Minisymposium:
A Unified Framework for Optimization Under Uncertainty

Informs, Seattle

October 20, 2019

Warren B. Powell
Princeton University
Department of Operations Research and Financial Engineering
Health applications

- Health sciences
  - Sequential design of experiments for drug discovery

- Drug delivery – Optimizing the design of protective membranes to control drug release

- Medical decision making – Optimal learning for medical treatments.

© 2019 Warren Powell
Laboratory sciences

Materials science

» Optimizing payloads: reactive species, biomolecules, fluorescent markers, …

» Controllers for robotic scientist for materials science experiments

» Optimizing nanoparticles to maximize photoconductivity

© 2019 Warren Powell
E-commerce

- Revenue management
  » Optimizing prices to maximize total revenue for a particulate night in a hotel.

- Ad-click optimization
  » How much to bid for ads on the internet.

- Personalized offer optimization
  » Designing offers for individual customers.

© 2019 Warren Powell
An energy generation portfolio
Energy from wind

- Wind power from all PJM wind farms

1 year

© 2019 Warren Powell
Schneider National
Multiagent supply chain management

Pratt&Whitney jet engines
- Over 1,000 parts
- Median lead time for a part is 120 days. Some lead times are over 300 days.
- Parts often require reworking.

Managing the supply chain
- Challenge is determining when to order parts given the long lead times, and production uncertainties.
- Suppliers work for multiple customers.
Oil spills

**Sensing**
- Drones fly over the ocean to detect the presence of oil.

**Communication**
- Drones coordinate by sharing information.

**Mitigation**
- Information from drones can be used to guide surface vehicles performing cleanup.
Emergency storm response

Hurricane Sandy
» Once in 100 years?
» Rare convergence of events
» But, meteorologists did an amazing job of forecasting the storm.

The power grid
» Loss of power creates cascading failures (lack of fuel, inability to pump water)
» How to plan?
» How to react?
Emergency storm response

Information state $I_t = \text{phone calls, weather information, flooding, …}

Physical state $R_t = \text{location of vehicle}$
Emergency storm response

Information state $I_t = \text{phone calls, weather information, flooding, …}$

Belief state $B_t = \text{probability of outage}$

Physical state $R_t = \text{location of vehicle}$
Derivative-based stochastic optimization

- Stochastic gradient algorithms:
  \[ x^{n+1} = x^n + \alpha_n \nabla F(x^n, W^{n+1}) \]
  are a form of sequential decision problem.
Sequential decision problems

All of these applications are sequential decision problems:

» States, decisions, new information …

\[ S_0, x_0, W_1, \ldots, S_t, x_t, W_{t+1}, \ldots \]

What we know (or believe) \hspace{1cm} What we observe (or learn)

What we control

» As we progress, we receive a “contribution” (cost, reward, …) \( C(S_t, x_t) \) (or \( C(S_t, x_t, W_{t+1}) \)).

» Decisions are made with a “policy” (or rule, or algorithm) \( x_t = X^\pi(S_t) \).

» The challenge is to find the best policy that optimizes the contributions.
A sequential decision problems

» We often need to write this using a counter $n$:

$$S^0, x^0, W^1, \ldots, S^n, x^n, W^{n+1}, \ldots$$

$n$ might index experiments, arrivals, or iterations of an algorithm.

» Sometimes we need to index both iterations and time

$$S^0_0, x^0_0, W^0_1, \ldots, S^0_t, x^0_t, W^0_{t+1}, \ldots, S^n_t, x^n_t, W^n_{t+1}$$

$t$ might index hour of week, while $n$ indexes the week.
Sequential decision problems

Major problem domains

» “Reinforcement learning” - Discrete
» Optimal control – Continuous
» Dynamic resource allocation
» Stochastic search (algorithms)
» Multiarmed bandit problems (“active learning”)
» Games (two agent)
» Multiagent systems
Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
Before we can solve complex problems, we have to know how to think about them.

The biggest challenge when making decisions under uncertainty is modeling.
Elements of a dynamic model

For deterministic problems, we speak the language of mathematical programming

» Linear programming:
\[
\begin{align*}
\min_x & \quad cx \\
Ax & = b \\
x & \geq 0
\end{align*}
\]

» For time-staged problems
\[
\begin{align*}
\min_{x_0, \ldots, x_T} & \quad \sum_{t=0}^T c_t x_t \\
A_t x_t - B_{t-1} x_{t-1} & = b_t \\
D_t x_t & \leq u_t \\
x_t & \geq 0
\end{align*}
\]

Arguably Dantzig’s biggest contribution, more so than the simplex algorithm, was his articulation of optimization problems in a standard format, which has given algorithmic researchers a common language.
Elements of a dynamic model

All sequential decision problems can be modeled using five core components:

» State variables \( S_t = (R_t, I_t, B_t) \)
  - Physical state \( R_t \), other information \( I_t \), belief state \( B_t \).

» Decision variables \( (x_t, a_t, u_t) \)
  - Made with policy \( X^\pi (S_t | \theta) \) (or \( A^\pi (S_t) \) or \( U^\pi (S_t) \))

» Exogenous information \( W_{t+1} \)
  - What do we learn for the first time between \( t \) and \( t + 1 \)?

» Transition function \( S_{t+1} = S^M (S_t, x_t, W_{t+1}) \)
  - How do the state variables evolve over time?

» Objective function
  - \( \max \pi \mathbb{E}_{S_0 \mathbb{E}_{W_1, \ldots, W_T | S_0} \sum_{t=0}^T C(S_t, X^\pi (S_t))} \)
Energy storage example

Battery arbitrage – When to charge, when to discharge, given stochastic prices

© 2019 W.B. Powell
Elements of a dynamic model

- The state variable:
  - Controls community
    - \( x_t = "\text{Information state}" \)
  - Operations research/MDP/Computer science
    - \( S_t = (R_t, I_t, B_t) \) = System state, where:
      - \( R_t \) = Resource state (physical state)
        - Location/status of truck/train/plane
        - Inventories (product, fleets, supplies)
      - \( I_t \) = Other (deterministic) information
        - Prices
        - Weather
      - \( B_t \) = Belief state ("state of knowledge")
        - Belief about market response to prices
        - Belief about the status of equipment
Energy storage example

State variables $S_t$

- $R_t =$ Energy stored in the battery
- $D_t =$ Current demand
- $p_t =$ Current grid price
- State variable $S_t = (R_t, D_t, p_t)$
Elements of a dynamic model

Decisions:

- Markov decision processes/Computer science
  - \( a_t = \text{Discrete action} \)
- Control theory
  - \( u_t = \text{Low-dimensional continuous vector} \)
- Operations research
  - \( x_t = \text{Usually a discrete or continuous but high-dimensional vector of decisions.} \)

At this point, we do not specify how to make a decision. Instead, we define the function \( X^\pi(S_t) \) (or \( A^\pi(S_t) \) or \( U^\pi(S_t) \)), where \( \pi \) specifies the type of policy. "\( \pi \)" carries information about the type of function \( f \), and any tunable parameters \( \theta \in \Theta^f \).
Energy storage example

Decision variables

» $x_t =$How much to sell ($x_t > 0$) or buy ($x_t < 0$) to/from the grid.

» Constraints:
  • $x_t \leq R_t$
  • $-x_t \leq R_{max} - R_t$

» Policy: $x_t = X^\pi(S_t)$
Elements of a dynamic model

- **Styles of decisions**
  - Binary
    \[ x \in X = \{0,1\} \]
  - Finite
    \[ x \in X = \{1,2,...,M\} \]
  - Continuous scalar
    \[ x \in X = [a,b] \]
  - Continuous vector
    \[ x = (x_1,...,x_K), \quad x_k \in \mathbb{R} \]
  - Discrete vector
    \[ x = (x_1,...,x_K), \quad x_k \in \mathbb{Z} \]
  - Categorical
    \[ x = (a_1,...,a_I), \quad a_i \text{ is a category (e.g. red/green/blue)} \]
Elements of a dynamic model

- Exogenous information:

\[ W_t = \text{New information that first became known at time } t \]

\[ = \left( \hat{R}_t, \hat{D}_t, \hat{p}_t, \hat{E}_t \right) \]

\[ \hat{R}_t = \text{Equipment failures, delays, new arrivals} \]

New drivers being hired to the network

\[ \hat{D}_t = \text{New customer demands} \]

\[ \hat{p}_t = \text{Changes in prices} \]

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

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.

\[ W_{t+1} \text{ may be a function of the state } S_t \text{ and/or the action } x_t, \text{ so we may write it as } W_{t+1}(S_t, x_t). \]
Energy storage example

- Exogenous information

\[ W_{t+1} = \]

