Miscelánea Matemática **66** (2018) 57-80 SMM

Del algoritmo de programación dinámica a los procesos Markovianos de decisión

David González-Sánchez
Departamento de Matemáticas
CONACYT-Universidad de Sonora
dgonzalezsa@conacyt.mx,

Saúl Díaz-Infante
Departamento de Matemáticas
CONACYT-Universidad de Sonora
sdinfante@conacyt.mx y

Francisco Peñuñuri
Departamento de Ingeniería Física
Universidad Autónoma de Yucatán
francisco.pa@correo.uady.mx

A Juan González Hernández, una excelente persona

Resumen

Presentamos una introducción a la solución de problemas de optimización multietapa. Partiendo del algoritmo de programación dinámica, consideramos aspectos teóricos y computacionales, principalmente de problemas deterministas y discutimos cómo generalizar algunos de los resultados a los procesos Markovianos de decisión.

1. Introducción

El problema clásico con el que se pueden introducir las ideas esenciales de la programación dinámica, es el problema de la ruta óptima (ruta más corta pensando en distancias, o ruta más barata pensando en costos) [2, 4]. Una versión sencilla de este problema se muestra esquemáticamente en la figura 1. En dicho problema, se desea determinar la ruta óptima para ir desde el nodo etiquetado con 0 hasta el nodo etiquetado con 7. Los costos entre cada par de nodos conectados por una flecha, se representan con un número junto a ésta. Por ejemplo, el costo para

pasar del nodo 0 al nodo 2 es 6. La ruta óptima será aquella para la cual la suma de sus costos sea mínima. Esta ruta óptima se puede obtener mediante una enumeración exhaustiva; generando todas las rutas posibles desde el nodo 0 hasta el nodo 7 y escogiendo la ruta de costo mínimo. En la tabla 1 se muestran tales rutas y sus respectivos costos. Se observa que la trayectoria óptima es $0 \to 1 \to 3 \to 5 \to 7$, con un costo de 16.

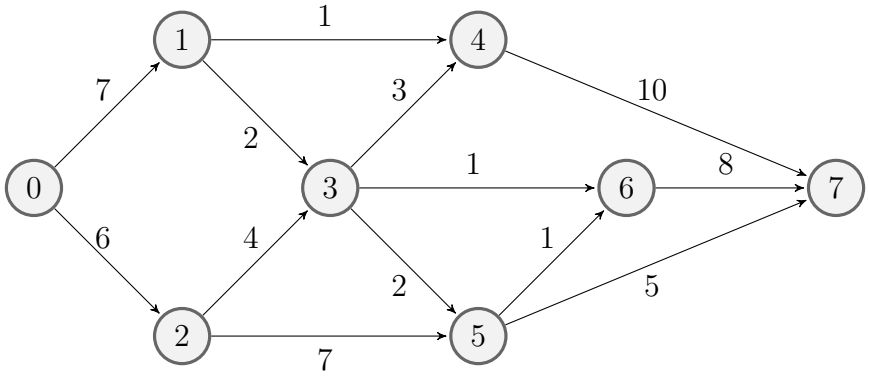

Figura 1. Ruta más corta, ejemplo de Bradimarte [4, p. 497]

En problemas similares al descrito en la figura 1, a medida que el número n de nodos crece, el número de rutas posibles aumenta y el número de operaciones necesarias para comparar los costos de todas las rutas es exponencial como función de n; ver, por ejemplo, Held y Karp [10]. Lo anterior hace que la solución por enumeración exhaustiva sea impráctica. El Algoritmo de Programación Dinámica (APD) ofrece

Ruta	Costo
$0 \to 1 \to 4 \to 7$	18
$0 \to 1 \to 3 \to 4 \to 7$	22
$0 \to 1 \to 3 \to 6 \to 7$	18
$0 \to 1 \to 3 \to 5 \to 6 \to 7$	20
$\mathbf{0 \to 1 \to 3 \to 5 \to 7}$	**16**
$0 \to 2 \to 3 \to 4 \to 7$	23
$0 \to 2 \to 3 \to 6 \to 7$	19
$0 \to 2 \to 3 \to 5 \to 6 \to 7$	21
$0 \to 2 \to 3 \to 5 \to 7$	17
$0 \to 2 \to 5 \to 6 \to 7$	22
$0 \to 2 \to 5 \to 7$	18

Cuadro 1. Posibles rutas y sus costos, en negritas la ruta óptima.

una forma transparente y elegante para resolver este problema; veamos las ideas.

