APPROXIMATE DYNAMIC PROGRAMMING

A SERIES OF LECTURES GIVEN AT

TSINGHUA UNIVERSITY

JUNE 2014

DIMITRI P. BERTSEKAS

Based on the books:

(1) "Neuro-Dynamic Programming," by DPB and J. N. Tsitsiklis, Athena Scientific, 1996

(2) "Dynamic Programming and Optimal Control, Vol. II: Approximate Dynamic Programming," by DPB, Athena Scientific, 2012

(3) "Abstract Dynamic Programming," by DPB, Athena Scientific, 2013

http://www.athenasc.com

For a fuller set of slides, see

http://web.mit.edu/dimitrib/www/publ.html

*Athena is MIT's UNIX-based computing environment. OCW does not provide access to it.

APPROXIMATE DYNAMIC PROGRAMMING

BRIEF OUTLINE I

- **Our subject:**
  - Large-scale DP based on approximations and in part on simulation.
  - This has been a research area of great interest for the last 25 years known under various names (e.g., reinforcement learning, neuro-dynamic programming)
  - Emerged through an enormously fruitful cross-fertilization of ideas from artificial intelligence and optimization/control theory
  - Deals with control of dynamic systems under uncertainty, but applies more broadly (e.g., discrete deterministic optimization)
  - A vast range of applications in control theory, operations research, artificial intelligence, and beyond ...
  - The subject is broad with rich variety of theory/math, algorithms, and applications. Our focus will be mostly on algorithms ... less on theory and modeling

APPROXIMATE DYNAMIC PROGRAMMING

BRIEF OUTLINE II

- **Our aim:**
  - A state-of-the-art account of some of the major topics at a graduate level
  - Show how to use approximation and simulation to address the dual curses of DP: dimensionality and modeling
- **Our 6-lecture plan:**
  - Two lectures on exact DP with emphasis on infinite horizon problems and issues of large-scale computational methods
  - One lecture on general issues of approximation and simulation for large-scale problems
  - One lecture on approximate policy iteration based on temporal differences (TD)/projected equations/Galerkin approximation
  - One lecture on aggregation methods
  - One lecture on Q-learning, and other methods, such as approximation in policy space

APPROXIMATE DYNAMIC PROGRAMMING

LECTURE 1

LECTURE OUTLINE

- Introduction to DP and approximate DP
- Finite horizon problems
- The DP algorithm for finite horizon problems
- Infinite horizon problems
- Basic theory of discounted infinite horizon problems

DP AS AN OPTIMIZATION METHODOLOGY

- Generic optimization problem:

$$\min_{u \in U} g(u)$$

where u is the optimization/decision variable, $g(u)$ is the cost function, and U is the constraint set

- Categories of problems:
  - Discrete (U is finite) or continuous
  - Linear (g is linear and U is polyhedral) or nonlinear
  - Stochastic or deterministic: In stochastic problems the cost involves a stochastic parameter w, which is averaged, i.e., it has the form

$$g(u) = E_w\{G(u, w)\}$$

    where w is a random parameter.

- DP deals with multistage stochastic problems
  - Information about w is revealed in stages
  - Decisions are also made in stages and make use of the available information
  - Its methodology is "different"

BASIC STRUCTURE OF STOCHASTIC DP

- Discrete-time system

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad k = 0, 1, \ldots, N-1$$

  - k: Discrete time
  - x_k: State; summarizes past information that is relevant for future optimization
  - u_k: Control; decision to be selected at time k from a given set
  - w_k: Random parameter (also called "disturbance" or "noise" depending on the context)
  - N: Horizon or number of times control is applied

