6.231 DYNAMIC PROGRAMMING

LECTURE 4

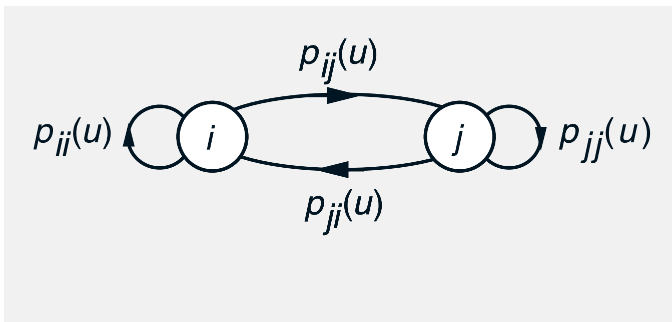
LECTURE OUTLINE

- Review of approximation in value space
- Approximate VI and PI
- Projected Bellman equations
- Matrix form of the projected equation
- Simulation-based implementation
- LSTD and LSPE methods
- Optimistic versions
- Multistep projected Bellman equations
- Bias-variance tradeoff

REVIEW

DISCOUNTED MDP

- System: Controlled Markov chain with states $i = 1, \ldots, n$, and finite control set $U(i)$ at state i
- Transition probabilities: $p_{ij}(u)$



- Cost of a policy $\pi = \{\mu_0, \mu_1, \ldots\}$ starting at state i:

$$J_\pi(i) = \lim_{N \to \infty} E\left\{ \sum_{k=0}^{N} \alpha^k g\big(i_k, \mu_k(i_k), i_{k+1}\big) \mid i_0 = i \right\}$$

with $\alpha \in [0, 1)$

- Shorthand notation for DP mappings

$$(TJ)(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i, u, j) + \alpha J(j)\big), \quad i = 1, \ldots, n,$$

$$(T_\mu J)(i) = \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\Big(g\big(i, \mu(i), j\big) + \alpha J(j)\Big), \quad i = 1, \ldots, n$$

"SHORTHAND" THEORY – A SUMMARY

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

$$J^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha J^*(j)\big), \quad \forall\ i$$

$$J_\mu(i) = \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\Big(g\big(i,\mu(i),j\big) + \alpha J_\mu(j)\Big), \quad \forall\ i$$

- Optimality condition:

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

i.e.,

$$\mu(i) \in \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha J^*(j)\big), \quad \forall\ i$$

THE TWO MAIN ALGORITHMS: VI AND PI

- Value iteration: For any $J \in \Re^n$

$$J^*(i) = \lim_{k \to \infty} (T^k J)(i), \qquad \forall\ i = 1, \ldots, n$$

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

$$J_{\mu^k}(i) = \sum_{j=1}^{n} p_{ij}\big(\mu^k(i)\big)\Big(g\big(i, \mu^k(i), j\big) + \alpha J_{\mu^k}(j)\Big),\ \ i = 1, \ldots, n$$

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

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

$$\mu^{k+1}(i) \in \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i, u, j) + \alpha J_{\mu^k}(j)\big), \quad \forall\ i$$

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

- Policy evaluation is equivalent to solving an $n \times n$ linear system of equations
- For large n, exact PI is out of the question (even though it terminates finitely)

APPROXIMATION IN VALUE SPACE

- Approximate J^* or J_μ from a parametric class $\tilde{J}(i;r)$, where i is the current state and $r = (r_1, \ldots, r_s)$ is a vector of "tunable" scalars weights

- Think n: HUGE, s: (Relatively) SMALL

- Many types of approximation architectures [i.e., parametric classes $\tilde{J}(i;r)$] to select from

- Any $r \in \Re^s$ defines a (suboptimal) one-step lookahead policy

$$\tilde{\mu}(i) = \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha \tilde{J}(j;r)\big), \quad \forall\, i$$

- We want to find a "good" r

- We will focus mostly on linear architectures

$$\tilde{J}(r) = \Phi r$$

where Φ is an $n \times s$ matrix whose columns are viewed as basis functions

LINEAR APPROXIMATION ARCHITECTURES

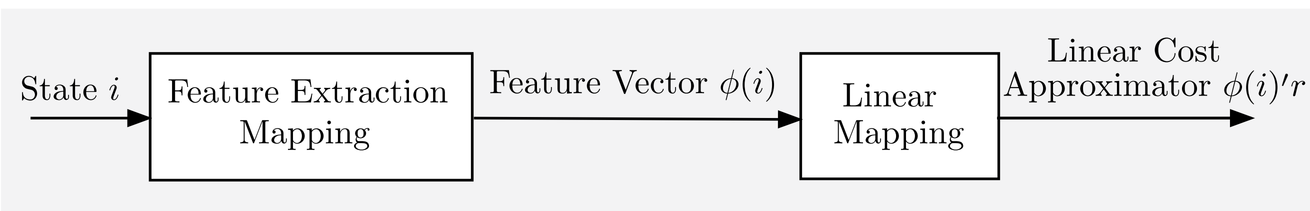
- We have

$$\tilde{J}(i;r) = \phi(i)'r, \qquad i = 1,\ldots,n$$

where $\phi(i)'$, $i = 1,\ldots,n$ is the ith row of Φ, or

$$\tilde{J}(r) = \Phi r = \sum_{j=1}^{s} \Phi_j r_j$$

where Φ_j is the jth column of Φ



- This is approximation on the subspace

$$S = \{\Phi r \mid r \in \Re^s\}$$

spanned by the columns of Φ (basis functions)