Denotemos por $i \xrightarrow{*} 7$, la ruta óptima desde el nodo i hasta el nodo 7, y por V_i el costo de ésta. Supongamos que dado un nodo inicial i, el nodo j está en la ruta óptima $i \xrightarrow{*} 7$, entonces la ruta óptima desde j, es decir, $j \xrightarrow{*} 7$, formará parte de $i \xrightarrow{*} 7$ y la llamaremos subruta de $i \xrightarrow{*} 7$. Por ejemplo, de la solución anterior, la ruta óptima $3 \xrightarrow{*} 7$ es una subruta de

$$0 \to 1 \to 3 \to 5 \to 7. \tag{1}$$

Asimismo, denotemos por $A(i)$ al conjunto de sucesores admisibles al nodo i y por c_{ij} al costo para ir del nodo i al nodo sucesor j. Por ejemplo, en la figura 1, los nodos 3 y 4 son todos los sucesores de 1, de donde $A(1) = \{3, 4\}$. Con esto, el costo de la ruta óptima $i \xrightarrow{*} 7$ será la suma del costo de ir desde i hasta un sucesor óptimo denotado por j^*, mas el costo de la ruta óptima desde j^*, es decir,

$$V_i = \min_{j \in A(i)} \{c_{ij} + V_j\}. \tag{2}$$

En el contexto de programación dinámica (PD), V_i es la función de valor y se define de forma recursiva a partir de la ecuación (2). Para cualquier nodo i, esta ecuación nos dice cómo calcular el costo mínimo para ir desde el nodo i al nodo 7: comparando las sumas del costo de ir a cualquier nodo $j \in A(i)$ y el costo mínimo V_j. Implícitamente, la ecuación (2) nos da una regla para decidir de forma óptima hacia donde ir en cada nodo. Por lo tanto, para encontrar la ruta óptima desde el sitio 0, basta con resolver dicha ecuación en forma recursiva hacia atrás, es decir, empezando por los predecesores del nodo final. Para este propósito, por consistencia imponemos la restricción $V_7 = 0$ y resolvemos para $i = 6, 5, \ldots, 0$, guardando las decisiones óptimas.

i	Sucesores	$V_i = \min_{j \in A(i)} \{c_{ij} + V_j\}$	Costo	Decisión óptima
7	———	———	0	———
6	7	$V_6 = c_{67} + V_7$	8	$6 \to 7$
5	6,7	$V_5 = \min\{c_{56} + V_6, c_{57} + V_7\}$	5	$5 \to 7$
4	7	$V_4 = c_{47} + V_7$	10	$4 \to 7$
3	4,5,6	$V_3 = \min\{c_{34} + V_4, c_{35} + V_5, c_{36} + V_6\}$	7	$3 \to 5$
2	3,5	$V_2 = \min\{\ c_{23} + V_3, c_{25} + V_5\}$	11	$2 \to 3$
1	3,4	$V_1 = \min\{\ c_{13} + V_3, c_{14} + V_4\}$	9	$1 \to 3$
0	1,2	$V_0 = \min\{c_{01} + V_1, c_{02} + V_2\}$	16	$0 \to 1$

Cuadro 2. Ruta más corta aplicando el APD.

La tabla 2 muestra, para cada nodo i, los sucesores admisibles, el costo mínimo calculado usando (2) y la decisión óptima. Por ejemplo, para el nodo 3 los sucesores admisibles son 4, 5 y 6; aunque ir de 3 a 6 es más barato que ir de 3 a 5, ir de 3 hasta 7 es más barato pasando por 5, luego la decisión óptima es ir al nodo 5. Usando la última columna de la tabla 2, a partir del nodo 0 vemos que el conjunto de decisiones óptimas (que llamaremos política óptima) es $0 \to 1 \to 3 \to 5 \to 7$. Esta ruta coincide con la solución (1) que encontramos mediante una búsqueda exhaustiva. La búsqueda recursiva de la ruta óptima ilustra la idea principal del APD: dividir un problema de optimización en subproblemas que se resuelven de manera recursiva.