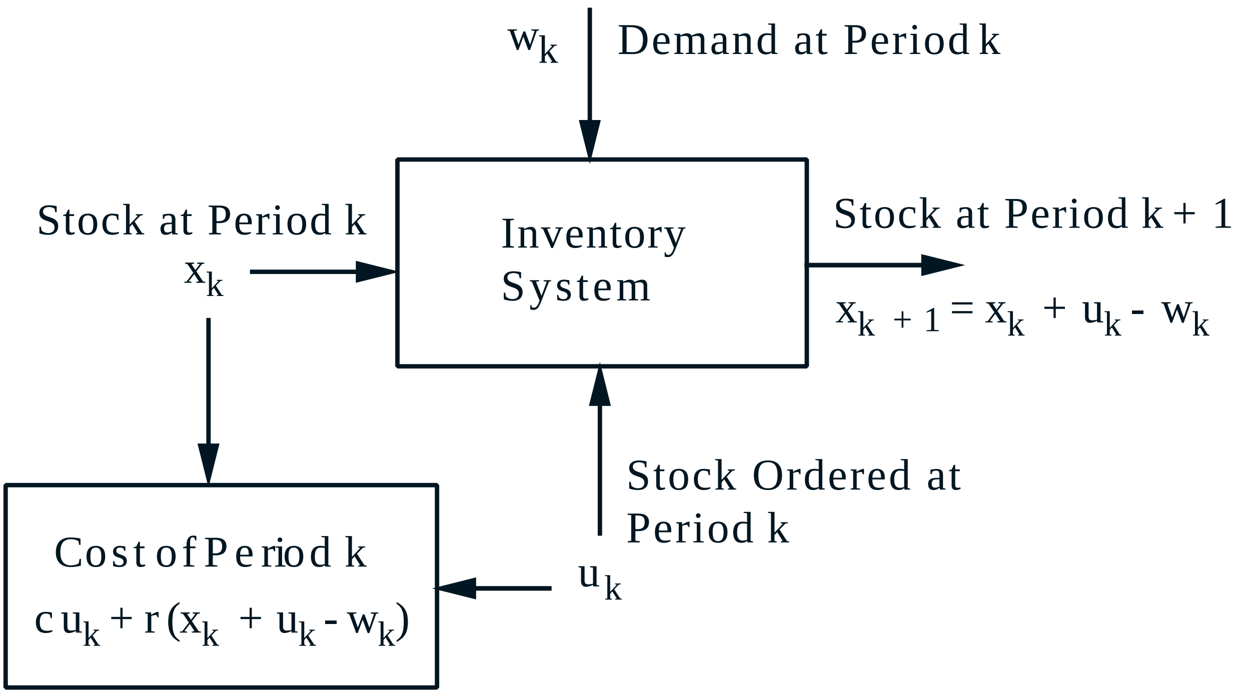
- Cost function that is additive over time

$$E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}$$

- Alternative system description: $P(x_{k+1} \mid x_k, u_k)$

$$x_{k+1} = w_k \quad \text{with } P(w_k \mid x_k, u_k) = P(x_{k+1} \mid x_k, u_k)$$

INVENTORY CONTROL EXAMPLE



- Discrete-time system

$$x_{k+1} = f_k(x_k, u_k, w_k) = x_k + u_k - w_k$$

- Cost function that is additive over time

$$E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}$$

$$= E\left\{ \sum_{k=0}^{N-1} \big(cu_k + r(x_k + u_k - w_k)\big) \right\}$$

ADDITIONAL ASSUMPTIONS

- Probability distribution of w_k does not depend on past values $w_{k-1}, \ldots, w_0$, but may depend on x_k and u_k
  - Otherwise past values of w, x, or u would be useful for future optimization
- The constraint set from which u_k is chosen at time k depends at most on x_k, not on prior x or u
- Optimization over policies (also called feedback control laws): These are rules/functions

$$u_k = \mu_k(x_k), \qquad k = 0, \ldots, N-1$$

that map state/inventory to control/order (closed-loop optimization, use of feedback)

