Approximate Dynamic Programming Based on Value and Policy Iteration

Dimitri Bertsekas
Dept. of Electrical Engineering
and Computer Science
M.I.T.

November 2006

BELLMAN AND THE DUAL CURSES

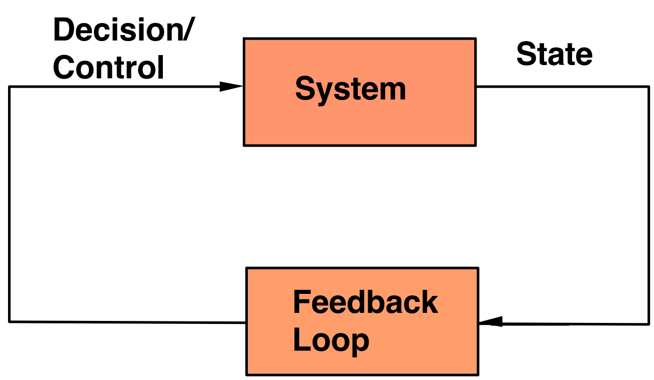
- Dynamic Programming (DP) is very broadly applicable, but it suffers from:
  - Curse of dimensionality
  - Curse of modeling
- We address “complexity” by using low-dimensional parametric approximations
- We allow simulators in place of models
- Unlimited applications in planning, resource allocation, stochastic control, discrete optimization
- Application is an art … but guided by substantial theory

OUTLINE

- **Main NDP framework**
- **Primary focus on approximation in value space, and value and policy iteration-type methods**
  - Rollout
  - Projected value iteration/LSPE for policy evaluation
  - Temporal difference methods
- **Methods not discussed: approximate linear programming, approximation in policy space**
- **References:**
  - Neuro-Dynamic Programming (1996, Bertsekas + Tsitsiklis)
  - Reinforcement Learning (1998, Sutton + Barto)
  - Dynamic Programming: 3rd Edition (Jan. 2007, Bertsekas)
  - Recent papers with V. Borkar, A. Nedic, and J. Yu
- **Papers and this talk can be downloaded from http://web.mit.edu/dimitrib/www/home.html**

DYNAMIC PROGRAMMING / DECISION AND CONTROL

- **Main ingredients:**
  - Dynamic system; state evolving in discrete time
  - Decision/control applied at each time
  - Cost is incurred at each time
  - There may be noise & model uncertainty
  - There is state feedback used to determine the control



APPLICATIONS

- Extremely broad range
- Sequential decision contexts
  - Planning (shortest paths, schedules, route planning, supply chain)
  - Resource allocation over time (maintenance, power generation)
  - Finance (investment over time, optimal stopping/option valuation)
  - Automatic control (vehicles, machines)
- Nonsequential decision contexts
  - Combinatorial/discrete optimization (breakdown solution into stages)
  - Branch and Bound/ Integer programming
- Applies to both deterministic and stochastic problems

KEY DP RESULT: BELLMAN'S EQUATION

- Optimal decision at the current state minimizes the expected value of

  Current stage cost

  + Future stages cost

  (starting from the next state

  - using opt. policy)
- Extensive mathematical methodology
- Applies to both discrete and continuous systems (and hybrids)
- Dual curses of dimensionality/modeling

APPROXIMATION IN VALUE SPACE

- Use one-step lookahead with an approximate cost
- At the current state select decision that minimizes the expected value of

  Current stage cost

  + Approximate future stages cost

  (starting from the next state)

- Important issues:
  - How to approximate/parametrize cost of a state
  - How to understand and control the effects of approximation
- Alternative (will not be discussed): Approximation in policy space (direct parametrization/optimization of policies)

METHODS TO COMPUTE AN APPROXIMATE COST

- Rollout algorithms
  - Use the cost of the heuristic (or a lower bound) as cost approximation
  - Use simulation to obtain this cost, starting from the state of interest
- Parametric approximation algorithms
  - Use a functional approximation to the optimal cost; e.g., linear combination of basis functions
  - Select the weights of the approximation
  - Systematic DP-related policy and value iteration methods (TD-Lambda, Q-Learning, LSPE, LSTD, etc)

APPROXIMATE POLICY ITERATION

- Given a current policy, define a new policy as follows:

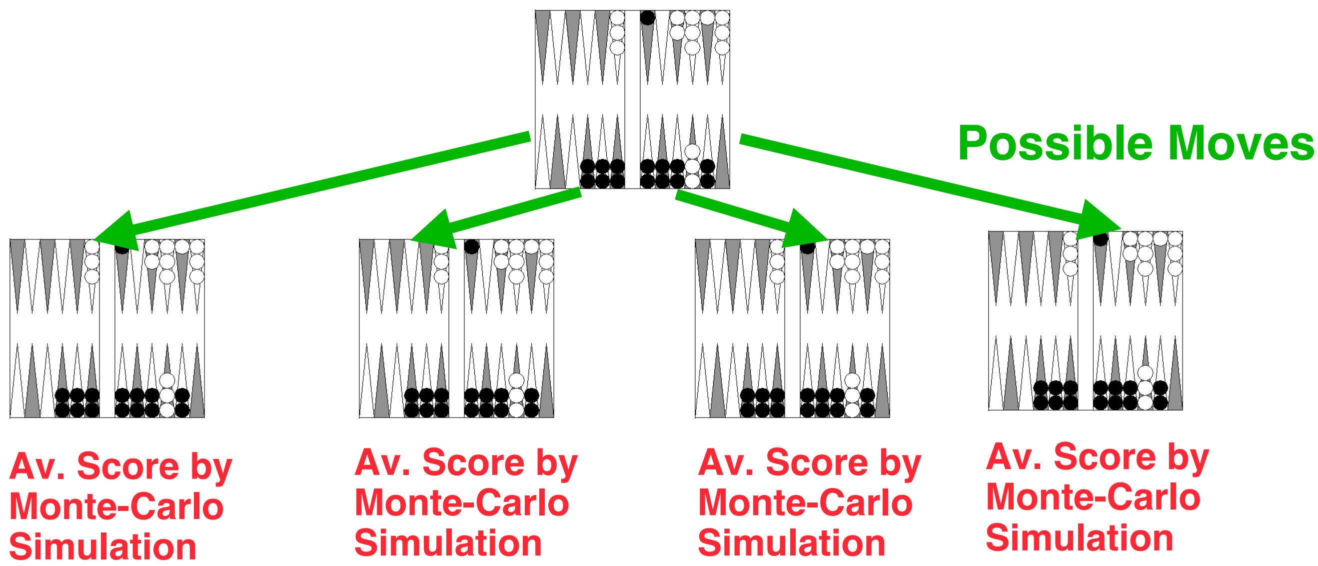
  At each state minimize

  Current stage cost + cost-to-go of current policy (starting from the next state)

- Policy improvement result: New policy has improved performance over current policy
- If the cost-to-go is approximate, the improvement is "approximate"
- Oscillation around the optimal; error bounds

ROLLOUT ONE-STEP POLICY ITERATION

- On-line (approximate) cost-to-go calculation by simulation of some base policy (heuristic)
- Rollout: Use action w/ best simulation results
- Rollout is one-step policy iteration

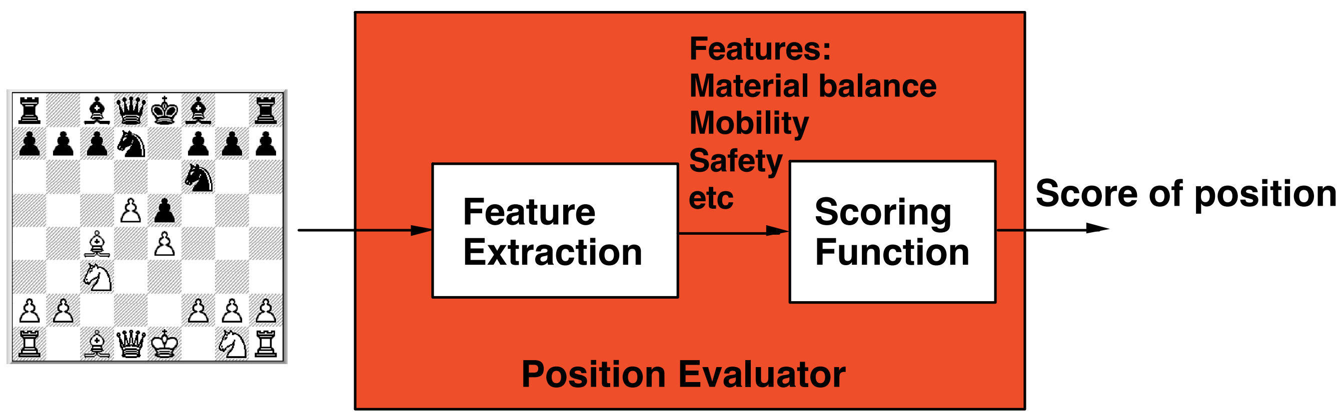


COST IMPROVEMENT PROPERTY

- Generic result: Rollout improves on base heuristic
- In practice, substantial improvements over the base heuristic(s) have been observed
- Major drawback: Extensive Monte-Carlo simulation (for stochastic problems)
- Excellent results with (deterministic) discrete and combinatorial problems
- Interesting special cases:
  - The classical open-loop feedback control policy (base heuristic is the optimal open-loop policy)
  - Model predictive control (major applications in control systems)

