Markov Decision Processes – Basics

(1) Definition: Let {Xn} be a stochastic process with state space E and let {Dn} be a stochastic process with action space A. The process {(Xn, Dn) : n=0, 1, … } is called a *Markov decision process* if

$$P\{X_{n+1} = j \mid (X_0,D_n), \ldots, (X_n, D_n)\} = P\{ X_{n+1} = j \mid (X_n, D_n)\}$$

for all n=0, 1, …, $j \in E$.

Furthermore, a Markov decision process has defined a Markov matrix, $\mathbf{P}_k$, and a cost (or profit) function f_k for each $k \in A$ where at each step, $\mathbf{f}_k(i)$ is a cost (or profit) incurred whenever $X_n=i$ and $D_n=k$ and

$$P\{ X_{n+1} = j \mid X_n = i, D_n = k \} = P_k(i, j).$$

(2) Definition: A policy is any rule, using current information, past information, and/or randomization that specifies which action to take at each point in time. The set of all possible policies is denoted by $\boldsymbol{\mathcal{D}}$.

(3) Definition: An action function, **a**, is a vector that maps the state space into the action space, i.e., an action function assigns an action to each state. (Thus, $\mathbf{a} : E \to A$.)

(4) Definition: A stationary policy is a policy that can be defined by an action function. In other words, if $X_n=i$, then a stationary policy sets $D_n=a(i)$ independent of previous states, previous actions, and time n.

Two common criteria used to evaluate policies: (1) expected total discounted cost (or profits) and (2) average long-run cost (or profit).

Expected Total Discounted Cost.

$$v_d(\alpha, i) = E_d\left[\sum_{n=0}^{\infty} \alpha^n f_{D_n}(X_n) \mid X_0 = 0 \right] \qquad \text{for } d \in \boldsymbol{\mathcal{D}}.$$

Our goal will be to find $d^* \in \boldsymbol{\mathcal{D}}$ such that $v_{d^*}(\alpha, i) = v^{\alpha}(i) = \min \{ v_d(\alpha, i) : d \in \boldsymbol{\mathcal{D}} \}$.

Average Long-Run Cost

$$\varphi_d = \lim_{n\to\infty}\left[f_{D_0}(X_0) + \cdots + f_{D_{n-1}}(X_{n-1}) \right] / n \qquad \text{for } d \in \boldsymbol{\mathcal{D}}.$$

Our goal will be to find $d^* \in \boldsymbol{\mathcal{D}}$ such that $\varphi_{d^*} = \varphi^* = \min \{\varphi_d : d \in \boldsymbol{\mathcal{D}} \}$.

Example for Lecture of April 15

Let $\{X_n\}$ be a stochastic process with state space E={a,b,c,d}. This process represents a machine that can be in one of four operating conditions denoted by the states a through d indicating increasing levels of deterioration. As the machine deteriorates (moves to a lower state), not only is it more expensive to operate, but also production is lost. Standard maintenance activities are always carried out in states b though d so that the machine may improve due to maintenance; however, improvement is not guaranteed. In addition to the state space, there is an action space which gives the decisions possible at each step. (We sometimes use the words "decisions" and "actions" interchangeably referring to the elements of the action space.) In this example, we shall assume that actions space is A={1,2}; that is, at each step there are two possible actions: use an inexperienced operator (Action 1) or use an experienced operator (Action 2). To complete the description of a Markov decision problem, we need a cost vector and a transition matrix for each possible action in the action space. For our example, define the two cost vectors and two Markov matrices as

$$\mathbf{f}_1 = (100, 125, 150, 500)^T$$

$$\mathbf{f}_2 = (300, 325, 350, 600)^T$$

$$\mathbf{P}_1 = \begin{array}{c} a \\ b \\ c \\ d \end{array} \begin{bmatrix} 0.1 & 0.3 & 0.6 & 0.0 \\ 0.0 & 0.2 & 0.5 & 0.3 \\ 0.0 & 0.1 & 0.2 & 0.7 \\ 0.8 & 0.1 & 0.0 & 0.1 \end{bmatrix}$$

$$\mathbf{P}_2 = \begin{array}{c} a \\ b \\ c \\ d \end{array} \begin{bmatrix} 0.6 & 0.3 & 0.1 & 0.0 \\ 0.75 & 0.1 & 0.1 & 0.05 \\ 0.8 & 0.2 & 0.0 & 0.0 \\ 0.9 & 0.1 & 0.0 & 0.0 \end{bmatrix}$$