- MAJOR DISTINCTION: We minimize over sequences of functions (mapping inventory to order)

$$\{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$$

NOT over sequences of controls/orders

$$\{u_0, u_1, \ldots, u_{N-1}\}$$

GENERIC FINITE-HORIZON PROBLEM

- System $x_{k+1} = f_k(x_k, u_k, w_k)$, $k = 0, \ldots, N-1$
- Control contraints $u_k \in U_k(x_k)$
- Probability distribution $P_k(\cdot \mid x_k, u_k)$ of w_k
- Policies $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, where μ_k maps states x_k into controls $u_k = \mu_k(x_k)$ and is such that $\mu_k(x_k) \in U_k(x_k)$ for all x_k
- Expected cost of π starting at x_0 is

$$J_\pi(x_0) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\}$$

- Optimal cost function

$$J^*(x_0) = \min_\pi J_\pi(x_0)$$

- Optimal policy π^* satisfies

$$J_{\pi^*}(x_0) = J^*(x_0)$$

When produced by DP, π^* is independent of x_0.

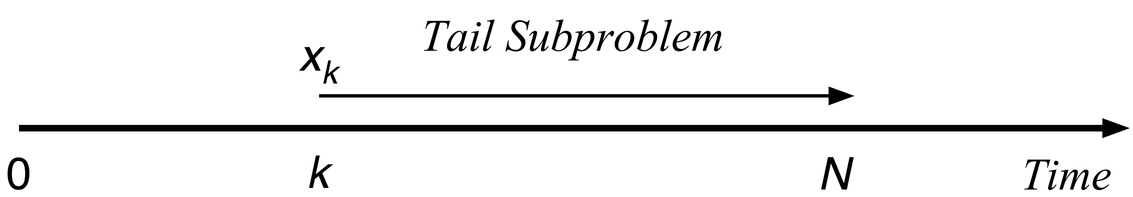
PRINCIPLE OF OPTIMALITY

- Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ be optimal policy

- Consider the "tail subproblem" whereby we are at x_k at time k and wish to minimize the "cost-to-go" from time k to time N

$$E\left\{ g_N(x_N) + \sum_{\ell=k}^{N-1} g_\ell\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big) \right\}$$

and the "tail policy" $\{\mu_k^*, \mu_{k+1}^*, \ldots, \mu_{N-1}^*\}$



- Principle of optimality: The tail policy is optimal for the tail subproblem (optimization of the future does not depend on what we did in the past)

- DP solves ALL the tail subroblems

- At the generic step, it solves ALL tail subproblems of a given time length, using the solution of the tail subproblems of shorter time length

DP ALGORITHM

- Computes for all k and states x_k:

  $J_k(x_k)$: opt. cost of tail problem starting at x_k

- Initial condition:

$$J_N(x_N) = g_N(x_N)$$

Go backwards, $k = N-1, \ldots, 0$, using

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big)\big\},$$

- To solve tail subproblem at time k minimize

kth-stage cost + Opt. cost of next tail problem starting from next state at time $k+1$

- Then $J_0(x_0)$, generated at the last step, is equal to the optimal cost $J^*(x_0)$. Also, the policy

$$\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$$

where $\mu_k^*(x_k)$ minimizes in the right side above for each x_k and k, is optimal

- Proof by induction

PRACTICAL DIFFICULTIES OF DP

- The curse of dimensionality
  - Exponential growth of the computational and storage requirements as the number of state variables and control variables increases
  - Quick explosion of the number of states in combinatorial problems
- The curse of modeling
  - Sometimes a simulator of the system is easier to construct than a model
- There may be real-time solution constraints
  - A family of problems may be addressed. The data of the problem to be solved is given with little advance notice
  - The problem data may change as the system is controlled – need for on-line replanning
- All of the above are motivations for approximation and simulation

A MAJOR IDEA: COST APPROXIMATION

- Use a policy computed from the DP equation where the optimal cost-to-go function J_{k+1} is replaced by an approximation $\tilde{J}_{k+1}$.
- Apply $\overline{\mu}_k(x_k)$, which attains the minimum in

$$\min_{u_k \in U_k(x_k)} E\Big\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\}$$

