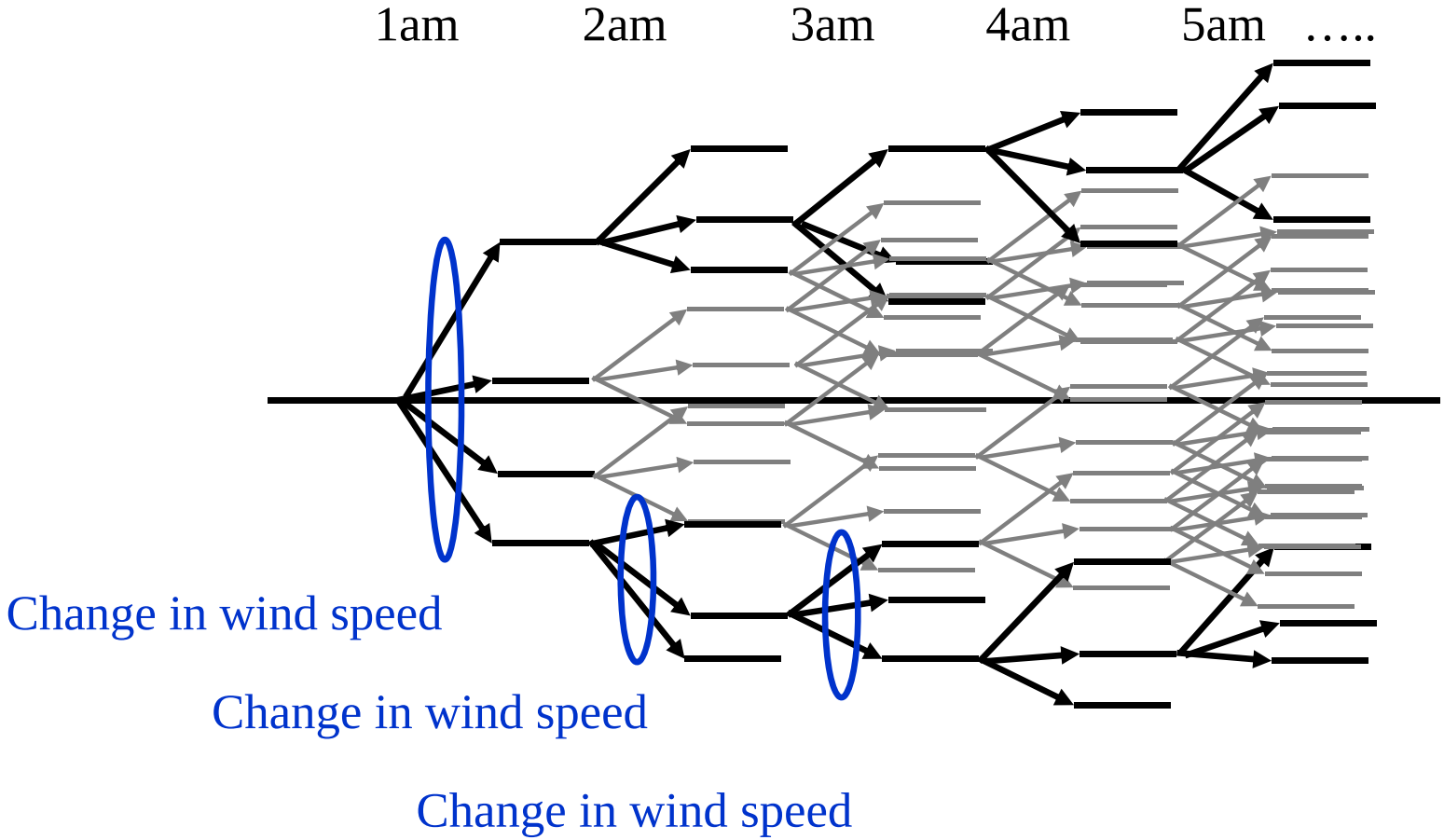
$$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » ***Stochastic lookahead (e.g. stochastic trees)***

$$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

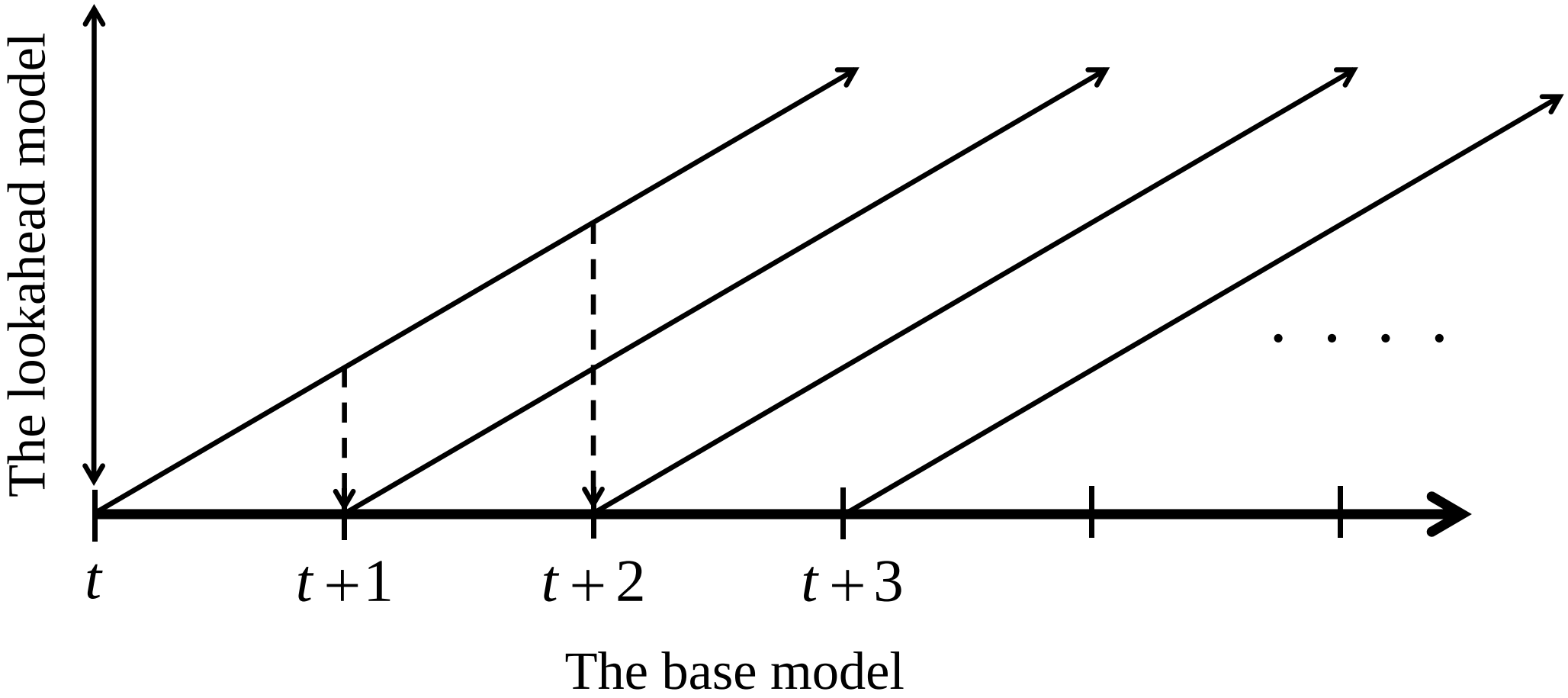
Stochastic lookahead policies

- Stochastic lookahead
  - » Here, we approximate the information model by using a Monte Carlo sample to create a scenario tree:



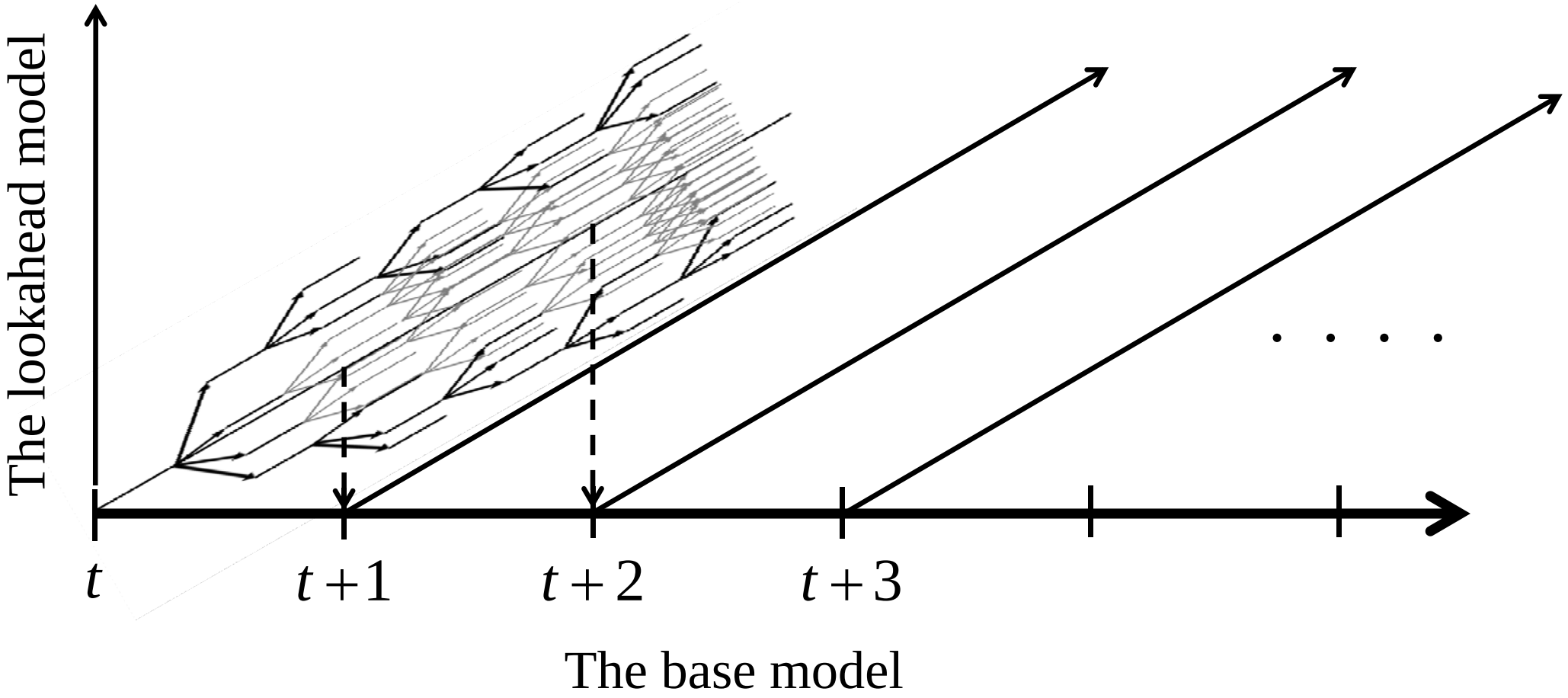
Stochastic lookahead policies

- We can then simulate this *lookahead policy* over time:



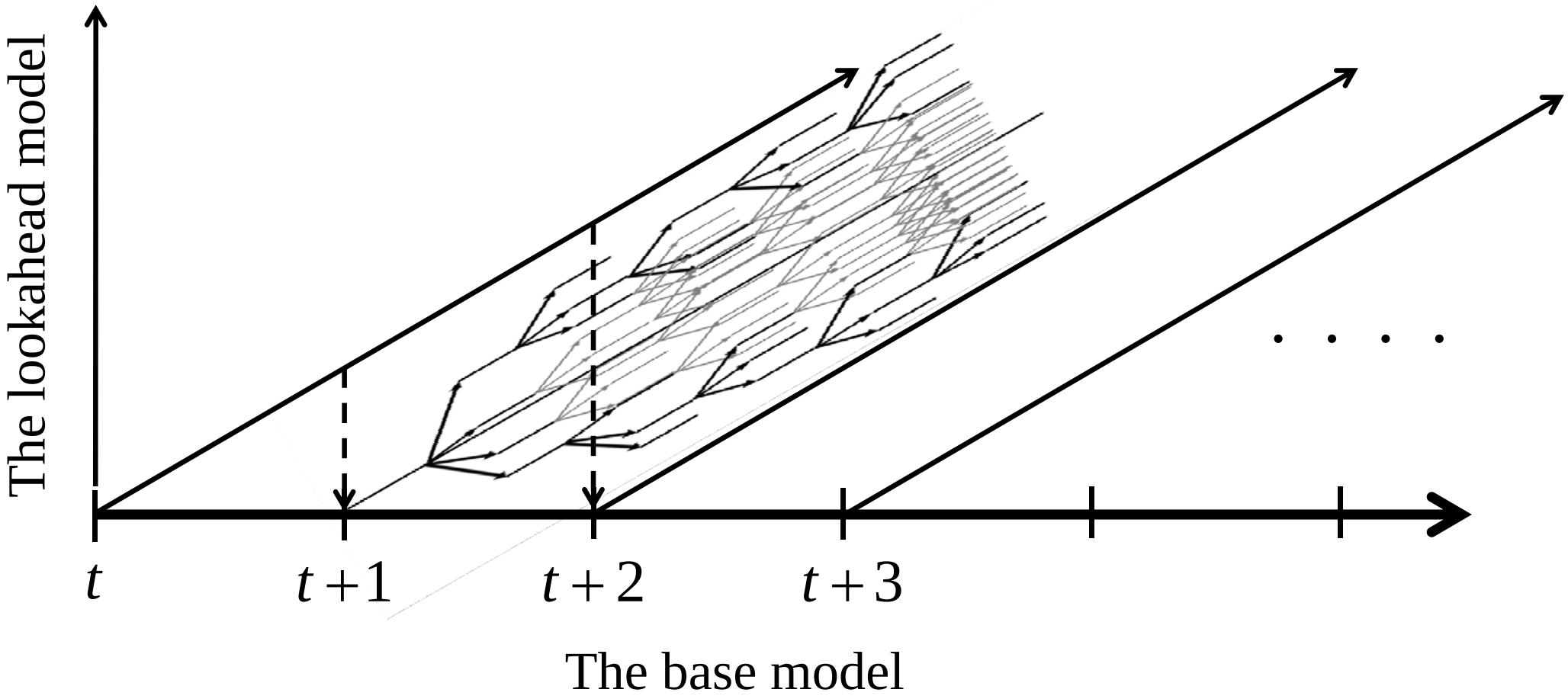
Stochastic lookahead policies

- We can then simulate this *lookahead policy* over time:



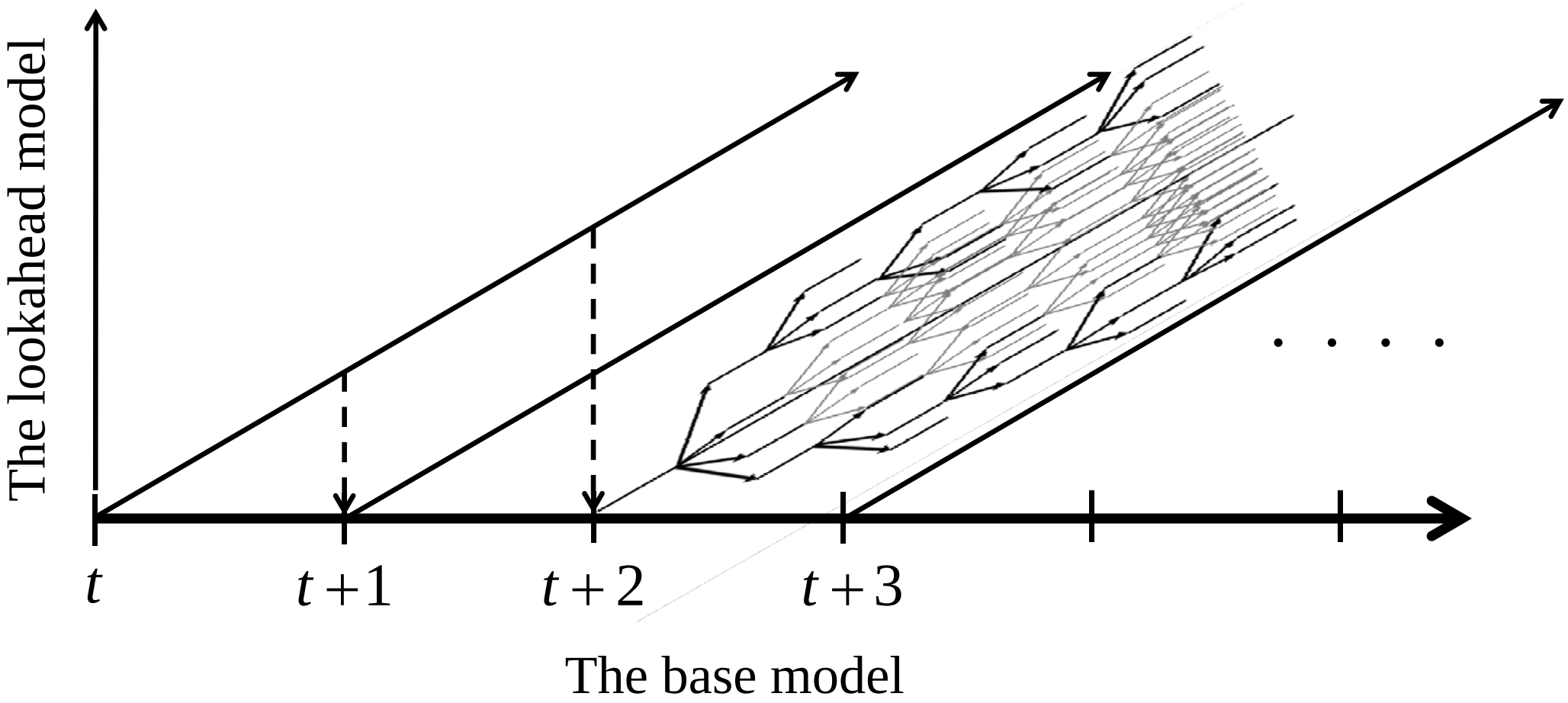
Stochastic lookahead policies

- We can then simulate this *lookahead policy* over time:



Stochastic lookahead policies

- We can then simulate this *lookahead policy* over time:



Lecture outline

- How to identify the right class of policy?

How to choose?

- Policy function approximations
  - Best for simple decisions
    - Whether to charge or discharge a battery
    - When to pump water in/our of a reservoir
    - When to replace an aging transformer
  - For many problems, the structure of the policy function is obvious:
    - Charge the battery when the price is below some point, discharge when it is above a higher point.
    - Pump water into the reservoir during off-peak hours, put water out during peak hours
    - Replace the transformer when age indicators (gases, fibers) are above some level.

How to choose?

- Robust cost function approximations
  - » Because there is a minimization, these can handle vector-valued decisions (such as unit commitment)
  - » They work well when minimizing a known cost (e.g. the cost over some horizon) works “pretty well,” and where it is clear what it might do wrong.
    - We need to schedule more generating capacity because the loads might be higher than expected, or renewables might be less than expected.
    - We might need to fire up the generator earlier because of potential delays.
  - » We will need to tune these adjustments using a base cost model.

How to choose?

- Value function approximations
  - These policies make decisions by balancing the cost now, plus the estimated future cost from being in the state that a decision now puts us in.
  - VFAs (value function approximations) work well when approximating the value of the downstream state is easy to estimate.
  - VFAs easily handle uncertainty, because it is possible to just average over the costs of different outcomes.
  - Estimating VFAs works well when we know something about the structure of the value function. Properties such as convexity or monotonicity really help.

How to choose?

- Lookahead policies
  - » This is the option of last resort, but we often need to turn to this.
  - » We need to turn to lookahead policies when we do not know the structure of a policy function or value function. We tend to need them for complex problems.
  - » Lookahead policies naturally handle forecasts – the other policies do not.
  - » The big decision:
    - Deterministic lookahead? This is what is widely used in engineering practice.
    - Stochastic lookahead? Popular with the research community, but not so widely used in practice. Typically very hard to solve.

How to choose?

- From lookahead to value functions:

$$X^{LA}(S_t) = \arg\min_{x_t, x_{t+1}, \ldots, x_{t+T}} C(S_t, x_t) + \underbrace{\sum_{t'=t+1}^{T} C(S_{t'}, x_{t'})}$$

$$X^{LA}(S_t) = \arg\min_{x_t, (x_{t+1}, \ldots, x_{t+T})(\omega)} C(S_t, x_t) + \sum_{\omega \in \Omega} p(\omega) \underbrace{\sum_{t'=t+1}^{T} C(S_{t'}(\omega), x_{t'}(\omega))}$$

$$V_{t+1}\left(S_{t+1}(S_t, x_t, W_t(\omega))\right)$$

$$\Downarrow$$

$$X^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \mathbb{E}\bar{V}_{t+1}(S_{t+1})\right)$$

» The VFA is just a way of approximating a lookahead model. If the approximation is easy, you should use it! But often, it is not.

How to choose?

- Building hybrids – you can mix and match:
  - » Lookahead policy for H time periods, followed by a VFA.
  - » Myopic/lookahead plus a low-dimensional PFA which is used to guide a high-dimensional linear program:
    - Solve the unit commitment, but force the pumped hydro to operate at certain times of day.
    - Or, just put bonuses for trying to follow an external rule (which is a form of PFA)
  - » Robust CFAs – Use a deterministic lookahead, modified by bonuses or constraints.
  - » Stochastic lookahead – Since these depend on a sample of the future, this works best when the first stage decision is low-dimensional.

How to choose?

- Most important:
  - Consider *all* four classes of policies.
  - It will usually be possible to restrict your search to one or two classes. This is where you need a feel for the problem, and a sense of how each class handles different problem classes.
  - Remember that the most important "stochastic model" is the base model, not the lookahead model.
  - Deterministic lookahead policies *can* work well, especially modified lookahead policies (we call these "robust cost function approximations") when they are tested using a stochastic simulator (or in the real world).
  - In the next series of slides, we present a simple energy storage problem, and then describe four variations, where each of the four classes of policies is best on certain problems.

An energy storage problem

- Consider a basic energy storage problem:

  - » We are going to show that with minor variations in the characteristics of this problem, we can make *each* class of policy work best.

An energy storage problem

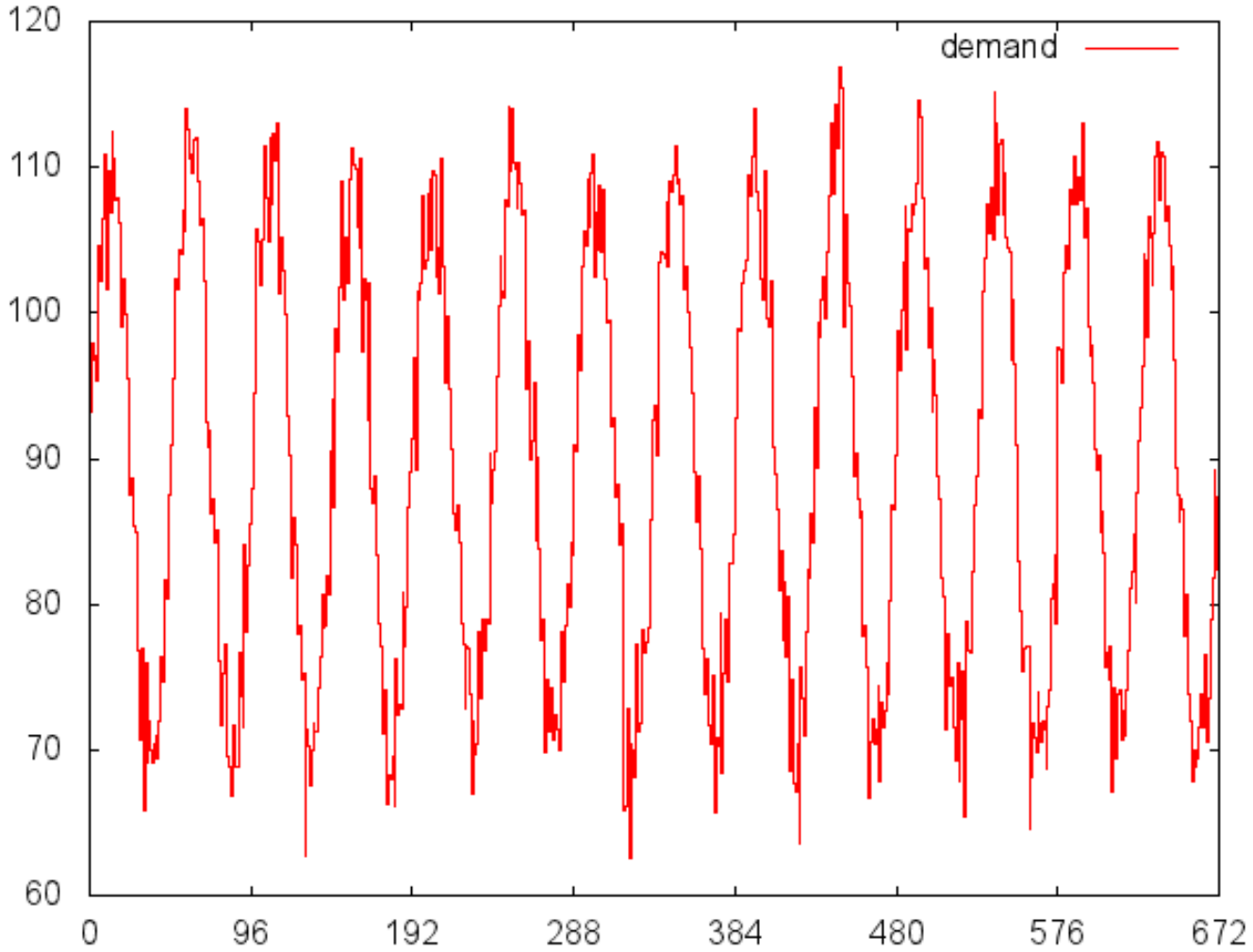
- We can create distinct flavors of this problem:
  - Problem class 1 – best for PFAs
    - Highly stochastic electricity prices
    - Stationary data
  - Problem class 2 – best for CFAs
    - Stochastic prices and wind
    - Time varying loads and reasonably accurate wind forecasts
  - Problem class 3 - best for VFAs
    - Stochastic wind and prices (but not too random)
    - Time varying loads, but inaccurate wind forecasts
  - Problem class 4 – best for deterministic lookaheads
    - Relatively low noise problem with accurate forecasts

An energy storage problem

- Notes:
  - In the next few slides, we show some illustrations of demands (loads), energy generation and prices.
  - We have created a family of six problems (summarized after the next few slides), all of which could easily arise in practice, which bring out the strengths in all four classes of policies.
  - This work is new and still evolving. In time we hope to post more detailed summaries of the characteristics of each of the six problems that bring out the strengths of each policy.