- \( \hat{\rho}_{t+1} \) = Change in the electricity price \( p_t \) between \( t \) and \( t + 1 \).
- \( \hat{D}_{t+1} \) = Demand for energy between time \( t \) and \( t + 1 \)
Elements of a dynamic model

- The transition function

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

\[
R_{t+1} = R_t + x_t + \hat{R}_{t+1}
\]

\[
p_{t+1} = p_t + \hat{p}_{t+1}
\]

\[
D_{t+1} = \hat{D}_{t+1}
\]

Inventories
Spot prices
Market demands

Also known as the:

- “System model”
- “State transition model”
- “Plant model”
- “Plant equation”
- “State equation”

- “Transfer function”
- “Transformation function”
- “Law of motion”
- “Model”
- “transition function”
Energy storage example

Transition function

\[
\begin{align*}
R_{t+1} &= R_t + \eta x_t \\
p_{t+1} &= p_t + \hat{p}_{t+1} \\
D_{t+1} &= \hat{D}_{t+1}
\end{align*}
\]

For many problems, the transition function can be very complex ("500 lines of Matlab code").
Elements of a dynamic model

- **Objective functions**
  - Cumulative reward ("online learning")
    \[
    \max_{\pi} \mathbb{E}\left\{ \sum_{t=0}^{T} C_t \left( S_t, X_\pi (S_t), W_{t+1} \right) \mid S_0 \right\}
    \]
    - Policies have to work well over time.
  - Final reward ("offline learning")
    \[
    \max_{\pi} \mathbb{E}\left\{ F(x_\pi^N, \hat{W}) \mid S_0 \right\}
    \]
    - We only care about how well the final decision \( x_\pi^N \) works.
  - Risk
    \[
    \max_{\pi} \rho\left\{ C(S_0, X_0^\pi (S_0)), C(S_1, X_1^\pi (S_1)), \ldots, C(S_T, X_T^\pi (S_T)) \mid S_0 \right\}
    \]
Energy storage example

Objective function

\[ C(S_t, x_t) = p_t x_t \]

\[ \max_{\pi} \mathbb{E}\left\{ \sum_{t=0}^{T} C(S_t, X^\pi(S_t)) \mid S_0 \right\} \]
Energy arbitrage

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

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

Energy in storage:

Price of electricity:
Energy arbitrage

Objective function

» Now the policy search is to find the best parameters $\rho$:

$$\max_{\rho} \mathbb{E} \left\{ \sum_{t=0}^{T} C(S_t, X^\pi(S_t | \rho)) | S_0 \right\}$$
Elements of a dynamic model

- The complete model:
  - Objective function
    - Cumulative reward ("online learning")
      \[
      \max_\pi \mathbb{E}\left\{\sum_{t=0}^{T} C_t\left(S_t, X^\pi_t(S_t), W_{t+1}\right) \mid S_0\right\}
      \]
    - Final reward ("offline learning")
      \[
      \max_\pi \mathbb{E}\left\{F(x^{\pi,N}, \hat{W}) \mid S_0\right\}
      \]
    - Risk:
      \[
      \max_\pi \rho\left\{C(S_0, X^\pi_0(S_0)), C(S_1, X^\pi_1(S_1)), \ldots, C(S_T, X^\pi_T(S_T)) \mid S_0\right\}
      \]
  - Transition function:
    \[
    S_{t+1} = S^M (S_t, x_t, W_{t+1})
    \]
  - Exogenous information:
    \[
    (S_0, W_1, W_2, \ldots, W_T)
    \]
Elements of a dynamic model

**Deterministic**

- Objective function
  \[
  \min \sum_{t=0}^{T} c_t x_t
  \]
- Decision variables:
  \[(x_0, \ldots, x_T)\]
- Constraints:
  - at time \(t\)
    \[A_t x_t = R_t\]
  - \(x_t \geq 0\)
  - Transition function
    \[R_{t+1} = b_{t+1} + B_t x_t\]

**Stochastic**

- Objective function
  \[
  \max_{\pi} E \left\{ \sum_{t=0}^{T} C_t \left( S_t, X_t^{\pi}(S_t), W_{t+1} \right) \right\} \mid S_0
  \]
- Policy
  \[X^{\pi} : S \rightarrow \mathcal{X}\]
- 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
  \[(S_0, W_1, W_2, \ldots, W_T)\]
Modeling with uncertainty

Stochastic prices

Charge/discharge decisions

\[ R_{\text{total}} = \max \left( \sum_{t=0}^{N} P(t) \left( x_{\text{generation}}(t) + x_{\text{discharge}}(t) - x_{\text{charge}}(t)/\eta \right) \right) \]

subject to:

\[ 0 \leq x_{\text{discharge}} \leq \hat{E}_{\text{max}} \]

\[ 0 \leq x_{\text{charge}} \leq \min(\eta x_{\text{generation}}(t), \eta \hat{E}_{\text{max}}) \]

\[ 0 \leq \sum_{t=0}^{N} (x_{\text{charge}}(t) - x_{\text{discharge}}(t)) \leq h\hat{E}_{\text{max}} \]
Imagine that you would like to solve the time-dependent linear program:

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

» subject to

$$A_0 x_0 = b_0$$
$$A_t x_t - B_{t-1} x_{t-1} = b_t, \quad t \geq 1.$$ 

We can convert this to a proper stochastic model by replacing $x_t$ with $X^\pi_t (S_t)$ and taking an expectation:

$$\max_{\pi} \mathbb{E} \sum_{t=0}^{T} c_t X^\pi_t (S_t) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=0}^{T} c_t X^\pi_t (S^n_t (\omega^n))$$

The policy $X^\pi_t (S_t)$ has to satisfy $A_t x_t = R_t$ with transition function:

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

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
An energy storage example

A more complex model

© 2019 W.B. Powell
An energy storage problem

State variables

- Wind speed
- Electricity prices
- Demand

» We will present the full model, accumulating the information we need in the state variable.
» We will highlight information we need as we proceed. This information will make up our state variable.
An energy storage problem

Decision variables

\[ x_t = \left( x_t^{EL}, x_t^{EB}, x_t^{GL}, x_t^{GB}, x_t^{BL} \right) \]

» Constraints;

\[ x_t^{EL} + x_t^{EB} \leq E_t \]

\[ x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t \]

\[ x_t^{BL} \leq R_t \]
An energy storage problem

Exogenous information

\[
W_t = \begin{cases} 
\hat{E}_t = \text{Change in energy from wind between } t - 1 \text{ and } t \\
\hat{D}_t = \text{Observed demand at time } t \\
\hat{p}_t = \text{Change in price between } t - 1 \text{ and } t 
\end{cases}
\]
An energy storage problem

Transition function

\[ R_{t+1}^{\text{battery}} = R_t^{\text{battery}} + x_t \]

\[ E_{t+1} = E_t + \hat{E}_{t+1} \]

\[ p_{t+1} = p_t + \hat{p}_{t+1} \]

\[ D_{t+1} = \hat{D}_{t+1} \]
An energy storage problem

Objective function

\[ C(S_t, x_t) = \mathbb{E}_{p_t} \left( x_t^{GB} + x_t^{GL} \right) \]

\[ \min_{\pi} \mathbb{E} \left\{ \sum_{t=0}^{T} C_t(S_t, X_t^\pi(S_t)) \mid S_0 \right\} \]
An energy storage problem

- **State variables**
  - Cost function
    - $p_t = \text{Price of electricity}$
  - Decision function
  - Constraints:
    - $x_t^{EL} + x_t^{EB} \leq E_t$
    - $x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t$
    - $x_t^{BL} \leq R_t$
  - Transition function
    - $p_{t+1} = p_t + \hat{p}_{t+1}$

$$S_t = (R_t, D_t, E_t, p_t)$$
An energy storage example

A time series price model
An energy storage problem

Transition function

» ARIMA price model:

\[ p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \epsilon_{t+1} \]
An energy storage problem

State variables

» Cost function
  \[ p_t = \text{Price of electricity} \]

» Decision function

Constraints:

\[ x_t^{EL} + x_t^{EB} \leq E_t \]
\[ x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t \]
\[ x_t^{BL} \leq R_t \]

» Transition function

\[ p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \epsilon_{t+1} \]

\[ S_t = (R_t, D_t, E_t, (p_t, p_{t-1}, p_{t-2})) \]
An energy storage example

(Passive) learning the price process
An energy storage problem

Transition function

» ARIMA price model with learning:

\[ p_{t+1} = \bar{\theta}_0 p_t + \bar{\theta}_1 p_{t-1} + \bar{\theta}_2 p_{t-2} + \varepsilon_{t+1} \]

Need to learn \( \bar{\theta}_{ti} \)