- Some approaches:
  - (a) Problem Approximation: Use $\tilde{J}_k$ derived from a related but simpler problem
  - (b) Parametric Cost-to-Go Approximation: Use as $\tilde{J}_k$ a function of a suitable parametric form, whose parameters are tuned by some heuristic or systematic scheme (we will mostly focus on this)
    - This is a major portion of Reinforcement Learning/Neuro-Dynamic Programming
  - (c) Rollout Approach: Use as $\tilde{J}_k$ the cost of some suboptimal policy, which is calculated either analytically or by simulation

ROLLOUT ALGORITHMS

- At each k and state x_k, use the control $\overline{\mu}_k(x_k)$ that minimizes in

$$\min_{u_k \in U_k(x_k)} E\big\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big)\big\},$$

where $\tilde{J}_{k+1}$ is the cost-to-go of some heuristic policy (called the base policy).

- Cost improvement property: The rollout algorithm achieves no worse (and usually much better) cost than the base policy starting from the same state.

- Main difficulty: Calculating $\tilde{J}_{k+1}(x)$ may be computationally intensive if the cost-to-go of the base policy cannot be analytically calculated.
  - May involve Monte Carlo simulation if the problem is stochastic.
  - Things improve in the deterministic case (an important application is discrete optimization).
  - Connection w/ Model Predictive Control (MPC).

INFINITE HORIZON PROBLEMS

- Same as the basic problem, but:
  - The number of stages is infinite.
  - The system is stationary.
- Total cost problems: Minimize

$$J_\pi(x_0) = \lim_{N \to \infty} \mathop{E}_{\substack{w_k \\ k=0,1,\ldots}} \left\{ \sum_{k=0}^{N-1} \alpha^k g\big(x_k, \mu_k(x_k), w_k\big) \right\}$$

  - Discounted problems ($\alpha < 1$, bounded g)
  - Stochastic shortest path problems ($\alpha = 1$, finite-state system with a termination state) - we will discuss sparringly
  - Discounted and undiscounted problems with unbounded cost per stage - we will not cover
- Average cost problems - we will not cover
- Infinite horizon characteristics:
  - Challenging analysis, elegance of solutions and algorithms
  - Stationary policies $\pi = \{\mu, \mu, \ldots\}$ and stationary forms of DP play a special role

DISCOUNTED PROBLEMS/BOUNDED COST

- Stationary system

$$x_{k+1} = f(x_k, u_k, w_k), \qquad k = 0, 1, \ldots$$

- Cost of a policy $\pi = \{\mu_0, \mu_1, \ldots\}$

$$J_\pi(x_0) = \lim_{N \to \infty} \mathop{E}_{\substack{w_k \\ k=0,1,\ldots}} \left\{ \sum_{k=0}^{N-1} \alpha^k g\big(x_k, \mu_k(x_k), w_k\big) \right\}$$

with $\alpha < 1$, and g is bounded [for some M, we have $|g(x, u, w)| \leq M$ for all (x, u, w)]

- Optimal cost function: $J^*(x) = \min_\pi J_\pi(x)$

- Boundedness of g guarantees that all costs are well-defined and bounded: $\big|J_\pi(x)\big| \leq \frac{M}{1-\alpha}$

- All spaces are arbitrary - only boundedness of g is important (there are math fine points, e.g. measurability, but they don't matter in practice)

- Important special case: All underlying spaces finite; a (finite spaces) Markovian Decision Problem or MDP

- All algorithms ultimately work with a finite spaces MDP approximating the original problem

SHORTHAND NOTATION FOR DP MAPPINGS

- For any function J of x, denote

$$(TJ)(x) = \min_{u \in U(x)} \mathop{E}_{w} \Big\{ g(x,u,w) + \alpha J\big(f(x,u,w)\big) \Big\}, \ \forall\, x$$

- TJ is the optimal cost function for the one-stage problem with stage cost g and terminal cost function αJ.

- T operates on bounded functions of x to produce other bounded functions of x

- For any stationary policy μ, denote

$$(T_\mu J)(x) = \mathop{E}_{w} \Big\{ g\big(x,\mu(x),w\big) + \alpha J\big(f(x,\mu(x),w)\big) \Big\}, \ \forall\, x$$

- The critical structure of the problem is captured in T and T_μ

- The entire theory of discounted problems can be developed in shorthand using T and T_μ

- True for many other DP problems.