PARAMETRIC APPROXIMATION: CHESS PARADIGM

- Chess playing computer programs
- State = board position
- Score of position: "Important features" appropriately weighted



COMPUTING WEIGHTS TRAINING

- **In chess: Weights are "hand-tuned"**
- **In more sophisticated methods: Weights are determined by using simulation-based training algorithms**
- **Temporal Differences TD(λ), Least Squares Policy Evaluation LSPE(λ), Least Squares Temporal Differences LSTD(λ)**
- **All of these methods are based on DP ideas of policy iteration and value iteration**

FOCUS ON APPROX. POLICY EVALUATION

- Consider stationary policy μ w/ cost function J
- Satisfies Bellman's equation:

$$J = T(J) = g_\mu + \alpha P_\mu J \quad \text{(discounted case)}$$

- Subspace approximation

$$J \sim \Phi r$$

Φ: matrix of basis functions

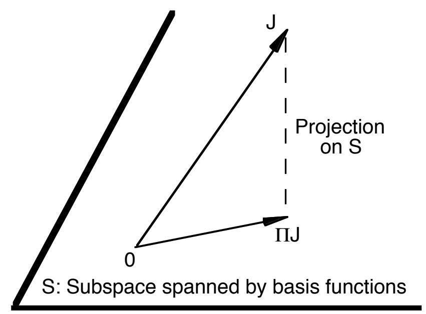
r: parameter vector

DIRECT AND INDIRECT APPROACHES

- **Direct:** Use simulated cost samples and least-squares fit

  $J \sim \Pi J$

  Approximate the cost

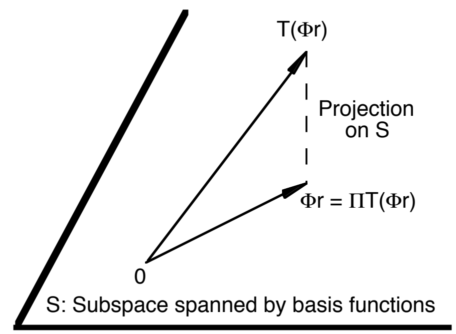


Direct Mehod: Projection of cost vector J

- **Indirect:** Solve a projected form of Bellman’s equation

  $\Phi r = \Pi T(\Phi r)$

  Approximate the equation



Indirect method: Solving a projected form of Bellman’s equation

DIRECT APPROACH

- **Minimize over r; least squares**

  $$\sum \left(\text{Simulated cost sample of } J(i) - (\Phi r)_i \right)^2$$

- **Each state is weighted proportionally to its appearance in the simulation**
- **Works even with nonlinear function approximation (in place of Φr)**
- **Gradient or special least squares methods can be used**
- **Problem with large error variance**

INDIRECT POLICY EVALUATION

- **Simulation-based methods that solve the Projected Bellman Equation (PBE):**
  - **TD(λ): (Sutton 1988) - stochastic approximation method, convergence (Tsitsiklis and Van Roy, 1997)**
  - **LSTD(λ): (Barto & Bradtke 1996, Boyan 2002) - solves by matrix inversion a simulation generated approximation to PBE, convergence (Nedic, Bertsekas, 2003), optimal convergence rate (Konda 2002)**
  - **LSPE(λ): (Bertsekas w/ Ioffe 1996, Borkar, Nedic , 2003, 2004, Yu 2006) - uses projected value iteration to find fixed point of PBE**
- **Key questions:**
  - **When does the PBE have a solution?**
  - **Convergence, rate of convergence, error bounds**

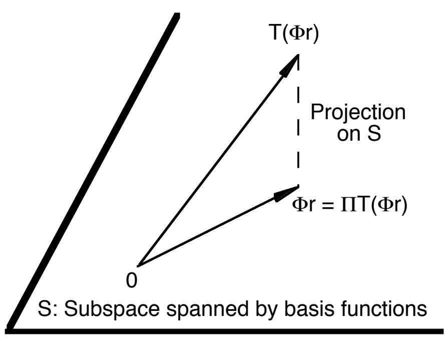
LEAST SQUARES POLICY EVALUATION (LSPE)