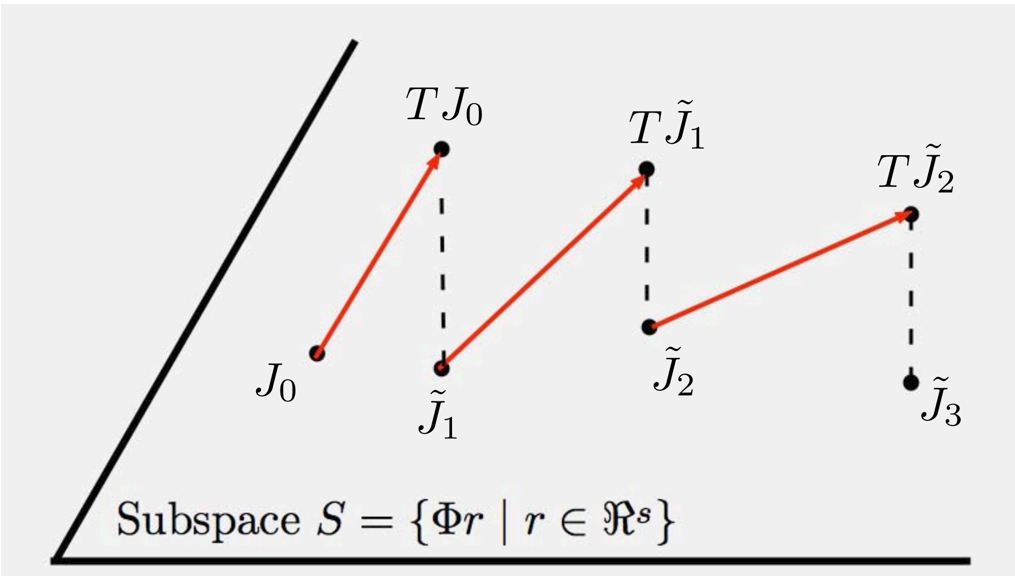
- Many examples of feature types: Polynomial approximation, radial basis functions, etc

- Instead of computing J_μ or J^*, which is huge-dimensional, we compute the low-dimensional $r = (r_1,\ldots,r_s)$ using low-dimensional calculations

APPROXIMATE VALUE ITERATION

APPROXIMATE (FITTED) VI

- Approximates sequentially $J_k(i) = (T^k J_0)(i)$, $k = 1, 2, \ldots$, with $\tilde{J}_k(i; r_k)$
- The starting function J_0 is given (e.g., $J_0 \equiv 0$)
- Approximate (Fitted) Value Iteration: A sequential "fit" to produce $\tilde{J}_{k+1}$ from $\tilde{J}_k$, i.e., $\tilde{J}_{k+1} \approx T\tilde{J}_k$ or (for a single policy μ) $\tilde{J}_{k+1} \approx T_\mu \tilde{J}_k$



Fitted Value Iteration

- After a large enough number N of steps, $\tilde{J}_N(i; r_N)$ is used as approximation $\tilde{J}(i; r)$ to $J^*(i)$
- Possibly use (approximate) projection Π with respect to some projection norm,

$$\tilde{J}_{k+1} \approx \Pi T \tilde{J}_k$$

WEIGHTED EUCLIDEAN PROJECTIONS

- Consider a weighted Euclidean norm

$$\|J\|_\xi = \sqrt{\sum_{i=1}^{n} \xi_i \big(J(i)\big)^2},$$

where $\xi = (\xi_1, \ldots, \xi_n)$ is a positive distribution ($\xi_i > 0$ for all i).

- Let Π denote the projection operation onto

$$S = \{\Phi r \mid r \in \Re^s\}$$

with respect to this norm, i.e., for any $J \in \Re^n$,

$$\Pi J = \Phi r^*$$

where

$$r^* = \arg\min_{r \in \Re^s} \|\Phi r - J\|_\xi^2$$

- Recall that weighted Euclidean projection can be implemented by simulation and least squares, i.e., sampling $J(i)$ according to ξ and solving

$$\min_{r \in \Re^s} \sum_{t=1}^{k} \big(\phi(i_t)' r - J(i_t)\big)^2$$

FITTED VI - NAIVE IMPLEMENTATION

- Select/sample a "small" subset I_k of representative states
- For each $i \in I_k$, given $\tilde{J}_k$, compute

$$(T\tilde{J}_k)(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha \tilde{J}_k(j;r)\big)$$

- "Fit" the function $\tilde{J}_{k+1}(i;r_{k+1})$ to the "small" set of values $(T\tilde{J}_k)(i)$, $i \in I_k$ (for example use some form of approximate projection)
- Simulation can be used for "model-free" implementation
- Error Bound: If the fit is uniformly accurate within $\delta > 0$, i.e.,

$$\max_i |\tilde{J}_{k+1}(i) - T\tilde{J}_k(i)| \le \delta,$$

then

$$\limsup_{k \to \infty} \max_{i=1,\ldots,n} \big(\tilde{J}_k(i,r_k) - J^*(i)\big) \le \frac{2\alpha\delta}{(1-\alpha)^2}$$

- But there is a potential problem!