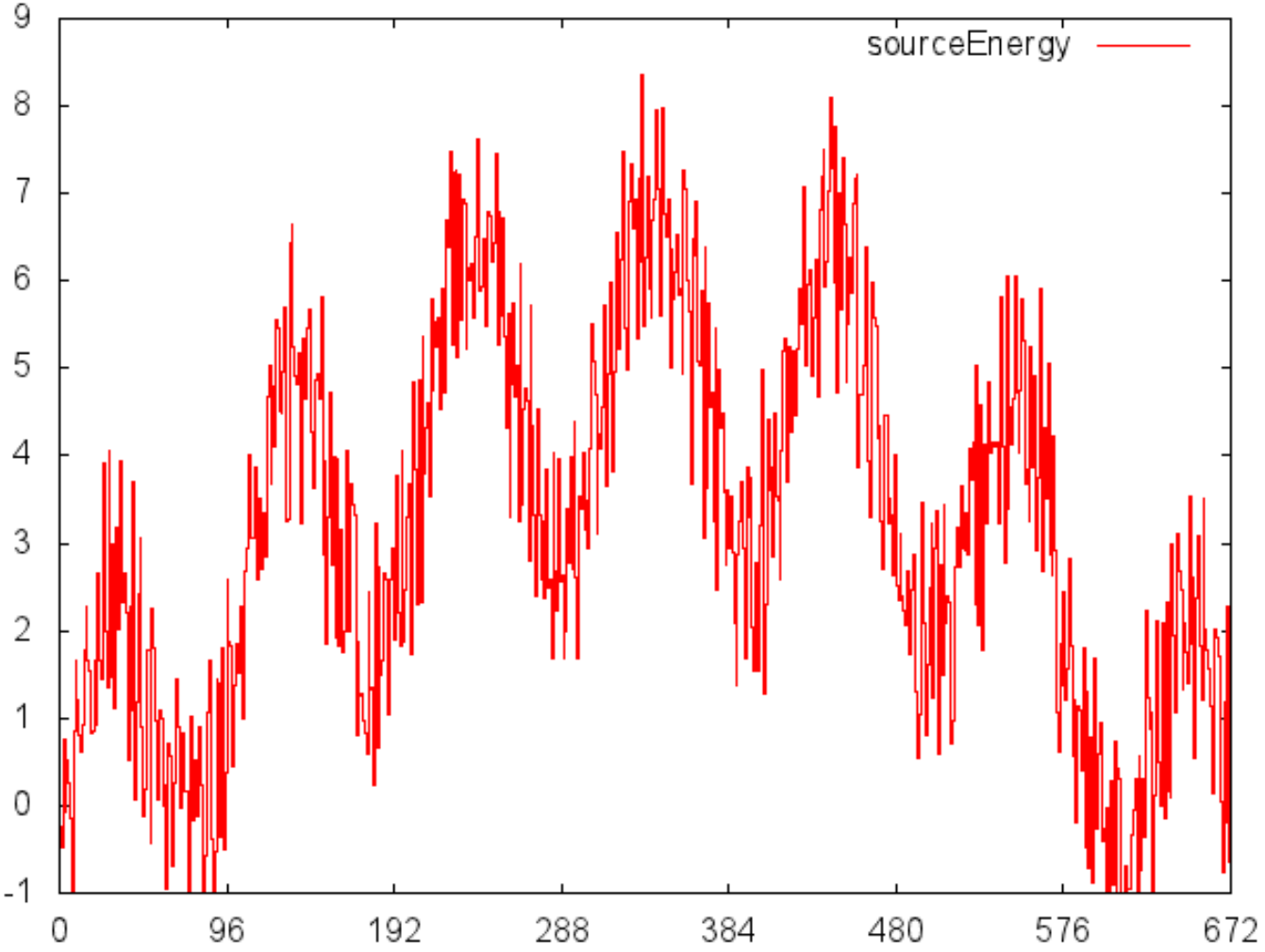
An energy storage problem

- A demand sample path



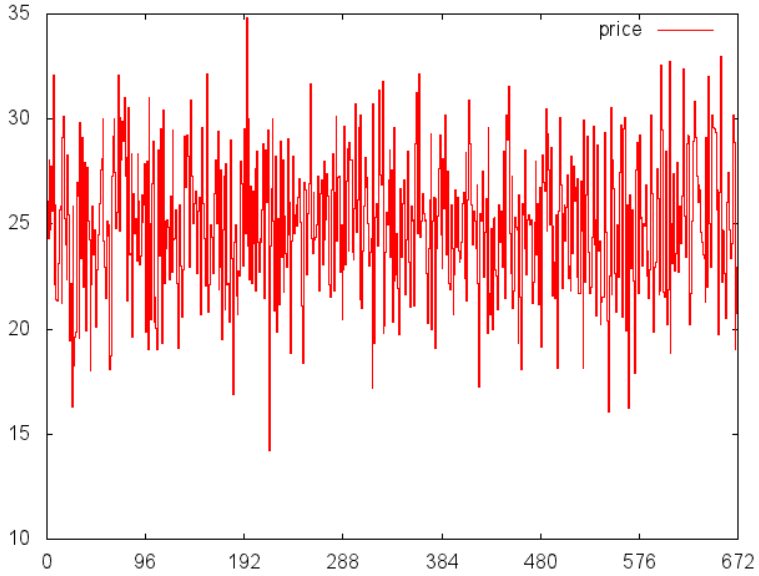
An energy storage problem

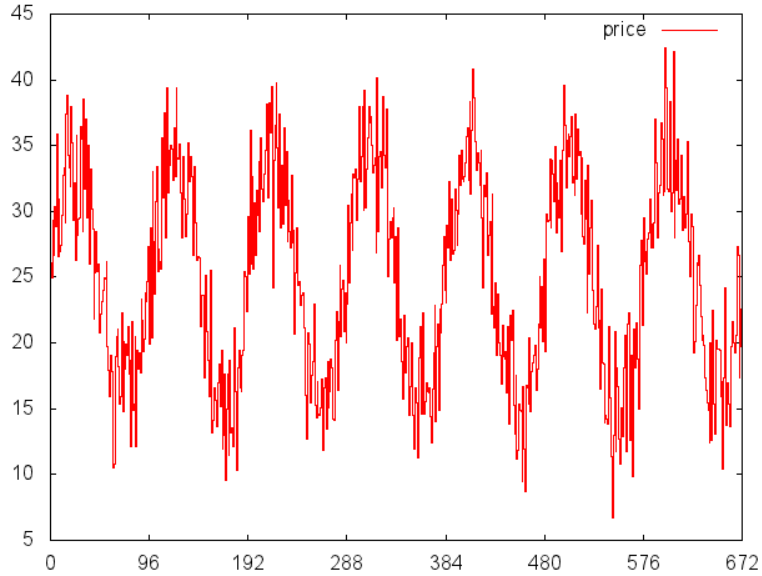
- A sample path for energy flows

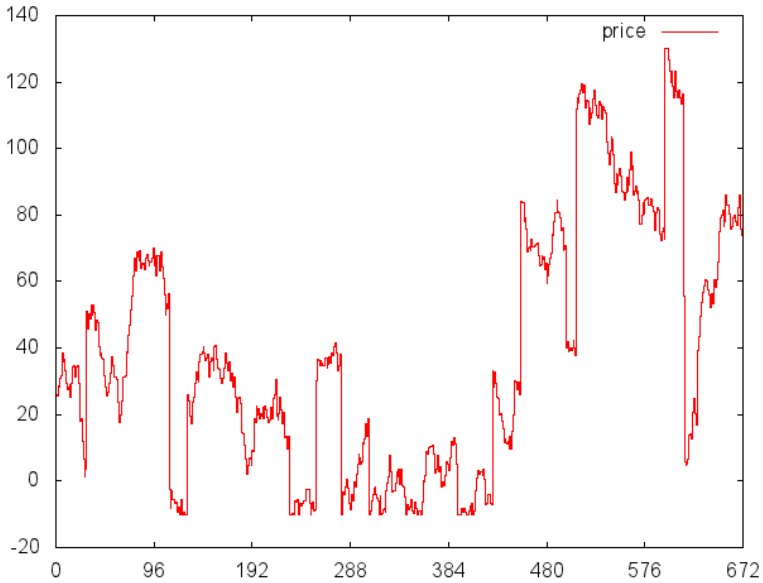


An energy storage problem

- Price sample paths







An energy storage problem

- The policies
  - The PFA:
    - Charge battery when price is below p1
    - Discharge when price is above p2
  - The CFA
    - Optimize over a horizon H; maintain upper and lower bounds (u, l) for every time period except the first (note that this is a hybrid with a lookahead).
  - The VFA
    - Piecewise linear, concave value function in terms of energy, indexed by time.
  - The lookahead (deterministic)
    - Optimize over a horizon H (only tunable parameter) using forecasts of demand, prices and wind energy

An energy storage problem

- Each policy is best on certain problems
  - » Results are percent of *posterior* optimal solution

Percent of posterior bound	PFA (20, 30)	CFA (12h; 0.2, 0.9)	VFA (0.1, 0.01)	detLH (12h)
MP02-331-000-10_1_10	97.7%	93.8%	*93.1%*	94.6%
MP02-331-000-10_1_1000	91.6%	93.0%	62.8%	91.3%
MP02-332-000-10_1_10	85.3%	91.0%	*74.2%*	97.4%
MP02-332-000-10_1_1000	86.3%	92.6%	62.6%	92.0%
MP02-33x-000-10_1_10	77.0%	68.9%	*76.3%*	70.2%
MP02-33x-000-10_1_1000	66.5%	39.4%	68.5%	43.5%
Average	84.1%	79.8%	72.9%	81.5%

  - » This means – if you have a favorite policy, I have a problem where another policy will work better.

Joint research with Prof. Stephan Meisel, University of Muenster, Germany.

Lecture outline

- The fields of stochastic optimization

The fields of stochastic optimization

- So, what is the difference between:
  - » Stochastic programming
  - » Dynamic programming, generally practiced as
    - Approximate dynamic programming
    - Reinforcement learning
  - » Robust optimization

The fields of stochastic optimization

- Stochastic programming
  - » For sequential problems, "stochastic programming" is equivalent to a stochastic lookahead policy.
  - » Stochastic programming as a field has focused primarily on a problem known as a "two stage stochastic program" which consists of:
    - Make a decision x_0 (e.g. schedule steam)
    - See random information W_1 (e.g. see the wind, outages).
    - Make final decision x_1 (e.g. schedule gas turbines)
  - » A small number of papers attempt to solve multistage problems (decision, information, decision, information, …), but these are rarely practical.
  - » A lookahead policy, using a two-stage approximation of the lookahead model, is written

$$X_t^{LA-S}(S_t \mid \theta) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

The fields of stochastic optimization

- Dynamic programming
  - » At the heart of dynamic programming is Bellman's equation, which means expressing a policy in terms of a value function (generally a value function approximation):

$$X_t^{\pi}(S_t \mid \theta) = \arg\min_{x_t} \left(C(S_t^n, x_t) + \bar{V}_t^{n-1}(S^{M,x}(S_t^n, x_t)) \right)$$

    - This is a policy based on a value function approximation
  - » But there are many papers in the dynamic programming literature which recognize that a policy might have a special structure (rule based, linear in the state, …) where the parameters can be optimized directly.
    - This would be a policy function approximation
  - » There is a substantial research literature working on approximating value functions and policy functions.

The fields of stochastic optimization

- Robust optimization
  - » Robust optimization emerged in engineering design, where we need to choose parameters x (e.g. the design of an airplane wing) that work across the worst set of random forces w:

$$\min_x \max_{w \in \mathcal{W}} F(x,w)$$

  - » In the last 10 years, researchers have proposed using this as a form of lookahead policy for sequential problems (such as unit commitment). Let $\mathcal{W}$ be an *uncertainty set* parameterized by θ (that might be the tail of the probability distribution). This gives us the policy

$$X_t^{LA-SP}(S_t \mid \theta) = \arg\min_{\tilde{x}_t,\ldots,\tilde{x}_{t+H}} \max_{w \in \mathcal{W}} \sum_{t'=t}^{T} C(\tilde{S}_{tt'}(w), \tilde{x}_{tt'}(w))$$

  - » Surprisingly, the policy is tuned using expectations:

$$\min_\theta E\left\{\sum_{t=0}^{T} C\left(S_t, X_t^{LA-SP}(S_t \mid \theta)\right)\right\}$$

The fields of stochastic optimization

- Robust cost function approximation

  - This is what is done in industry. It is fair to say that current industry practice is described as follows:

    ISOs use a parametric cost function approximation optimized using online learning (real world observations) to produce a robust policy.

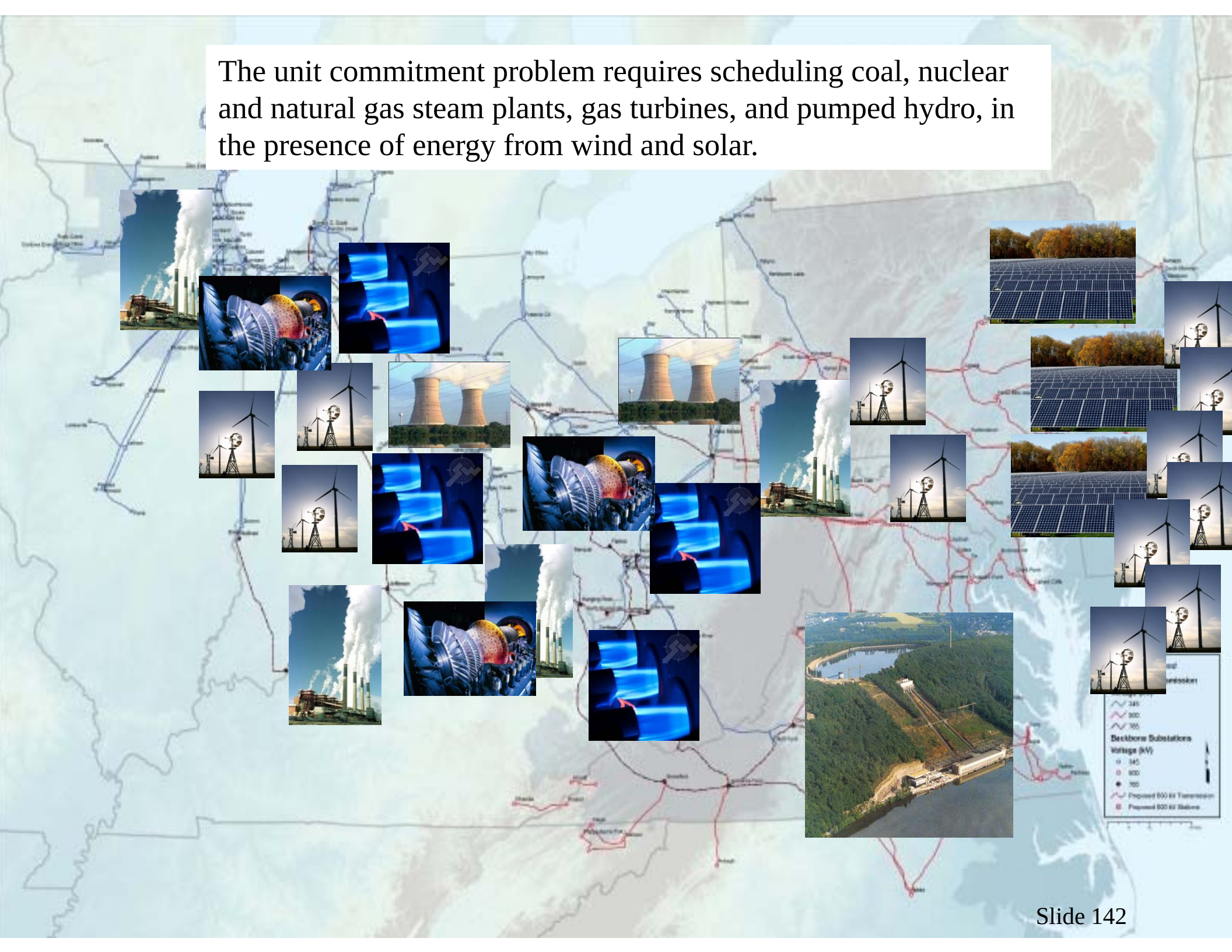
  - This entire strategy has been overlooked by the academic community as a potential algorithmic strategy.

  - We believe that for the unit commitment problem, a robust CFA is going to be very difficult to beat.

Lecture outline

- The stochastic unit commitment problem

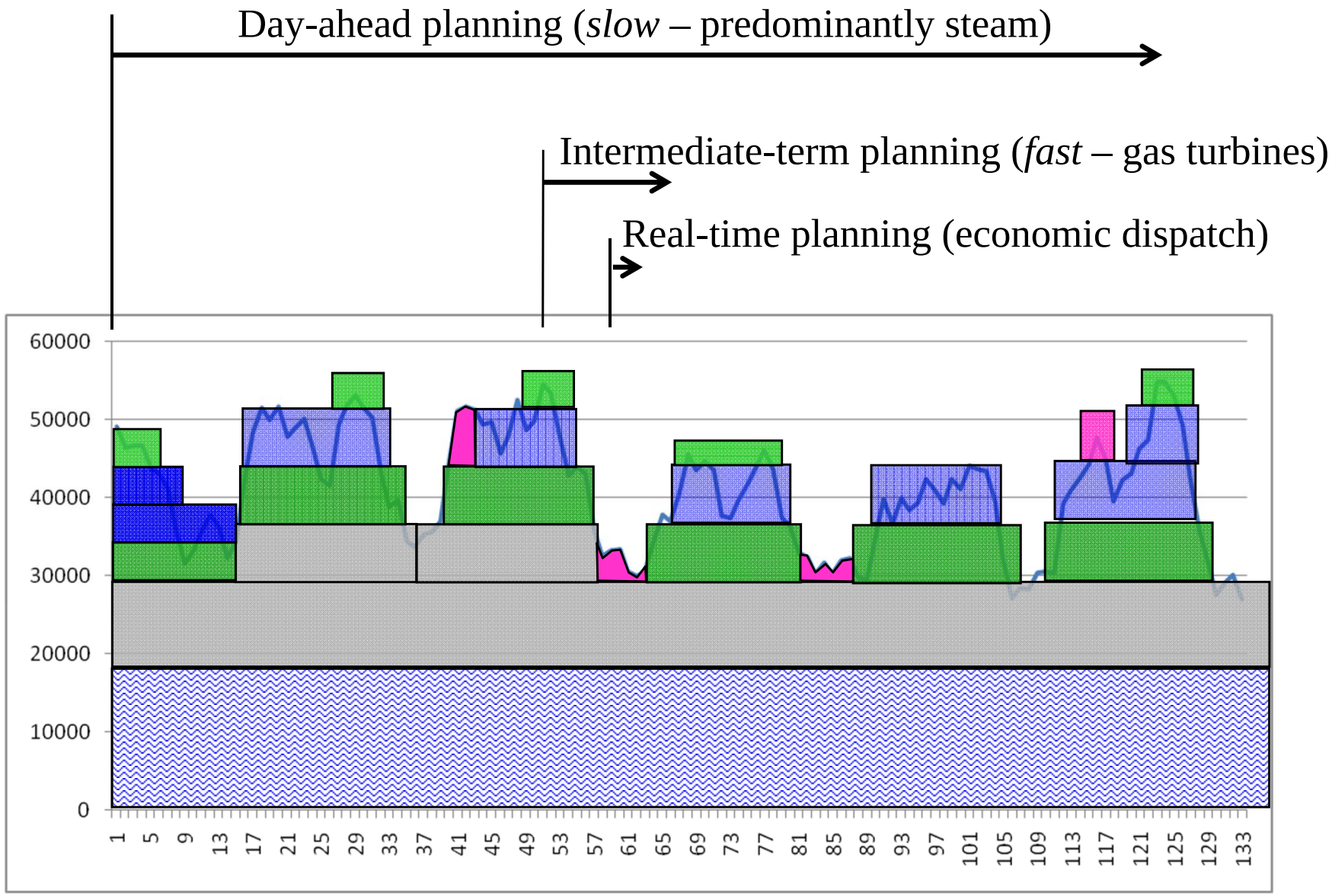
The unit commitment problem requires scheduling coal, nuclear and natural gas steam plants, gas turbines, and pumped hydro, in the presence of energy from wind and solar.



The stochastic unit commitment problem

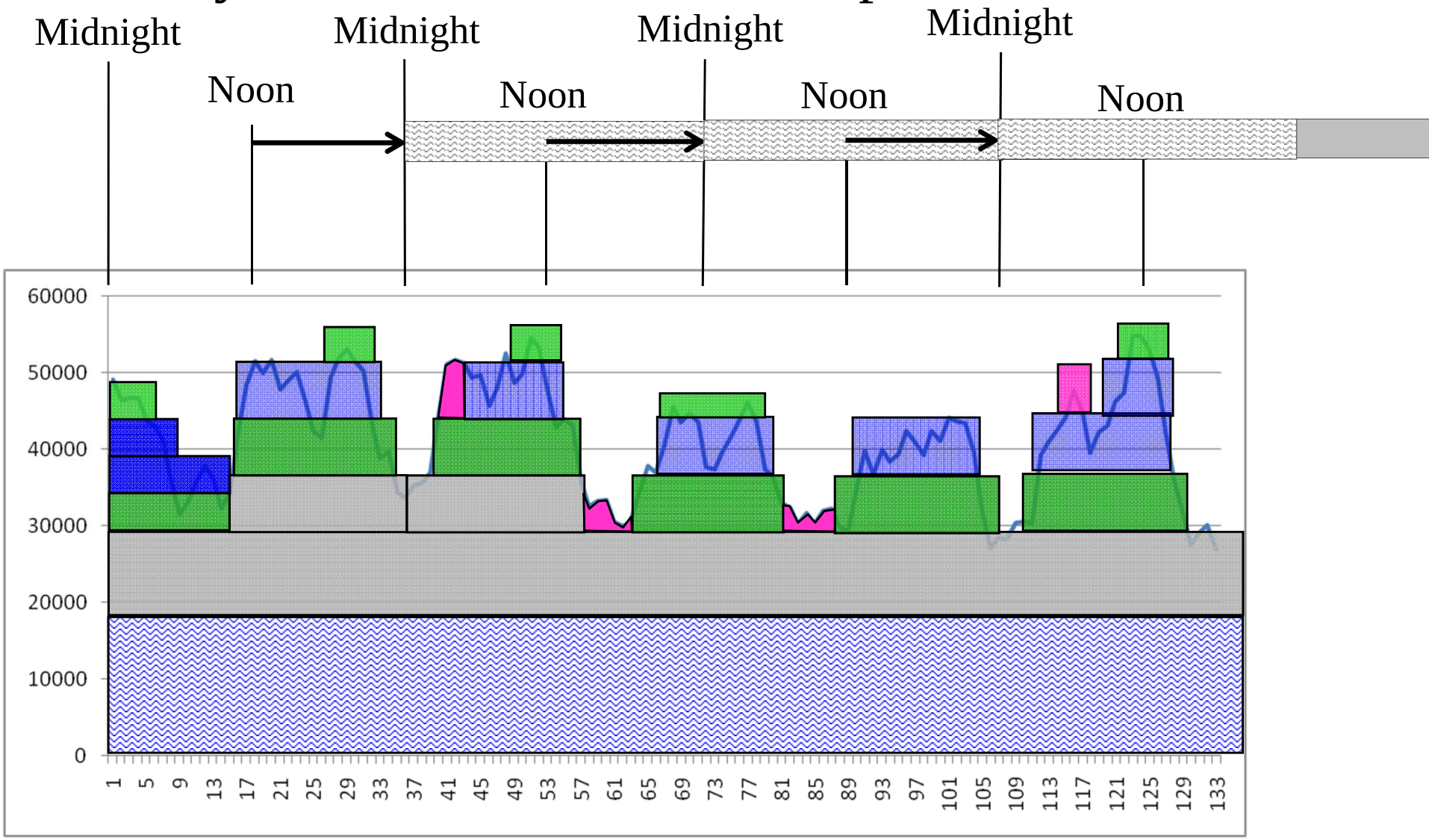
- Overview
  - Energy generators (nuclear, coal and gas steam generators, gas turbines, pumped hydro) all have to be scheduled while respecting different notification times.
  - There are three primary planning processes:
    - The day-ahead unit commitment problem – This is where steam generators are planned
    - The intermediate-term unit commitment problem – Run every 15-30 minutes, this is primarily where we make commitments to gas turbines
    - The real-time economic dispatch – Run every five minutes, this process can ramp generators up and down, but cannot turn them on or off.