En este artículo presentamos una introducción a los problemas de optimización multietapa usando el algoritmo de programación dinámica. La organización del artículo es la siguiente. En la sección 2 se demuestra que el algoritmo de programación dinámica funciona para resolver una familia de problemas de minimización, deterministas y con un número finito de etapas. Se dice que este tipo de problemas tienen horizonte temporal finito. La sección 3 está dedicada a problemas con horizonte temporal infinito y en los cuales el costo total toma una forma especial que se conoce como costo total descontado. En este caso la función de costo mínimo satisface una ecuación funcional, conocida como ecuación de Bellman. Recíprocamente, bajo cierta condición de crecimiento, si hay una solución de la ecuación de Bellman entonces esta solución tiene que ser la función de costo mínimo. Se verá también que el algoritmo de programación dinámica puede usarse para aproximar la función de costo mínimo, a este proceso se le conoce como iteración de valores. La sección 4 ilustra el algoritmo de iteración de valores mediante un problema de descarga óptima de residuos contaminantes en un lago. En la sección 5 se muestra cómo extender el algoritmo de PD a procesos Markovianos de decisión, es decir a problemas cuya dinámica sigue un proceso de Markov. En la sección 6 se presentan algunos comentarios finales, mencionando algunas referencias tanto teóricas como numéricas.

El lector puede entender gran parte del artículo si sabe qué es sucesión convergente. Si además está familiarizado con el teorema de punto fijo de Banach, entonces puede seguir todas las demostraciones. Sólo para la sección 5 se requieren algunos conceptos de probabilidad. Incluimos ejemplos numéricos usando MATLAB como herramienta de cómputo; el código puede descargarse de [9].

2. El algoritmo de programación dinámica

Sean $X \subseteq \mathbb{R}^n$, $A \subseteq \mathbb{R}^m$ y $\{A(x) \subseteq A \mid x \in X\}$. Nos referiremos a X como el conjunto de estados, A es el conjunto de control y la familia $\{A(x),\ x \in X\}$ denota al conjunto de acciones factibles si el sistema está en el estado x. El sistema evoluciona de acuerdo con

$$x_{k+1} = f(x_k, a_k), \quad k = 0, 1, \ldots, N-1, \tag{3}$$

donde x_0 está dado y $f : X \times A \to X$. Decimos que $\{a_0, a_1, \ldots, a_{N-1}\}$ es una sucesión (finita) de acciones admisibles si $a_k \in A(x_k)$ y (3) se satisface para todo $k = 0, 1, \ldots, N-1$. Observemos que cada sucesión $\{a_k\}$ de acciones admisibles, genera, a través de (3), una única sucesión de estados $\{x_k\}$.

Las sucesiones de acciones admisibles pueden escogerse siguiendo ciertas «reglas», por ejemplo, la acción a_k puede escogerse tomando en cuenta toda la *historia* $(x_0, a_0, \ldots, a_{k-1}, x_k)$. De forma precisa, $a_k = g_k(x_0, a_0, \ldots, a_{k-1}, x_k)$, donde g_k es una función. A la sucesión $\pi = \{g_k\}$ se le conoce como *política* o *estrategia*. El APD genera políticas que sólo dependen del último estado, es decir, $a_k = g_k(x_k)$. A estas políticas se les conoce como *estrategias Markovianas*.

El problema de optimización que nos interesa consiste en encontrar una sucesión $\pi = \{a_k\}$ de acciones admisibles que minimice el costo

$$C_0(x, \pi) := \sum_{k=0}^{N-1} c_k(x_k, a_k) + c_N(x_N), \tag{4}$$

donde $c_k : X \times A \to \mathbb{R}$ para $k \leq N-1$ y $c_N : X \to \mathbb{R}$. Sea $\Pi(x)$ el conjunto de todas las sucesiones de acciones admisibles con la condición inicial $x_0 = x$. En términos de la *función de valor*

$$v(x) := \inf_{\pi \in \Pi(x)} C_0(x, \pi), \quad x \in X,$$

queremos encontrar $\hat{\pi}$ tal que $v(x) = C_0(x, \hat{\pi})$ para cada estado inicial x en X. Si el ínfimo se alcanza dentro del conjunto $\Pi(x_0)$, reemplazamos ínf por mín.

Para resolver este problema de minimización, asumiendo que existe una solución, consideremos las siguientes funciones

$$J_N(x) := c_N(x), \tag{5}$$

y para $k = N-1, N-2, \ldots, 0$,

$$J_k(x) := \min_{a \in A(x)} \{c_k(x, a) + J_{k+1}(f(x, a))\}. \tag{6}$$

Teorema 2.1. *Sean $J_N, \ldots, J_0$ las funciones definidas en* (5)–(6). *Supongamos que para cada $k = 0, \ldots, N-1$, existe una función $h_k : X \to A$ tal que $h_k(x) \in A(x)$ y*

$$\begin{aligned} J_k(x) &= \min_{a \in A(x)} \{c_k(x,a) + J_{k+1}(f(x,a))\} \\ &= c_k(x, h_k(x)) + J_{k+1}(f(x, h_k(x))) \end{aligned} \tag{7}$$

para todo $x \in X$. Entonces $\hat{\pi} := \{h_0(x_0), h_1(x_1), \ldots, h_{N-1}(x_{N-1})\}$ minimiza (4). *Más aún, J_0 es igual a la función de valor v.*