© 2019 W.B. Powell
Learning in stochastic optimization

Updating the demand parameter

Let \( p_{t+1} \) be the new price and let

\[
\overline{F}_t^{\text{price}} (\overline{p}_t \mid \overline{\theta}_t) = (\overline{\theta}_t)^T \overline{p}_t = \overline{\theta}_{t0} p_t + \overline{\theta}_{t1} p_{t-1} + \overline{\theta}_{t2} p_{t-2}
\]

We update our estimate \( \overline{\theta}_t \) using our recursive least squares equations:

\[
\overline{\theta}_{t+1} = \overline{\theta}_t + \frac{1}{\gamma_t} M_t \overline{p}_t \varepsilon_{t+1}
\]

\[
\varepsilon_{t+1} = \overline{F}_t^{\text{price}} (\overline{p}_t \mid \overline{\theta}_t) - p_{t+1}
\]

\[
M_{t+1} = M_t - \frac{1}{\gamma_t} M_t \overline{p}_t (\overline{p}_t)^T M_t
\]

\[
\gamma_{t+1} = 1 - (\overline{p}_t)^T M_t \overline{p}_t
\]
An energy storage problem

State variables

» Cost function

\[ p_t \text{ = Price of electricity} \]

» Decision function

Constraints:

\[ x_t^{EL} + x_t^{EB} \leq E_t \]
\[ x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t \]
\[ x_t^{BL} \leq R_t \]

» Transition function

\[ p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1}^p \]
\[ \bar{\theta}_{t+1} = \bar{\theta}_t + \frac{1}{\gamma_t} M_t \bar{p}_t \varepsilon_{t+1} \]

\[ S_t = (R_t, D_t, E_t, (p_t, p_{t-1}, p_{t-2}), (\bar{\theta}_t, M_t)) \]

Physical state variables

Other information \( I_t \)

Belief state \( B_t \)

© 2019 W.B. Powell
An energy storage problem

Types of learning:

» No learning (θ’s are known)

\[ p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1} \]

» Passive learning (learn θs from price data)

\[ p_{t+1} = \overline{\theta}_0 p_t + \overline{\theta}_1 p_{t-1} + \overline{\theta}_2 p_{t-2} + \varepsilon_{t+1} \]

» Active learning (“bandit problems”)

\[ p_{t+1} = \bar{\theta}_0 p_t + \bar{\theta}_1 p_{t-1} + \bar{\theta}_2 p_{t-2} + \bar{\theta}_3 x_t^{GB} + \varepsilon_{t+1} \]

Modeling the active learning is the same as passive, but the choice of policy will change.

© 2019 W.B. Powell
An energy storage problem

Learning a demand response function

Sampling to maximize revenue:

Sampling to learn demand response:
An energy storage example

Planning with rolling forecasts
Parametric cost function approximation

- An energy storage problem:

![Sample Paths of Spot Price (P)](image)

- Wind speed
- Electricity prices
- Demand

© 2019 W.B. Powell
Parametric cost function approximation

Forecasts evolve over time as new information arrives:

- Rolling forecasts, updated each hour.
- Forecast made at midnight:
- Actual
An energy storage problem

- Exogenous information

\[ W_t = \begin{cases} \varepsilon_{t,t'}^f = \text{Difference between forecasted energy } E_t, \text{ at time } t' \text{ made at } t - 1 \text{ and } t \end{cases} \]
An energy storage problem

Transition function

\[ E_{t+1} = f_{t+1}^E + \varepsilon_{t+1,t+1}^E \]

\[ f_{t+1,t'}^E = f_{t,t'}^E + \varepsilon_{t+1,t'}^E, \quad t' > t + 1 \]

\[ \varepsilon_{t+1,t+1}^E \sim N(0, \sigma_\varepsilon^2) \]

\[ \varepsilon_{t+1,t'}^E \sim N(0, (t' - (t + 1))\sigma_\varepsilon^2) \]

© 2019 W.B. Powell
An energy storage problem

- **State variables**
  - Cost function
    - \( p_t \) = Price of electricity
  - Decision function
  - Constraints:
    - \( x_t^{EL} + x_t^{EB} \leq E_t \)
    - \( x_t^{GL} + x_t^{EL} + x_t^{BL} = D_t \)
    - \( x_t^{BL} \leq R_t \)

- **Transition function**
  - \( p_{t+1} = \theta_0 p_t + \theta_1 p_{t-1} + \theta_2 p_{t-2} + \varepsilon_{t+1}^p \)
  - \( \overline{\theta}_{t+1} = \overline{\theta}_t + \frac{1}{\gamma_t} M_t \overline{p}_t \varepsilon_{t+1} \)
  - \( f_{t+1,t'}^E = f_{t,t'}^E + \varepsilon_{t+1,t'}^E \)

\( S_t = (R_t, D_t, E_t, (p_t, p_{t-1}, p_{t-2}), (\overline{\theta}_t, M_t), f_t^E) \)
Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
Modeling uncertainty

Classes of uncertainty

» Observational uncertainty
» Prognostic uncertainty (forecasting)
» Experimental noise/variability
» Transitional uncertainty
» Inferential uncertainty
» Model uncertainty
» Systematic exogenous uncertainty
» Control/implementation uncertainty
» Algorithmic noise
» Goal uncertainty

Modeling uncertainty in the context of stochastic optimization is a relatively untapped area of research.
Outline

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
Designing policies

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

  \[
  A \text{ policy is a mapping from a state to an action.}
  \]

  ... any mapping.

- How do we search over an arbitrary space of policies?
Designing policies

“Policies” and the English language

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Formula</th>
<th>Principle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belief</td>
<td>Habit</td>
<td>Procedure</td>
</tr>
<tr>
<td>Bias</td>
<td>Laws/bylaws</td>
<td>Process</td>
</tr>
<tr>
<td>Commandment</td>
<td>Manner</td>
<td>Protocols</td>
</tr>
<tr>
<td>Conduct</td>
<td>Method</td>
<td>Recipe</td>
</tr>
<tr>
<td>Control law</td>
<td>Mode</td>
<td>Ritual</td>
</tr>
<tr>
<td>Convention</td>
<td>Mores</td>
<td>Rule</td>
</tr>
<tr>
<td>Culture</td>
<td>Patterns</td>
<td>Style</td>
</tr>
<tr>
<td>Customs</td>
<td>Plans</td>
<td>Technique</td>
</tr>
<tr>
<td>Dogma</td>
<td>Policies</td>
<td>Tenet</td>
</tr>
<tr>
<td>Etiquette</td>
<td>Practice</td>
<td>Tradition</td>
</tr>
<tr>
<td>Fashion</td>
<td>Prejudice</td>
<td>Way of life</td>
</tr>
</tbody>
</table>
Designing policies

Two fundamental strategies:

**Policy search** – Search over a class of functions for making decisions to optimize some metric.

$$\max_{\pi = (f \in F, \theta^f \in \Theta^f)} \mathbb{E}\left\{ \sum_{t=0}^{T} C(S_t, X^\pi_t(S_t | \theta)) | S_0 \right\}$$

**Lookahead approximations** – Approximate the impact of a decision now on the future.

$$X^*_t(S_t) = \arg\max_{x_t} \left( C(S_t, x_t) + \mathbb{E}\left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E}\left\{ \sum_{t'=t+1}^{T} C(S_{t'}, X^\pi_{t'}(S_{t'})) | S_{t+1} \right\} | S_t, x_t \right\} \right\} \right)$$
Designing policies

Policy search:

1) Policy function approximations (PFAs) \( x_t = X^{PFA} (S_t | \theta) \)
   - Lookup tables
     - “when in this state, take this action”
   - Parametric functions
     - Order-up-to policies: if inventory is less than \( s \), order up to \( S \).
     - Affine policies - \( x_t = X^{PFA} (S_t | \theta) = \sum_{f \in F} \theta_f \phi_f (S_t) \)
     - Shallow neural networks
   - Locally/semi/non parametric
     - Kernel regression
     - Deep neural networks

2) Cost function approximations (CFAs)
   - Optimizing a deterministic model modified to handle uncertainty (buffer stocks, schedule slack)
   \[
   X^{CFA} (S_t | \theta) = \arg \max_{x_t \in \bar{X}_t (\theta)} \bar{C}^{\pi} (S_t, x_t | \theta)
   \]
Designing policies

- The ultimate lookahead policy is optimal

\[ X_t^*(S_t) = \arg \max_{x_t} \left\{ C(S_t, x_t) + \mathbb{E} \left[ \max_{\pi \in \Pi} \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^\pi(S_{t'})) \mid S_{t+1} \right] \mid S_t, x_t \right\} \]
Designing policies

- **Lookahead approximations** – Approximate the impact of a decision now on the future:

  » An optimal policy (based on looking ahead):

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

3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

\[
X^*_t(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \tilde{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)
\]

\[
X^{VFA}_t(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \tilde{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)
\]

\[
= \arg \max_{x_t} \left( C(S_t, x_t) + \tilde{V}^x_t(S_t^x) \right)
\]

\[
= \arg \max_{x_t} \tilde{Q}_t(S_t, x_t) \quad ("Q-learning")
\]
Designing policies