The timing of decisions



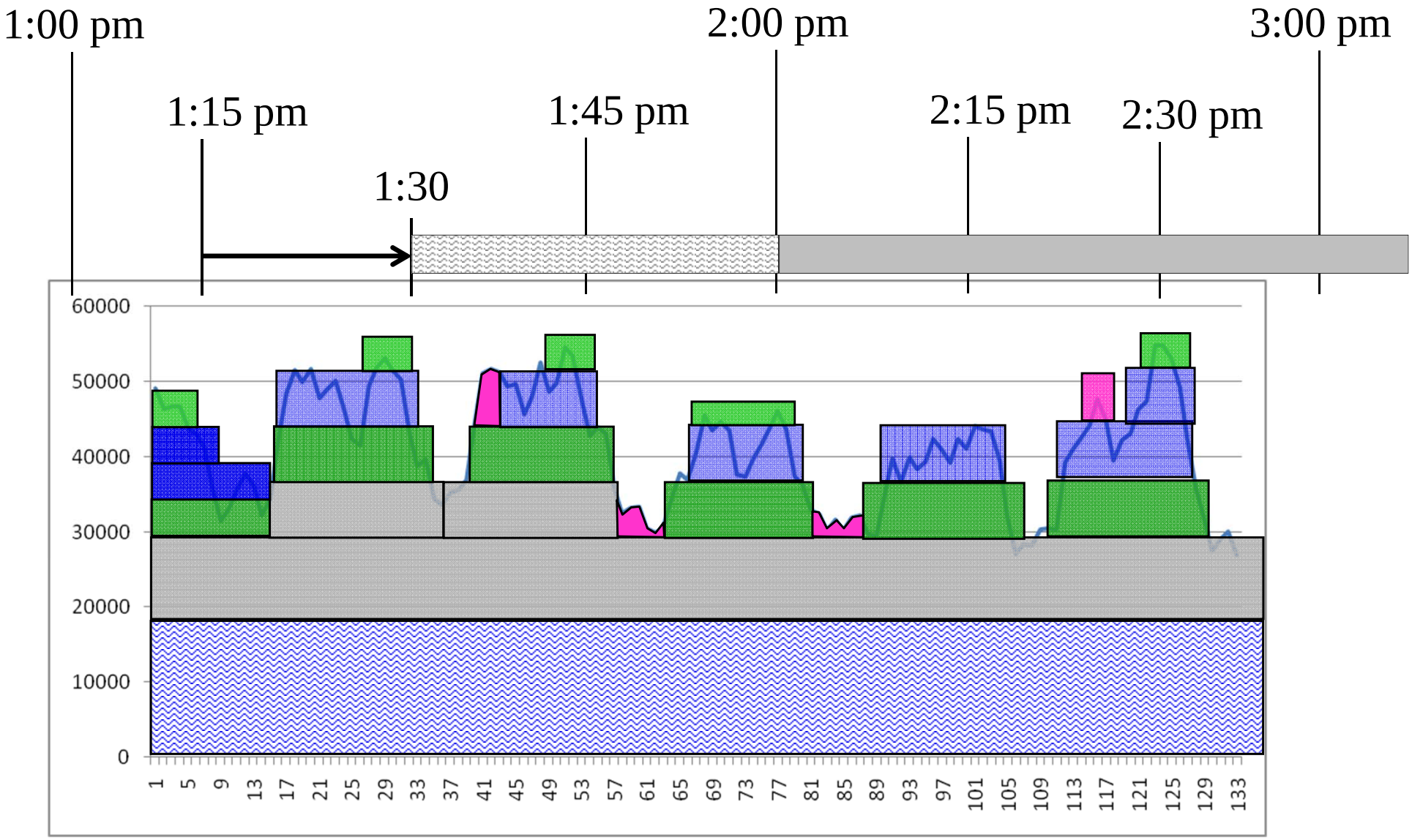
The timing of decisions

- The day-ahead unit commitment problem



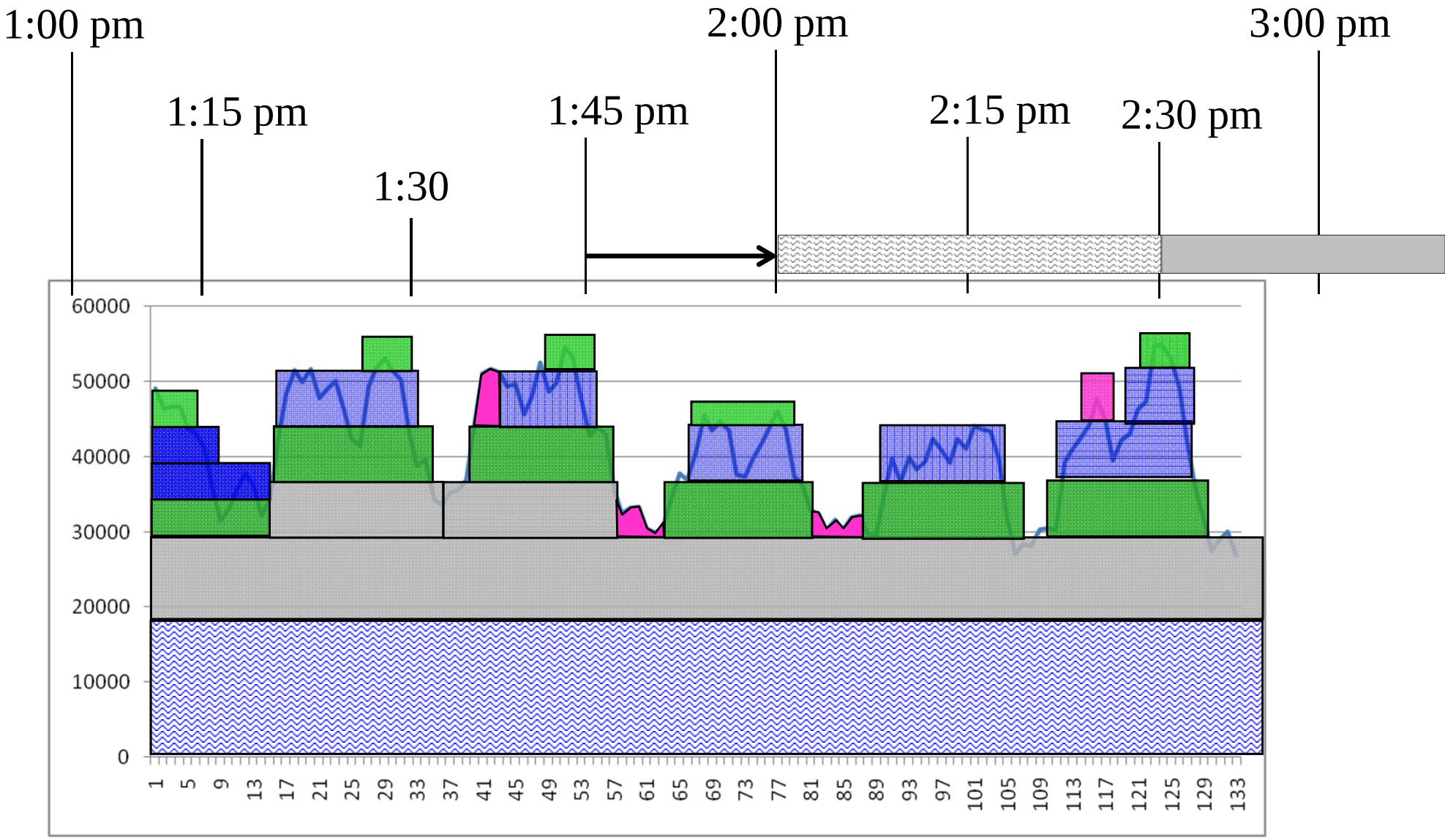
The timing of decisions

- Intermediate-term unit commitment problem



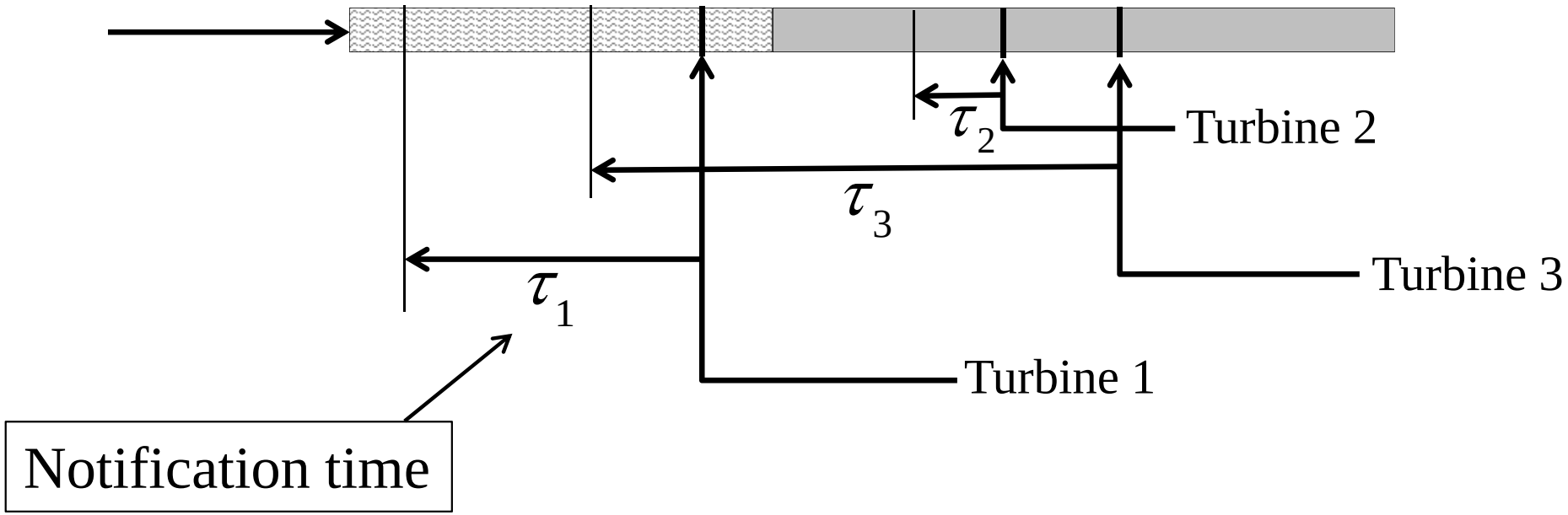
The timing of decisions

- Intermediate-term unit commitment problem



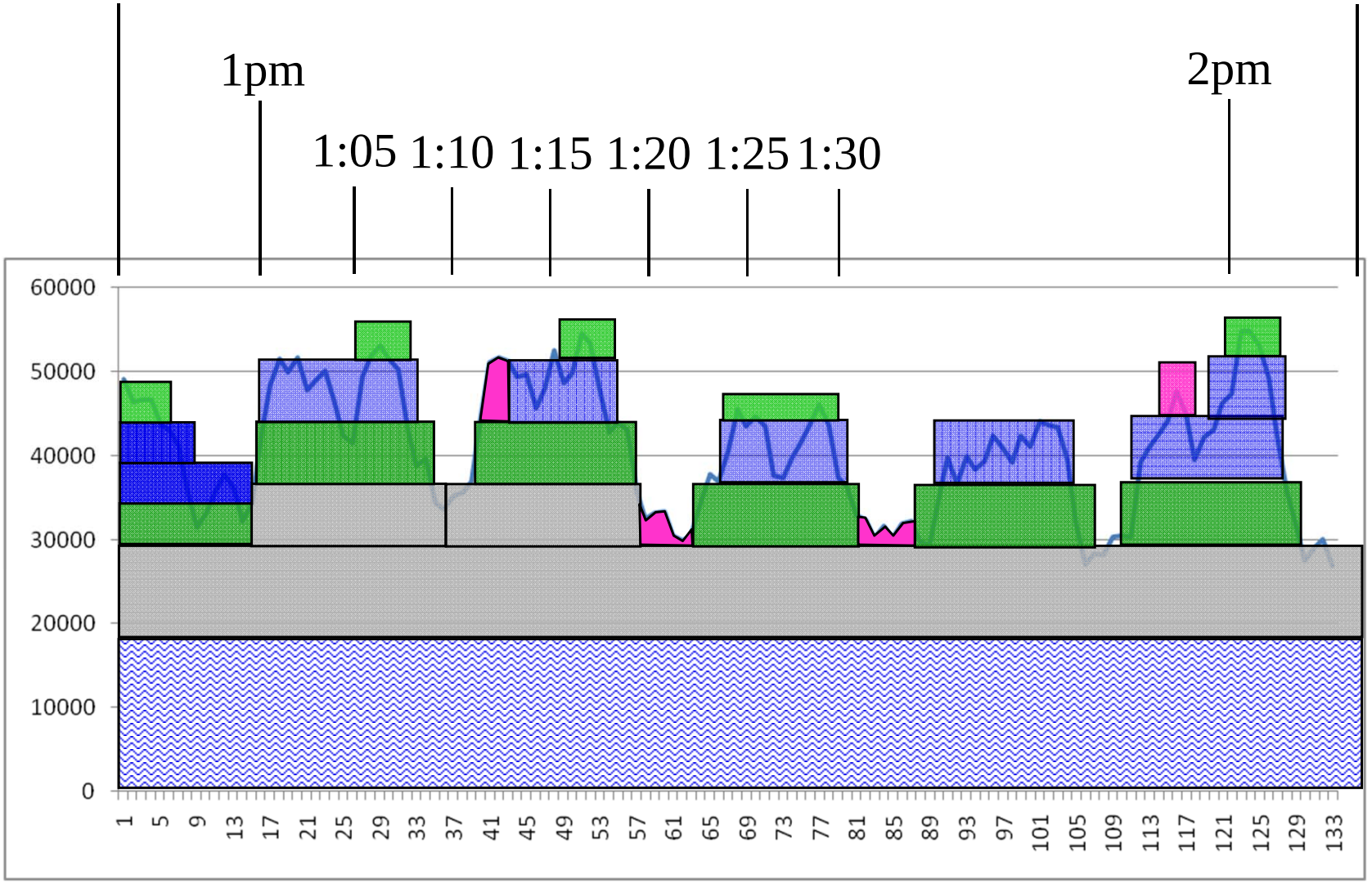
The timing of decisions

- Intermediate-term unit commitment problem
  - » There are a range of notification times for different generators.
  - » Generators are only called if their commitment time falls in the hatched area.



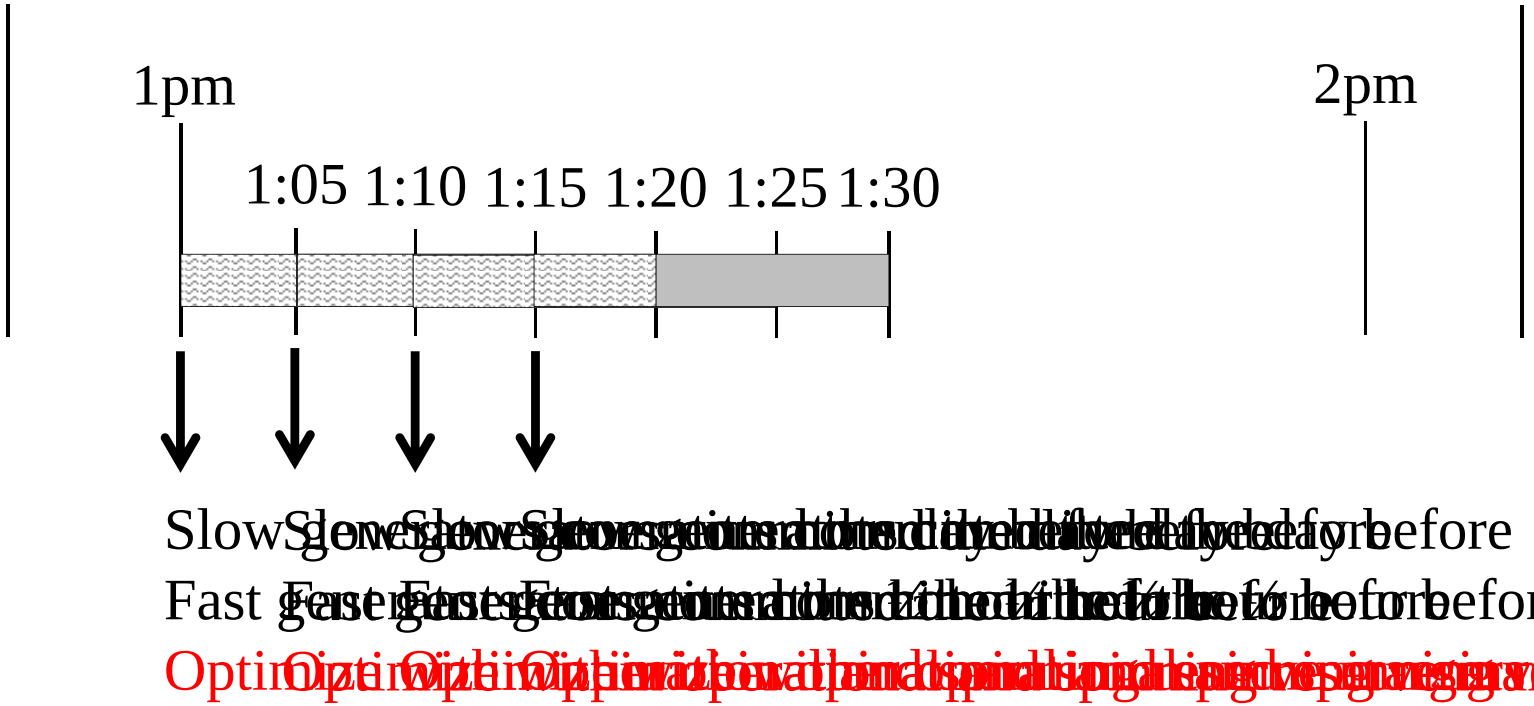
The timing of decisions

- Real-time economic dispatch problem



The timing of decisions

- Real-time economic dispatch problem



» Generators may be ramped up and down, but may not be turned on or off.

Stochastic unit commitment

- We have just described our *base model*:
  - The base model is a computer simulation, which we need to create settings (such as high penetrations of renewables) which do not exist today.
  - It should model the problem at a high level of detail.
  - It is very important to calibrate the base model carefully against history.
  - The base model should not be confused with a lookahead model, which may need to introduce simplifications for computational efficiency.

SMART-ISO: Calibration

- The base model consists of three core components:
  - The policy (which determines how decisions are made)

$$x_t = X^{\pi}(S_t)$$

  - The exogenous information process:

$$W_1, W_2, ..., W_t, ...$$

    Contains information about changes in renewables, temperature, equipment failures, and market behavior.
  - The transition function, which captures all the dynamics of the problem:

$$S_{t+1} = S^M\left(S_t, x_t, W_{t+1}\right)$$

  - So what policy should we use?

Four classes of policies

1) Policy function approximations (PFAs)
   - » Lookup tables, rules, parametric functions
2) Cost function approximation (CFAs)
   - » $X^{CFA}(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$
3) Policies based on value function approximations (VFAs)
   - » $X_t^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) Lookahead policies
   - » ***Deterministic lookahead:***

$$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » ***Stochastic lookahead (e.g. stochastic trees)***

$$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\omega \in \tilde{\Omega}_t} p(\omega) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\omega), \tilde{x}_{tt'}(\omega))$$

Stochastic unit commitment

- The stochastic unit commitment and stochastic programming
  - In recent years, the “stochastic unit commitment problem” has become synonymous with an algorithmic strategy called “stochastic programming.”
  - This is only one of our four classes of policies.
  - Below, we take a close look at using a stochastic lookahead policy for unit commitment (“stochastic programming”)
  - We are going to show the results of research that suggests that a stochastic lookahead model (using scenario trees) is not suited for handling the uncertainty of renewables.

Four classes of policies

1) Policy function approximations (PFAs)
   - » Lookup tables, rules, parametric functions
2) Cost function approximation (CFAs)
   - » $X^{CFA}(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$
3) Policies based on value function approximations (VFAs)
   - » $X_t^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^{x} \left(S_t^{x}(S_t, x_t) \right) \right)$
4) Lookahead policies
   - » ***Deterministic lookahead:***

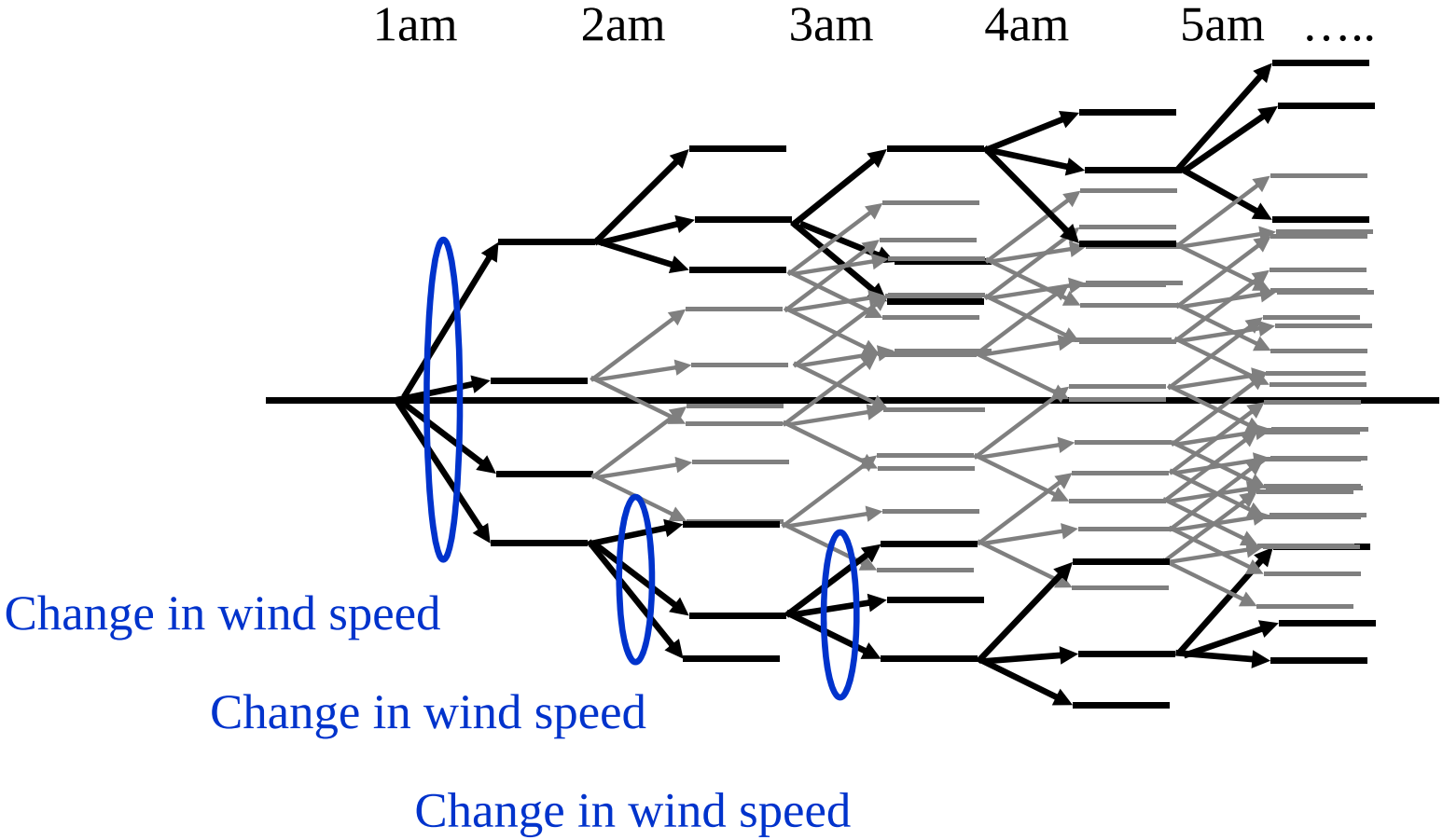
$$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$

   - » ***Stochastic lookahead (e.g. stochastic trees)***

$$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\omega \in \tilde{\Omega}_t} p(\omega) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\omega), \tilde{x}_{tt'}(\omega))$$

Stochastic lookahead policies

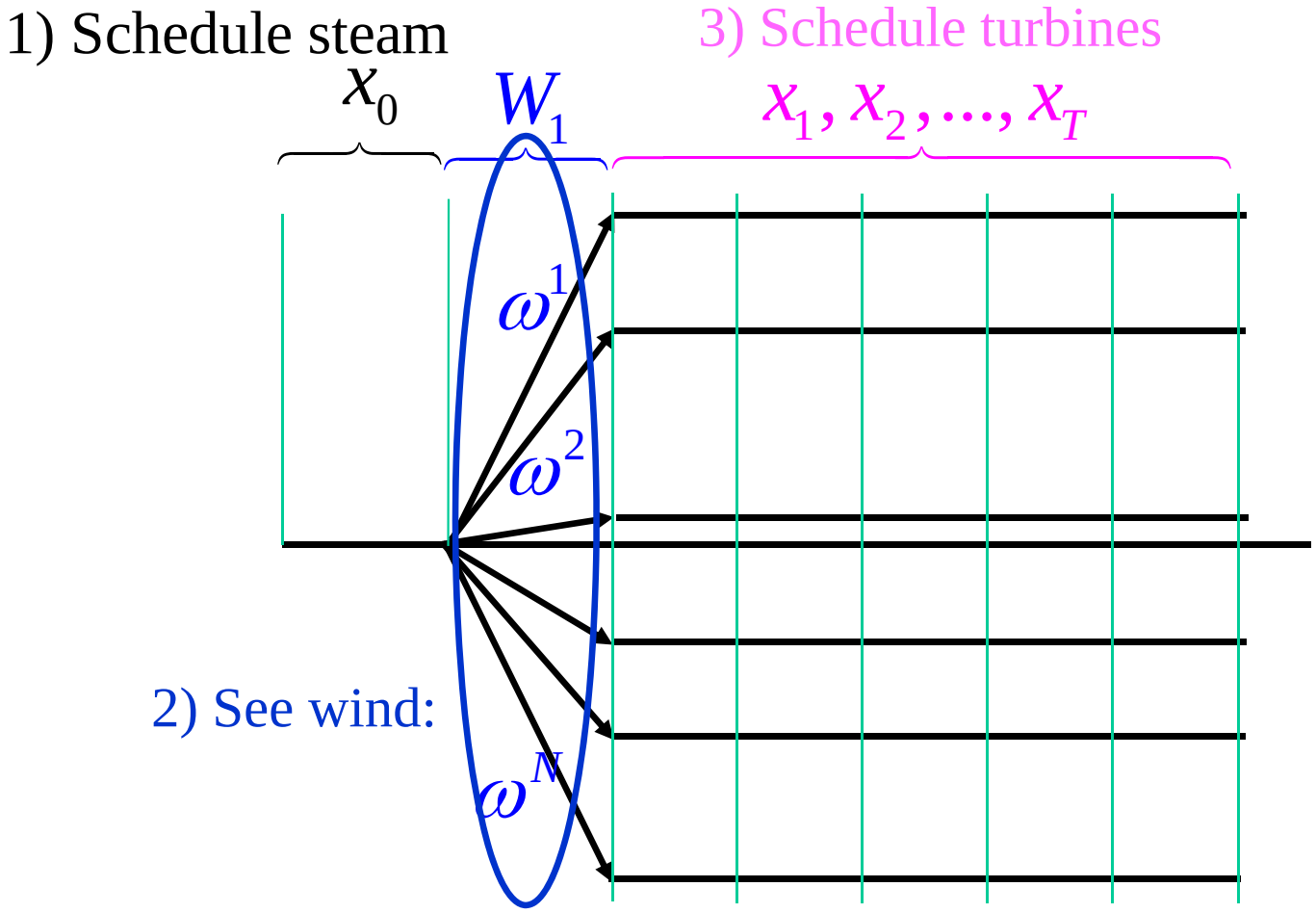
- Stochastic lookahead
  - » Here, we approximate the information model by using a Monte Carlo sample to create a scenario tree:



Multistage unit commitment models are completely unsolvable.