Demostración. Sea $\pi' = \{a'_0, \ldots, a'_{N-1}\}$ cualquier sucesión de acciones admisibles. Para cada $k = 0, \ldots, N-1$, definimos el *costo de k hacia adelante*

$$C_k(x, \pi') := \sum_{j=k}^{N-1} c_j(x'_j, a'_j) + c_N(x'_N),$$

donde $x'_k = x$ y, para $j = k, \ldots, N-1$,

$$x'_{j+1} = f(x'_j, a'_j).$$

Definimos también $C_N(x, \pi') := c_N(x)$ para cada $x \in X$. Usando *inducción hacia atrás*, vamos a demostrar que

$$C_k(x, \pi') \geq J_k(x) \quad \forall\, x \in X,\ k = N, N-1, \ldots, 0, \tag{8}$$

con igualdad si $\pi' = \hat{\pi}$. Para $k = N$, (8) se cumple con igualdad. Supongamos que $C_{k+1}(y, \pi') \geq J_{k+1}(y)$ para todo $y \in X$, con igualdad si $\pi' = \hat{\pi}$. Entonces

$$\begin{aligned} C_k(x, \pi') &= c_k(x, a'_k) + C_{k+1}(f(x, a'_k), \pi') \\ &\geq c_k(x, a'_k) + J_{k+1}(f(x, a'_k)) \qquad (9) \\ &\geq c_k(x, h_k(x)) + J_{k+1}(f(x, h_k(x))) \qquad (10) \\ &= J_k(x). \end{aligned}$$

La desigualdad (9) se tiene por la hipótesis de inducción, con $y = f(x, a'_k)$, mientras que (10) se sigue de (7). Más aún, notemos que (10) y (7) se cumplen con igualdad si $\pi' = \hat{\pi}$. Esto demuestra, en particular, que la función J_0 obtenida en el último paso de este algoritmo coincide con la función de valor v. □

A continuación presentamos el pseudocódigo del APD (5).

Algoritmo 1: Algoritmo de Programación Dinámica (APD)

Entrada: Funciones de costo $c_0, \ldots, c_N$, restricciones $A(x)$ y dinámica del sistema f.
Salida: Funciones de costo mínimo $J_0, \ldots, J_N$ y política óptima $\hat{\pi}$.

$J_N(x) \leftarrow c_N(x)$
para cada $k = N-1, N-2, \ldots, 0$ **hacer**
  para cada $x \in X$ **hacer**
    $J_k(x) \leftarrow \min_{a \in A(x)} \{c_k(x,a) + J_{k+1}(f(x,a))\}$
    $h_k(x) \leftarrow \arg\min_{a \in A(x)} \{c_k(x,a) + J_{k+1}(f(x,a))\}$
  fin
fin

Observación 2.1. El APD también funciona para resolver problemas de maximización. En tal caso, hay que cambiar mín por máx en (6). En la literatura sobre optimización, tanto teórica como computacional, es más común trabajar con problemas de minimización que de maximización. En algunos lenguajes de programación como MATLAB, por convención, los problemas de optimización se formulan para buscar un mínimo.

Ejemplo 2.1 (Decisiones de ahorro)**.** Supongamos que Ricardo ha ganado el premio mayor de la Lotería Nacional y decide meter su dinero al banco que le ofrece una tasa de interés anual i. Al inicio del año k decide retirar a_k pesos para sus gastos y planea repetir este proceso N veces, es decir, $k = 0, 1, \ldots, N-1$. Si x_k denota la cantidad de dinero al inicio del año k, entonces la siguiente ecuación

$$x_{k+1} = (1+i)(x_k - a_k), \quad k = 0, 1, \ldots, N-1, \tag{11}$$

describe cómo va cambiando la fortuna de Ricardo en función de los retiros que haga. *Utility theory*, una empresa de consultoría, le recomienda a Ricardo que escoja $a_0, \ldots, a_{N-1}$ de tal manera que maximice

$$\beta^N (x_N)^{1-\gamma} + \sum_{k=0}^{N-1} \beta^k (a_k)^{1-\gamma}. \tag{12}$$