- **Lookahead approximations** – Approximate the impact of a decision now on the future:

  » An optimal policy (based on looking ahead):

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

  3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

  \[ X^*_t(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ V_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right) \]

  \[ X^{VFA}_t(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \bar{V}_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right) \]

  \[ = \arg \max_{x_t} \left( C(S_t, x_t) + \bar{V}_t^x(S_t^x) \right) \]

  \[ = \arg \max_{x_t} Q_t(S_t, x_t) \quad ("Q-learning") \]
Designing policies

- Lookahead approximations – Approximate the impact of a decision now on the future:
  » An optimal policy (based on looking ahead):

\[
X_t^*(S_t) = \arg\max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^\pi(S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right)
\]

3) Approximating the value of being in a downstream state using machine learning ("value function approximations")

\[
X_t^*(S_t) = \arg\max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ V_{t+1}(S_{t+1}) \mid S_t, x_t \right\} \right)
\]

\[
X_t^{VFA}(S_t) = \arg\max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \overline{V}_t(S_{t+1}) \mid S_t, x_t \right\} \right)
\]

\[
= \arg\max_{x_t} \left( C(S_t, x_t) + \overline{V}_t^x(S_t^x) \right)
\]

\[
= \arg\max_{x_t} \overline{Q}_t(S_t, x_t) \quad ("Q-learning")
\]
Designing policies

The ultimate lookahead policy is optimal

\[ X_t^*(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \max_{\pi \in \Pi} \mathbb{E} \left( \sum_{t'=t+1}^{T} C(S_{t'}, X^\pi_{t'}(S_{t'})) \mid S_{t+1} \right) \mid S_t, x_t \right\} \right) \]
Designing policies

The ultimate lookahead policy is optimal

\[ X_t^* (S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \max_{\pi \in \Pi} \left\{ \mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^\pi (S_{t'})) \mid S_{t+1} \right\} \mid S_t, x_t \right\} \right) \]

4) Instead, we have to create an approximation called the lookahead model:

\[ \left( S_t, x_t, \tilde{W}_{t,t+1}, \tilde{S}_{t,t+1}, \tilde{x}_{t,t+1}, \tilde{W}_{t,t+2}, \ldots, \tilde{S}_{t,t'}, \tilde{x}_{t,t'}, \tilde{W}_{t,t'+1}, \ldots \right) \]

» A direct lookahead policy (DLA) works by solving the lookahead model:

\[ X_t^* (S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \mathbb{E} \left\{ \max_{\tilde{\pi} \in \tilde{\Pi}} \left\{ \mathbb{E} \sum_{t'=t+1}^{t+H} C(\tilde{S}_{t'}, \tilde{X}_{t'}^{\tilde{\pi}} (\tilde{S}_{t'})) \mid \tilde{S}_{t+1} \right\} \mid S_t, x_t \right\} \right) \]
Designing policies

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1)</td>
<td><strong>Policy function approximations (PFAs)</strong></td>
</tr>
<tr>
<td></td>
<td>» Lookup tables, rules, parametric/nonparametric functions</td>
</tr>
<tr>
<td>2)</td>
<td><strong>Cost function approximation (CFAs)</strong></td>
</tr>
<tr>
<td></td>
<td>» ( X^{CFA}(S_t</td>
</tr>
<tr>
<td>3)</td>
<td><strong>Policies based on value function approximations (VFAs)</strong></td>
</tr>
<tr>
<td></td>
<td>» ( X^{VFA}(S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \bar{V}_t^x(S_t^x(S_t, x_t)) \right) )</td>
</tr>
<tr>
<td>4)</td>
<td><strong>Direct lookahead policies (DLAs)</strong></td>
</tr>
<tr>
<td></td>
<td>» Deterministic lookahead rolling horizon proc./model predictive control</td>
</tr>
<tr>
<td></td>
<td>( X_t^{LA-D}(S_t) = \arg \max_{\tilde{x}<em>t, \ldots, \tilde{x}</em>{t,H}} C(\tilde{S}<em>t, \tilde{x}<em>t) + \sum</em>{t'=t+1} C(\tilde{S}</em>{t'}, \tilde{x}_{t'}) )</td>
</tr>
<tr>
<td></td>
<td>» Chance constrained programming</td>
</tr>
<tr>
<td></td>
<td>( P[A_t x_t \leq f(W)] \leq 1 - \delta )</td>
</tr>
<tr>
<td></td>
<td>» Stochastic lookahead /stochastic prog/Monte Carlo tree search</td>
</tr>
<tr>
<td></td>
<td>( X_t^{LA-S}(S_t) = \arg \max_{\tilde{x}<em>t, \ldots, \tilde{x}</em>{t+H}} C(\tilde{S}<em>t, \tilde{x}<em>t) + \sum</em>{t'=t+1} C(\tilde{S}</em>{t'}, (\tilde{\omega}), \tilde{x}_{t'}(\tilde{\omega})) )</td>
</tr>
<tr>
<td></td>
<td>» “Robust optimization”</td>
</tr>
<tr>
<td></td>
<td>( X_t^{LA-RO}(S_t) = \arg \max_{\tilde{x}<em>t, \ldots, \tilde{x}</em>{t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}<em>t, \tilde{x}<em>t) + \sum</em>{t'=t+1} C(\tilde{S}</em>{t'}, (w), \tilde{x}_{t'}(w)) )</td>
</tr>
</tbody>
</table>
Designing policies

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

2) Cost function approximation (CFAs)
   » \( X^{CFA}(S_t \mid \theta) = \arg \max_{x_t \in X_{\pi}^t(\theta)} \overline{C}^\pi (S_t, x_t \mid \theta) \)

3) Policies based on value function approximations (VFAs)
   » \( X_t^{VFA}(S_t) = \arg \max_{\pi} \left( C(S_t, x_t) + \overline{V}^t_x (S_t^x(S_t, x_t)) \right) \)

4) Direct lookahead policies (DLAs)
   » Deterministic lookahead/rolling horizon proc./model predictive control
     \( X_t^{LA-D}(S_t) = \arg \max_{\tilde{x}_t, \ldots, \tilde{x}_{t+H}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, \tilde{x}_{tt'}) \)
   » Chance constrained programming
     \( P[A_t x_t \leq f(W)] \leq 1 - \delta \)
   » Stochastic lookahead /stochastic prog/Monte Carlo tree search
     \( X_t^{LA-S}(S_t) = \arg \max_{\tilde{x}_t, \ldots, \tilde{x}_{t+T}} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} p(\tilde{\omega}) \sum_{\tilde{\omega} \in \tilde{\Omega}_t} C(\tilde{S}_{tt'}, (\tilde{\omega}), \tilde{x}_{tt'}(\tilde{\omega})) \)
   » “Robust optimization”
     \( X_t^{LA-RO}(S_t) = \arg \max_{\tilde{x}_t, \ldots, \tilde{x}_{t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_{tt}, \tilde{x}_{tt}) + \sum_{t'=t+1}^{T} C(\tilde{S}_{tt'}, (w), \tilde{x}_{tt'}(w)) \)
Designing policies

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

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

3) Policies based on value function approximations (VFAs)
   » \( X^{VFA}_t (S_t) = \arg \max_{x_t} \left( C(S_t, x_t) + \bar{V}^X_t \left( S^x_t (S_t, x_t) \right) \right) \)

4) Direct lookahead policies (DLAs)
   » Deterministic lookahead rolling horizon proc./model predictive control
     \( X^{LA-D}_t (S_t) = \arg \max_{\tilde{x}_{t+1}, \ldots, \tilde{x}_{t+H}} C(\tilde{S}_t, \tilde{x}_t) + \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, \tilde{x}_{t'}) \)
   » Chance constrained programming
     \( P[A_t x_t \leq f(W)] \leq 1 - \delta \)
   » Stochastic lookahead /stochastic prog/Monte Carlo tree search
     \( X^{LA-S}_t (S_t) = \arg \max_{\tilde{x}_t, \ldots, \tilde{x}_{t+H}} C(\tilde{S}_t, \tilde{x}_t) + \sum_{\tilde{\omega} \in \Omega_t} p(\tilde{\omega}) \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, (\tilde{\omega}), \tilde{x}_{t'}(\tilde{\omega})) \)
     » “Robust optimization”
     \( X^{LA-RO}_t (S_t) = \arg \max_{\tilde{x}_t, \ldots, \tilde{x}_{t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_t, \tilde{x}_t) + \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, (w), \tilde{x}_{t'}(w)) \)

Imbedded optimization
The Universal Framework for Sequential Decisions

Warren B. Powell, Princeton University

\[
\max_{\pi} \mathbb{E} \left\{ \sum_{t=0}^{T} C(S_t, X^{\pi}(S_t)) | S_0 \right\}
\]

where \( S_{t+1} = S^{M}(S_t, X^{\pi}(S_t), W_{t+1}) \)

and given \((S_0, W_1, W_2, ..., W_t, ...)\)

The four classes of policies (PFAs, CFAs, VFAs and DLAs) are *universal*.