Sequence of events in a Markov decision process with $i \in E$ and $k \in A$

Observe state	➔	Take action	➔	Incur cost	➔	Transition to next state
$X_n = i$		$D_n = k$		$f_k(i)$		$P_k(i, j)$

Discounted Cost Problem

(1) Property: *Fixed-Point Theorem for Markov Decision Processes*. Let $\mathbf{v}^\alpha$ be the optimal value function for the expected total discounted cost problem with $0<\alpha<1$. The function $\mathbf{v}^\alpha$ satisfies, for each $i\in E$, the following

$$v^\alpha(i) = \min_{k\in A} \{f_k(i) + \alpha \sum_{j\in E} P_k(i,j)v^\alpha(j)\} .$$

Furthermore, it is the only function satisfying this property.

(2) Property: *Characteristic of the Optimal Policy*. Let $\mathbf{v}^\alpha$ be the optimal value function for the expected total discounted cost problem with $0<\alpha<1$. The action function, for each $i\in E$, defined as

$$a(i) = \operatorname{argmin}_{k\in A} \{f_k(i) + \alpha \sum_{j\in E} P_k(i,j)v^\alpha(j)\} .$$

is an optimal policy for this problem.

(3) Algorithm: *The Value Improvement Algorithm*. The following iteration procedure will yield an approximation to the optimal value function for the expected total discounted cost problem with $0<\alpha<1$.

```
Step 1.  Make sure that $\alpha<1$, choose a small positive value for $\varepsilon$, set $n=0$, and let $v_0(i)=0$ for each $i\in E$.
Step 2.  For each $i\in E$, define $v_{n+1}(i) = \min_{k\in A} \{f_k(i) + \alpha \sum_{j\in E} P_k(i,j)v_n(j)\}$ .
Step 3.  Define $\delta$ by $\delta = \min_{i\in E} \{ | v_{n+1}(i) - v_n(i) | \}$.
Step 4.  If $\delta < \varepsilon$, let $\mathbf{v}^\alpha = \mathbf{v}_{n+1}$ and stop; otherwise, increment n by one and return to Step 2.
```

(4) Algorithm: *The Policy Improvement Algorithm*. The following iteration procedure will yield the optimal (stationary) policy and its optimal value function for the expected total discounted cost problem with $0<\alpha<1$.

```
Step 1.  Make sure that $\alpha<1$ and set $n=0$. For each $i\in E$, define the action function, $\mathbf{a}_0$ by
             $a_0(i) = \operatorname{argmin}_{k\in A} \{f_k(i)\}$
Step 2.  For each $i,j\in E$, define the matrix $\mathbf{P}$ and the vector $\mathbf{f}$ by
             $P(i, j) = P_{an(i)}(i, j)$
             $f(i) = f_{an(i)}(i)$
Step 3.  Define the value function $\mathbf{v}$ by
             $\mathbf{v} = (\mathbf{I} - \alpha\mathbf{P})^{-1}\mathbf{f}$.
Step 4.  For each $i\in E$, define the action $\mathbf{a}_{n+1}$ by
             $a_{n+1}(i) = \operatorname{argmin}_{k\in A} \{f_k(i) + \alpha \sum_{j\in E} P_k(i,j)v(j)\}$ .
Step 5.  If $\mathbf{a}_{n+1} = \mathbf{a}_n$ , let $\mathbf{v}^\alpha = \mathbf{v}$ and $\mathbf{a}^\alpha = \mathbf{a}$ and stop; otherwise, increment n and return to Step 2.
```

Note for Step 3. If you are writing a computer code for the policy improvement algorithm, Step 3 is to find $\mathbf{v}$ the satisfies $(\mathbf{I} - \alpha\mathbf{P})\mathbf{v} = \mathbf{f}$ so you may want to use a more efficient procedure than using the inverse.