La cantidad final x_N será heredada a sus descendientes. Los parámetros $\beta, \gamma \in (0,1)$ fueron estimados por *Utility theory* (véase [8]). Usando el APD encontramos que cada función J_k es de la forma $J_k(x) = A_k \beta^k x^{1-\gamma}$, donde $A_N = 1$ y para $k = N-1, \ldots, 0$,

$$A_k = [1 + ((1+i)\beta A_{k+1})^{1/\gamma}]^{\gamma}.$$

La estrategia óptima está dada por

$$h_k(x) = \frac{x}{A_k^{1/\gamma}}, \quad k = 0, \ldots, N-1.$$

Este tipo de estrategias tienen la ventaja de ser óptimas en cada uno de los subproblemas, tal como vimos en la demostración del teorema 2.1. Supongamos que, por alguna razón, en el año $k-1$, Ricardo retiró $a'_{k-1} > h_{k-1}(x_{k-1})$, es decir hizo un retiro mayor a lo previsto. Entonces para maximizar su utilidad a partir del año k, Ricardo va a usar la estrategia $h_k, \ldots, h_{N-1}$. En particular, en el año k deberá retirar $h_k(x'_k)$ con $x'_k = (1+i)(x_{k-1} - a'_{k-1})$. Esta propiedad de las estrategias óptimas, encontradas mediante el APD, hace que el APD se pueda usar en procesos Markovianos de decisión.

Ejemplo 2.2 (Un problema de pesca [26])**.** Sea $x(t) \in [0, 1000]$ la cantidad de peces a tiempo t, supongamos que la cantidad de peces a capturar $a(t)$, durante un día, está restringida a 10, que la cantidad inicial de peces es 250 y que después de 200 días, la secretaría de pesca impone la restricción de una población no menor a 750 peces. Con esto en mente, el objetivo del problema es determinar cuántos peces se deberán capturar al día, de tal manera que se cumplan las restricciones mencionadas y que se maximice la pesca.

Notemos que la cantidad inicial de peces evolucionará durante estos 200 días de acuerdo a cómo se reproduzcan, mueran y capturen. El modelo logístico [15], describe la dinámica poblacional de los peces. Supongamos una tasa de reproducción de $r = 2/100$ peces por día y una capacidad de carga $K = 1000$ peces. Considerando a los pescadores como a un depredador, las siguientes expresiones

$$\frac{dx}{dt} = \frac{2}{100} x \left(1 - \frac{x}{1000}\right) - a(t), \tag{13}$$

$$x(0) = 250, \quad t \in [0, 200], \tag{14}$$

describen la dinámica de los peces. Luego, de acuerdo a las restricciones del problema se busca maximizar la pesca durante los 200 días, es decir, se desea maximizar

$$\int_0^{200} a(s)ds. \tag{15}$$

Este problema puede resolverse usando cálculo de variaciones o el principio del máximo de Pontryagin. Otra alternativa es discretizar las ecuaciones (13) a (15) y aplicar el APD que ofrece un esquema de aproximación muy claro de formular. Aunque computacionalmente costoso, el APD es muy robusto pues regresa una solución que depende solamente del estado actual x_k, entonces se tiene una regla de decisión para controlar la transición al siguiente estado de forma óptima.

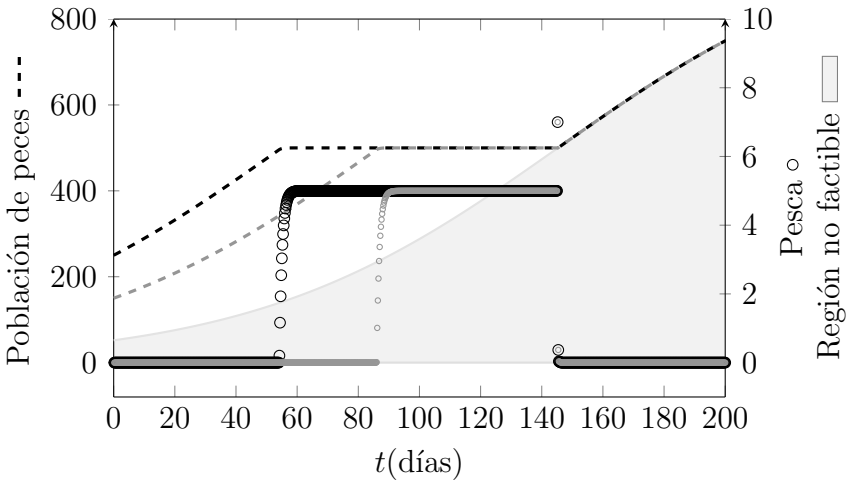

Figura 2. Población de peces y política óptima de pesca del ejemplo 2.2. La escala para la población de peces está a la izquierda; a la derecha, la correspondiente para la política de pesca.