Any sequential decision problem will use one of these four classes (or a hybrid), including whatever you might be doing now.

The optimal policy (if we could solve it) is given by

\[
X^*(S_t) = \arg \max_x \left( C(S_t, x) + \mathbb{E} \left( \max_{\pi} \mathbb{E} \left\{ \sum_{t'=t+1}^{t+H} C(S_{t'}, X^{\pi}(S_{t'})) | S_{t+1} \right\} | S_t, x_t \right) \right)
\]

Policy Function Approximation (PFA)

\[
X^{PFA}(S_t | \theta) = \sum_{f \in F} \theta_f \phi_f(S_t)
\]

Neural network

Cost Function Approximation (CFA)

\[
X^{CFA}(S_t | \theta) = \arg \max_x \left( c_t x_t + \sum_{f} \theta_f \phi_f(S_t) \right)
\]

\[
= \arg \max_x \left( \mu_t x_t + \theta^E \bar{\sigma}_t x_t \right)
\]

Value Function Approximation (VFA)

\[
X^{VFA}(S_t | \theta) = \arg \max_x \left( C(S_t, x) + \mathbb{E}\{V_{t+1}(S_{t+1}) | S_t, x_t\} \right)
\]

\[
= \arg \max_x \left( C(S_t, x) + \bar{V}_t^x(S_t^x) \right)
\]

\[
= \arg \max \{Q(S_t, x) \}
\]

Direct Lookahead (DLA)

\[
X^{DLA}(S_t | \theta) = \arg \max_x \left( c_t x_t + \sum_{t'=t+1}^{t+H} \tilde{c}_{tt'} \bar{x}_{tt'} \right)
\]

Available at jungle.princeton.edu
Designing policies

Notes:

» Policies in the “policy search class” are simpler, and as a result this is what you are most likely going to see (and use) in practice.

» “The price of simplicity is tunable parameters”

» Tuning is hard!

• ... but you do tuning offline.
Learning problems

Classes of machine learning problems in stochastic optimization

1) Approximating the objective

\[ \bar{F}(x|\theta) \approx \mathbb{E}F(x, W). \]

2) Designing a policy \( X^\pi(S|\theta) \).

3) A value function approximation

\[ \bar{V}_t(S_t|\theta) \approx V_t(S_t). \]

4) Designing a cost function approximation:
   - The objective function \( \bar{C}^\pi(S_t, x_t|\theta) \).
   - The constraints \( X^\pi(S_t|\theta) \)

5) Approximating the transition function

\[ \bar{S}^M(S_t, x_t, W_{t+1}|\theta) \approx S^M(S_t, x_t, W_{t+1}) \]
The four classes of policies

» Policy function approximations (PFAs)
» Cost function approximations (CFAs)
» Value function approximations (VFAs)
» Direct lookahead policies (DLAs)
» Hybrid direct lookahead/CFA

» Any of the four classes may work best!
The four classes of policies

- Policy function approximations (PFAs)
- Cost function approximations (CFAs)
- Value function approximations (VFAs)
- Direct lookahead policies (DLAs)
- Hybrid direct lookahead/CFA
- Any of the four classes may work best!
Policy function approximations

Battery arbitrage – When to charge, when to discharge, given volatile LMPs
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 $\theta_{\text{Charge}}$ and $\theta_{\text{Discharge}}$. 
Policy function approximations

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

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

Energy in storage:

Price of electricity:
Policy function approximations

Finding the best policy

» We need to maximize

\[ \max_\theta F(\theta) = \mathbb{E} \sum_{t=0}^{T} \gamma^t C(S_t, X^\pi_t (S_t | \theta)) \]

» We cannot compute the expectation, so we run simulations:
Policy function approximations

How do we search for the best $\theta$?

» Derivative-based
  • Use classical stochastic gradient methods:
    $$x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1})$$
    • The gradient $\nabla_x (F x^n, W^{n+1})$ may be computed analytically, but more often it is computed numerically.

» Derivative-free
  • Build a belief model $\bar{F}(x) \approx \mathbb{E} F(x, W)$ that approximates our function.

» Both of these approaches are sequential decision problems!
Policy function approximations

- Derivative-based stochastic search
  - Basic problem:
    \[ \max_x \mathbb{E}\{F(x,W) \mid S_0\} \]
  - Stochastic gradient algorithm
    \[ x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1}) \]
  - Let \( \pi \) be our “algorithm” and let \( x^{\pi,N} \) be the solution that “algorithm” \( \pi \) produces after \( N \) iterations.
  - Now we would state the optimization problem as
    \[ \max_{\pi} \mathbb{E}F(x^{\pi,N}, W) \]
    where \( \pi \) is our stochastic gradient algorithm.
  - In other words, we want the optimal algorithm.
Policy function approximations

- Derivative-based stochastic search:
  - Assume that our stepsize rule

\[ \alpha_n = \frac{\theta}{\theta + N^n} \]

where \( N^n \) = number of times the solution has not improved:

\[ N^{n+1} = \begin{cases} 
N^n + 1 & \text{If } \nabla_x F(x^{n-1}, W^n) \nabla_x F(x^n, W^{n+1}) < 0 \\
N^n & \text{Otherwise}
\end{cases} \]

- The stepsize \( \alpha_n \) is the “decision.” The stepsize rule is the “policy” which is a form of policy function approximation.
Derivative-based, finite horizon

- Derivative-based stochastic search – finite horizon
  - State variables
    \[ S^n = (x^n, N^n) \]
  - Decision variable: \( \alpha_n \) made using policy
    \[ \alpha^n(S^n | \theta) = \frac{\theta}{\theta + N^n} \]
  - Exogenous information: \( W^{n+1} \)
  - Transition function
    \[ x^{n+1} = x^n + \alpha_n \nabla_x F(x^n, W^{n+1}) \]
    \[ N^{n+1} = N^n + \begin{cases} 1 & \text{if } \nabla_x F(x^n, W^{n+1}) \nabla_x F(x^{n-1}, W^n) < 0 \\ 0 & \text{Otherwise} \end{cases} \]
  - Our objective is to find the best stepsize policy that solves
    \[ \max_\theta \mathbb{E}F(x^n, W, \hat{W}) \]
The four classes of policies

- Policy function approximations (PFAs)
- Cost function approximations (CFAs)
- Value function approximations (VFAs)
- Direct lookahead policies (DLAs)
- Hybrid direct lookahead/CFA

Any of the four classes may work best!
Cost function approximations

Materials science

We need to find the best of seven catalysts to maximize the strength of materials.

\[ \text{Cov}(\mu_x, \mu_{x'}) \]

<table>
<thead>
<tr>
<th></th>
<th>1.4 nm Fe</th>
<th>1 nm Fe</th>
<th>2 nm Fe</th>
<th>10nm ALD Al2O3+1.2 nm IBS Fe</th>
<th>2 nm Ni</th>
<th>Ni 0.6 nm</th>
<th>10nm ALD Al2O3+1 nm Ni</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 nm Fe</td>
<td>1</td>
<td>0.7</td>
<td>0.7</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>1 nm Fe</td>
<td>0.7</td>
<td>1</td>
<td>0.7</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>2 nm Fe</td>
<td>0.7</td>
<td>0.7</td>
<td>1</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>10nm ALD Al2O3+1.2 nm IBS Fe</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>2 nm Ni</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
<td>1</td>
<td>1</td>
<td>0.7</td>
<td>0.6</td>
</tr>
<tr>
<td>Ni 0.6 nm</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
<td>0.3</td>
<td>0.7</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>10nm ALD Al2O3+1 nm Ni</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0</td>
<td>0.6</td>
<td>0.6</td>
<td>1</td>
</tr>
</tbody>
</table>
Cost function approximations

- Correlated beliefs: Testing one material teaches us about other materials
Cost function approximations

Cost function approximations (CFA)

» Upper confidence bounding

\[ X^{UCB}(S^n | \theta^{UCB}) = \arg \max_x \left( \bar{\mu}_x^n + \theta^{UCB} \sqrt{\frac{\log n}{N_x^n}} \right) \]

» Interval estimation

\[ X^{IE}(S^n | \theta^{IE}) = \arg \max_x \left( \bar{\mu}_x^n + \theta^{IE} \bar{\sigma}_x^n \right) \]