Stochastic lookahead policies

- Two stage lookahead approximation

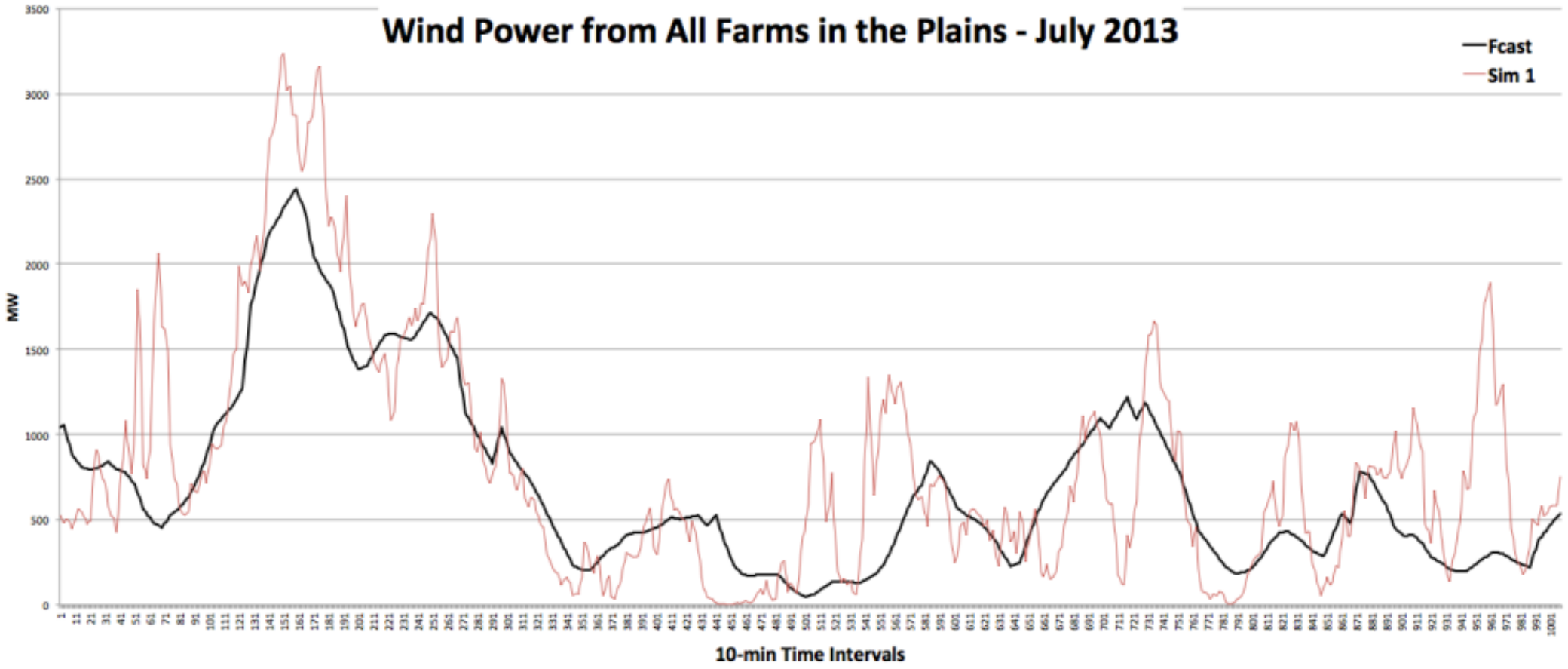


Stochastic unit commitment

- We have to construct *scenarios* of what might happen.
  - » "Scenarios" are possible stochastic outcomes in a lookahead model (not to be confused with "sample paths" that represent realizations in a base model)
  - » For our problem, assume the only uncertainty is the energy generated by wind.
  - » We developed a mathematical model that models the errors in forecast. We then sample errors from this model, and add it to the forecast to get a realization of the energy that *might* be generated from wind over the day.
  - » Below we show five scenarios.

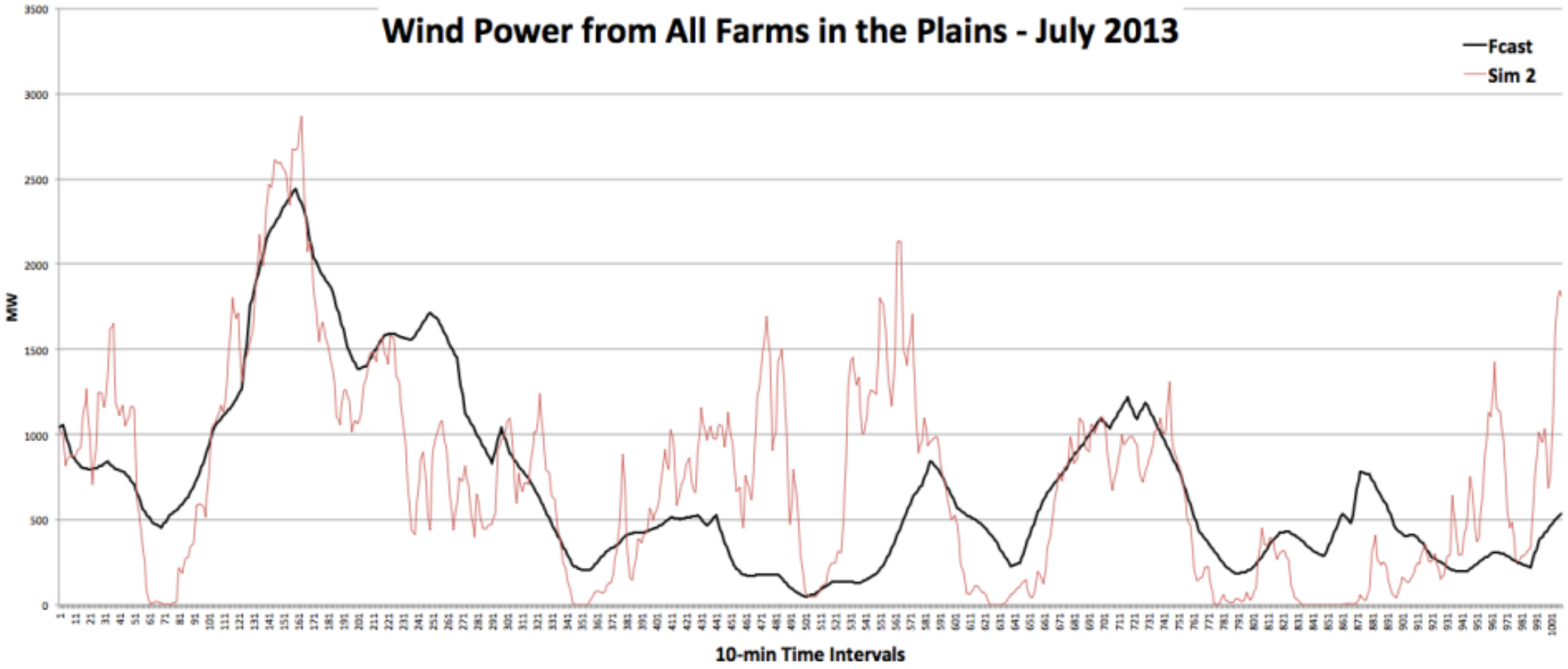
Stochastic lookahead policies

- Creating wind scenarios (Scenario #1)



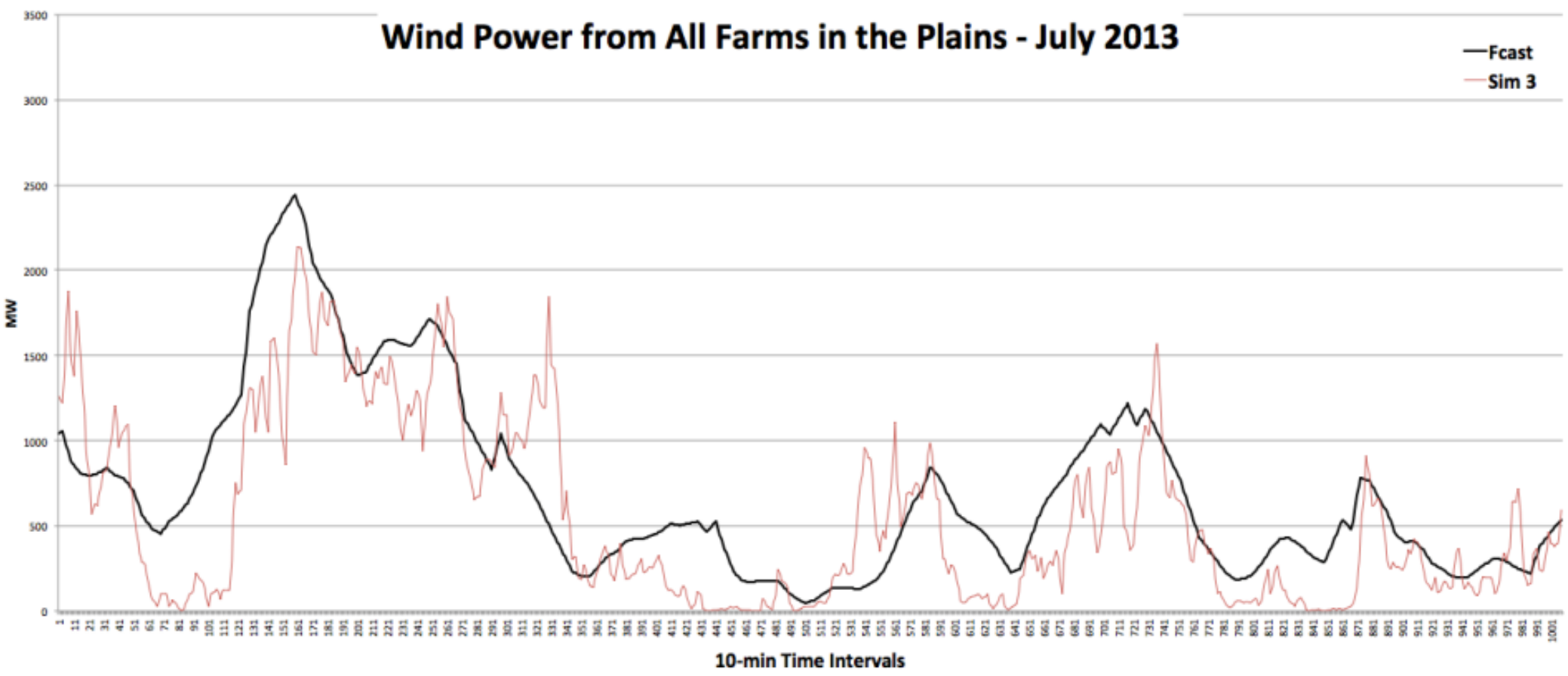
Stochastic lookahead policies

- Creating wind scenarios (Scenario #2)



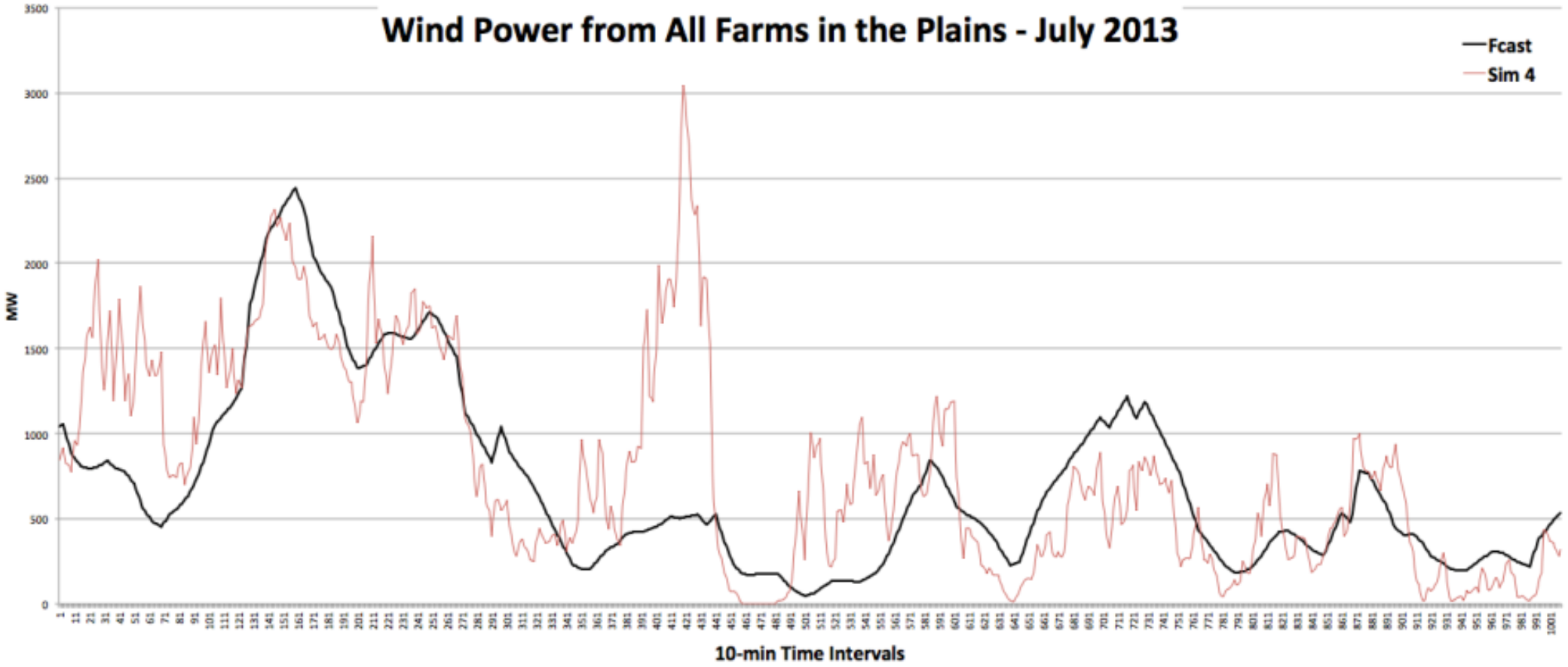
Stochastic lookahead policies

- Creating wind scenarios (Scenario #3)



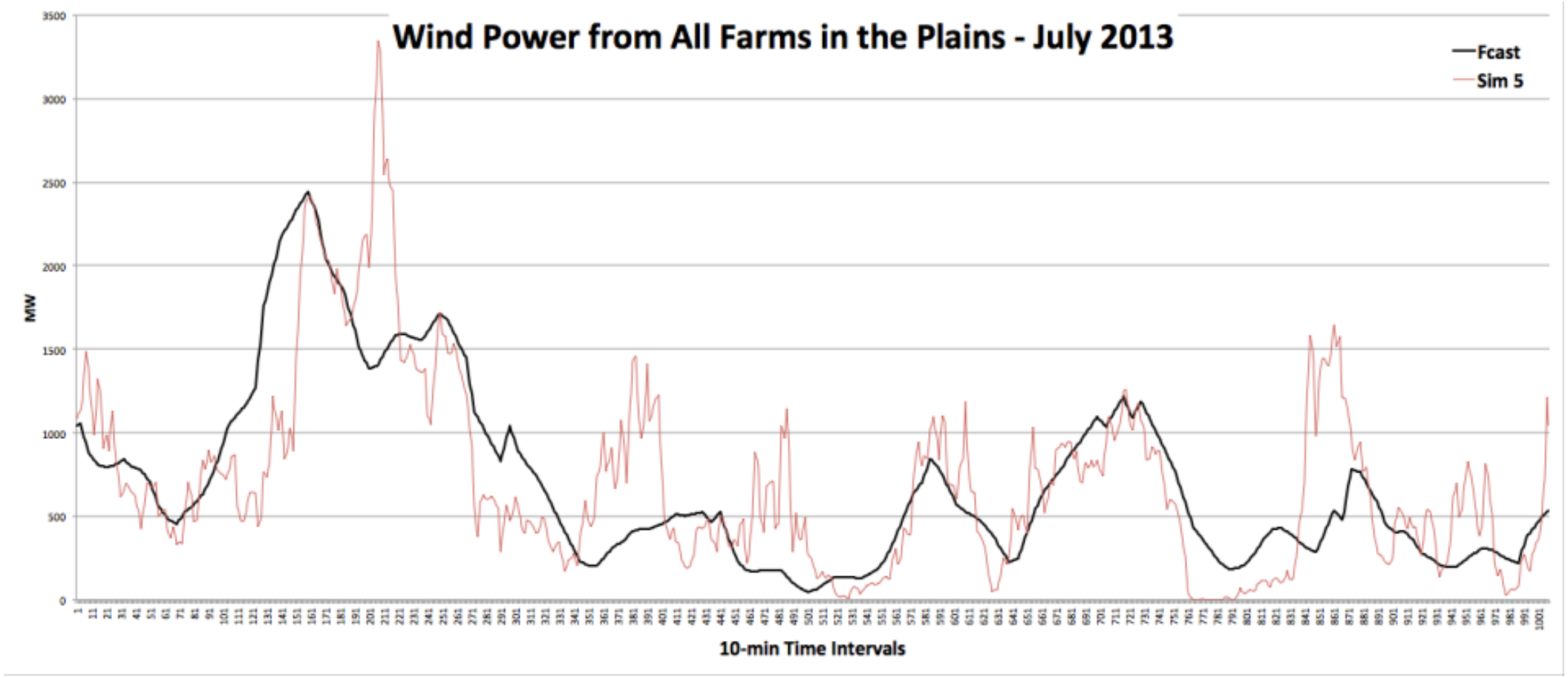
Stochastic lookahead policies

- Creating wind scenarios (Scenario #4)



Stochastic lookahead policies

- Creating wind scenarios (Scenario #5)

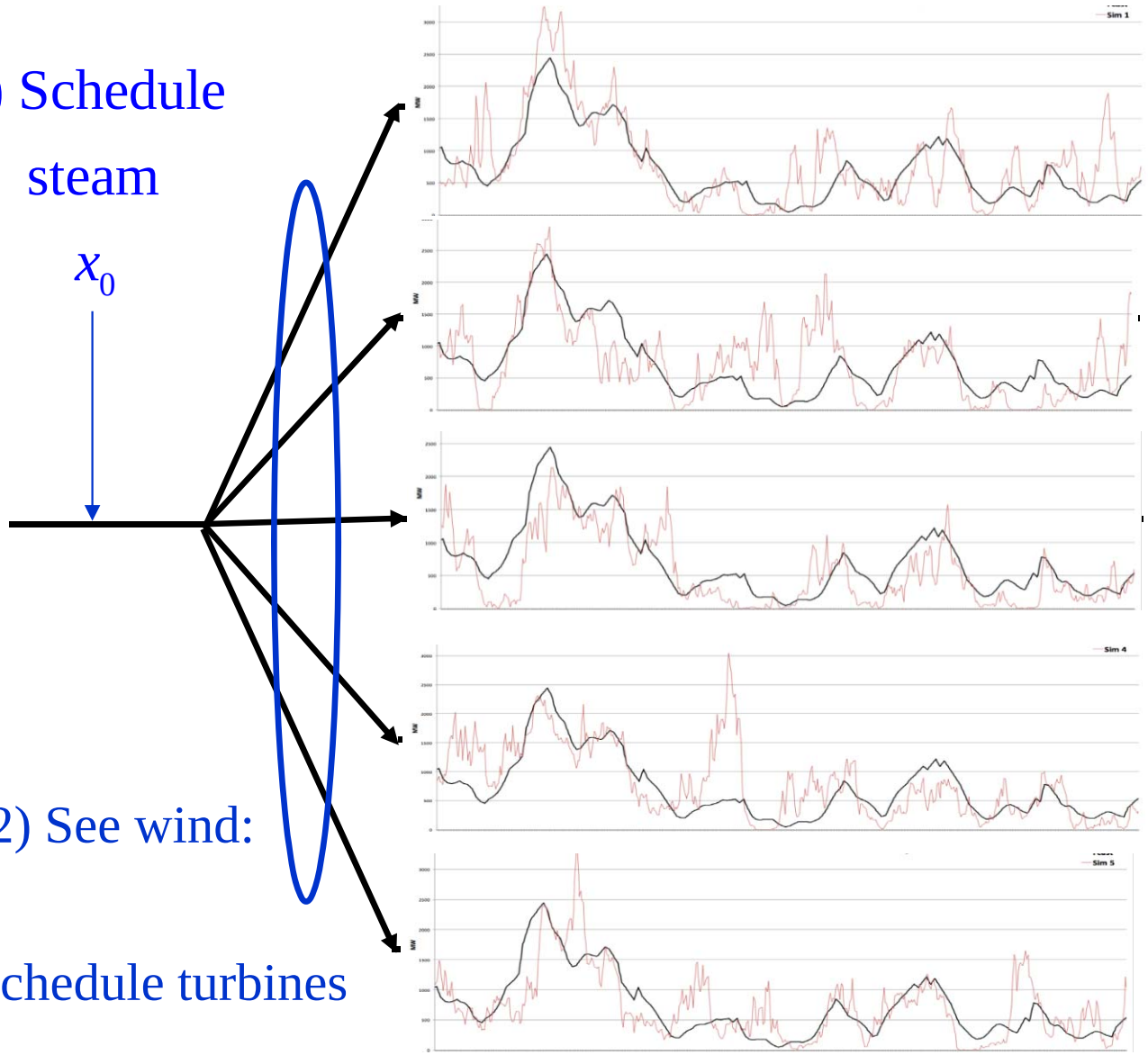


Stochastic unit commitment

- Forming the lookahead model
  - Now we are going to create a single model that combines:
    - The decision of which steam generators we should schedule today.
    - The five wind energy scenarios.
    - For each scenario, we create a complete schedule of all of the gas turbines that would be scheduled tomorrow.
  - The two-stage model means that the decision to schedule a gas turbine at 10am is done knowing the entire wind profile tomorrow.
  - This creates a single, very large unit commitment problem. If we have K scenarios, then we have K gas turbine scheduling problems, plus the steam scheduling problem, all as a single, large integer program.

Stochastic lookahead policies

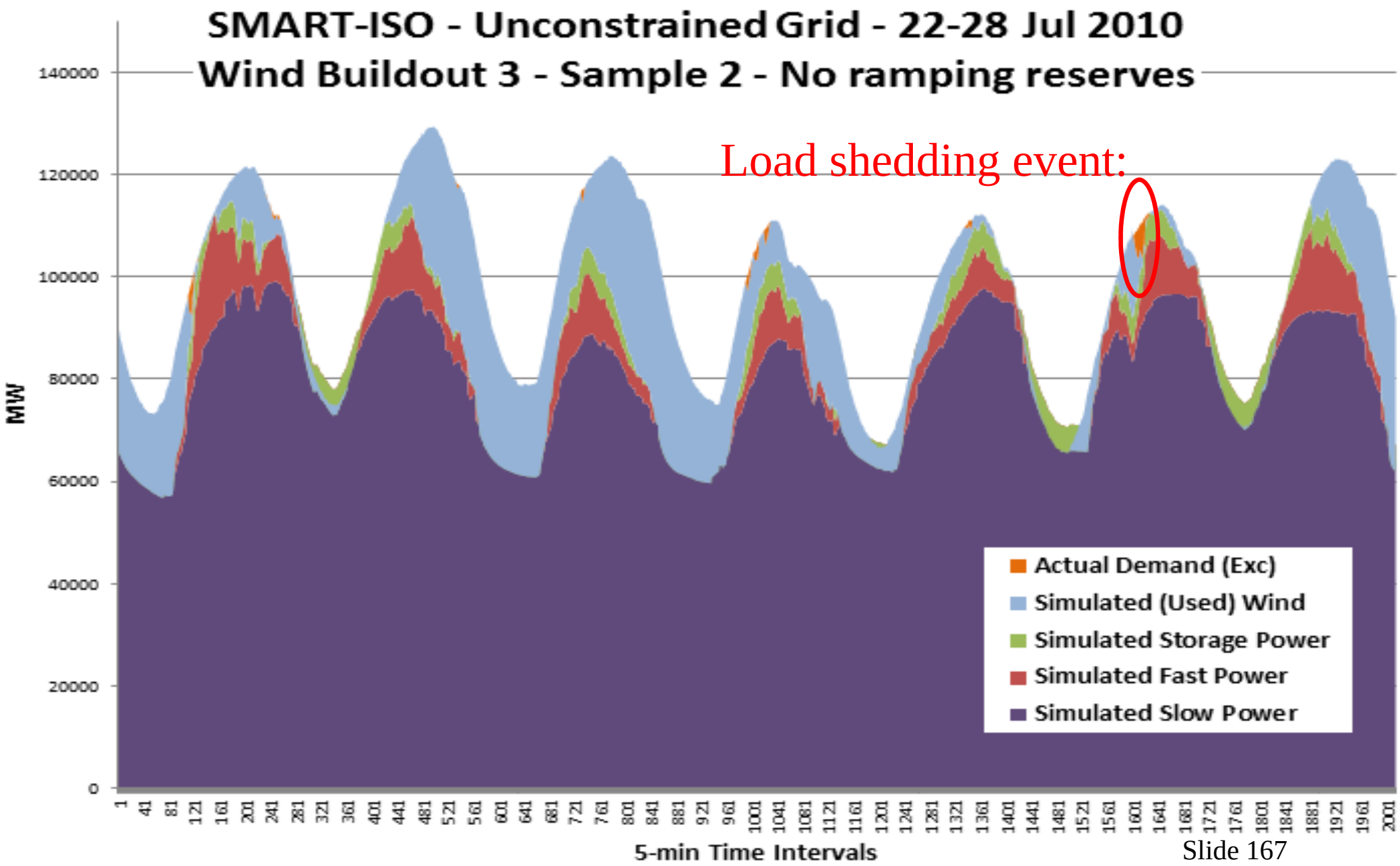
- The two-stage approximation



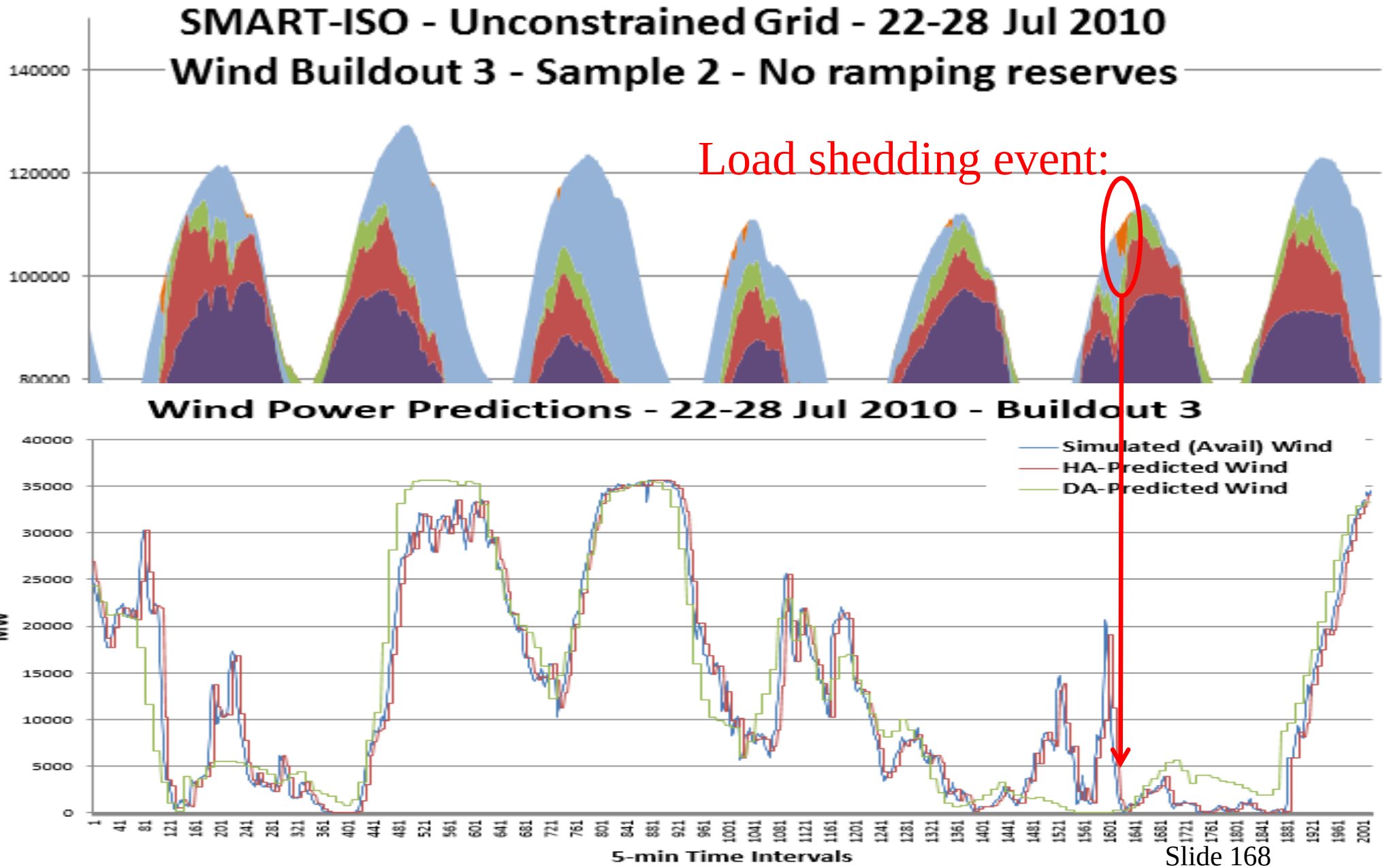
Stochastic unit commitment

- What is next:
  - » We have done some analysis of load shedding in our simulator.
  - » It appears that a specific type of wind event creates problems for our model.
  - » We examined what happens when we allow the day-ahead decision to see what was going to happen tomorrow.
  - » We are going to show that this advance information allows the model to schedule extra steam *but only when it is needed*.