Para aplicar el APD, aproximamos la solución de las ecuaciones (13) a (15) por el método de Euler con tamaño de paso δ y reescribimos el problema de optimización como

$$\begin{aligned} &\min_{a_k\in[0,10]} \sum_{k=0}^{N-1} -ha_k, \quad N = \frac{200}{h} + 1 \\ \text{sujeto a} \quad &x_{k+1} = x_k + \frac{2}{100}x_k\left(1 - \frac{x_k}{1000}\right)h - a_k h, \\ &x_0 = 250, \quad x_N = 750, \quad x_k \in [0, 1000]. \end{aligned} \tag{16}$$

Para encontrar la estrategia de pesca óptima del problema (16), de acuerdo al APD empleamos la función `dpm` de MATLAB reportada en [26]. Los detalles pueden consultarse en el repositorio [9]. La función `dpm`, este y otros ejemplos pueden encontrarse en [25].

La figura 2 muestra la evolución óptima del número de peces junto con la política de pesca para dos condiciones iniciales. El color negro denota el problema para una condición inicial de 250 peces y el color gris para una de 150 peces. Para la condición inicial de 250 peces, se observa que la decisión óptima que maximiza la pesca es no pescar hasta el día 54, durante los días 55–141 capturar 5 peces y finalizar la temporada de captura el día 142 con 7 peces. Al tomar una condición inicial de 150 peces, se tiene que el tiempo de pesca se reduce alrededor de 20 días. En gris claro se denota la región no factible, es decir el conjunto de trayectorias que no satisfacen las condiciones de frontera $x_{200} = 750$ peces.

3. Problemas descontados estacionarios

En esta sección consideramos problemas *estacionarios descontados*, es decir, problemas en los que se quiere minimizar

$$C(x,\pi) := \lim_{K\to\infty} \sum_{k=0}^{K} \beta^k c(x_k, a_k), \quad \pi \in \Pi(x),$$

donde $c : X \times A \to \mathbb{R}$ y $\beta \in (0,1)$ se conoce como *factor de descuento*. Asumimos que para cada $x \in X$, existe $\pi \in \Pi(x)$ tal que $C(x,\pi) < \infty$. Definimos la función de valor

$$v(x) := \inf_{\pi\in\Pi(x)} C(x,\pi), \quad x \in X.$$

3.1 La ecuación de Bellman

Teorema 3.1. *Supongamos que la función de valor es finita, es decir $v(x) > -\infty$ para todo $x \in X$. Entonces v satisface la siguiente ecuación, conocida como* **ecuación de Bellman**,

$$v(x) = \inf_{a\in A(x)} \{c(x,a) + \beta v(f(x,a))\} \quad \forall x \in X. \tag{17}$$

Demostración. Sea $x_0 = x$ un estado inicial arbitrario. Notemos que si $\pi = \{a_0, a_1, a_2 \ldots\} \in \Pi(x)$, entonces $\pi_1 := \{a_1, a_2, \ldots\} \in \Pi(x_1)$ con $x_1 = f(x, a_0)$. Luego

$$\begin{aligned} C(x,\pi) &= c(x,a_0) + \beta C(x_1,\pi_1) \\ &\geq c(x,a_0) + \beta v(x_1) \\ &= c(x,a_0) + \beta v(f(x,a_0)) \\ &\geq \inf_{a\in A(x)} \{c(x,a) + \beta v(f(x,a))\}, \end{aligned}$$

de donde obtenemos

$$v(x) = \inf_{\pi\in\Pi(x)} C(x,\pi) \geq \inf_{a\in A(x)} \{c(x,a) + \beta v(f(x,a))\}. \tag{18}$$

Por otro lado, sea $a_0 \in A(x)$ arbitrario. Entonces para cualquier $\tilde{\pi} \in \Pi(f(x,a_0))$ se cumple que $v(x) \leq c(x,a_0) + \beta C(f(x,a_0),\tilde{\pi})$. Es decir,

$$\beta^{-1}[v(x) - c(x,a_0)] \leq C(f(x,a_0),\tilde{\pi}) \qquad \forall \tilde{\pi} \in \Pi(f(x,a_0)),$$

de donde $\beta^{-1}[v(x) - c(x,a_0)] \leq v(f(x,a_0))$. Por ser $a_0 \in A(x)$ arbitrario,

$$v(x) \leq c(x,a_0) + \beta v(f(x,a_0)) \qquad \forall a_0 \in A(x).$$

Por lo tanto $v(x) \leq \inf_{a\in A(x)}\{c(x,a) + \beta v(f(x,a))\}$. Esta última desigualdad y (18) implican que v es una solución de la ecuación de Bellman. □

Se acaba de demostrar que la función de valor es una solución de la ecuación de Bellman. Sin embargo, como podemos ver en el siguiente ejemplo, no cualquier solución de la ecuación de Bellman coincide con la función de valor.