» Thompson sampling

\[ x^n = \arg \max_x \bar{\mu}_x^n \quad \bar{\mu}_x^n \sim N(\bar{\mu}_x^n, \theta^{TS} \bar{\sigma}_x^{2,n}) \]
Cost function approximations

- Picking $\theta^{IE} = 0$ means we are evaluating each choice at the mean.
Cost function approximations

- Picking $\theta^{IE} = 2$ means we are evaluating each choice at the 95th percentile.
Cost function approximations

- Optimizing the policy
  - We optimize $\theta^{IE}$ to maximize:

$$\max_{\theta^{IE}} F(\theta^{IE}) = \mathbb{E} F\left(x^{\pi,N}, W\right)$$

where

$$x^n = X^{IE} (S^n | \theta^{IE}) = \arg \max_x \left( \mu^n_x + \theta^{IE} \sigma^n_x \right) \quad S^n = (\mu^n_x, \sigma^n_x)$$

- Notes:
  - This can handle any belief model, including correlated beliefs, nonlinear belief models.
  - All we require is that we be able to simulate a policy.
Cost function approximations

Drivers

Demands
The assignment of drivers to loads evolves over time, with new loads being called in, along with updates to the status of a driver.
Cost function approximations

A purely myopic policy would solve this problem using

$$\min_x \sum_d \sum_l c_{tdl} x_{idl}$$

where

$$x_{tdl} = \begin{cases} 1 & \text{If we assign driver } d \text{ to load } l \\ 0 & \text{Otherwise} \end{cases}$$

$$c_{tdl} = \text{Cost of assigning driver } d \text{ to load } l \text{ at time } t$$

What if a load is not assigned to any driver, and has been delayed for a while? This model ignores the fact that we eventually have to assign someone to the load.
Cost function approximations

We can minimize delayed loads by solving a modified objective function:

$$\min_x \sum_d \sum_l \left( c_{tdl} - \theta \tau_{tl} \right) x_{idl}$$

where

$$\tau_{tl} = \text{How long load } l \text{ has been delayed by time } t$$

$$\theta = \text{Bonus for moving a delayed load}$$

We refer to our modified objective function as a cost function approximation.
We now have to tune our policy, which we define as:

\[ X^\pi (S_t \mid \theta) = \arg \min_x \sum_{d} \sum_{l} \left( c_{tdl} - \theta \tau_{tl} \right) x_{tdl} \]

\[ \bar{C}^\pi (S_t, x_t \mid \theta) \]

We can now optimize \( \theta \), another form of policy search, by solving

\[ \min_\theta F^\pi (\theta) = \mathbb{E} \sum_{t=0}^{T} C(S_t, X^\pi_t (S_t \mid \theta)) \]
Cost function approximations

Other applications

» Airlines optimizing schedules with schedule slack to handle weather uncertainty.

» Manufacturers using buffer stocks to hedge against production delays and quality problems.

» Grid operators scheduling extra generation capacity in case of outages.

» Adding time to a trip planned by Google maps to account for uncertain congestion.
Outline

The four classes of policies

- Policy function approximations (PFAs)
- Cost function approximations (CFAs)
- Value function approximations (VFAs)
- Direct lookahead policies (DLAs)
- Hybrid direct lookahead/CFA
- Any of the four classes may work best!
Driverless EV optimization

- Current Uber logic:
  - Show nearest 8 drivers.
  - Contact closest driver to confirm assignment.
  - If driver does not confirm, contact second closest driver.

- Limitations:
  - Ignores potential future opportunities for each driver.
Driverless EV optimization
Driverless EV optimization
Driverless EV optimization
The assignment of cars to riders evolves over time, with new riders arriving, along with updates of cars available.
Driverless EV optimization

While we are optimizing the drivers, we also have to learn the response of drivers and riders to price.

» We assume a parameterized response curve for buyers, and a separate one for sellers

\[
P^Y(p, a | \theta) = \frac{e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}{1 + e^{\theta_{ij}^0 + \theta_{ij} p + \theta_{ij}^a a}}
\]

» We do not know the right values of the parameters, so we have a learning problem.

» Our belief state is our distribution of belief on \( \theta \).
Driverless EV optimization

Time Location Battery Fleet

$C_{ta_1r_1}$

$v(a_1')$

$v(a_1'')$
Driverless EV optimization

\[ c_{i,a_1} + v(a_1') \]

\[ c_{i,a_1} + v(a_1') \]

\[ c_{i,a_1} + v(a_1') \]

\[ c_{i,a_1} + v(a_1') \]
Driverless EV optimization
Driverless EV optimization

Original solution

- $a_1$
- $a_2$
- $a_3$
- $a_4$

$\mathbf{v}(a_1)$
$\mathbf{v}(a_2)$
$\mathbf{v}(a_3)$
$\mathbf{v}(a_4)$
Driverless EV optimization
Driverless EV optimization

\[ \text{Difference} = \hat{v}(a_1) \]

\[ v(a_1') - v(a_2') \]

\[ v(a_2') - v(a_3') \]

\[ v(a_3') - v(a_4') \]
Driverless EV optimization

Estimating average values:

\[ \bar{v}^n(a) = (1 - \alpha)\bar{v}^{n-1}(a) + (\alpha)\hat{v}^n(a) \]
Driverless EV optimization

Step 1: Start with a pre-decision state $S^n_t$

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

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

to obtain $x^n$.

Step 3: Update the value function approximation

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

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

$$S^{n}_{t+1} = S^M_t(S^n_t, x^n_t, W_{t+1}(\omega^n))$$

Step 5: Return to step 1.
Driverless EV optimization

Estimating average values:

» The attributes of a car might include:
  • Location
  • Type of car
  • Charge level
  • Driver attributes?

» We estimate values at different levels of aggregation, and then combine them:

\[ \bar{v}^n(a) \approx \sum_{g \in G} w_{a}^{g,n} \bar{v}^{g,n}(a) \]

» The weights are proportional to the inverse of the estimate of the variance plus square of the bias.

» This is a form of variable dimensional learning.
Driverless EV optimization

Variable-dimensional learning

Dimensionality

Weight on each level

Log iterations
Driverless EV optimization

- The value of a vehicle in the future
  - Value function approximation captures charge level, as well as time and location.
  - Hierarchical aggregation accelerated the learning process.
Driverless EV optimization

- Value functions exhibited a variety of shapes
  - Depends on the patterns of trips out of a zone
  - This discouraged the use of simple parametric approximations
Driverless EV optimization

- 22000 zones
- 31560 Trips
- 1000-2000 cars
- Battery capacity: 50-100 KWh
Driverless EV optimization

- No aggregation – (!!) Solution gets worse!

Graph showing profit versus number of iterations.
Driverless EV optimization

Three levels of aggregation - Better

Profit versus number of iterations

$1.6 \times 10^6$
Driverless EV optimization

- Five levels of aggregation

Profit versus number of iterations

$1.6 \times 10^6$
Driverless EV optimization

- Trip requests over time
  - Challenge is to recharge during off-peak periods
Driverless EV optimization

- Heuristic dispatch vs. ADP-based policies
  - Effect of value function approximations on recharging

Heuristic recharging logic

- Recharging during peak period

Recharging controlled by approximate dynamic programming

- Trip requests
- No recharging during peak period

Total battery charge

Actual trips
Outline

The four classes of policies

» Policy function approximations (PFAs)
» Cost function approximations (CFAs)
» Value function approximations (VFAs)
» Direct lookahead policies (DLAs)
» Hybrid direct lookahead/CFA

» Any of the four classes may work best!
Lookahead policies

Planning your next chess move:

» You put your finger on the piece while you think about moves into the future. This is a lookahead policy, illustrated for a problem with discrete actions.
Lookahead policies

- Decision trees:
Lookahead policies

Modeling lookahead policies

» Lookahead policies solve a *lookahead model*, which is an approximation of the future.

» It is important to understand the difference between the:

  • Base model – this is the model we are trying to solve by finding the best policy. This is usually some form of simulator.

  • The lookahead model, which is our approximation of the future to help us make better decisions now.

» The base model is typically a simulator, or it might be the real world.
Lookahead policies

- Lookahead models
  - Use tilde variables with double time indices.

The lookahead model

\[ \tilde{S}_{t}, \tilde{x}_{t}, \tilde{W}_{t}, \ldots, \tilde{S}_{t+1}, \tilde{x}_{t+1}, \tilde{W}_{t+1}, \ldots, \tilde{S}_{t+2}, \tilde{x}_{t+2}, \tilde{W}_{t+2}, \ldots, \tilde{S}_{t+n}, \tilde{x}_{t+n}, \tilde{W}_{t+n}, \ldots \]

\[ (S_0, x_0, W_1, \ldots, S_t, x_t, W_t, \ldots) \]
Lookahead policies

Lookahead models use five classes of approximations:

» Horizon truncation – Replacing a longer horizon problem with a shorter horizon.