AN EXAMPLE OF FAILURE

- Consider two-state discounted MDP with states 1 and 2, and a single policy.
  - Deterministic transitions: $1 \to 2$ and $2 \to 2$
  - Transition costs $\equiv 0$, so $J^*(1) = J^*(2) = 0$.
- Consider (exact) fitted VI scheme that approximates cost functions within $S = \big\{(r, 2r) \mid r \in \Re\big\}$ with a weighted least squares fit; here $\Phi = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$
- Given $\tilde{J}_k = (r_k, 2r_k)$, we find $\tilde{J}_{k+1} = (r_{k+1}, 2r_{k+1})$, where $\tilde{J}_{k+1} = \Pi_\xi(T\tilde{J}_k)$, with weights $\xi = (\xi_1, \xi_2)$:

$$r_{k+1} = \arg\min_r \Big[\xi_1\big(r - (T\tilde{J}_k)(1)\big)^2 + \xi_2\big(2r - (T\tilde{J}_k)(2)\big)^2\Big]$$

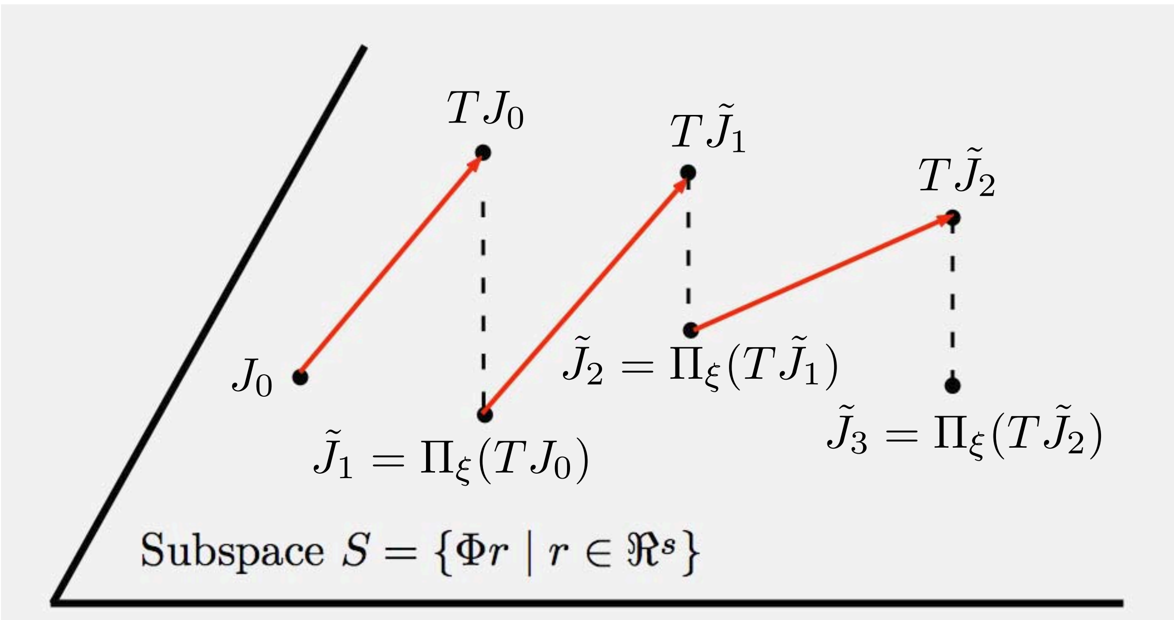
- With straightforward calculation

$$r_{k+1} = \alpha\beta r_k, \qquad \text{where } \beta = 2(\xi_1 + 2\xi_2)/(\xi_1 + 4\xi_2) > 1$$

- So if $\alpha > 1/\beta$ (e.g., $\xi_1 = \xi_2 = 1$), the sequence $\{r_k\}$ diverges and so does $\{\tilde{J}_k\}$.
- Difficulty is that T is a contraction, but $\Pi_\xi T$ ($=$ least squares fit composed with T) is not.

NORM MISMATCH PROBLEM

- For the method to converge, we need $\Pi_\xi T$ to be a contraction; the contraction property of T is not enough



Fitted Value Iteration with Projection

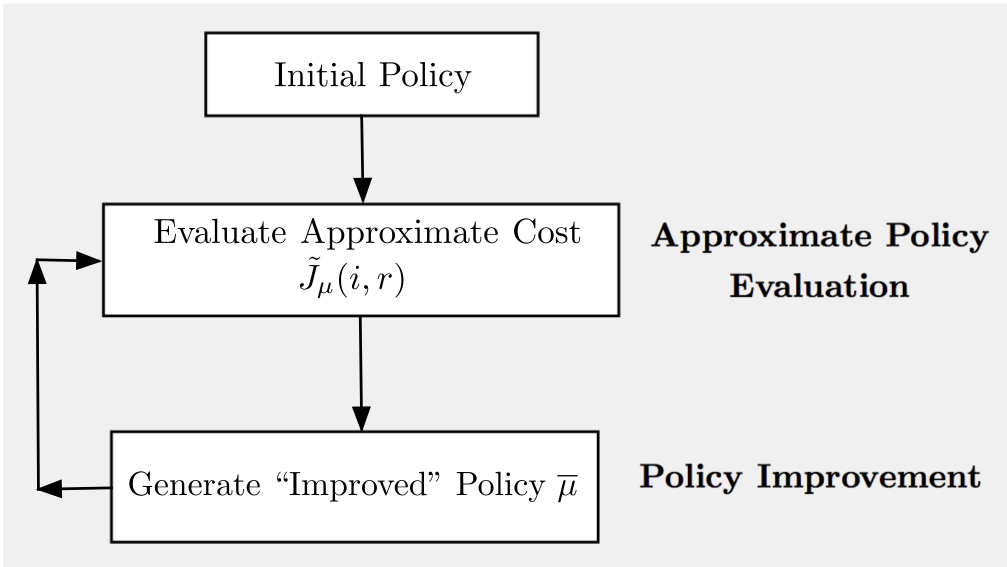
- We need a vector of weights ξ such that T is a contraction with respect to the weighted Euclidean norm $\|\cdot\|_\xi$