Ejemplo 3.1. Consideremos $X = [0, \infty)$, $A(x) = [0, \beta^{-1}]$ con $\beta \in (0, 1)$, $c(x, a) = x(\beta^{-1} - a)$ y $f(x, a) = xa$. Puesto que las acciones factibles son números no negativos, entonces para cualquier estado inicial $x_0 \geq 0$ y cualquier $\pi = \{a_k\}$ tenemos que $x_k \geq 0$ para todo $k \geq 0$ y, más aún, $C(x_0, \pi) \geq 0$. Si escogemos $a_k = \beta^{-1}$ para cada $k \geq 0$, entonces concluimos que $v(x) = 0$ para todo $x \in X$.

Por otro lado, la función $w(x) = \beta^{-1}x$ es una solución de la ecuación de Bellman. En efecto, para cualquier $x \in X$,

$$\begin{aligned} \inf_{a \in A(x)} \{c(x, a) + \beta w(f(x, a))\} &= \inf_{a \in [0, \beta^{-1}]} \{x(\beta^{-1} - a) + xa\} \\ &= w(x). \end{aligned}$$

Es decir, la ecuación de Bellman puede tener soluciones distintas a la función de valor.

Teorema 3.2. *Sea $w : X \to \mathbb{R}$ cualquier solución de la ecuación de Bellman, es decir,*

$$w(x) = \inf_{a \in A(x)} \{c(x, a) + \beta w(f(x, a))\} \quad \forall x \in X. \tag{19}$$

Si para cada $x \in X$ y cualquier $\pi \in \Pi(x)$, la trayectoria de estados $\{x_k\}$ generada por π satisface

$$\lim_{k \to \infty} \beta^k w(x_k) = 0, \tag{20}$$

entonces $w(x) = \inf_{\pi \in \Pi(x)} C(x, \pi)$.

Demostración. Sea $x_0 = x$ un estado inicial arbitrario en X. Si $\pi = \{a_k\} \in \Pi(x)$ y $\{x_k\}$ es la correspondiente trayectoria de estados, entonces (19) implica que el estado inicial satisface $w(x) \leq c(x, a_0) + \beta w(x_1)$. Usando repetidamente (19) con cada uno de los estados x_k vemos que

$$w(x) \leq \sum_{k=0}^{K-1} \beta^k c(x_k, a_k) + \beta^K w(x_K).$$

Por lo tanto, usando (20), tenemos que

$$w(x) \leq C(x, \pi) \qquad \forall \pi \in \Pi(x). \tag{21}$$

Para terminar la demostración, vamos a probar que $w(x)$ es la máxima cota inferior. Es decir, dado $\varepsilon > 0$, vamos a construir una sucesión de acciones admisibles $\pi^\varepsilon = \{a_k^\varepsilon\}$ en $\Pi(x)$ tal que $C(x, \pi^\varepsilon) < w(x) + \varepsilon$.

De (19) tenemos que, para $\varepsilon(1-\beta)/2 > 0$ y el estado inicial x, existe $a_0^\varepsilon \in A(x)$ tal que

$$c(x, a_0^\varepsilon) + \beta w(f(x, a_0^\varepsilon)) < w(x) + \varepsilon(1-\beta)/2.$$

Para $\varepsilon(1-\beta)/2 > 0$ y $x_1^\varepsilon := f(x, a_0^\varepsilon)$, usamos (19) para escoger $a_1^\varepsilon \in A(x_1^\varepsilon)$ tal que

$$c(x_1^\varepsilon, a_1^\varepsilon) + \beta w(f(x_1^\varepsilon, a_1^\varepsilon)) < w(x_1^\varepsilon) + \varepsilon(1-\beta)/2.$$

Continuando con este proceso, de manera inductiva tenemos una sucesión de acciones admisibles $\pi^\varepsilon := \{a_k^\varepsilon\}$ tal que

$$c(x_k^\varepsilon, a_k^\varepsilon) + \beta w(x_{k+1}^\varepsilon) < w(x_k^\varepsilon) + \varepsilon(1-\beta)/2 \qquad \forall k. \tag{22}$$