Average Cost Problem

(1) Property: *Key Theorem for Long-Run Average Cost Markov Decision Processes*. Assume that every stationary policy yields a Markov chain with only one irreducible set. There exists a scalar φ^* and a vector $\mathbf{h}$ such that, for all $i \in E$,

$$\varphi^* + h(i) = \min_{k\in A} \{f_k(i) + \textstyle\sum_{j\in E} P_k\,(i,j)h(j)\}\ .$$

The scalar φ^* is the optimal cost for the long-run average cost problem, and the optimal action function is defined by

$$a(i) = \operatorname{argmin}_{k\in A} \{f_k(i) + \textstyle\sum_{j\in E} P_k\,(i,j)h(j)\}\ .$$

Furthermore, the vector $\mathbf{h}$ is unique up to an additive constant.

(2) Property: *Relationship between the Two Problems*. Let $\mathbf{v}^{\alpha}$ be the optimal value function for the expected total discounted cost problem with $0<\alpha<1$ and let φ^* be the optimal cost for the long-run average cost problem. Assume that every stationary policy yields a Markov chain with only one irreducible set, then $\lim_{\alpha\to 1-} (1 - \alpha)\, v^{\alpha}(i) = \varphi^*$, for every $i \in E$.

(3) Property: *Intuitive Meaning of the Vector **h***. Let $\mathbf{v}^{\alpha}$ be the optimal value function for the expected total discounted cost problem with $0<\alpha<1$ and let $\mathbf{h}$ be the vector from the Key Theorem for the long-run average problem. Then $\lim_{\alpha\to 1-} v^{\alpha}(i) - v^{\alpha}(j) = h(i) - h(j)$ for every $i \in E$. (In other words, h(i) is the relative advantage of starting in state i instead of state j.)

(4) Algorithm: *The Policy Improvement Algorithm*. The following iteration procedure will yield an optimal policy for the long-run cost problem.

Step 1. Set n=0 and let state 1 denote the first state in the state space. For each $i \in E$, define the action function, $\mathbf{a}_0$ by

$$a_0(i) = \operatorname{argmin}_{k\in A} \{f_k(i)\}$$

Step 2. For each $i,j \in E$, define the matrix $\mathbf{P}$ and the vector $\mathbf{f}$ by

$$P(i, j) = P_{an(i)}(i, j)$$
$$f(i) = f_{an(i)}(i)$$

Step 3. Determine the values for φ and $\mathbf{h}$ by solving the system of equations given by

$$\varphi + \mathbf{h} = \mathbf{f} + \mathbf{Ph},$$

where $h(1) = 0$.

Step 4. For each $i \in E$, define the action $\mathbf{a}_{n+1}$ by

$$a_{n+1}(i) = \operatorname{argmin}_{k\in A} \{f_k(i) + \textstyle\sum_{j\in E} P_k\,(i,j)h(j)\}\ .$$

Step 5. If $\mathbf{a}_{n+1} = \mathbf{a}_n$, let $\varphi^* = \varphi$, and $\mathbf{a}^* = \mathbf{a}$ and stop; otherwise, increment n and return to Step 2.

Note for Step 3. Using an inverse is not always the most efficient approach; however, when using Excel, it is often very convenient. For Excel, let the matrix A = (I–P), *except*, replace the first column by a vector of all ones. Let the vector $\mathbf{x} = \mathbf{A}^{-1}\mathbf{f}$; and we have $\varphi = x(1)$ and $h(i) = x(i)$ for $i \neq 1$.

Comparison of the Total Discounted Cost and Long-Run Average Cost Problems

Discount	**Vector**	**i=a**	**i=b**	**i=c**	**i=d**
α=0.95	v^α	4287	4382	4441	4613
	$(1-\alpha)v^\alpha$	214.35	219.10	222.05	230.65
	$v^\alpha(i) - v^\alpha(a)$	0	94	154	326
α=0.99	v^α	21827	21923	21978	22150
	$(1-\alpha)v^\alpha$	218.27	219.23	219.78	221.50
	$v^\alpha(i) - v^\alpha(a)$	0	97	151	323
α=0.999	v^α	219.36	219233	219286	219459
	$(1-\alpha)v^\alpha$	219.13	219.23	219.28	219.45
	$v^\alpha(i) - v^\alpha(a)$	0	97	150	323