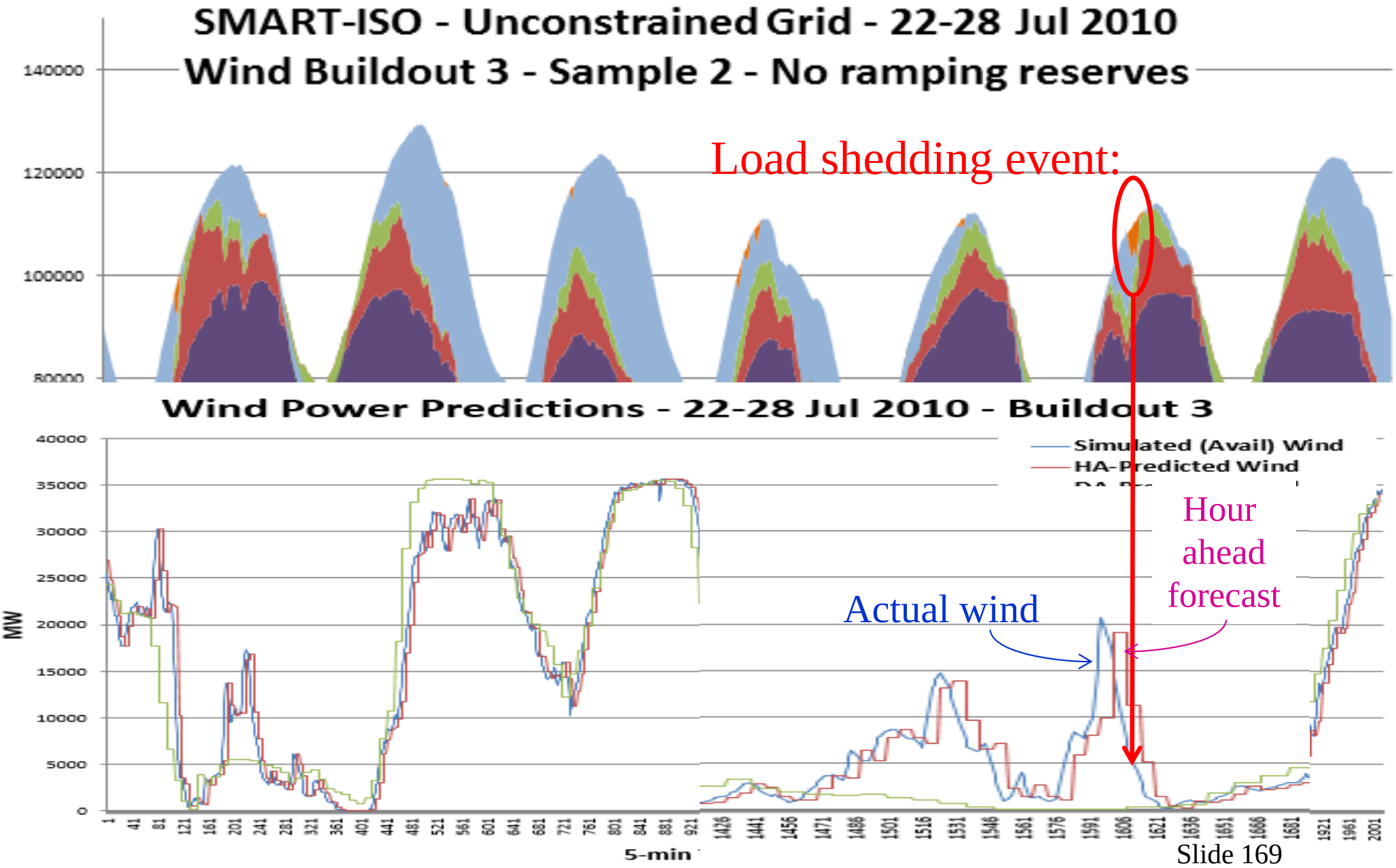
Stochastic lookahead policies



Stochastic lookahead policies



Stochastic lookahead policies



Stochastic unit commitment

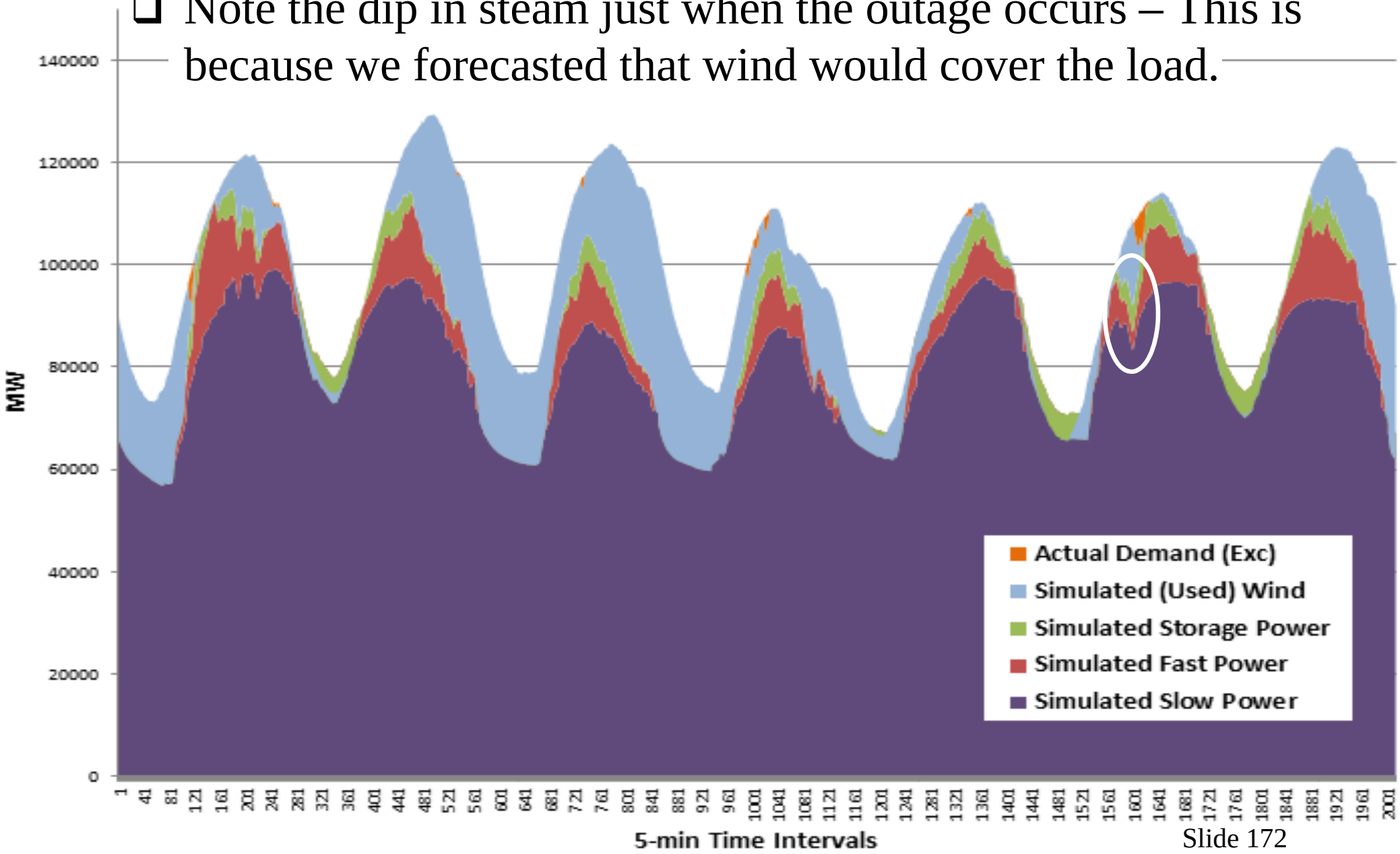
- The outage appears to happen under the following conditions:
  - It is July (we are maxed out on gas turbines – we were unable to schedule more gas reserves than we did).
  - The day-ahead forecast of energy from wind was low (it is hard to see, but the day-ahead forecast of wind energy was virtually zero).
  - The energy from wind was above the forecast, but then dropped suddenly.
  - It was the sudden drop, and the lack of sufficient turbine reserves, that created the load shedding event.
  - Note that it is very important to capture the actual behavior of wind.

Stochastic unit commitment

- What is next:
  - We are going to try to fix the outage by allowing the day-ahead model to see the future.
  - This will fix the problem, by scheduling additional steam, but it does so by scheduling extra steam at *exactly the time that it is needed*.
  - This is the power of the sophisticated tools that are used by the ISOs (which we are using). If you allow a code such as Cplex (or Gurobi) to optimize across hundreds of generators, you will get almost exactly what you need.

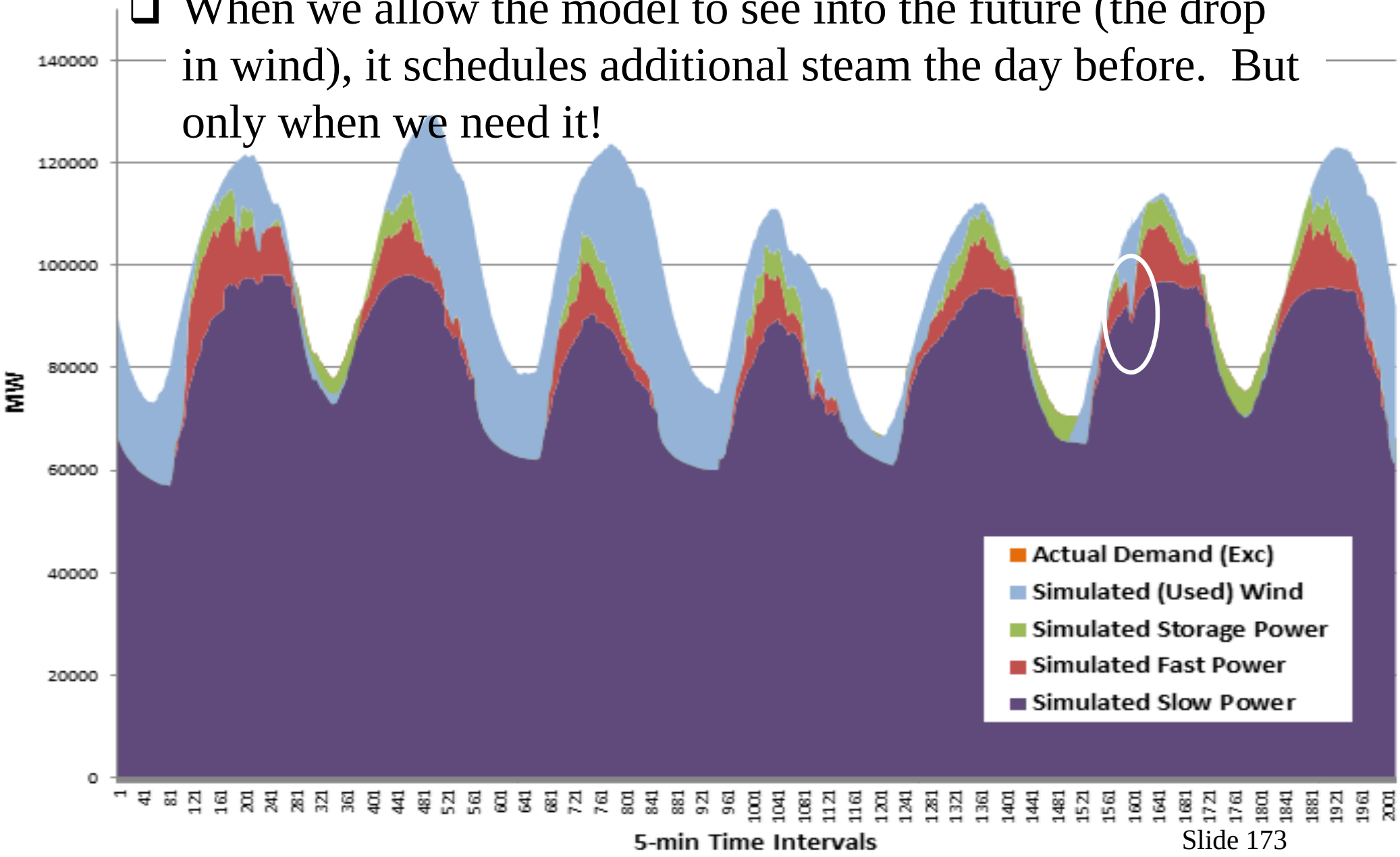
Stochastic lookahead policies

- Note the dip in steam just when the outage occurs – This is because we forecasted that wind would cover the load.



Stochastic lookahead policies

- When we allow the model to see into the future (the drop in wind), it schedules additional steam the day before. But only when we need it!



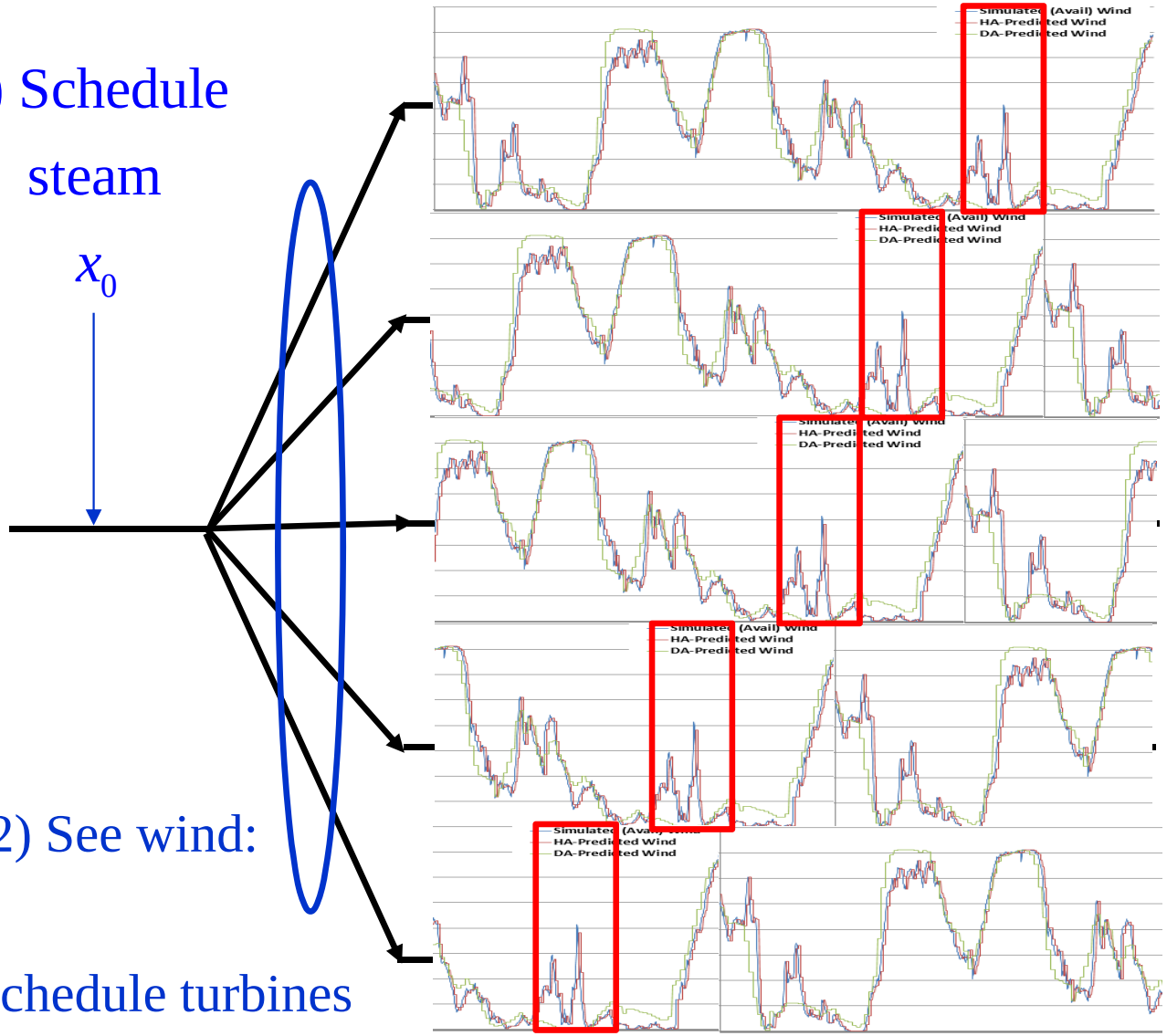
Stochastic lookahead

- Observations:
  - » Our unit commitment model uses advance information to schedule additional steam at just the time when it is needed.
  - » But, these shifts could happen at almost any time!
  - » This means that we need this extra steam at all points in time!
  - » The only way to accomplish this with scenarios is to create scenarios where this wind event happens at *all* points in time. In the next slide, we create a series of scenarios that accomplish this, by shifting the set of events to create the wind dip at every point in time.

Stochastic lookahead policies

- The two-stage approximation

Downward wind shift



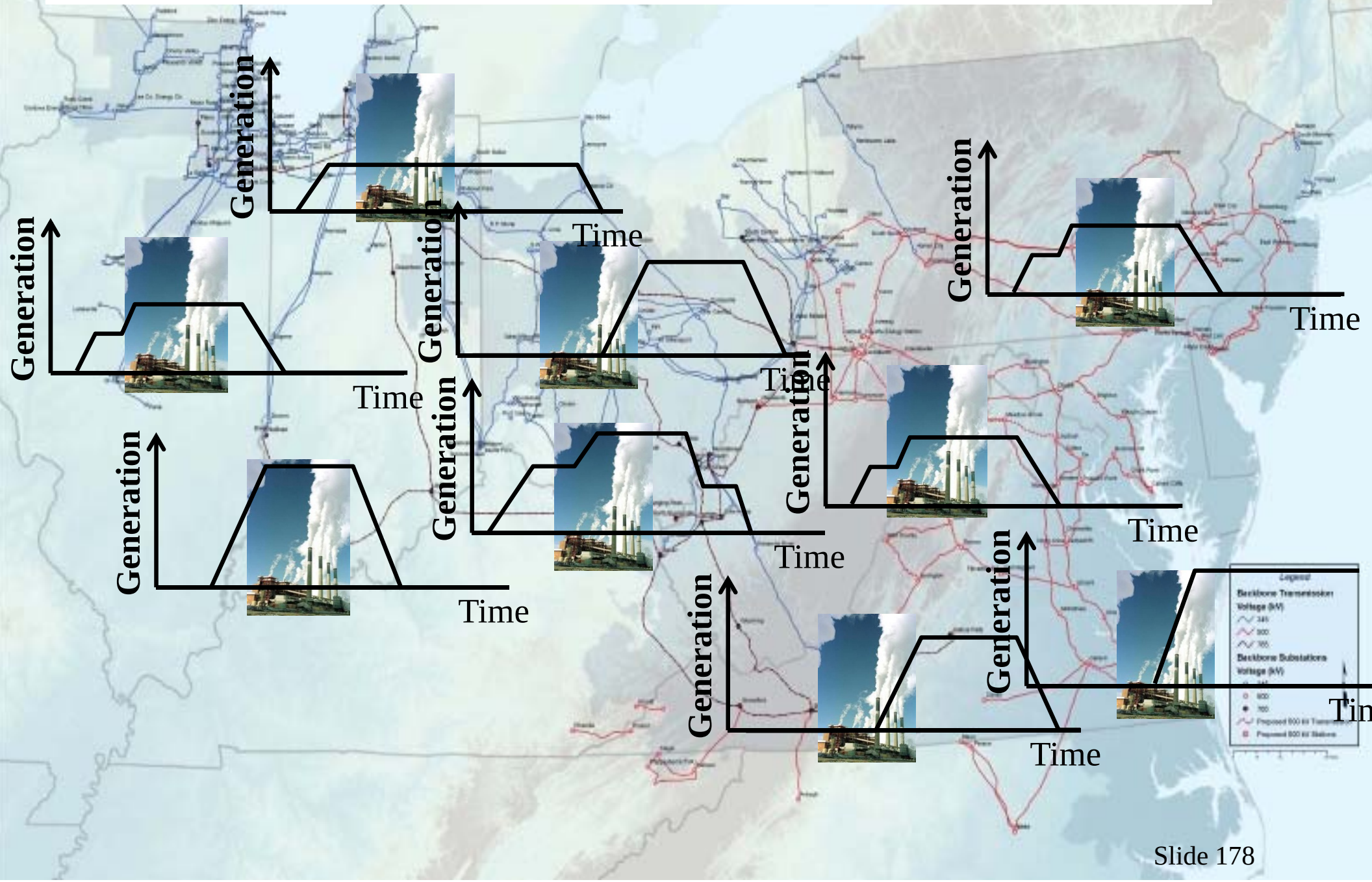
Stochastic unit commitment

- First issue: How many scenarios?
  - It is not enough to create these shifts at every point in time.
  - We have to handle the fact that PJM needs to have proper reserves across the different control zones they use to manage their network.
  - PJM plans for a 1-in-10 years outage rate. We cannot assure this even using 200 scenarios (most discussions of the number of scenarios seems to focus on 10-20 scenarios).
  - To properly test any policy, it would have to be tested on 1000's of possible sample realizations.

Stochastic unit commitment

- Second issue: Dimensionality
  - We believe that a major problem is that the first stage decision (scheduling steam) is very high dimensional (almost 10,000 integer variables just for the steam).
  - Question: can we get a robust solution, scheduling 10,000 integer variables, using 20 (or 200) scenarios?
  - We are talking about operational models, so the solutions have to be implementable. The ISOs have to be sure that the solution works across the entire network, under every conceivable scenario (to the 1-in-10,000 standard).
  - We feel that too little attention has been devoted to analyzing the robustness of the solution at an operational level.

- The decision to schedule steam generation is very high dimensional:

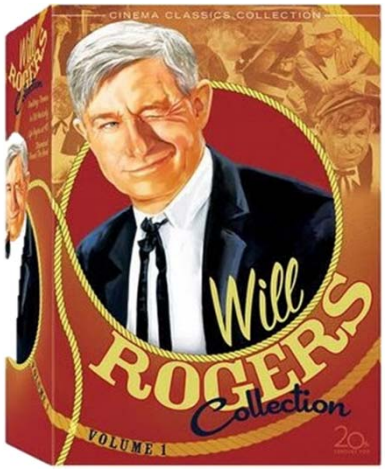


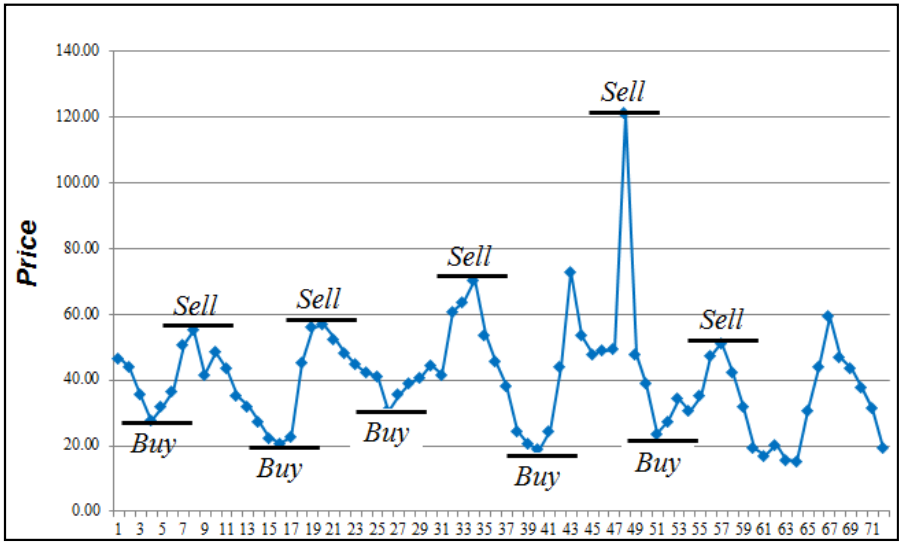
Stochastic unit commitment

- Third issue: The two-stage assumption
  - » Virtually no attention has been given to the errors caused by using a two-stage approximation.
  - » This means that the second stage scheduling of gas turbines are assumed to be able to see the *entire* day.
  - » However, we have found that the most important errors are in the hour-ahead forecast, which is done using "persistence" forecasting, which assumes nothing will change over short time frames.
  - » In our simulations, eliminating the short-term errors would reduce outages by 80 percent.
  - » This is not fixed by going to three-stages (which is dramatically harder than two-stages). It needs a full multistage model.

Stochastic unit commitment

- Third issue: The two-stage assumption
  - » We already saw an example of the effect of the “two stage” approximation. This was our “Will Rogers” policy.
  - » Peeking into the future introduces significant errors for stochastic unit commitment because short-term forecasting of wind is very poor.