- Then we can show that $\Pi_\xi T$ is a contraction with respect to $\|\cdot\|_\xi$

- We will come back to this issue

APPROXIMATE POLICY ITERATION

APPROXIMATE PI



- Evaluation of typical policy μ: Linear cost function approximation $\tilde J_\mu(r) = \Phi r$, where Φ is full rank $n \times s$ matrix with columns the basis functions, and ith row denoted $\phi(i)'$.

- Policy "improvement" to generate $\overline{\mu}$:

$$\overline{\mu}(i) = \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha\phi(j)'r\big)$$

- Error Bound (same as approximate VI): If

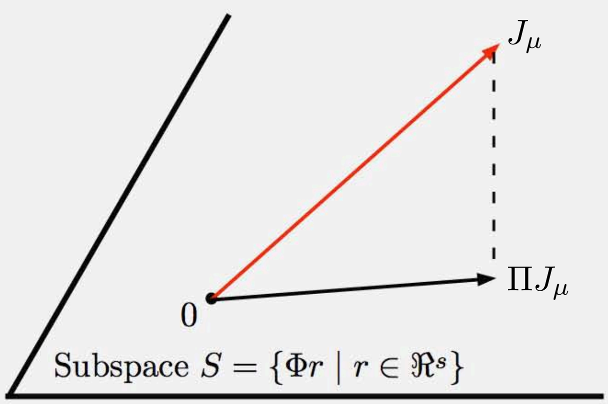
$$\max_i |\tilde J_{\mu^k}(i,r_k) - J_{\mu^k}(i)| \le \delta, \qquad k = 0,1,\ldots$$

the sequence $\{\mu^k\}$ satisfies

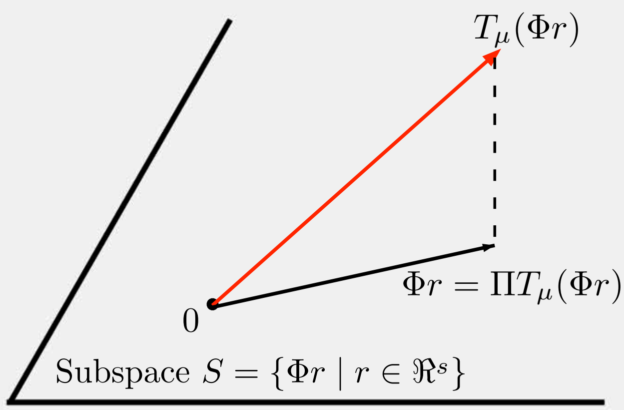
$$\limsup_{k\to\infty} \max_i \big(J_{\mu^k}(i) - J^*(i)\big) \le \frac{2\alpha\delta}{(1-\alpha)^2}$$

POLICY EVALUATION

- Let's consider approximate evaluation of the cost of the current policy by using simulation.
  - Direct policy evaluation - Cost samples generated by simulation, and optimization by least squares
  - Indirect policy evaluation - solving the projected equation $\Phi r = \Pi T_\mu(\Phi r)$ where Π is projection w/ respect to a suitable weighted Euclidean norm



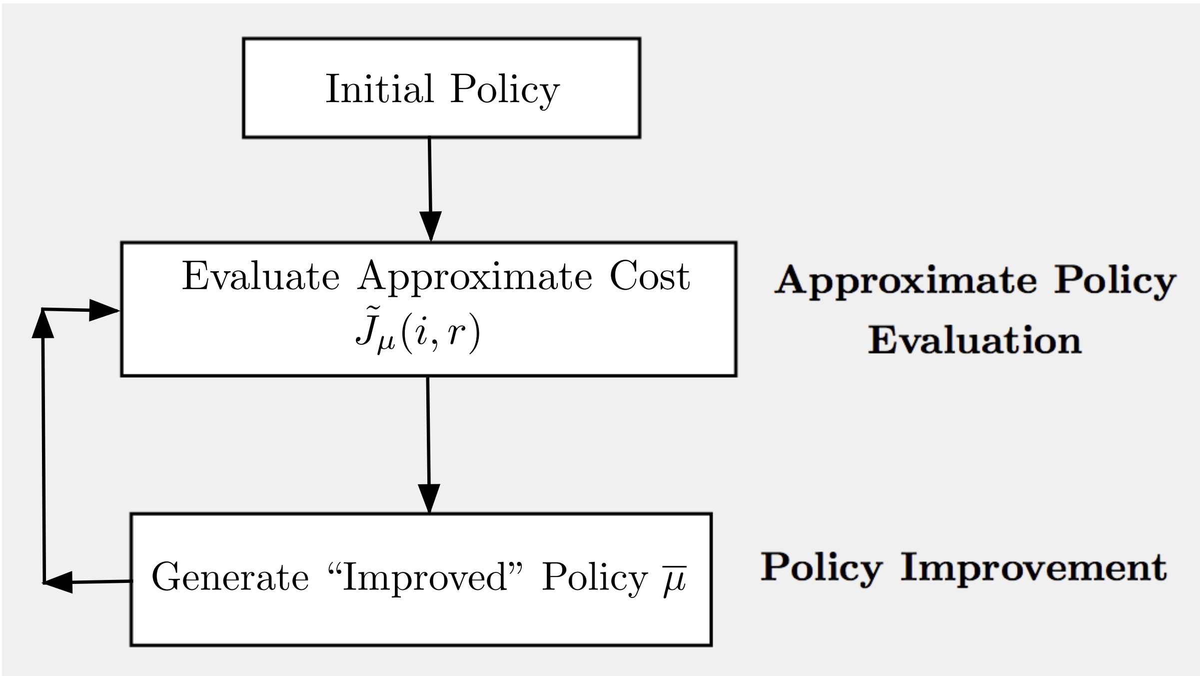
Direct Method: Projection of cost vector J_μ