» Stage aggregation – Replacing multistage problems with two-stage approximation.

» Outcome aggregation/sampling – Simplifying the exogenous information process.

» Discretization – Of time, states and decisions.

» Dimensionality reduction – We may ignore some variables (such as forecasts) in the lookahead model that we capture in the base model (these become latent variables in the lookahead model).
Lookahead policies

- The lookahead state variable

\[ \tilde{S}_{tt'} = (\tilde{R}_{tt'}, \tilde{D}_{tt'}, \tilde{E}_{tt'}, (\tilde{p}_{tt'}, \tilde{p}_{t', t'-1}, \tilde{p}_{t', t'-2}), (\tilde{\theta}_{tt'}, \tilde{M}_{tt'}), \tilde{f}_{tt'}) \]

Latent variables

- Common simplifications in lookahead models:
  - Ignore the updating of estimates of parameters \( \tilde{\theta}_{tt'} = \tilde{\theta}_t \)
  - Ignore the adaptive updating of forecasts \( \tilde{f}_{tt'}^{D} = f_t^{D} \)
Lookahead policies

We can use this notation to create a policy based on our *lookahead model*:

$X_t^*(S_t) = \arg\max C(S_t, x_t) + \tilde{E} \left\{ \max_{\tilde{\pi} \in \Pi} \tilde{E} \left\{ \sum_{t'=t+1}^{t+H} C(\tilde{S}_{t'}, \tilde{X}_t^*(\tilde{S}_{t'})) \mid \tilde{S}_{t,t+1} \right\} \mid S_t, x_t \right\}$

» Simplest lookahead is deterministic.
Dynamic shortest paths
Dynamic shortest paths
Dynamic shortest paths

The problem

» Finding the best path through a stochastic dynamic network.

The policy

» We can try to solve a stochastic lookahead model (perhaps using approximate dynamic programming).
» We can solve a deterministic lookahead model using point estimates of travel times that are updated from time to time.
» We can solve a parameterized lookahead model, where we use the $\theta$-percentile of the travel time on each link.
Dynamic shortest paths

Modeling:

» Objective function
  • Cost per period
    \[
    C(S_t, X_t^\pi (S_t)) = (X_t^\pi (S_t))^T \hat{c}_t
    \]
    \[
    = \sum_j x_{t,i_t,j}^\pi \hat{c}_{t,i_t,j}
    \]
    = Costs incurred at time t.
  • Total costs:
    \[
    \min_{\pi} \mathbb{E} \sum_{t=0}^T C(S_t, X_t^\pi (S_t))
    \]

» This is the base model.

» We are going to build a policy based on a lookahead model.
Dynamic shortest paths

- Estimates of travel times over the network as of time $t$. These estimates are updated over time.
Dynamic shortest paths

- A time-dependent, deterministic network
Dynamic shortest paths

- A time-dependent, deterministic lookahead network

The base model:

- $t'$ for $t + 4$
- $t'$ for $t + 3$
- $t'$ for $t + 2$
- $t'$ for $t + 1$
- $t'$ for $t$

The lookahead model:
Dynamic shortest paths

- A time-dependent, deterministic lookahead network

The base model

The lookahead model

\[ t' = t + 4 \]
\[ t' = t + 3 \]
\[ t' = t + 2 \]
\[ t' = t + 1 \]
\[ t' = t \]
Dynamic shortest paths

- A time-dependent, deterministic lookahead network
Dynamic shortest paths

Simulating a lookahead policy

We would like to compute

\[ F^\pi = \mathbb{E} \sum_{t=0}^{T} \sum_{i,j} X_{t,ij}^\pi (S_t) \hat{c}_{t,ij} \]

but this is intractable.

Let \( \omega \) be a sample realization of costs

\[ \hat{c}_{t,t',ij} (\omega), \hat{c}_{t+1,t',ij} (\omega), \hat{c}_{t+2,t',ij} (\omega), \ldots \]

Now simulate the policy

\[ \hat{F}^\pi (\omega^n) = \sum_{t=0}^{T} \sum_{i,j} X_{t,ij}^\pi (S_t (\omega^n)) \hat{c}_{t,ij} (\omega^n) \]

Finally, get the average performance

\[ \overline{F}^\pi = \frac{1}{N} \sum_{n=1}^{N} \hat{F}^\pi (\omega^n) \]
Outline

The four classes of policies

- Policy function approximations (PFAs)
- Cost function approximations (CFAs)
- Value function approximations (VFAs)
- Direct lookahead policies (DLAs)
- Hybrid direct lookahead/CFA

Any of the four classes may work best!
Hybrid direct lookahead/CFA

An energy storage problem:
Four (meta)classes of policies

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

2) Cost function approximation (CFAs)
   » \( X^{CFA}(S_t | \theta) = \arg\max_{x \in \tilde{x}(\theta)} C^{\pi}(S_t, x_t | \theta) \)

3) Policies based on value function approximations (VFAs)
   » \( X^{VFA}(S_t) = \arg\max_{x_t} \left( C(S_t, x_t) + V_t^x(S_t^x(S_t, x_t)) \right) \)

4) Direct lookahead policies (DLAs)
   » Deterministic lookahead rolling horizon proc. / model predictive control
     \( X^{LA-D}_t(S_t) = \arg\max_{\tilde{x}_t, \tilde{x}_{t+1}} C(\tilde{S}_t^u, \tilde{x}_t^u) + \sum_{t'=t+1} C(\tilde{S}_{t'}^u, \tilde{x}_{t'}^u) \)
   » Chance constrained programming
     \[ P[A_t x_t \leq f(W)] \leq 1 - \delta \]
   » Stochastic lookahead / stochastic prog. / Monte Carlo tree search
     \( X^{LA-S}_t(S_t) = \arg\max_{\tilde{x}_t, \tilde{x}_{t+1}, \ldots, \tilde{x}_{t+H}} C(\tilde{S}_t^u, \tilde{x}_t^u) + \sum_{\tilde{\omega} \in \Omega_t} p(\tilde{\omega}) \sum_{t'=t+1} C(\tilde{S}_{t'}^u, (\tilde{\omega}), \tilde{x}_{t'}^u(\tilde{\omega})) \)
   » “Robust optimization”
     \( X^{LA-RO}_t(S_t) = \arg\max_{\tilde{x}_t, \ldots, \tilde{x}_{t+H}} \min_{w \in W_t(\theta)} C(\tilde{S}_t^u, \tilde{x}_t^u) + \sum_{t'=t+1} C(\tilde{S}_{t'}^u, (w), \tilde{x}_{t'}^u(\omega)) \)
Hybrid direct lookahead/CFA

Benchmark policy – Deterministic lookahead

\[
X_t^{D-LA}(S_t) = \arg\min_{x_t, (\tilde{x}_{tt'}, t' = t+1, \ldots, t+H)} \left( C(S_t, x_t) + \sum_{t' = t+1}^{t+H} \tilde{c}_{tt'} \tilde{x}_{tt'} \right)
\]

\[
\tilde{x}_{tt'}^{wd} + \beta \tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{gd} \leq f_{tt'}^D
\]

\[
\tilde{x}_{tt'}^{gd} + \tilde{x}_{tt'}^{gr} \leq f_{tt'}^G
\]

\[
\tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{rg} \leq \tilde{R}_{tt'}
\]

\[
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} \leq R^{\text{max}} - \tilde{R}_{tt'}
\]

\[
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{wd} \leq f_{tt'}^E
\]

\[
\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} \leq \gamma^{\text{charge}}
\]

\[
\tilde{x}_{tt'}^{rd} + \tilde{x}_{tt'}^{rg} \leq \gamma^{\text{discharge}}
\]

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

- Lookahead policies peek into the future
  - Optimize over deterministic lookahead model
Hybrid direct lookahead/CFA

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

\[ X_{t+D}^i(S_t) = \arg \min_{\tilde{x}_{t+D}} C(\tilde{S}_t, \tilde{S}_{t+D}) + \sum_{i=1}^I \gamma^{t-i} C(\tilde{S}_{t-i}, \tilde{S}_{t+D}) \]
Hybrid direct lookahead/CFA

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

The real process

The lookahead model

\( X_{t+1}^{LA-P}(S_{t+1}) = \arg \min_{X_{t+1}} C(\tilde{S}_{t+1}, \tilde{X}_{t+1}) + \sum_{i=2}^{\gamma} \gamma^{-i-1} C(\tilde{S}_{t+i}, \tilde{X}_{t+i}) \)

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

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

\[ \gamma_{t+2}^{LA-D}(S_{t+2}) = \arg \min C(\tilde{S}_{t+2}, \tilde{F}_{t+2}) + \sum_{i=t+3}^{t+\infty} \gamma^{t+2} C(\tilde{S}_{i+2}, \tilde{F}_{i+2}) \]
Hybrid direct lookahead/CFA