- T and T_μ provide a powerful unifying framework for DP. This is the essence of the book "Abstract Dynamic Programming"

FINITE-HORIZON COST EXPRESSIONS

- Consider an N-stage policy $\pi_0^N = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ with a terminal cost J:

$$J_{\pi_0^N}(x_0) = E\left\{\alpha^N J(x_k) + \sum_{\ell=0}^{N-1} \alpha^\ell g\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big)\right\}$$
$$= E\left\{g\big(x_0, \mu_0(x_0), w_0\big) + \alpha J_{\pi_1^N}(x_1)\right\}$$
$$= (T_{\mu_0} J_{\pi_1^N})(x_0)$$

where $\pi_1^N = \{\mu_1, \mu_2, \ldots, \mu_{N-1}\}$

- By induction we have

$$J_{\pi_0^N}(x) = (T_{\mu_0} T_{\mu_1} \cdots T_{\mu_{N-1}} J)(x), \qquad \forall\ x$$

- For a stationary policy μ the N-stage cost function (with terminal cost J) is

$$J_{\pi_0^N} = T_\mu^N J$$

where T_μ^N is the N-fold composition of T_μ

- Similarly the optimal N-stage cost function (with terminal cost J) is $T^N J$

- $T^N J = T(T^{N-1} J)$ is just the DP algorithm

"SHORTHAND" THEORY – A SUMMARY

- Infinite horizon cost function expressions [with $J_0(x) \equiv 0$]

$$J_\pi(x) = \lim_{N\to\infty} (T_{\mu_0} T_{\mu_1} \cdots T_{\mu_N} J_0)(x), \quad J_\mu(x) = \lim_{N\to\infty} (T_\mu^N J_0)(x)$$

- Bellman's equation: $J^* = TJ^*, \ \ J_\mu = T_\mu J_\mu$
- Optimality condition:

$$\mu\text{: optimal} \quad <==> \quad T_\mu J^* = TJ^*$$

- Value iteration: For any (bounded) J

$$J^*(x) = \lim_{k\to\infty} (T^k J)(x), \qquad \forall \ x$$

- Policy iteration: Given μ^k,
  - Policy evaluation: Find J_{μ^k} by solving

$$J_{\mu^k} = T_{\mu^k} J_{\mu^k}$$

  - Policy improvement: Find μ^{k+1} such that

$$T_{\mu^{k+1}} J_{\mu^k} = TJ_{\mu^k}$$

TWO KEY PROPERTIES

- Monotonicity property: For any J and J' such that $J(x) \leq J'(x)$ for all x, and any μ

$$(TJ)(x) \leq (TJ')(x), \qquad \forall\, x,$$

$$(T_\mu J)(x) \leq (T_\mu J')(x), \qquad \forall\, x.$$

- Constant Shift property: For any J, any scalar r, and any μ

$$\big(T(J + re)\big)(x) = (TJ)(x) + \alpha r, \qquad \forall\, x,$$

$$\big(T_\mu(J + re)\big)(x) = (T_\mu J)(x) + \alpha r, \qquad \forall\, x,$$

where e is the unit function $[e(x) \equiv 1]$.

- Monotonicity is present in all DP models (undiscounted, etc)
- Constant shift is special to discounted models
- Discounted problems have another property of major importance: T and T_μ are contraction mappings (we will show this later)

CONVERGENCE OF VALUE ITERATION

- For all bounded J,

$$J^*(x) = \lim_{k\to\infty} (T^k J)(x), \qquad \text{for all } x$$

Proof: For simplicity we give the proof for $J \equiv 0$. For any initial state x_0, and policy $\pi = \{\mu_0, \mu_1, \ldots\}$,

$$\begin{aligned} J_\pi(x_0) &= E\left\{ \sum_{\ell=0}^{\infty} \alpha^\ell g\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big) \right\} \\ &= E\left\{ \sum_{\ell=0}^{k-1} \alpha^\ell g\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big) \right\} \\ &\qquad + E\left\{ \sum_{\ell=k}^{\infty} \alpha^\ell g\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big) \right\} \end{aligned}$$