Lecture outline

- The robust cost function approximation

The robust CFA for unit commitment

- The robust cost function approximation
  - The ISOs today use simple rules to obtain robust solutions, such as scheduling reserves.
  - These strategies are fairly sophisticated:
    - Spinning
    - Non-spinning
    - Distributed spatially across the regions.
  - This reserve strategy is a form of *robust cost function approximation*. They represent a parametric modification of a base cost policy, and have been tuned over the years in an online setting (called the real world).
  - The ISOs use what would be a hybrid: the robust CFA concept applied to a deterministic lookahead base model.

Four classes of policies

1) Policy function approximations (PFAs)
   - » Lookup tables, rules, parametric functions

2) Cost function approximation (CFAs)
   - » $X^{CFA}(S_t \mid \theta) = \arg\min_{x_t \in \bar{X}_t(\theta)} \bar{C}^{\pi}(S_t, x_t \mid \theta)$

3) Policies based on value function approximations (VFAs)
   - » $X_t^{VFA}(S_t) = \arg\min_{x_t} \left(C(S_t, x_t) + \gamma \bar{V}_t^x \left(S_t^x(S_t, x_t) \right) \right)$

4) Lookahead policies
   - » ***Deterministic lookahead:***
     $$X_t^{LA-D}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})$$
   - » ***Stochastic lookahead (e.g. stochastic trees)***
     $$X_t^{LA-S}(S_t) = \arg\min_{\tilde{x}_{tt}, \tilde{x}_{t,t+1}, \ldots, \tilde{x}_{t,t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{\tilde{\omega} \in \tilde{\Omega}_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} \gamma^{t'-t} C(\tilde{S}_{tt'}(\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega}))$$

A hybrid lookahead-CFA policy

- A deterministic lookahead model
  - » Optimize over all decisions at the same time

$$\min_{\substack{(x_{tt'})_{t'=1,\ldots,24} \\ (yt_{t'})_{t'=1,\ldots,24}}} \sum_{t'=t}^{t+H} C(x_{tt'}, y_{tt'})$$

Steam generation

Gas turbines

  - » In a deterministic model, we mix generators with different notification times:
    - Steam generation is made day-ahead
    - Gas turbines can be planned an hour ahead or less

A hybrid lookahead-CFA policy

- A deterministic lookahead policy
  - This is the policy produced by solving a deterministic lookahead model

$$X_t^{\pi}(S_t) = \min_{\substack{(x_{tt'})_{t'=1,\ldots,24} \\ (yt_{t'})_{t'=1,\ldots,24}}} \sum_{t'=t}^{t+H} C(x_{tt'}, y_{tt'})$$

Steam generation

Gas turbines

  - *No ISO uses a deterministic lookahead model. It would never work, and for this reason they have never used it. They always modify the model to produce a robust solution.*

A hybrid lookahead-CFA policy

- A robust CFA policy
  - The ISOs introduce reserves:

$$X_t^\pi(S_t \mid \theta) = \min_{\substack{(x_{tt'})_{t'=1,\ldots,24} \\ (y_{tt'})_{t'=1,\ldots,24}}} \sum_{t'=t}^{t+H} C(x_{tt'}, y_{tt'})$$

$$x_{t,t'}^{\max} - x_{t,t'} \geq \theta^{up} L_{tt'} \qquad \text{Up-ramping reserve}$$

$$x_{t,t'} - x_{t,t'}^{\max} \geq \theta^{down} L_{tt'} \qquad \text{Down-ramping reserve}$$

  - This modification is a form of parametric function (a parametric cost function approximation). It has to be tuned to produce a robust policy.

A hybrid lookahead-CFA policy

- A robust CFA policy
  - The ISOs introduce reserves:

$$X_t^\pi(S_t \mid \theta) = \min_{\substack{(x_{tt'})_{t'=1,\ldots,24} \\ (y_{tt'})_{t'=1,\ldots,24}}} \sum_{t'=t}^{t+H} C(x_{tt'}, y_{tt'})$$

$$x_{t,t'}^{\max} - x_{t,t'} \geq \theta^{up} L_{tt'} \qquad \text{Up-ramping reserve}$$

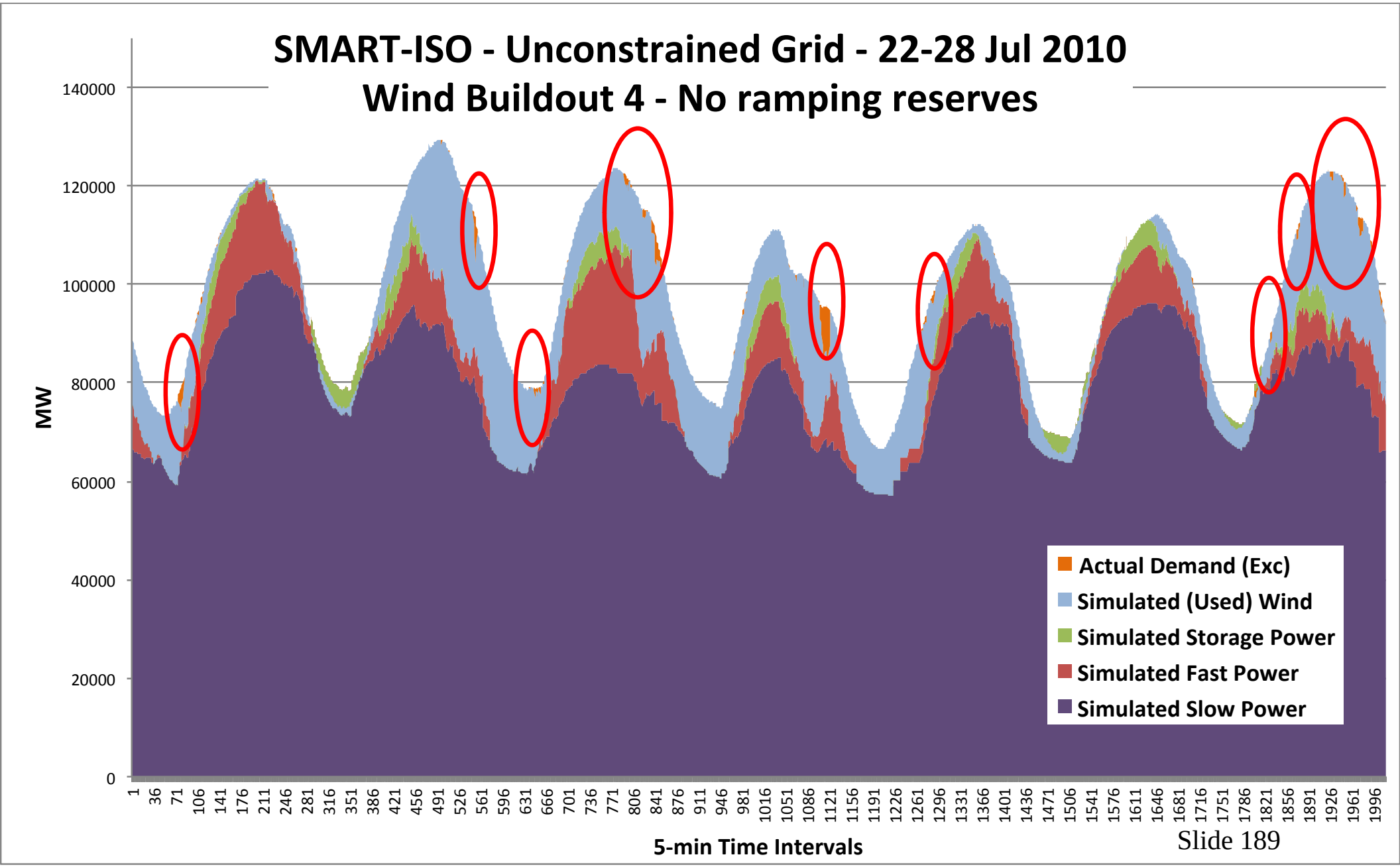
$$x_{t,t'} - x_{t,t'}^{\max} \geq \theta^{down} L_{tt'} \qquad \text{Down-ramping reserve}$$

  - It is easy to tune this policy when there are only two parameters $\theta = \left(\theta^{up}, \theta^{down}\right)$
  - But we might want the parameters to depend on information such as forecasts.

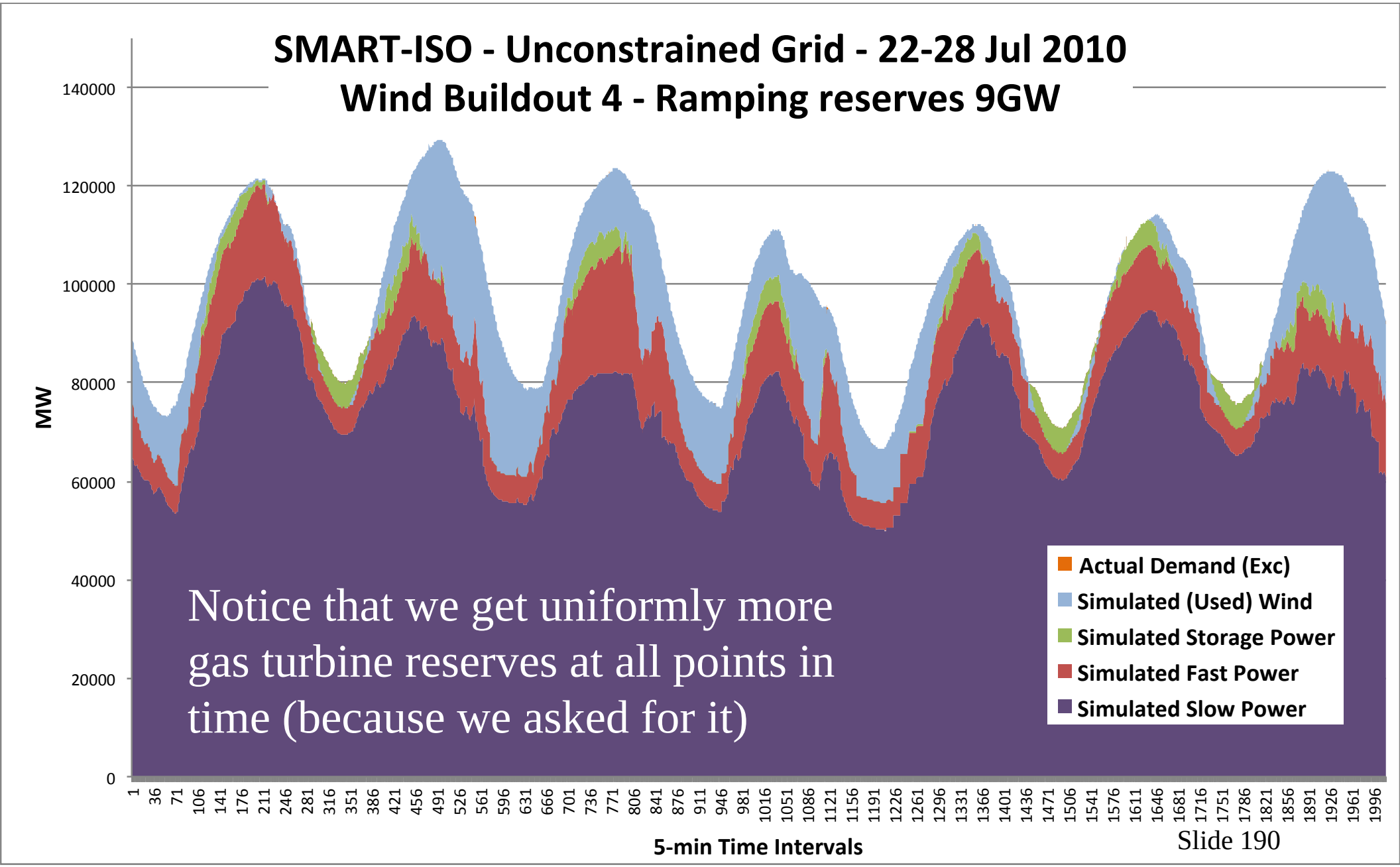
Stochastic unit commitment

- Properties of a parametric cost function approximation
  - » It allows us to use domain expertise to control the structure of the solution
  - » We can force the model to schedule extra reserve:
    - At all points in time (or just during the day)
    - Allocated across regions
  - » This is all accomplished without using a multiscenario model.
  - » The next slides illustrate the solution before the reserve policy is imposed, and after. Pay attention to the change pattern of gas turbines (in maroon).

SMART-ISO: Offshore wind study



SMART-ISO: Offshore wind study



Comparison of policies

- Robust cost function approximation
  - » Requires solving modified deterministic lookahead model
  - » Parameters have to be tuned, ideally with a highly realistic *base model* (“simulator”)
  - » User captures domain knowledge when specifying the structure of the CFA
- Stochastic lookahead
  - » Requires solving approximate stochastic lookahead model
  - » Tuning is generally not done (although in theory possible, but it is very expensive)
  - » No need for user-specified parametric approximation (and no ability to incorporate domain knowledge)

Stochastic unit commitment

- So how should we tune our robust cost function approximation? Two strategies:
  - » Offline learning – Inside a simulator such as SMART-ISO
    - Using a simulator, we can tune a policy under conditions that do not now exist.
    - We can also test changes in the implementation of the policy.
    - But these all depend on the accuracy of the simulator, and the events we choose to simulate.
  - » Online learning – Tune the policy in the real world
    - This is what the ISOs do now.
    - They do not wait for outages – they wait for operating margins to be compromised.
    - PJM also uses an elaborate “perfect dispatch” process (posterior optimization) to help tune policies.

Lecture outline

- Off-shore wind study

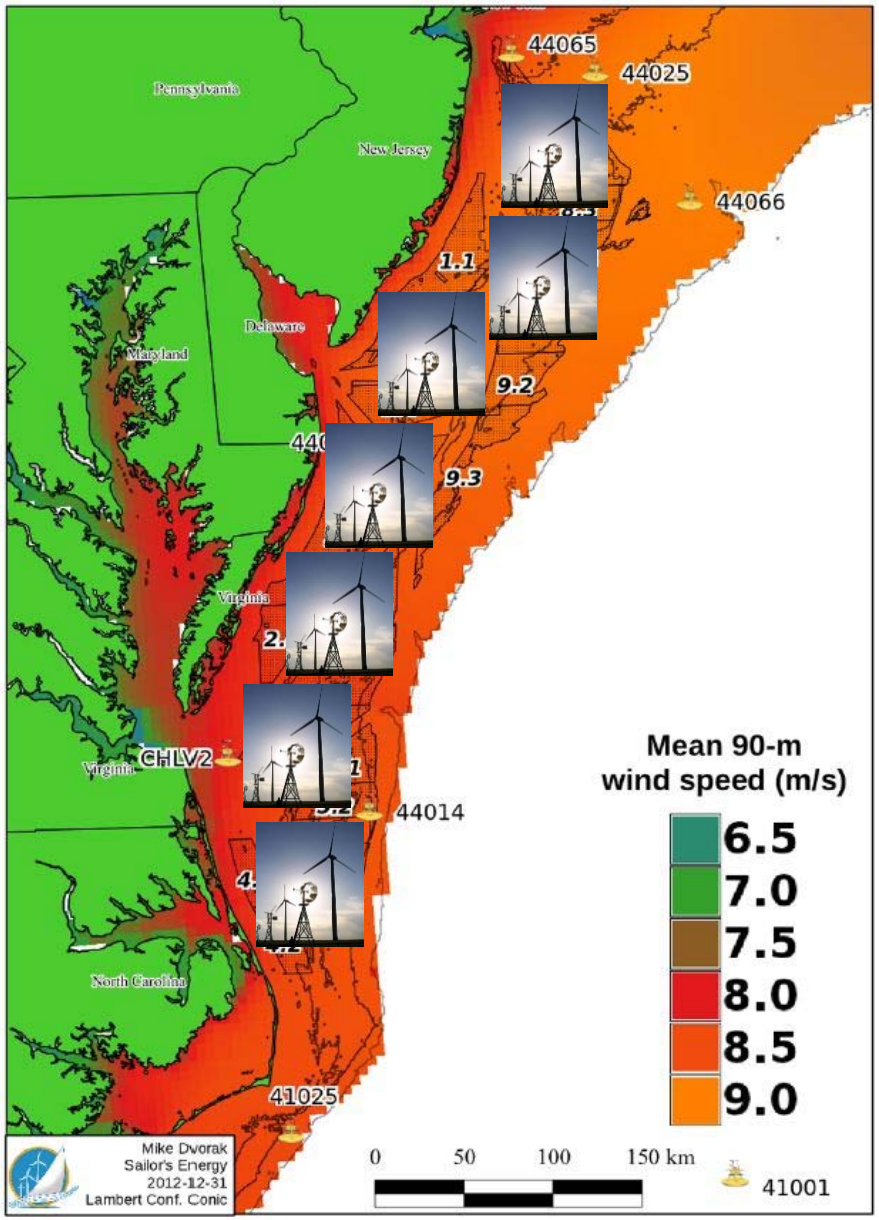
SMART-ISO: Offshore wind study

- Does it work?
  - » We are now going to show that our robust CFA can handle up to 28 GW of off-shore wind capacity by just tuning fast ramping reserves.
  - » Our study of off-shore wind for the mid-Atlantic states used SMART-ISO, a carefully calibrated simulator (“base model”) of PJM.
  - » We developed very realistic sample paths of wind, and simulated these over all four seasons.

SMART-ISO: Offshore wind study

- Mid-Atlantic Offshore Wind Integration and Transmission Study (U. Delaware & partners, funded by DOE)

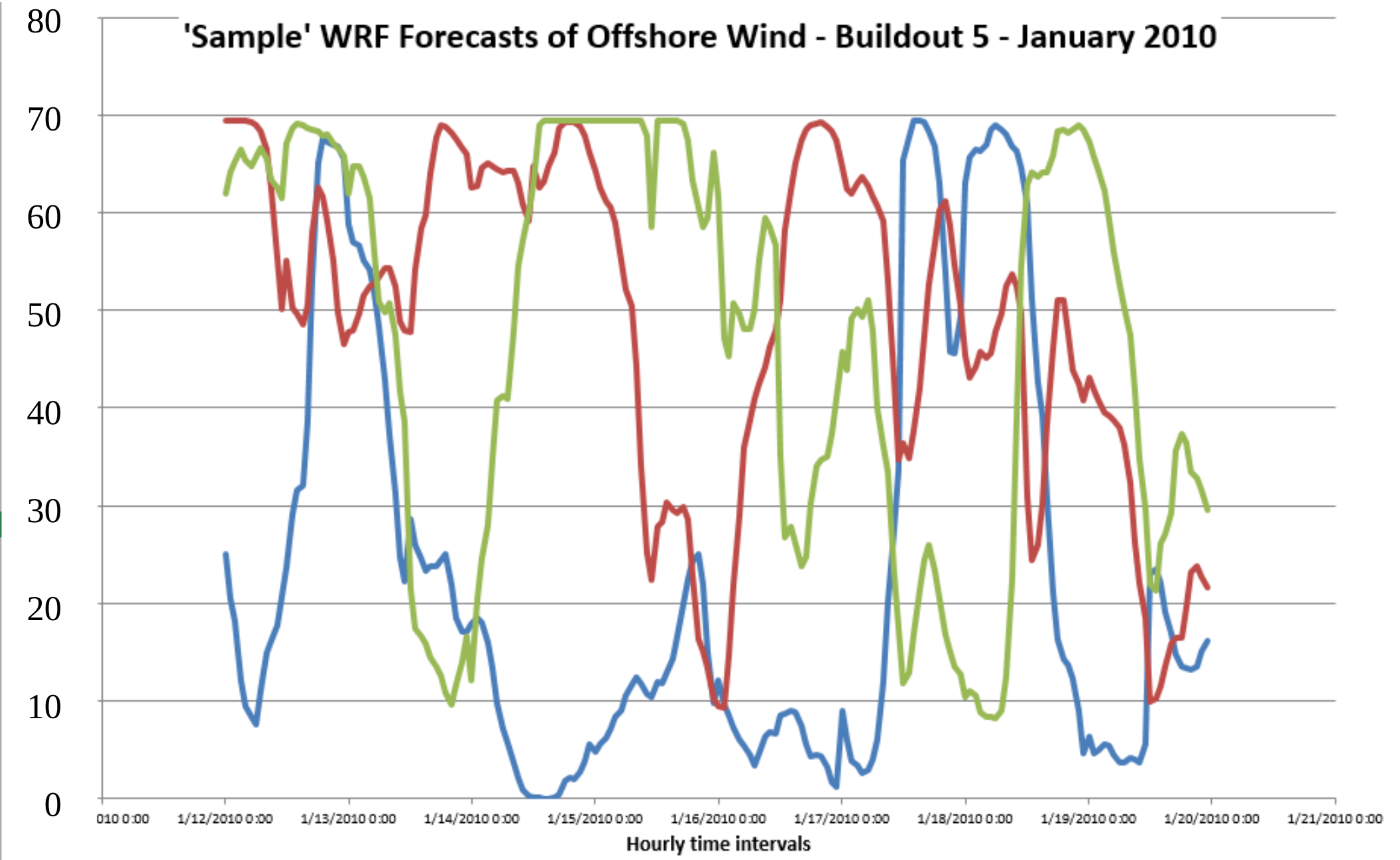
- 29 offshore sub-blocks in 5 build-out scenarios:
  - » 1: 8 GW
  - » 2: 28 GW
  - » 3: 40 GW
  - » 4: 55 GW
  - » 5: 78 GW



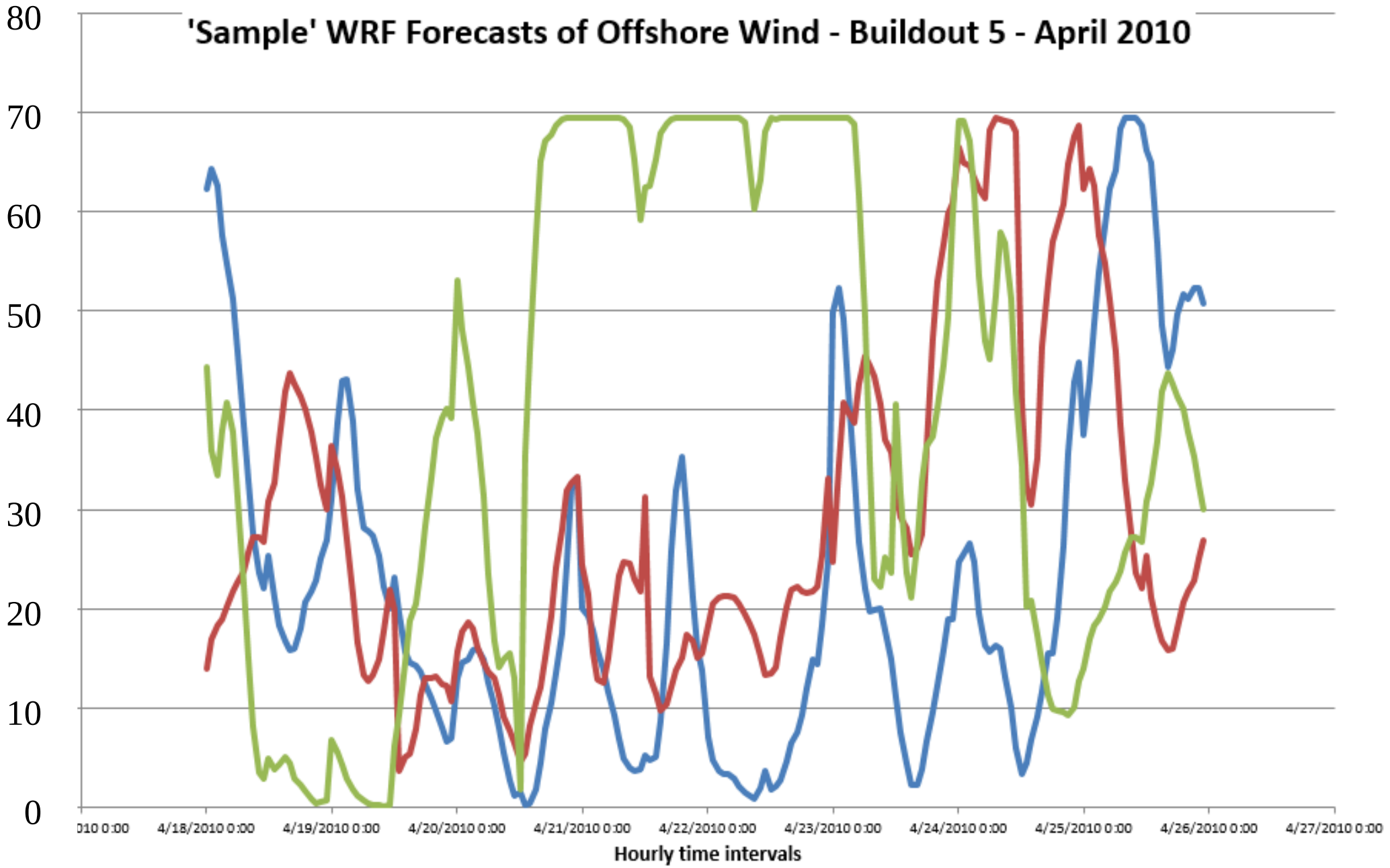
Generating wind sample paths

- Methodology for creating off-shore wind samples
  - Use actual forecasts from on-shore wind to develop a stochastic error model
  - Generate sample paths of wind by sampling errors from on-shore stochastic error model
  - Repeat this for base case and five buildout levels
- Our sample paths are based on:
  - Four months: January, April, July and October
  - WRF forecasts from three weeks each month
  - Seven sample paths generated around each forecast, creating a total of 84 sample paths.
  - The next four screens show the WRF forecasts for each month, followed by a screen with one forecast, and the seven sample paths generated around that forecast.