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

- Recall that projection can be implemented by simulation and least squares

PI WITH INDIRECT POLICY EVALUATION



- Given the current policy μ:
  - We solve the projected Bellman's equation

$$\Phi r = \Pi T_\mu(\Phi r)$$

  - We approximate the solution J_μ of Bellman's equation

$$J = T_\mu J$$

  with the projected equation solution $\tilde{J}_\mu(r)$

KEY QUESTIONS AND RESULTS

- Does the projected equation have a solution?

- Under what conditions is the mapping ΠT_μ a contraction, so ΠT_μ has unique fixed point?

- Assumption: The Markov chain corresponding to μ has a single recurrent class and no transient states, i.e., it has steady-state probabilities that are positive

$$\xi_j = \lim_{N \to \infty} \frac{1}{N} \sum_{k=1}^{N} P(i_k = j \mid i_0 = i) > 0$$

Note that ξ_j is the long-term frequency of state j.

- Proposition: (Norm Matching Property) Assume that the projection Π is with respect to $\|\cdot\|_\xi$, where $\xi = (\xi_1, \ldots, \xi_n)$ is the steady-state probability vector. Then:

  (a) ΠT_μ is contraction of modulus α with respect to $\|\cdot\|_\xi$.

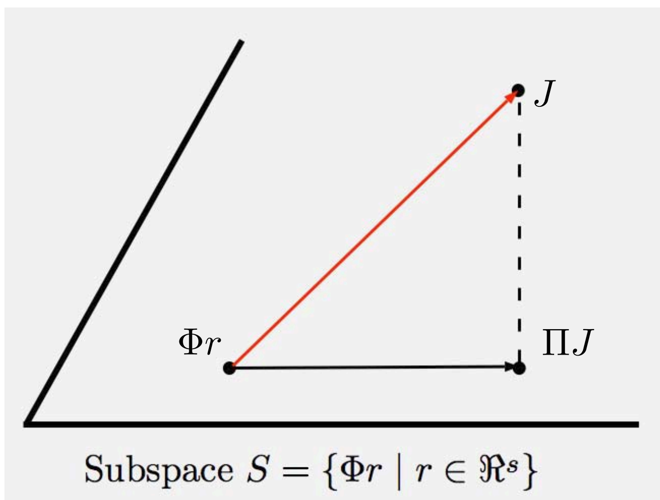
  (b) The unique fixed point Φr^* of ΠT_μ satisfies

$$\|J_\mu - \Phi r^*\|_\xi \leq \frac{1}{\sqrt{1-\alpha^2}} \|J_\mu - \Pi J_\mu\|_\xi$$

PRELIMINARIES: PROJECTION PROPERTIES

- Important property of the projection Π on S with weighted Euclidean norm $\|\cdot\|_\xi$. For all $J \in \Re^n$, $\Phi r \in S$, the Pythagorean Theorem holds:

$$\|J - \Phi r\|_\xi^2 = \|J - \Pi J\|_\xi^2 + \|\Pi J - \Phi r\|_\xi^2$$



- The Pythagorean Theorem implies that the projection is nonexpansive, i.e.,

$$\|\Pi J - \Pi \bar{J}\|_\xi \leq \|J - \bar{J}\|_\xi, \qquad \text{for all } J, \bar{J} \in \Re^n.$$

To see this, note that

$$\begin{aligned} \left\|\Pi(J - \overline{J})\right\|_\xi^2 &\leq \left\|\Pi(J - \overline{J})\right\|_\xi^2 + \left\|(I - \Pi)(J - \overline{J})\right\|_\xi^2 \\ &= \|J - \overline{J}\|_\xi^2 \end{aligned}$$

PROOF OF CONTRACTION PROPERTY

- Lemma: If P is the transition matrix of μ,

$$\|Pz\|_\xi \le \|z\|_\xi, \qquad z \in \Re^n$$

Proof: Let p_{ij} be the components of P. For all $z \in \Re^n$, we have

$$\|Pz\|_\xi^2 = \sum_{i=1}^n \xi_i \left(\sum_{j=1}^n p_{ij} z_j \right)^2 \le \sum_{i=1}^n \xi_i \sum_{j=1}^n p_{ij} z_j^2$$

$$= \sum_{j=1}^n \sum_{i=1}^n \xi_i p_{ij} z_j^2 = \sum_{j=1}^n \xi_j z_j^2 = \|z\|_\xi^2,$$

where the inequality follows from the convexity of the quadratic function, and the next to last equality follows from the defining property $\sum_{i=1}^n \xi_i p_{ij} = \xi_j$ of the steady-state probabilities.