The tail portion satisfies

$$\left| E\left\{ \sum_{\ell=k}^{\infty} \alpha^\ell g\big(x_\ell, \mu_\ell(x_\ell), w_\ell\big) \right\} \right| \le \frac{\alpha^k M}{1-\alpha},$$

where $M \ge |g(x,u,w)|$. Take min over π of both sides, then lim as $k \to \infty$. **Q.E.D.**

BELLMAN'S EQUATION

- The optimal cost function J^* is a solution of Bellman's equation, $J^* = TJ^*$, i.e., for all x,

$$J^*(x) = \min_{u \in U(x)} \mathop{E}_{w} \Big\{ g(x,u,w) + \alpha J^*\big(f(x,u,w)\big) \Big\}$$

Proof: For all x and k,

$$J^*(x) - \frac{\alpha^k M}{1-\alpha} \le (T^k J_0)(x) \le J^*(x) + \frac{\alpha^k M}{1-\alpha},$$

where $J_0(x) \equiv 0$ and $M \ge |g(x,u,w)|$. Applying T to this relation, and using Monotonicity and Constant Shift,

$$\begin{aligned}(TJ^*)(x) - \frac{\alpha^{k+1} M}{1-\alpha} &\le (T^{k+1} J_0)(x) \\ &\le (TJ^*)(x) + \frac{\alpha^{k+1} M}{1-\alpha}\end{aligned}$$

Taking the limit as $k \to \infty$ and using the fact

$$\lim_{k \to \infty} (T^{k+1} J_0)(x) = J^*(x)$$

we obtain $J^* = TJ^*$. **Q.E.D.**

THE CONTRACTION PROPERTY

- Contraction property: For any bounded functions J and J', and any μ,

$$\max_x \big|(TJ)(x) - (TJ')(x)\big| \leq \alpha \max_x \big|J(x) - J'(x)\big|,$$

$$\max_x \big|(T_\mu J)(x) - (T_\mu J')(x)\big| \leq \alpha \max_x \big|J(x) - J'(x)\big|.$$

Proof: Denote $c = \max_{x \in S} \big|J(x) - J'(x)\big|$. Then

$$J(x) - c \leq J'(x) \leq J(x) + c, \qquad \forall\ x$$

Apply T to both sides, and use the Monotonicity and Constant Shift properties:

$$(TJ)(x) - \alpha c \leq (TJ')(x) \leq (TJ)(x) + \alpha c, \qquad \forall\ x$$

Hence

$$\big|(TJ)(x) - (TJ')(x)\big| \leq \alpha c, \qquad \forall\ x.$$

Q.E.D.

- Note: This implies that J^* is the unique solution of $J^* = TJ^*$, and J_μ is the unique solution of $J^* = TJ^*$

NEC. AND SUFFICIENT OPT. CONDITION

- A stationary policy μ is optimal if and only if $\mu(x)$ attains the minimum in Bellman's equation for each x; i.e.,

$$TJ^* = T_\mu J^*,$$

or, equivalently, for all x,

$$\mu(x) \in \arg \min_{u \in U(x)} \mathop{E}_{w} \left\{ g(x, u, w) + \alpha J^*\big(f(x, u, w)\big) \right\}$$

Proof: If $TJ^* = T_\mu J^*$, then using Bellman's equation ($J^* = TJ^*$), we have

$$J^* = T_\mu J^*,$$

so by uniqueness of the fixed point of T_μ, we obtain $J^* = J_\mu$; i.e., μ is optimal.

- Conversely, if the stationary policy μ is optimal, we have $J^* = J_\mu$, so

$$J^* = T_\mu J^*.$$

Combining this with Bellman's Eq. ($J^* = TJ^*$), we obtain $TJ^* = T_\mu J^*$. **Q.E.D.**