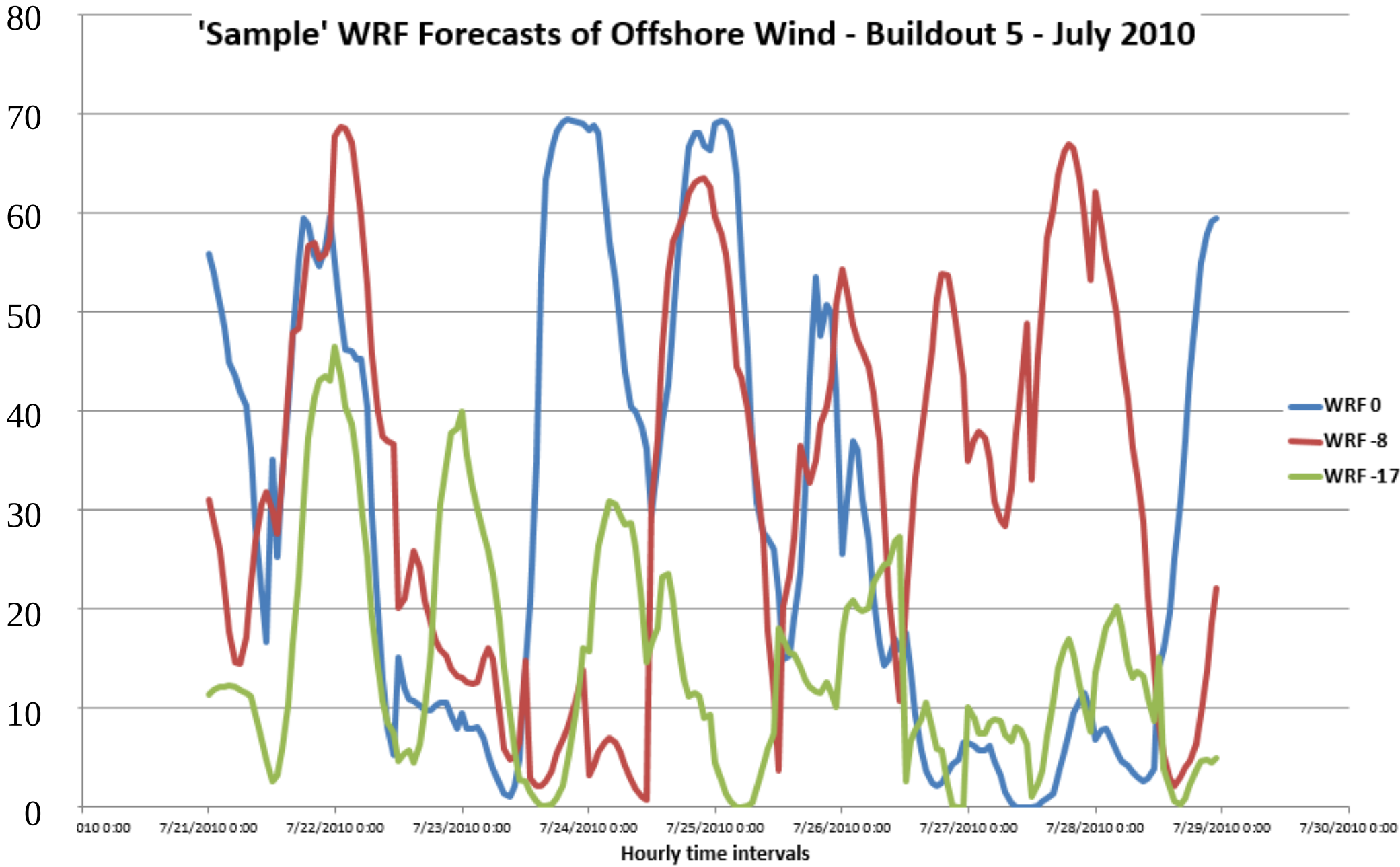
Wind forecast samples



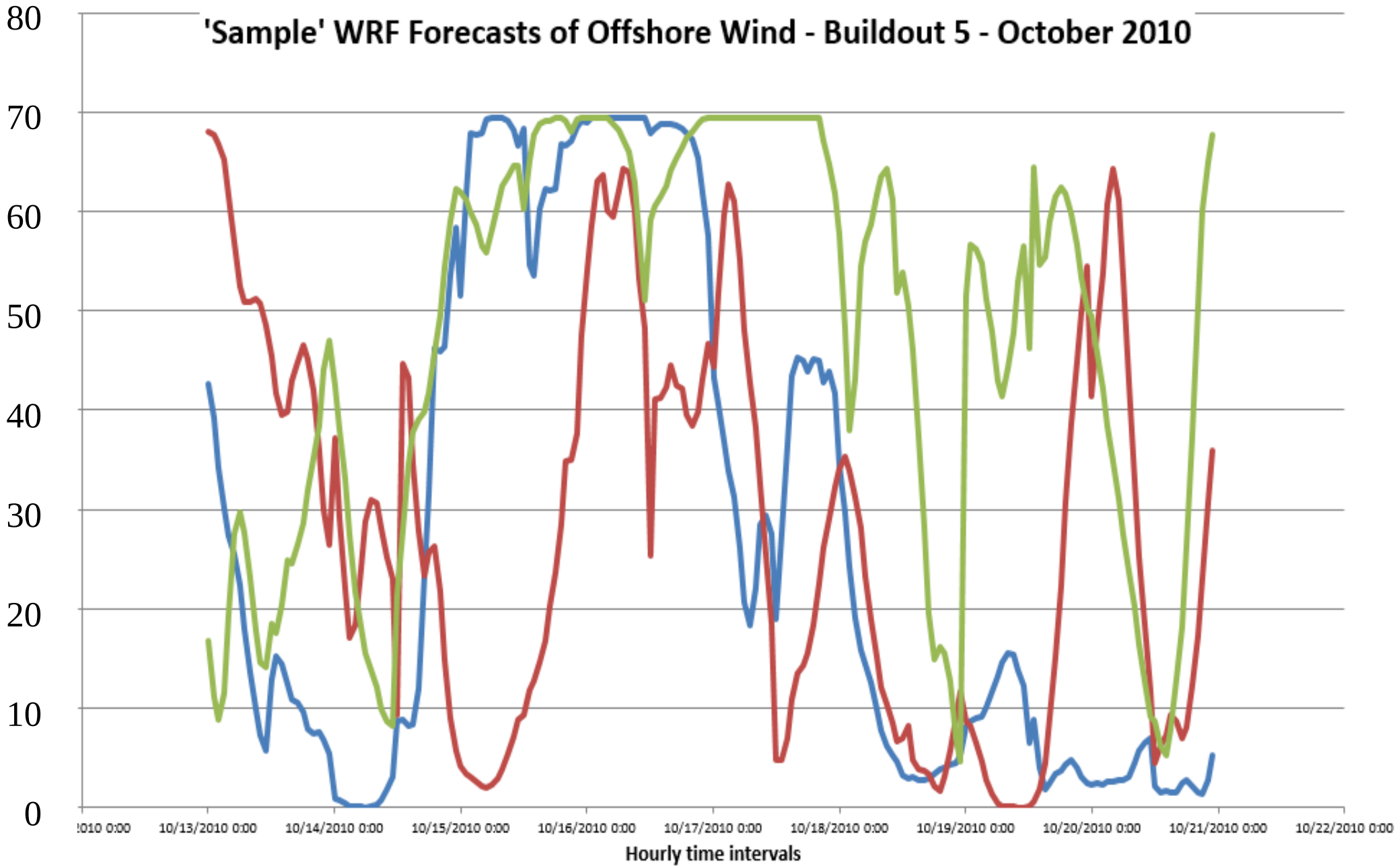
Wind forecast samples



Wind forecast samples

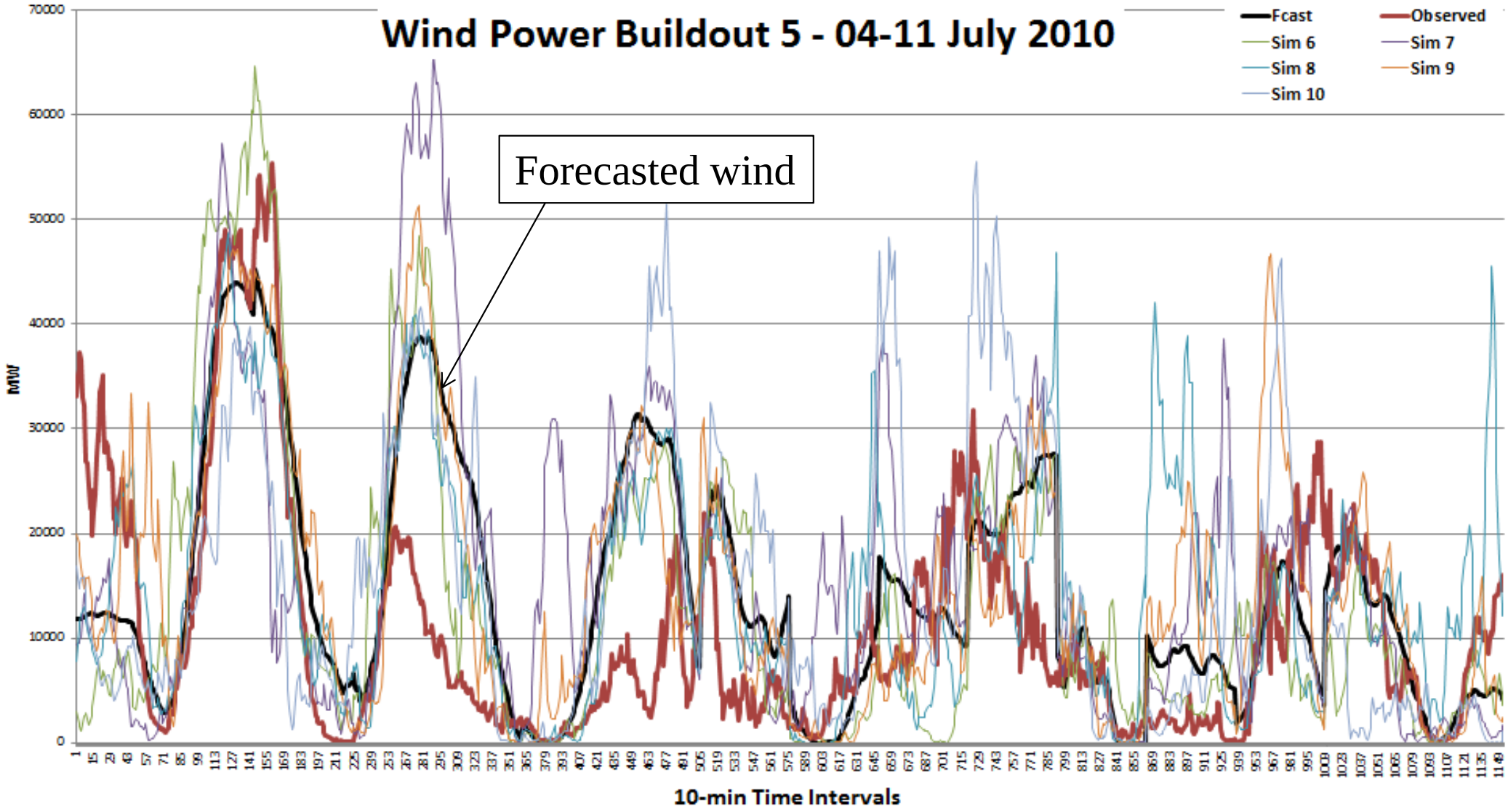


Wind forecast samples



Simulating offshore wind

- Offshore wind – Buildout level 5

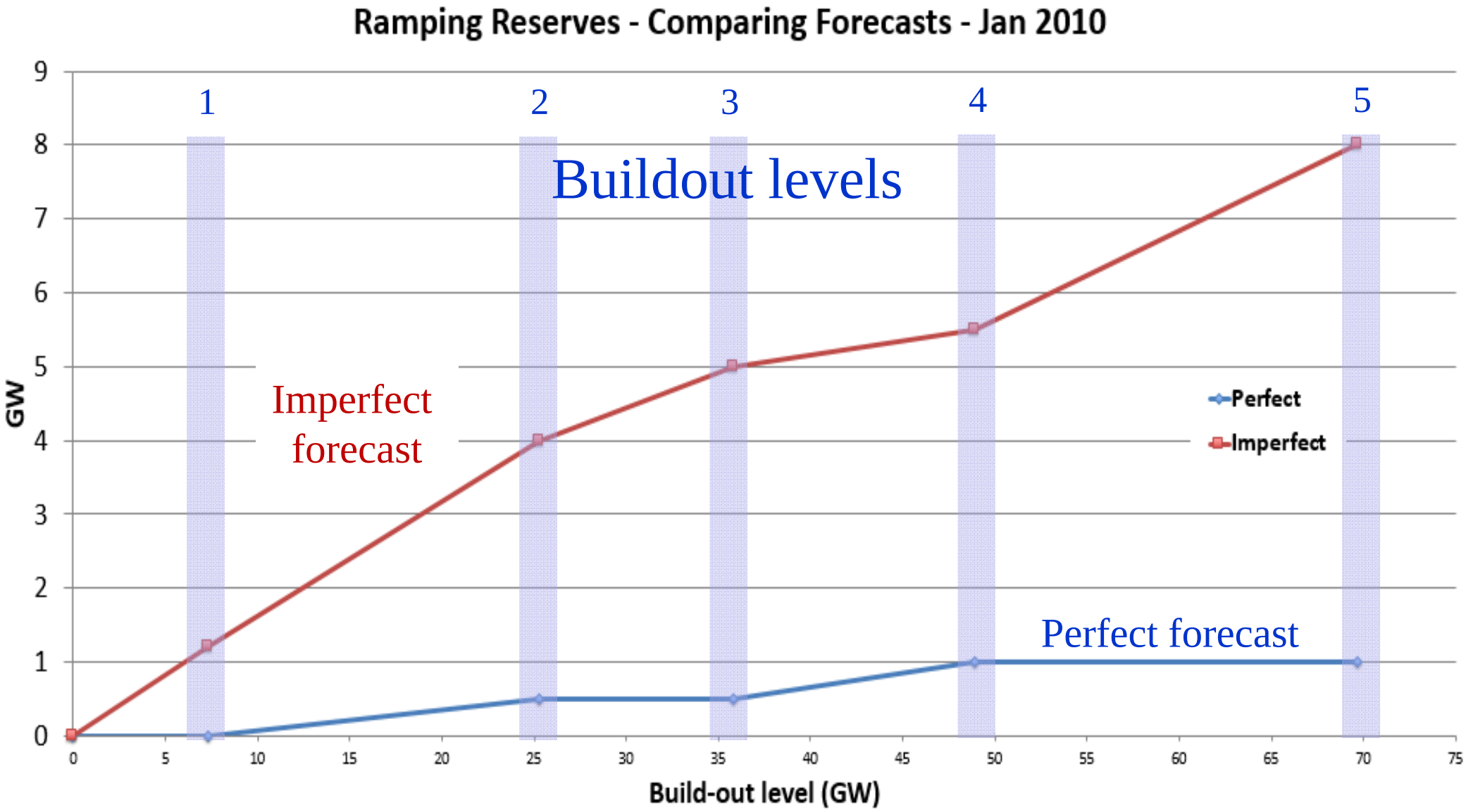


SMART-ISO: Offshore wind study

- Simulations:
  - » For each month, we tuned the reserve policy to try to avoid any load shedding for any of the 21 sample paths for that month.
  - » The next slide shows these reserve levels for each of the five build levels, using both a perfect forecast and an imperfect forecast.
  - » We then show the percentage of sample paths that produced an outage for each build level using:
    - The base PJM reserve policy (1.3 GW reserve)
    - The adjusted reserve to avoid load shedding
    - Perfect forecasts
  - » We were able to avoid all load shedding for January, April and October using suitably tuned reserves, for all build levels (that is, up to 78 GW).
  - » For July, we started seeing outages at 40GW.

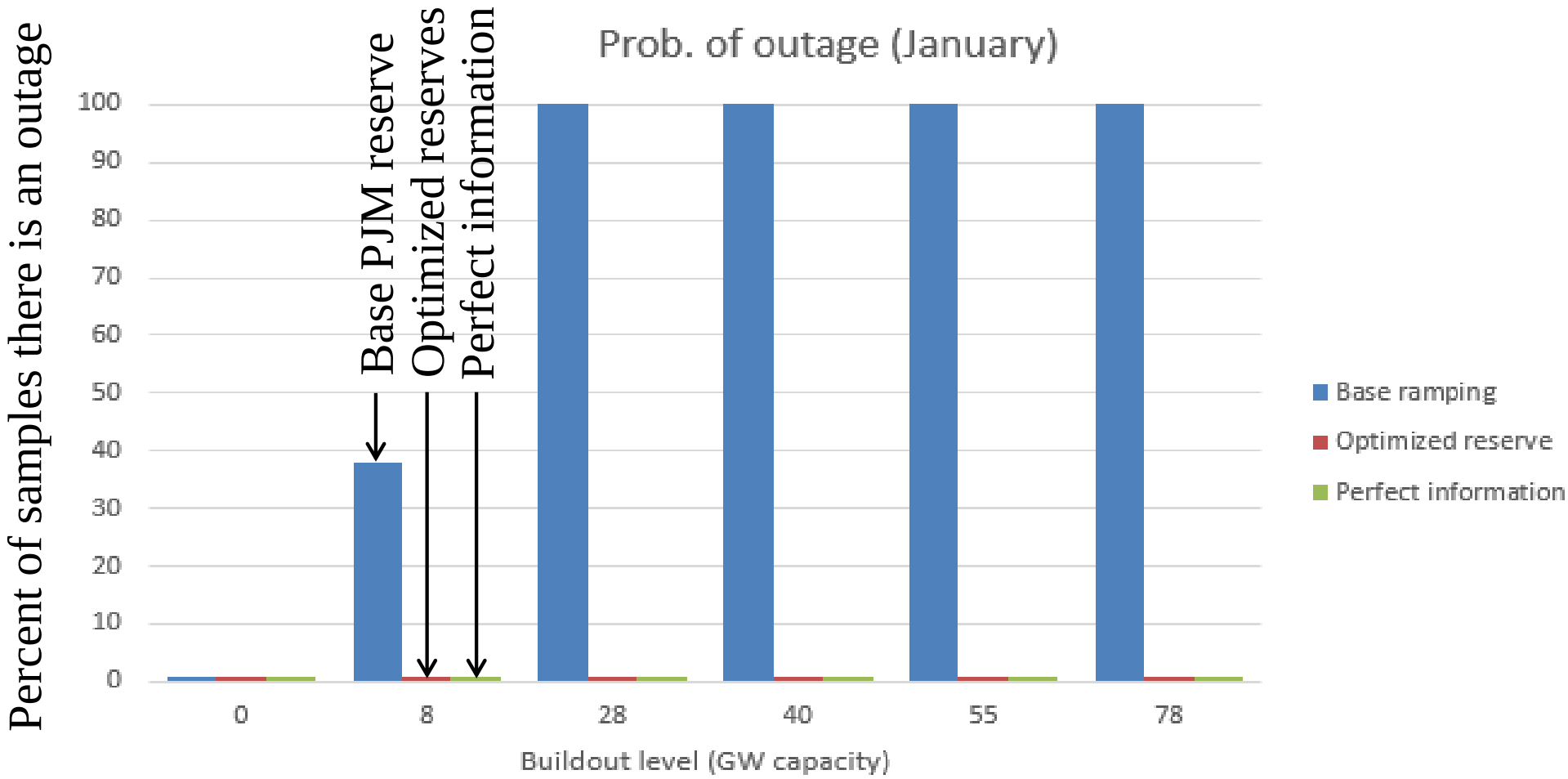
SMART-ISO: Offshore wind study

- Ramping reserves for January, 2010.



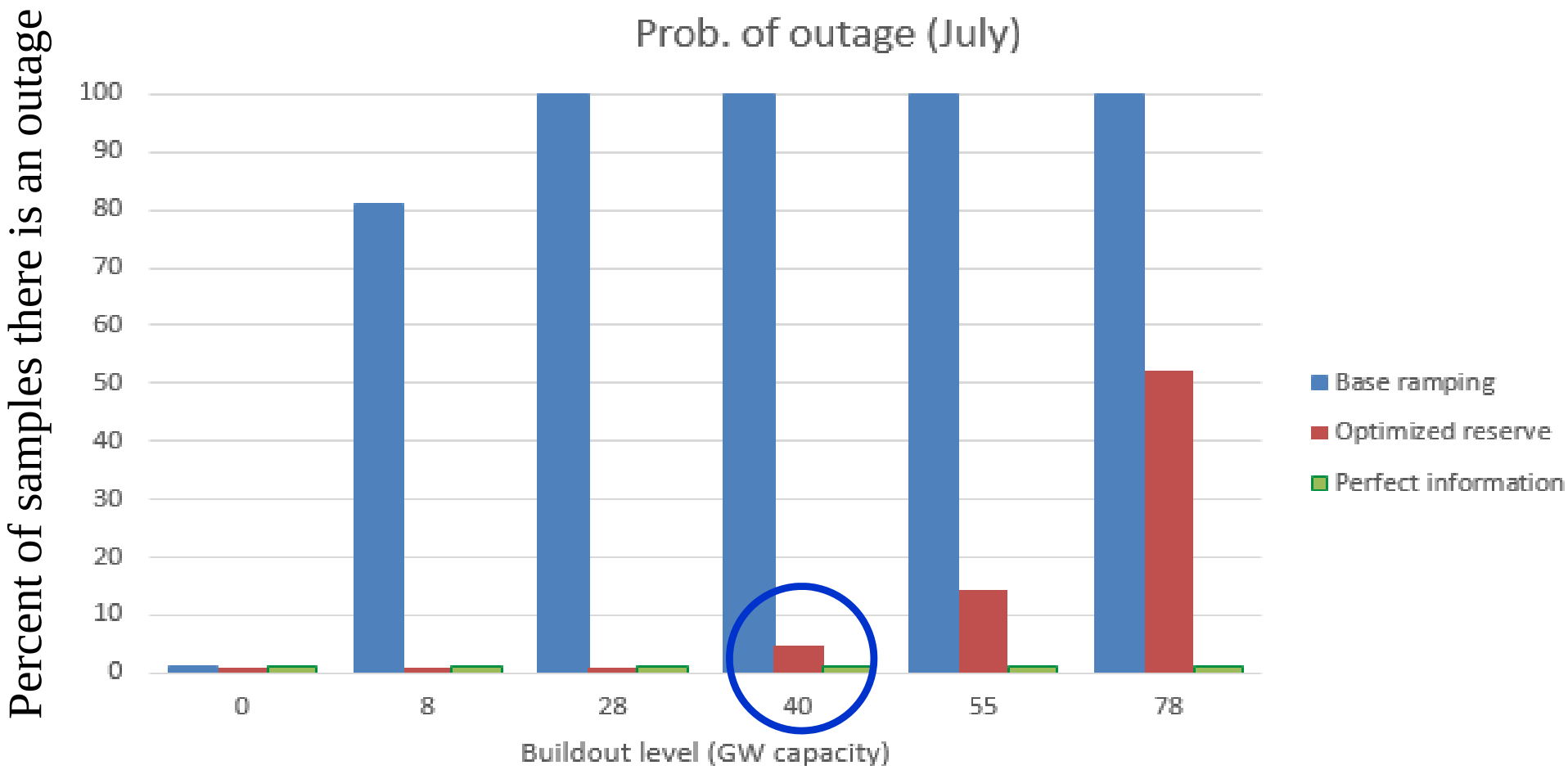
SMART-ISO: Offshore wind study

- Outage probabilities over 21 sample paths for January, April and October



SMART-ISO: Offshore wind study

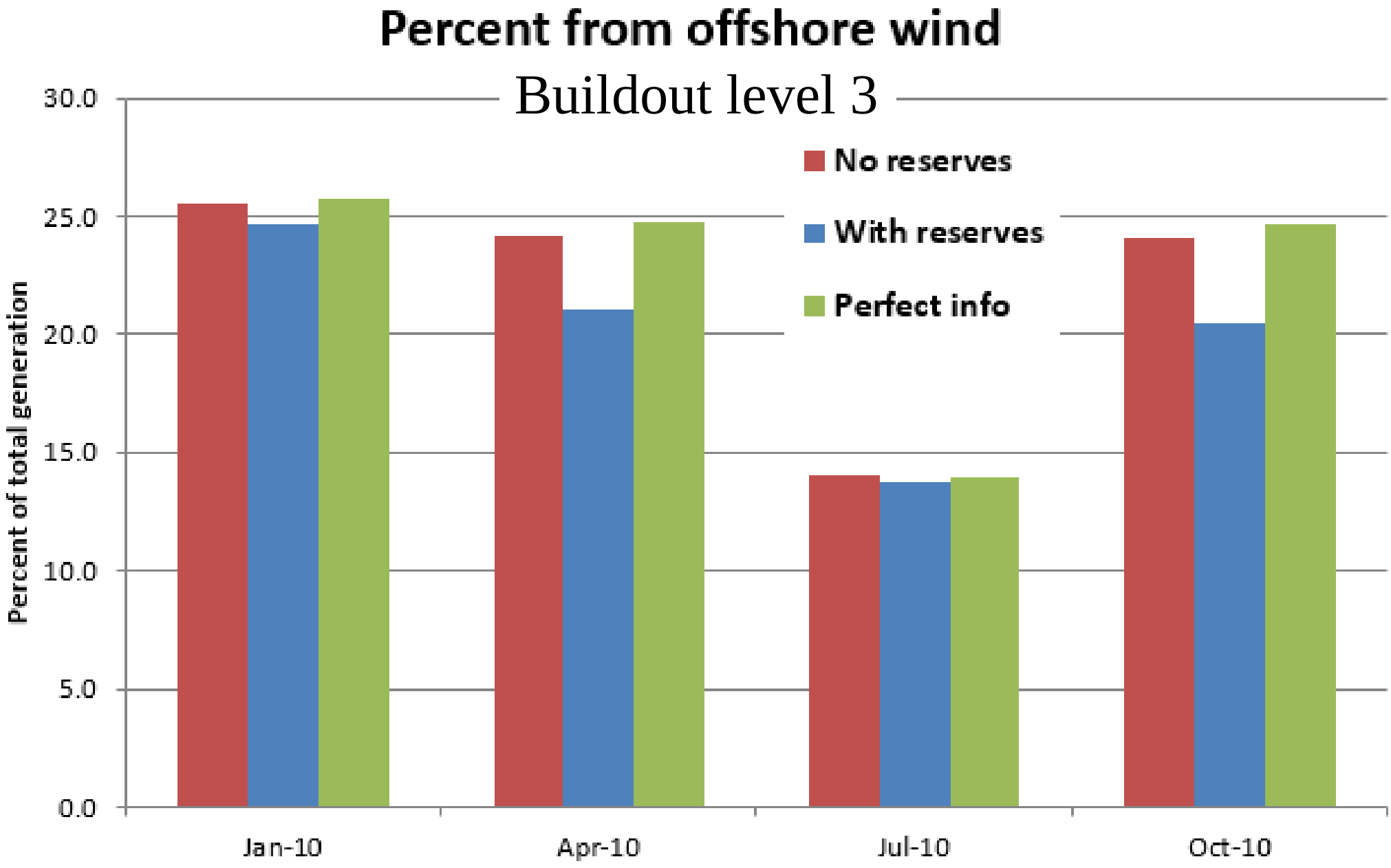
- Outage probabilities over 21 sample paths for July
  - » There is a load shedding event during one sample path at buildout level 3 (40GW of capacity)