- Using the lemma, the nonexpansiveness of Π, and the definition $T_\mu J = g + \alpha P J$, we have

$$\|\Pi T_\mu J - \Pi T_\mu \bar{J}\|_\xi \le \|T_\mu J - T_\mu \bar{J}\|_\xi = \alpha \|P(J - \bar{J})\|_\xi \le \alpha \|J - \bar{J}\|_\xi$$

for all $J, \bar{J} \in \Re^n$. Hence ΠT_μ is a contraction of modulus α.

PROOF OF ERROR BOUND

- Let Φr^* be the fixed point of ΠT. We have

$$\|J_\mu - \Phi r^*\|_\xi \leq \frac{1}{\sqrt{1-\alpha^2}} \|J_\mu - \Pi J_\mu\|_\xi.$$

Proof: We have

$$\begin{aligned}
\|J_\mu - \Phi r^*\|_\xi^2 &= \|J_\mu - \Pi J_\mu\|_\xi^2 + \left\|\Pi J_\mu - \Phi r^*\right\|_\xi^2 \\
&= \|J_\mu - \Pi J_\mu\|_\xi^2 + \left\|\Pi T J_\mu - \Pi T(\Phi r^*)\right\|_\xi^2 \\
&\leq \|J_\mu - \Pi J_\mu\|_\xi^2 + \alpha^2 \|J_\mu - \Phi r^*\|_\xi^2,
\end{aligned}$$

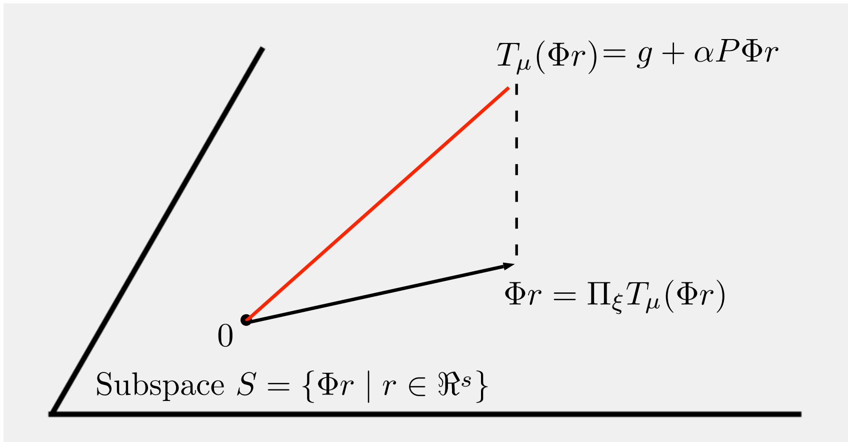
where

- The first equality uses the Pythagorean Theorem
- The second equality holds because J_μ is the fixed point of T and Φr^* is the fixed point of ΠT
- The inequality uses the contraction property of ΠT.

Q.E.D.

SIMULATION-BASED SOLUTION OF PROJECTED EQUATION

MATRIX FORM OF PROJECTED EQUATION



- The solution Φr^* satisfies the orthogonality condition: The error

$$\Phi r^* - (g + \alpha P \Phi r^*)$$

is "orthogonal" to the subspace spanned by the columns of Φ.

- This is written as

$$\Phi' \Xi \big(\Phi r^* - (g + \alpha P \Phi r^*) \big) = 0,$$

where Ξ is the diagonal matrix with the steady-state probabilities $\xi_1, \ldots, \xi_n$ along the diagonal.

- Equivalently, $Cr^* = d$, where

$$C = \Phi' \Xi (I - \alpha P) \Phi, \qquad d = \Phi' \Xi g$$

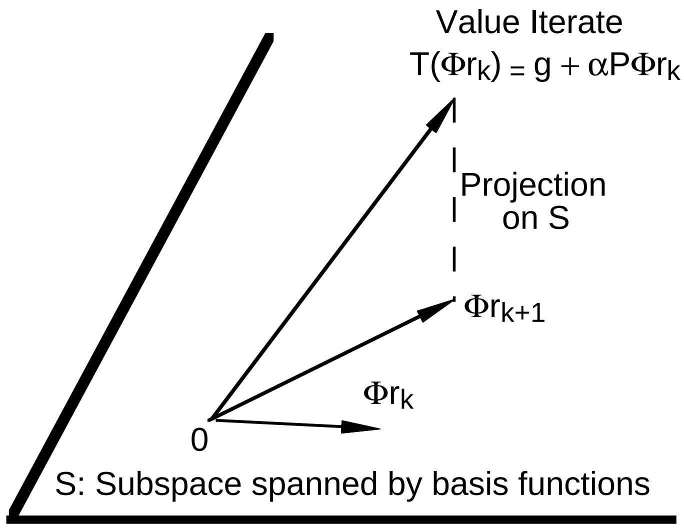
but computing C and d is HARD (high-dimensional inner products).