- Consider α-discounted Markov Decision Problem (finite state and control spaces)
- We want to approximate the solution of Bellman equation:

$$J = T(J) = g_\mu + \alpha P_\mu J$$

- We solve the projected Bellman equation

$$\Phi r = \Pi T(\Phi r)$$

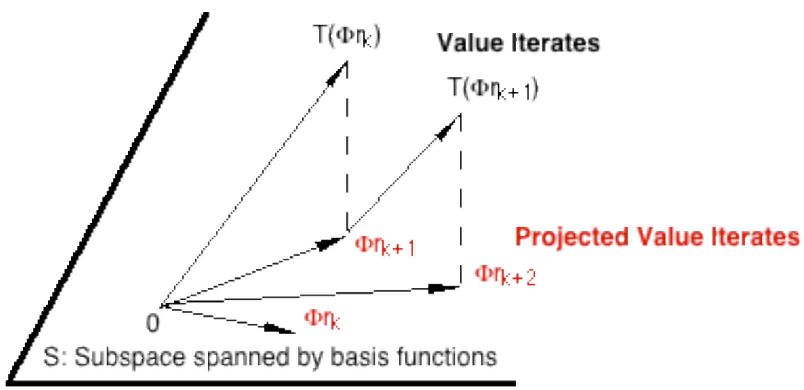


Indirect method: Solving a projected form of Bellman's equation

PROJECTED VALUE ITERATION (PVI)

- **Value iteration:** $J_{t+1} = T(J_t)$
- **Projected Value iteration:** $\Phi r_{t+1} = \Pi T(\Phi r_t)$

  where Φ is a matrix of basis functions and Π is projection w/ respect to some weighted Euclidean norm $||\cdot||$
- **Norm mismatch issue:**
  - Π is nonexpansive with respect to $||\cdot||$
  - T is a contraction w/ respect to the sup norm
- **Key Question:** When is ΠT a contraction w/ respect to some norm?



PROJECTION W/ RESPECT TO DISTRIBUTION NORM

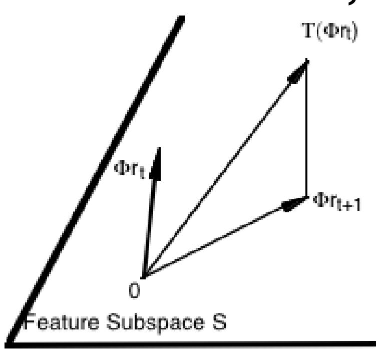
- Consider the **steady-state distribution norm** $||.||_\xi$
  - Weight of ith component: the steady-state probability ξ_j of state j in the Markov chain corresponding to the policy evaluated

- **Remarkable Fact:** If Π is projection w/ respect to the distribution norm, then ΠT is a contraction for discounted problems
- Key property

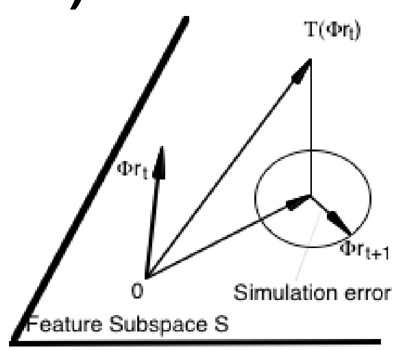
$$||Pz||_\xi \leq ||z||_\xi$$

LSPE: SIMULATION-BASED IMPLEMENTATION

- **Key Fact: $\Phi r_{t+1} = \Pi T(\Phi r_t)$ can be implemented by simulation**
- **$\Phi r_{t+1} = \Pi T(\Phi r_t)$ + Diminishing simulation noise**
- **Interesting convergence theory (see papers at www site)**
- **Optimal convergence rate; much better than TD(λ), same as LSTD (Yu and Bertsekas, 2006)**



Value Iteration with Linear Function Approximation



Simulation-Based Value Iteration with Linear Function Approximation

LSPE DETAILS

- PVI:

$$r_{k+1} = \arg\min_r \sum_{i=1}^{n} \xi_i \left(\phi(i)'r - \sum_{j=1}^{n} p_{ij} \big(g(i,j) + \alpha\phi(j)'r_k \big) \right)^2$$

- LSPE: Generate an infinitely long trajectory $(i_0, i_1, \ldots)$ and set

$$r_{k+1} = \arg\min_r \sum_{t=0}^{k} \big(\phi(i_t)'r - g(i_t, i_{t+1}) - \alpha\phi(i_{t+1})'r_k \big)^2$$

LSPE - PVI COMPARISON

- PVI:

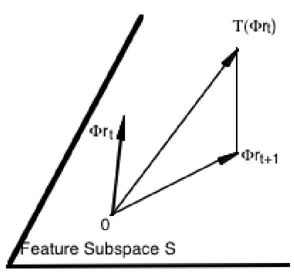
$$r_{k+1} = \left(\sum_{i=1}^{n} \xi_i \, \phi(i)\phi(i)'\right)^{-1} \left(\sum_{i=1}^{n} \xi_i \, \phi(i) \sum_{j=1}^{n} p_{ij} \big(g(i,j) + \alpha\phi(j)'r_k\big)\right)$$

- LSPE:

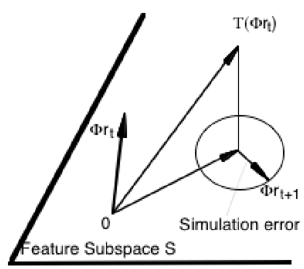
$$r_{k+1} = \left(\sum_{i=1}^{n} \hat{\xi}_{i,k} \, \phi(i)\phi(i)'\right)^{-1} \left(\sum_{i=1}^{n} \hat{\xi}_{i,k} \, \phi(i) \sum_{j=1}^{n} \hat{p}_{ij,k} \big(g(i,j) + \alpha\phi(j)'r_k\big)\right)$$

where $\hat{\xi}_{i,k}$ and $\hat{p}_{ij,k}$ are empirical frequencies

$$\hat{\xi}_{i,k} = \frac{\sum_{t=0}^{k} \delta(i_t = i)}{k+1}, \qquad \hat{p}_{ij,k} = \frac{\sum_{t=0}^{k} \delta(i_t = i, i_{t+1} = j)}{\sum_{t=0}^{k} \delta(i_t = i)}$$



Value Iteration with Linear Function Approximation



Simulation-Based Value Iteration with Linear Function Approximation

LSTD
LEAST SQUARES TEMPORAL DIFFERENCE METHODS

- Generate an infinitely long trajectory $(i_0, i_1, \ldots)$ and set

$$\hat{r} = \arg\min_{r \in \Re^s} \sum_{t=0}^{k} \left(\phi(i_t)'r - g(i_t, i_{t+1}) - \alpha\phi(i_{t+1})'\hat{r}\right)^2$$

Not a least squares problem, but can be solved as a linear system of equations

- Compare with LSPE

$$r_{k+1} = \arg\min_{r \in \Re^s} \sum_{t=0}^{k} \left(\phi(i_t)'r - g(i_t, i_{t+1}) - \alpha\phi(i_{t+1})'r_k\right)^2$$

- LSPE is one fixed point iteration for solving the LSTD system
- Same convergence rate; asymptotically coincide

LSPE(λ), LSTD(λ)

- For $\lambda \in [0, 1)$, define the mapping

$$T^{(\lambda)} = (1 - \lambda) \sum_{t=0}^{\infty} \lambda^t T^{t+1}$$

It has the same fixed point J_μ as T

- Apply PVI, LSPE, LSTD to $T^{(\lambda)}$

- $T^{(\lambda)}$ and $\Pi T^{(\lambda)}$ are contractions of modulus

$$\alpha_\lambda = \frac{\alpha(1 - \lambda)}{1 - \alpha\lambda}$$

ERROR BOUNDS

- Same convergence properties, fixed point depends on λ
- Error bounds

$$\|J_\mu - \Phi r_\lambda\|_\xi \leq \frac{1}{\sqrt{1-\alpha_\lambda^2}} \|J_\mu - \Pi J_\mu\|_\xi,$$

where Φr_λ is the fixed point of $\Pi T^{(\lambda)}$ and $\alpha_\lambda = \alpha(1-\lambda)/(1-\alpha\lambda)$

- As $\lambda \to 0$, error increases, but susceptibility to noise improves

EXTENSIONS

- Straightforward extension to stochastic shortest path problems (no discounting, but T is contraction)
- Not so straightforward extension to average cost problems (T is not a contraction, Tsitsiklis and Van Roy 1999, Yu and Bertsekas 2006)
- PVI/LSPE is designed for approx. policy evaluation. How does it work when embedded within approx. policy iteration?
- There are limited classes of problems where PVI/LSPE works with T: nonlinear in $\Phi r_{t+1} = \Pi T(\Phi r_t)$

CONCLUDING REMARKS

- NDP is a broadly applicable methodology; addresses large problems that are intractable in other ways
- No need for a detailed model; a simulator suffices
- Interesting theory for parametric approximation - challenging to apply
- Simple theory for rollout - consistent success (when Monte Carlo is not overwhelming)
- Successful application is an art
- Many questions remain