SMART-ISO: Offshore wind study

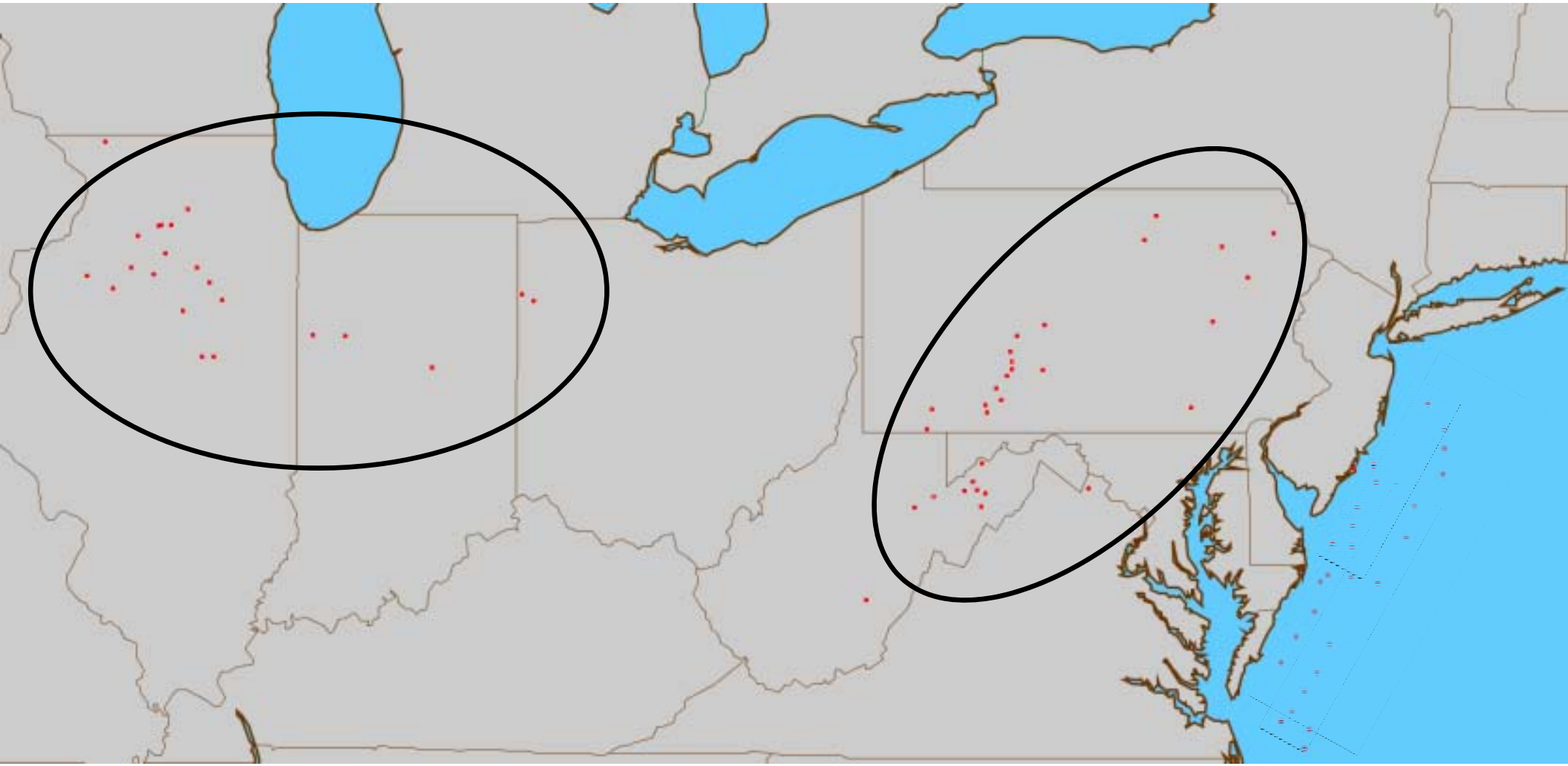
- Notes:
  - The outage at the 40 GW level occurs because we ran out of gas turbines for fast ramping reserves
  - This can be fixed by simply purchasing more gas turbines…
  - … or by scheduling extra steam reserve in the day-ahead market (this is how the problem is fixed with perfect information).
  - We believe we can cover wind energy from over 40GW of generating capacity using nothing more than standard reserve policies.
  - This observation simply means this level of wind energy is feasible, not that it is economical.

SMART-ISO: Offshore wind study



SMART-ISO – Onshore wind study

- Wind farms on the PJM system

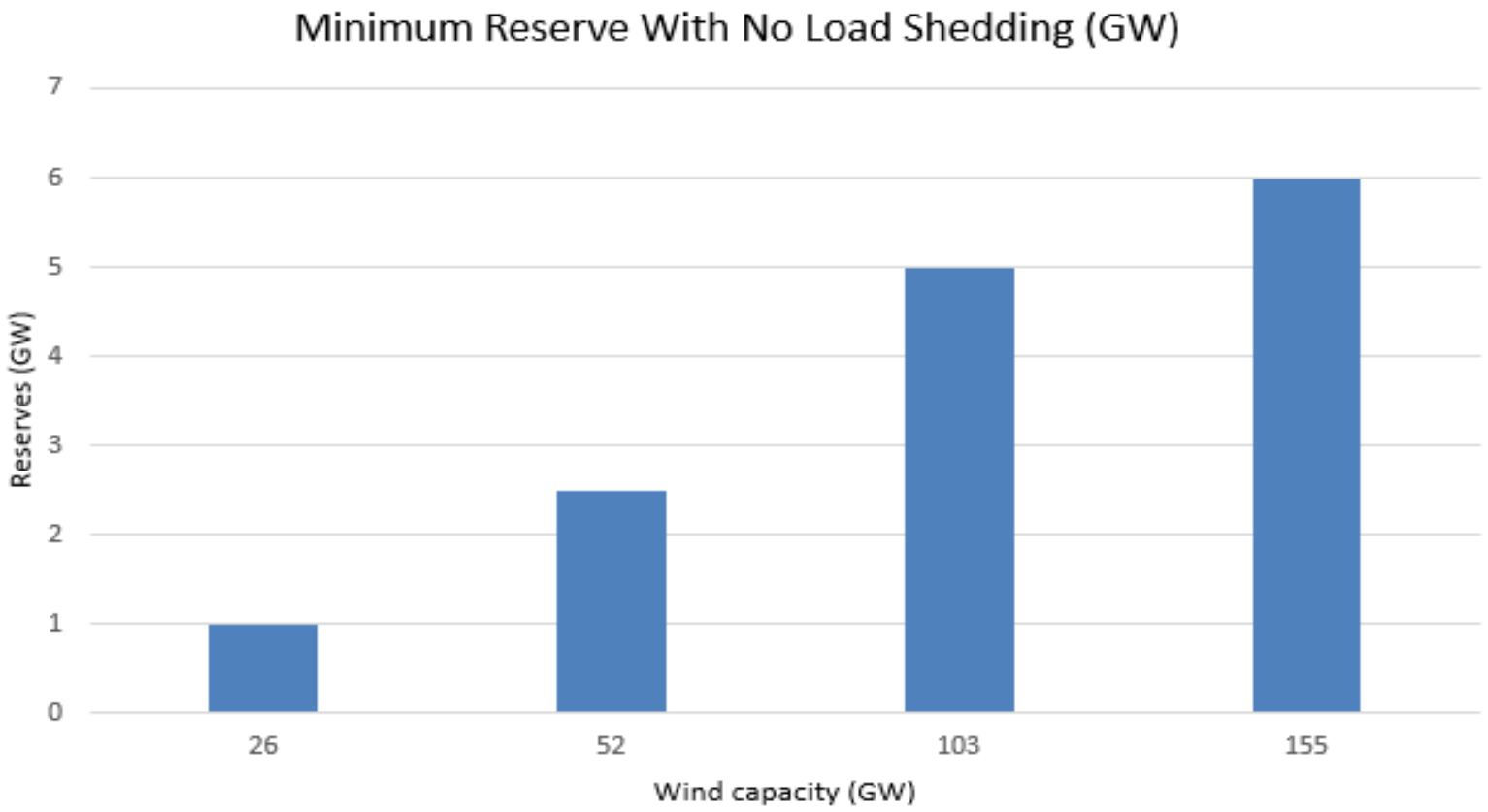


SMART-ISO – Onshore wind study

- Approach:
  - We used actual wind and wind forecasts from the PJM forecasting vendor spanning four seasons, nine weeks per season.
  - We did not do any stochastic modeling of wind…
  - … but we ignored any smoothing from using wind farms that are more spatially distributed.
  - We used the following buildout levels:
    - 6.5GW – Current installation capacity
    - 26GW
    - 52GW
    - 103GW
    - 155GW – Highest level where we could increase reserves to avoid any load shedding over entire 36 week simulation.

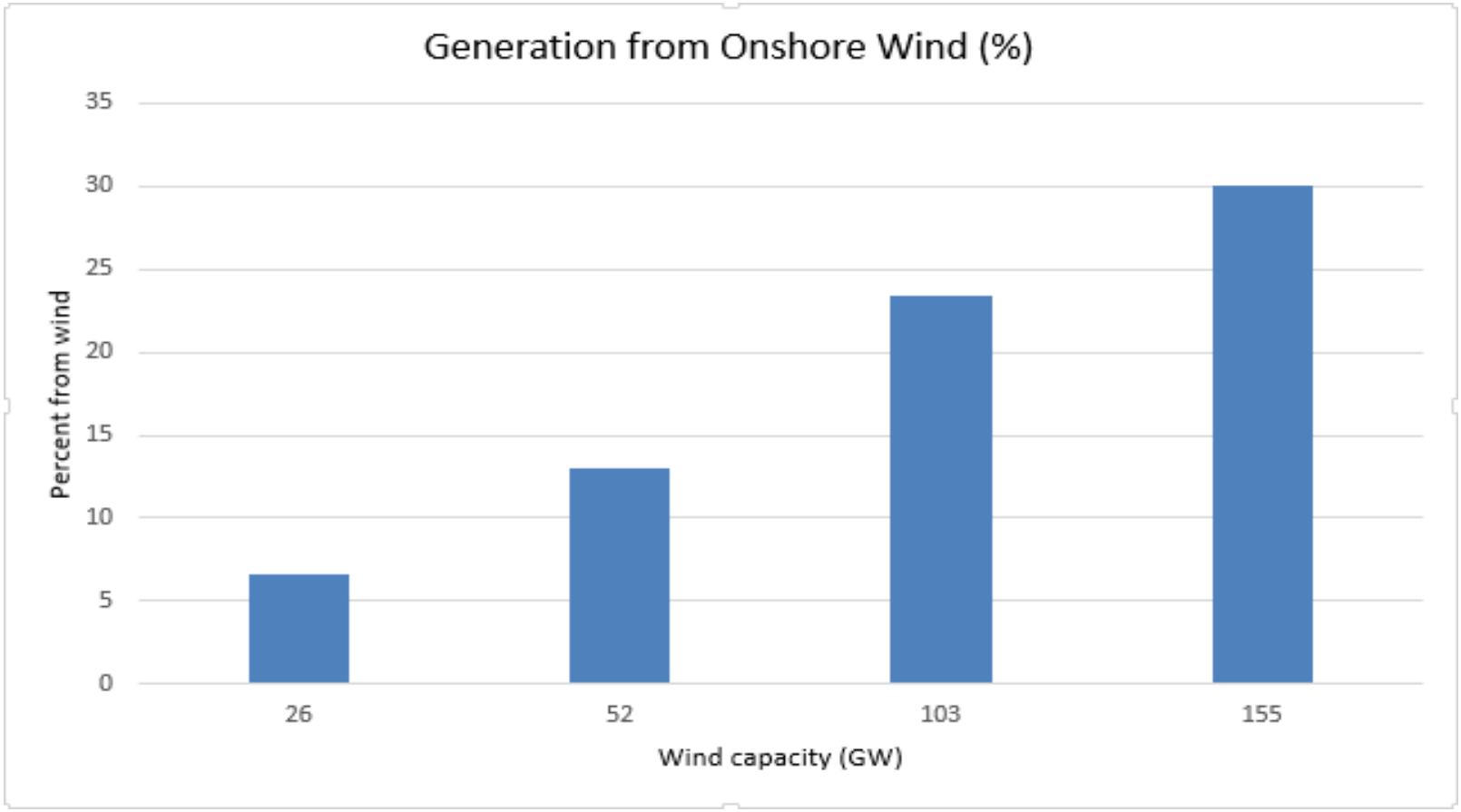
SMART-ISO – Onshore wind study

- Reserve levels to avoid load shedding



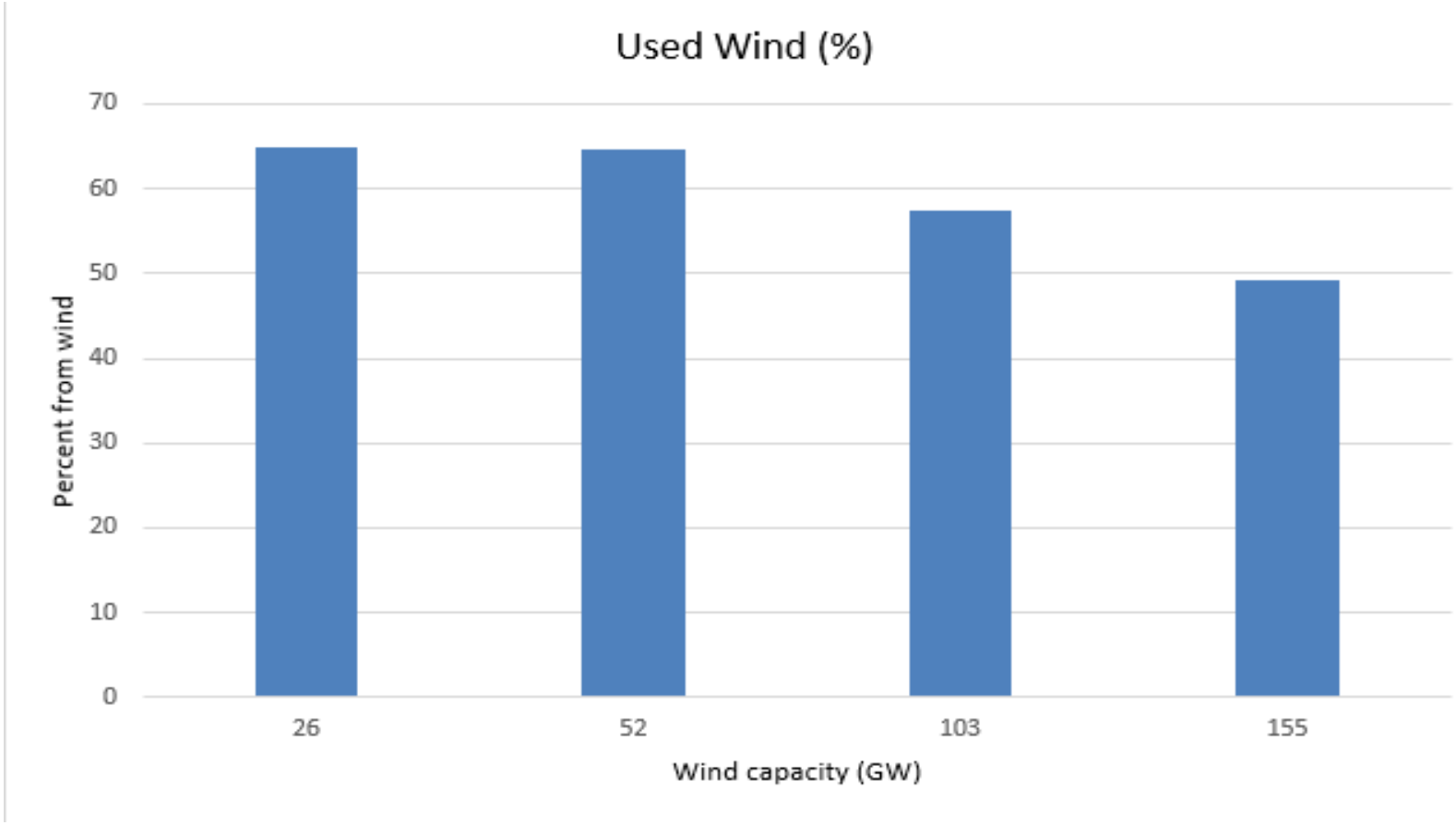
SMART-ISO – Onshore wind study

- Percent generation from onshore wind (all 36 weeks)



SMART-ISO – Onshore wind study

- Percent wind energy that is used (all 36 weeks)



SMART-ISO – Onshore wind study

- Remarks:
  - We were able to generate 30 percent of our electricity from onshore wind, without a single load shedding event over 36 weeks.
  - This was accomplished using only 6GW of fast ramping reserves.
  - … but it did require 155GW of installed capacity (might be expensive!).

Lecture outline

- Perspectives on robust policies

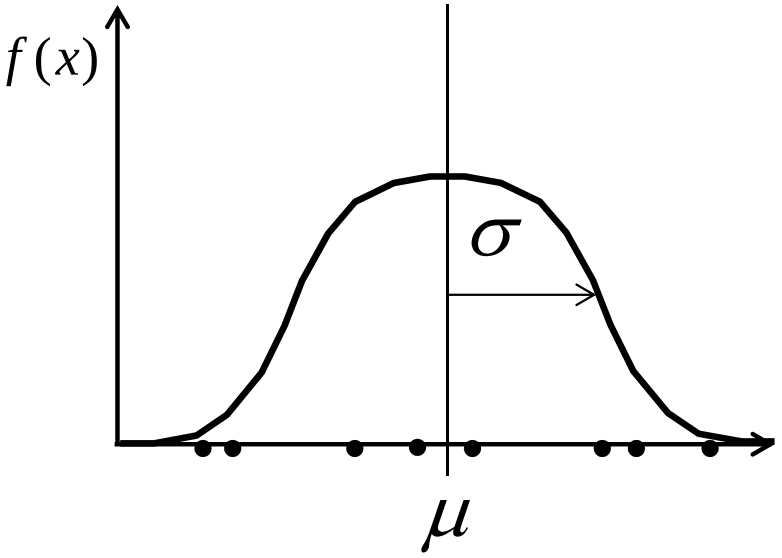
Perspectives

- Scenario trees vs cost function approximation
  - Scenario trees are a form of *nonparametric* representation of the probability model in the future.
  - The robust cost function approximation is a *parametric function*. It has to be specified by a domain expert, and tuned in a *stochastic base model* (or the real world).
  - The next slides illustrate both of these concepts.

Approximating distributions

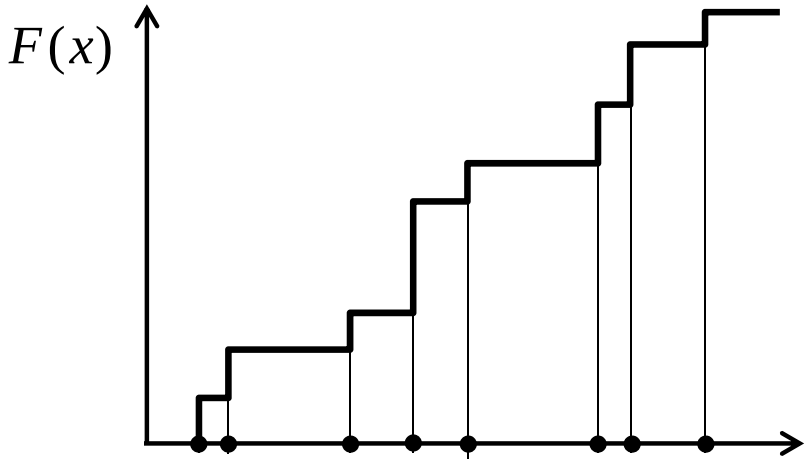
- Stochastic lookahead model
  - » Uses approximation of the *information process* in the lookahead model

Parametric distribution (pdf)



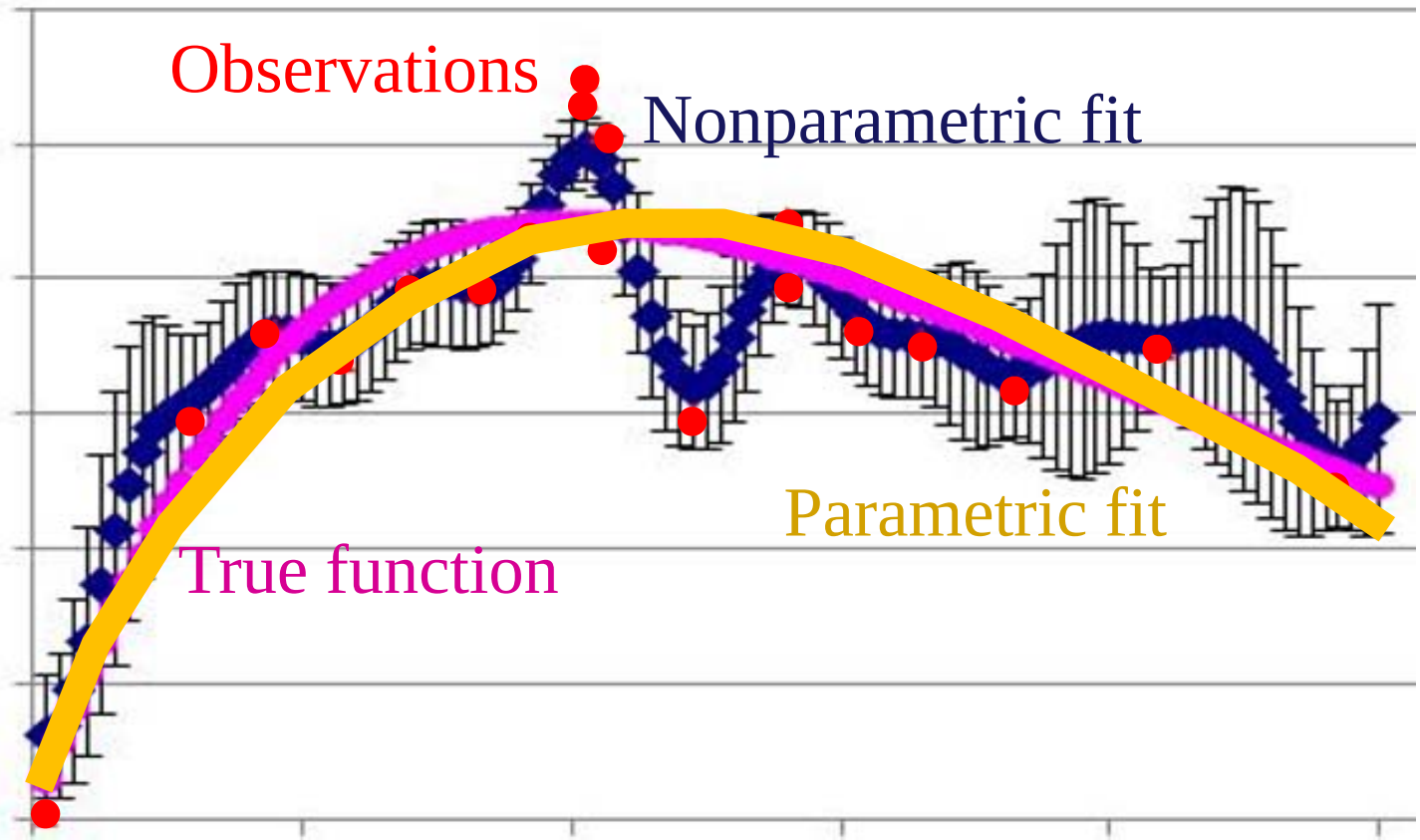
$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{(x-\mu)^2}{\sigma^2}\right)}$$

Nonparametric distribution (cdf)



Approximating a function

- Parametric vs. nonparametric



» We can use our understanding of the function to impose a shape.

Perspectives

- Strengths of nonparametric models
  - » You do not have to specify the structure.
  - » You let the data do all the work.

- Weakness of nonparametric models
  - » You are not *allowed* to specify the structure, even if you know it.
  - » Nonparametrics require a *lot* of data to get the structure right.

Perspectives

- Strengths of parametric models
  - You can use domain knowledge to specify the structure.
  - Exploiting the structure means fitting only a few parameters, which requires much less data.

- Weakness of nonparametric models
  - You have to specify the structure, which means you are limited to the strategies that you can think of.
  - Parametric policies are great when they are parameterized by a small number of parameters, but there is a strong desire to make policies a function of the state of the system, which complicates things. This is illustrated in the next slides.

Designing a policy

- A robust lookahead-CFA policy
  - A basic robust CFA might be parametrized by a small number of parameters:

$$F_t(S_t|\theta) = \min_{(x_{tt'})_{t'=1,\ldots,24}} \mathbb{E} \sum_{t'=t}^{t+48} C(x_{tt'}, Y^{\pi}(S_{tt'}))$$

π

$$x^{\max}_{t,t'} - x_{t,t'} \geq \theta^{up} L_{tt'} \qquad \text{Up-ramping reserve}$$

$$x_{t,t'} - x^{\max}_{t,t'} \geq \theta^{down} L_{tt'} \qquad \text{Down-ramping reserve}$$

  - It is easy to tune this policy when there are only two parameters $\theta = \left(\theta^{up}, \theta^{down}\right)$
  - But perhaps the ramping reserves should depend on other information?

Designing a policy

- A robust lookahead-CFA policy
  - » We imbed a policy for fast-response adjustments within a lookahead model for planning steam:

$$F_t(S_t|\theta) = \min_{(x_{tt'})_{t'=1,\ldots,24}} \mathbb{E} \sum_{t'=t}^{t+48} C(x_{tt'}, Y^{\pi}(S_{tt'}))$$

π

$$x^{\max}_{t,t'} - x_{t,t'} \geq \theta^{up}(S_t) L_{tt'} \qquad \text{Up-ramping reserve}$$

$$x_{t,t'} - x^{\max}_{t,t'} \geq \theta^{down}(S_t) L_{tt'} \qquad \text{Down-ramping reserve}$$

  - » The parameters might depend on a state variable that captures
    - Weather forecast (esp. change in weather)
    - Load forecast (indicates how close to capacity)
  - » Now the ramping parameters are *functions*. ☹

Designing a policy

- How do we compare policies?
  - » Let's agree on a base model (a simulator) and compare different policies!

$$\bar{F}^{\pi} \approx \sum_{n=1}^{N} p(\omega^n) \sum_{t=0}^{T} \gamma^t C\left(S_t(\omega^n), X^{\pi}(S_t(\omega^n))\right)$$

where ω represents a sample path capturing season, type of meteorology (stormy, calm), and stochastic variations around this base.

  - » Now compare
    - $\bar{F}^{Robust-CFA}(\theta^{Robust-CFA})$
    - $\bar{F}^{Stoch-LA}(\theta^{Stoch-LA})$

Thank you!

To download this tutorial, go to

http://www.castlelab.princeton.edu

and click on "Presentations." For additional tutorials on stochastic optimization, go to:

http://www.castlelab.princeton.edu/jungle.htm