SOLUTION OF PROJECTED EQUATION

- Solve $Cr^* = d$ by matrix inversion: $r^* = C^{-1}d$
- Projected Value Iteration (PVI) method:

$$\Phi r_{k+1} = \Pi T(\Phi r_k) = \Pi(g + \alpha P \Phi r_k)$$

Converges to r^* because ΠT is a contraction.



- PVI can be written as:

$$r_{k+1} = \arg\min_{r \in \Re^s} \left\| \Phi r - (g + \alpha P \Phi r_k) \right\|_\xi^2$$

By setting to 0 the gradient with respect to r,

$$\Phi' \Xi \big(\Phi r_{k+1} - (g + \alpha P \Phi r_k) \big) = 0,$$

which yields

$$r_{k+1} = r_k - (\Phi' \Xi \Phi)^{-1}(C r_k - d)$$

SIMULATION-BASED IMPLEMENTATIONS

- Key idea: Calculate simulation-based approximations based on k samples

$$C_k \approx C, \qquad d_k \approx d$$

- Matrix inversion $r^* = C^{-1}d$ is approximated by

$$\hat{r}_k = C_k^{-1} d_k$$

This is the LSTD (Least Squares Temporal Differences) Method.

- PVI method $r_{k+1} = r_k - (\Phi'\Xi\Phi)^{-1}(Cr_k - d)$ is approximated by

$$r_{k+1} = r_k - G_k(C_k r_k - d_k)$$

where

$$G_k \approx (\Phi'\Xi\Phi)^{-1}$$

This is the LSPE (Least Squares Policy Evaluation) Method.

- Key fact: C_k, d_k, and G_k can be computed with low-dimensional linear algebra (of order s; the number of basis functions).

SIMULATION MECHANICS

- We generate an infinitely long trajectory $(i_0, i_1, \ldots)$ of the Markov chain, so states i and transitions (i,j) appear with long-term frequencies ξ_i and p_{ij}.
- After generating each transition (i_t, i_{t+1}), we compute the row $\phi(i_t)'$ of Φ and the cost component $g(i_t, i_{t+1})$.
- We form

$$d_k = \frac{1}{k+1} \sum_{t=0}^{k} \phi(i_t) g(i_t, i_{t+1}) \approx \sum_{i,j} \xi_i p_{ij} \phi(i) g(i,j) = \Phi' \Xi g = d$$

$$C_k = \frac{1}{k+1} \sum_{t=0}^{k} \phi(i_t) \big(\phi(i_t) - \alpha \phi(i_{t+1})\big)' \approx \Phi' \Xi (I - \alpha P) \Phi = C$$

Also in the case of LSPE

$$G_k = \frac{1}{k+1} \sum_{t=0}^{k} \phi(i_t) \phi(i_t)' \approx \Phi' \Xi \Phi$$

- Convergence based on law of large numbers.
- C_k, d_k, and G_k can be formed incrementally. Also can be written using the formalism of temporal differences (this is just a matter of style)

OPTIMISTIC VERSIONS

- Instead of calculating nearly exact approximations $C_k \approx C$ and $d_k \approx d$, we do a less accurate approximation, based on few simulation samples

- Evaluate (coarsely) current policy μ, then do a policy improvement

- This often leads to faster computation (as optimistic methods often do)

- Very complex behavior (see the subsequent discussion on oscillations)

- The matrix inversion/LSTD method has serious problems due to large simulation noise (because of limited sampling) - particularly if the C matrix is ill-conditioned

- LSPE tends to cope better because of its iterative nature (this is true of other iterative methods as well)

- A stepsize $\gamma \in (0, 1]$ in LSPE may be useful to damp the effect of simulation noise

$$r_{k+1} = r_k - \gamma G_k(C_k r_k - d_k)$$

MULTISTEP PROJECTED EQUATIONS

MULTISTEP METHODS

- Introduce a multistep version of Bellman's equation $J = T^{(\lambda)}J$, where for $\lambda \in [0, 1)$,

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

Geometrically weighted sum of powers of T.

- Note that T^{ℓ} is a contraction with modulus α^{ℓ}, with respect to the weighted Euclidean norm $\|\cdot\|_{\xi}$, where ξ is the steady-state probability vector of the Markov chain.

- Hence $T^{(\lambda)}$ is a contraction with modulus

$$\alpha_{\lambda} = (1 - \lambda) \sum_{\ell=0}^{\infty} \alpha^{\ell+1} \lambda^{\ell} = \frac{\alpha(1 - \lambda)}{1 - \alpha\lambda}$$