Benchmark policy – Deterministic lookahead

\[ X_{DLA}^{D}(S_t | \theta) = \arg\min_{x_t, (x_{tt'}, t'=t+1, \ldots, t+H)} \left( C(S_t, x_t) + \left[ \sum_{t'=t+1}^{t+H} \bar{c}_{tt'} \bar{x}_{tt'} \right] \right) \]

\[ \bar{x}_{tt'}^{wd} + \beta \bar{x}_{tt'}^{rd} + \bar{x}_{tt'}^{gd} \leq f_{tt'}^{D} \]

\[ \bar{x}_{tt'}^{gd} + \bar{x}_{tt'}^{gr} \leq f_{tt'}^{G} \]

\[ \bar{x}_{tt'}^{rd} + \bar{x}_{tt'}^{rg} \leq \bar{R}_{tt'} \]

\[ \bar{x}_{tt'}^{wr} + \bar{x}_{tt'}^{gr} \leq R_{tt'}^{\max} - \bar{R}_{tt'} \]

\[ \bar{x}_{tt'}^{wr} + \bar{x}_{tt'}^{wd} \leq \theta_{t'-t} f_{tt'}^{E} \]

\[ \bar{x}_{tt'}^{wr} + \bar{x}_{tt'}^{gr} \leq \gamma_{\text{charge}}^{\text{charge}} \]

\[ \bar{x}_{tt'}^{rd} + \bar{x}_{tt'}^{rg} \leq \gamma_{\text{discharge}}^{\text{discharge}} \]
Hybrid direct lookahead/CFA

- Parametric cost function approximations
  - Replace the constraint
    \[
    \tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{wd} \leq f_{tt'}^E
    \]
    with:
    - Lookup table modified forecasts (one adjustment term for each time \( \tau = t' - t \) in the future):
      \[
      x_{tt'}^{wr} + x_{tt'}^{wd} \leq (\theta_{t' - t}) f_{tt'}^E
      \]
    - We can simulate the performance of a parameterized policy:
      \[
      \bar{F}(\theta, \omega) = \sum_{t=0}^{T} C \left( S_t(\omega), X_t^{\pi} \left( S_t(\omega) \mid \theta \right) \right)
      \]
    - The challenge is to optimize the parameters:
      \[
      \min_{\theta} \mathbb{E} \sum_{t=0}^{T} C \left( S_t, X_t^{\pi} \left( S_t \mid \theta \right) \right)
      \]

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

- One-dimensional contour plots – perfect forecast

$\theta^*_i = 1$ for perfect forecasts.
Hybrid direct lookahead/CFA

- One-dimensional contour plots - uncertain forecast
Hybrid direct lookahead/CFA

- 2-D contours for uncertain forecasts
Hybrid direct lookahead/CFA

Simultaneous perturbation stochastic approximation

» Let:
  • $x^n$ be a $p$ – dimensional vector.
  • $\delta^n$ be a scalar perturbation
  • $Z^n$ be a $p$ – dimensional vector, with each element drawn from a normal (0,1) distribution.

» We can obtain a sampled estimate of the gradient $\nabla_x F(x^n, W^{n+1})$ using two function evaluations: $F(x^n + \delta^n Z^n)$ and $F(x^n - \delta^n Z^n)$

$$
\nabla_x F(x^n, W^{n+1}) =
\begin{bmatrix}
\frac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z^n} \\
2\delta^n Z_1^n \\
\frac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z_2^n} \\
\vdots \\
\frac{F(x^n + \delta^n Z^n) - F(x^n - \delta^n Z^n)}{2\delta^n Z_p^n}
\end{bmatrix}
$$

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

Algorithmic issues:

» Stepsize rules
  • We use a standard gradient-based method:

\[ x^{n+1} = x^n + \alpha \nabla_x F(x^n, W^{n+1}) \]

  • We face the usual challenge of choosing stepsize rules (we used RMSProp), and tuning the parameter(s) in the stepsizerule (this is an optimization problem within the optimization problem).

» We use “mini-batches” to reduce the noise when evaluating the function in SPSA.

» We also used gradient smoothing:

\[ \bar{g}^{n+1} = (1 - \eta)\bar{g}^n + \eta \nabla_x F(x^n, W^{n+1}) \]
\[ x^{n+1} = x^n + \alpha_n \bar{g}^n \]
Hybrid direct lookahead/CFA

- Tuning the parameters
  - Stepsizes tuned for region [0,2].

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

Performance of the SAGF method with time independent lookup table policy

Mini-batch = 40
Mini-batch = 20
Mini-batch = 10
Mini-batch = 1
Mini-batch = 40

© 2019 W.B. Powell
Hybrid direct lookahead/CFA

Performance of the SAGF method with time independent lookup table policy

- Mini-batch = 20
- Mini-batch = 40
- Mini-batch = 10
The four classes of policies

- Policy function approximations (PFAs)
- Cost function approximations (CFAs)
- Value function approximations (VFAs)
- Direct lookahead policies (DLAs)
- Hybrid direct lookahead/CFA

Any of the four classes may work best!
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 (heavy tailed) electricity prices
  - Stationary data

- **Problem class 2** – Best for CFAs
  - Stochastic prices and wind (but not heavy tailed)
  - Stationary data

- **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

- **Problem class 5** – A hybrid policy worked best here
  - Stochastic prices and wind, nonstationary data, noisy forecasts.
An energy storage problem

**The policies**

- **The PFA:**
  - Charge battery when price is below $p_1$
  - Discharge when price is above $p_2$

- **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

- **The lookahead CFA**
  - Use a lookahead policy (deterministic), but with a tunable parameter that improves robustness.
An energy storage problem

Each policy is best on certain problems

» Results are percent of posterior optimal solution

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

- Elements of a dynamic model
- An energy storage illustration
- Modeling uncertainty
- Designing policies
- Educational materials
Educational materials

More information is available at

» Jungle.princeton.edu

Scroll down to “Educational materials” to find:

» A five page book chapter (very quick read)
» A 25 page article in EJOR
» Material for undergraduate course:
  • Sequential decision analytics and modeling
» Material for graduate level course
  • Reinforcement Learning and Stochastic Optimization
Educational materials

Undergraduate course: *Sequential Decision Analytics and Modeling*.

» There is an undergraduate level book that can be accessed at jungle.Princeton.edu or directly from:
  • http://tinyurl.com/sequentialdecisionanalytics

» This uses a teach-by-example style.

» Prerequisite is an undergraduate course in probability and statistics (one chapter uses linear programming, but can be read without a course in linear programming)

» There is an undergraduate course with complete lecture notes in powerpoint at
  • http://www.castlelab.princeton.edu/orf-411/
SEQUENTIAL DECISION ANALYTICS AND MODELING:
Modeling exercises with python

Warren B. Powell
This book can be accessed by going to
https://tinyurl.com/sequentialdecisionanalytics
If you would like to write your own chapter, please go to
https://tinyurl.com/sequentialdecisionanalyticspub

May 27, 2019
3  Adaptive market planning  
   3.1  Narrative  
   3.2  Basic model  
      3.2.1  State variables  
      3.2.2  Decision variables  
      3.2.3  Exogenous information  
      3.2.4  Transition function  
      3.2.5  Objective function  
   3.3  Uncertainty modeling  
   3.4  Designing policies  
   3.5  Policy evaluation  
      3.5.1  Cumulative reward  
      3.5.2  Final reward  
   3.6  Extensions  

4  Learning the best diabetes medication  
   4.1  Narrative  
   4.2  Basic model  
      4.2.1  State variables  
      4.2.2  Decision variables  
      4.2.3  Exogenous information  
      4.2.4  Transition function  
      4.2.5  Objective function  
   4.3  Modeling uncertainty  
   4.4  Designing policies  
   4.5  Policy evaluation  
   4.6  Extensions  
   Problems
Sequential decision analytics

In addition:

- There is a link to a python library of exercises on github. There are 10 modules for 10 different chapters.

- Each chapter follows a specific modeling style. You may write your own chapters at
  - http://tinyurl.com/sequentialdecisionanalyticspub

All material is available at jungle.Princeton.edu
Sequential decision analytics

- There is also a 700+ page graduate level book written entirely around the unified framework.
  - The book can be downloaded from jungle.princeton.edu
  - The prerequisite is also an undergraduate course in probability and statistics, but the presentation is much more methods driven.
  - This is a work in progress. I hope to finish the book by fall, 2020.
Stochastic Optimization and Reinforcement Learning: A unified framework for sequential decisions

Warren B Powell
Thank you!

More material is available at:

jungle.princeton.edu

Please be sure you are on the signup list so I can send you followup information (link is at the top of jungle.Princeton.edu).

Look under “Educational materials“

This presentation will be posted at jungle.Princeton.edu soon.