Note that $\alpha_{\lambda} \rightarrow 0$ as $\lambda \rightarrow 1$

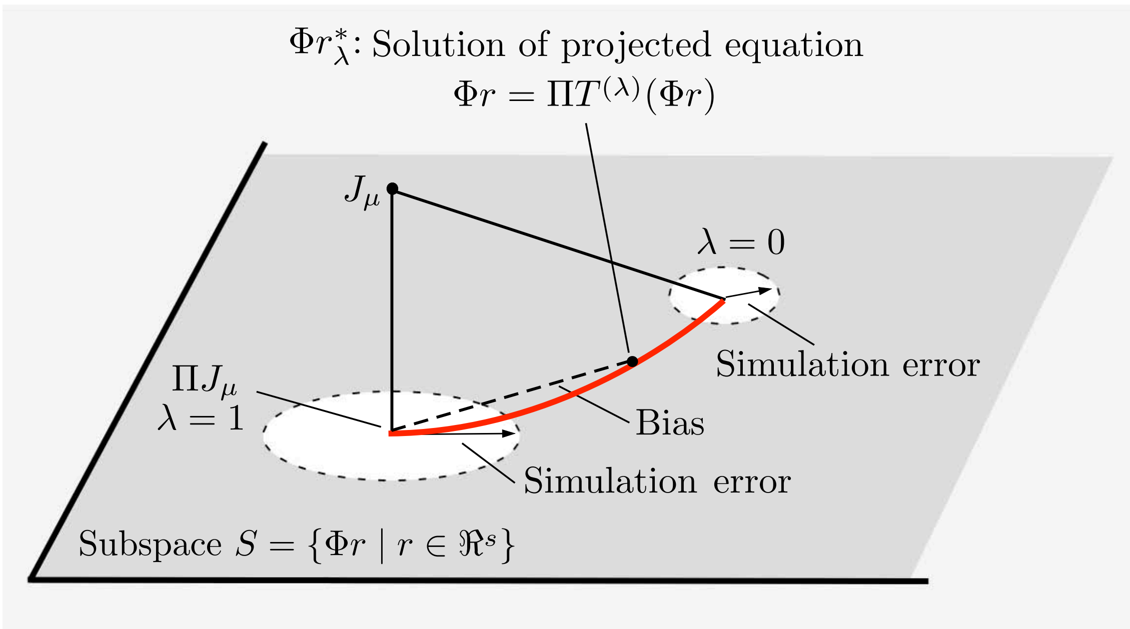
- T^{ℓ} and $T^{(\lambda)}$ have the same fixed point J_{μ} and

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

where Φr_{λ}^{*} is the fixed point of $\Pi T^{(\lambda)}$.

- The fixed point Φr_{λ}^{*} depends on λ.

BIAS-VARIANCE TRADEOFF



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

- As $\lambda \uparrow 1$, we have $\alpha_\lambda \downarrow 0$, so error bound (and the quality of approximation) improves as $\lambda \uparrow 1$. In fact

$$\lim_{\lambda \uparrow 1} \Phi r_\lambda^* = \Pi J_\mu$$

- But the simulation noise in approximating

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

increases

- Choice of λ is usually based on trial and error

MULTISTEP PROJECTED EQ. METHODS

- The projected Bellman equation is

$$\Phi r = \Pi T^{(\lambda)}(\Phi r)$$

- In matrix form: $C^{(\lambda)} r = d^{(\lambda)}$, where

$$C^{(\lambda)} = \Phi' \Xi \big(I - \alpha P^{(\lambda)}\big)\Phi, \qquad d^{(\lambda)} = \Phi' \Xi g^{(\lambda)},$$

with

$$P^{(\lambda)} = (1-\lambda)\sum_{\ell=0}^{\infty} \alpha^{\ell}\lambda^{\ell} P^{\ell+1}, \quad g^{(\lambda)} = \sum_{\ell=0}^{\infty} \alpha^{\ell}\lambda^{\ell} P^{\ell} g$$

- The LSTD(λ) method is

$$\big(C_k^{(\lambda)}\big)^{-1} d_k^{(\lambda)},$$

where $C_k^{(\lambda)}$ and $d_k^{(\lambda)}$ are simulation-based approximations of $C^{(\lambda)}$ and $d^{(\lambda)}$.

- The LSPE(λ) method is

$$r_{k+1} = r_k - \gamma G_k \big(C_k^{(\lambda)} r_k - d_k^{(\lambda)}\big)$$

where G_k is a simulation-based approx. to $(\Phi' \Xi \Phi)^{-1}$

- TD(λ): An important simpler/slower iteration [similar to LSPE(λ) with $G_k = I$ - see the text].

MORE ON MULTISTEP METHODS

- The simulation process to obtain $C_k^{(\lambda)}$ and $d_k^{(\lambda)}$ is similar to the case $\lambda = 0$ (single simulation trajectory $i_0, i_1, \ldots$, more complex formulas)

$$C_k^{(\lambda)} = \frac{1}{k+1} \sum_{t=0}^{k} \phi(i_t) \sum_{m=t}^{k} \alpha^{m-t} \lambda^{m-t} \big(\phi(i_m) - \alpha \phi(i_{m+1})\big)'$$

$$d_k^{(\lambda)} = \frac{1}{k+1} \sum_{t=0}^{k} \phi(i_t) \sum_{m=t}^{k} \alpha^{m-t} \lambda^{m-t} g_{i_m}$$

- In the context of approximate policy iteration, we can use optimistic versions (few samples between policy updates).
- Many different versions (see the text).
- Note the λ-tradeoffs:
  - As $\lambda \uparrow 1$, $C_k^{(\lambda)}$ and $d_k^{(\lambda)}$ contain more "simulation noise", so more samples are needed for a close approximation of r_λ (the solution of the projected equation)
  - The error bound $\|J_\mu - \Phi r_\lambda\|_\xi$ becomes smaller
  - As $\lambda \uparrow 1$, $\Pi T^{(\lambda)}$ becomes a contraction for arbitrary projection norm