Ahora multiplicamos cada término en (22) por β^k y sumamos para obtener

$$\sum_{k=0}^{K-1} \beta^k c(x_k^\varepsilon, a_k^\varepsilon) + \beta^K w(x_K^\varepsilon) < w(x) + [\varepsilon(1-\beta)/2] \sum_{k=0}^{K-1} \beta^k.$$

Haciendo $K \to \infty$ y usando (20), vemos que

$$C(x, \pi^\varepsilon) \leq w(x) + \varepsilon/2 < w(x) + \varepsilon.$$

De esta desigualdad y (21), concluimos que $w(x) = \text{ínf}_{\pi \in \Pi(x)} C(x, \pi)$. □

En el teorema anterior se establece, bajo la condición (20), que una solución w de la ecuación de Bellman coincide con la función de valor. Si w es acotada, entonces (20) se satisface trivialmente puesto que $\beta \in (0, 1)$.

3.2 Existencia de soluciones

A continuación damos condiciones suficientes para que exista una solución de la ecuación de Bellman.

Hipótesis 3.1. *Sea* $\mathbb{K} := \{(x, a) \in X \times A \mid a \in A(x)\}$.

(a) *Para cada* $x \in X$, $A(x)$ *es compacto.*

(b) *Para cada* $x_0 \in X$ *y cualquier sucesión* $\{x_k\}$ *en* X *que converge a* x_0 *se tiene que,*

  (b.1) *si* $(x_0, b) \in \mathbb{K}$, *entonces existe* $(x_k, b_k) \in \mathbb{K}$, $k \in \mathbb{N}$, *tal que* $b_k \to b$, *y*

  (b.2) *si* $(x_k, a_k) \in \mathbb{K}$, $k \in \mathbb{N}$, *entonces* $\{(x_k, a_k)\}$ *tiene una subsucesión que converge a algún punto* (x_0, a) *en* $\mathbb{K}$.

Lema 3.1. *Supongamos que* $\mathbb{K}$ *satisface la hipótesis 3.1. Si* $\phi : \mathbb{K} \to \mathbb{R}$ *es continua, entonces la función*

$$\varphi(x) := \text{mín}\{\phi(x, a) \mid a \in A(x)\}$$

es continua.

Demostración. Primero notemos que la función φ está bien definida puesto que cada conjunto $A(x)$ es compacto, es decir, para cada x existe $a_x \in A(x)$ tal que $\varphi(x) = \phi(x, a_x)$. Veamos ahora que φ es continua en cada punto $x_0 \in X$. Sea $\{x_k\}$ cualquier sucesión que converge a x_0. Si $b_0 \in A(x_0)$ tal que $\varphi(x_0) = \phi(x_0, b_0)$, entonces por la hipótesis 3.1(b.1) existe $(x_k, b_k) \in \mathbb{K}$, $k \in \mathbb{N}$, que converge a (x_0, b_0). Como ϕ es continua, $\lim_{k\to\infty} \phi(x_k, b_k) = \phi(x_0, b) = \varphi(x_0)$. Además, para cada $k \in \mathbb{N}$, $\varphi(x_k) \leq \phi(x_k, b_k)$ de donde tenemos

$$\limsup_{k\to\infty} \varphi(x_k) \leq \limsup_{k\to\infty} \phi(x_k, b_k) = \varphi(x_0). \tag{23}$$

Por otro lado, sea a_k tal que $\varphi(x_k) = \phi(x_k, a_k)$, $k \in \mathbb{N}$. Consideremos cualquier subsucesión $\{\phi(x_{k_l}, a_{k_l})\}$ convergente de $\{\varphi(x_k)\}$. Por la hipótesis 3.1(b.2), no hay pérdida de generalidad al suponer que $\{(x_{k_l}, a_{k_l})\}$ converge a algún punto $(x_0, a) \in \mathbb{K}$ (de lo contrario escogemos una subsucesión que sí lo sea). Entonces

$$\lim_{l\to\infty} \phi(x_{k_l}, a_{k_l}) = \phi(x_0, a) \geq \varphi(x_0),$$

es decir, $\lim_{l\to\infty} \varphi(x_{k_l}) \geq \varphi(x_0)$. Por ser $\{\varphi(x_{k_l})\}$ cualquier subsucesión convergente de $\{\varphi(x_k)\}$, concluimos que

$$\liminf_{k\to\infty} \varphi(x_k) \geq \varphi(x_0).$$

De esta desigualdad y (23) vemos que $\lim_{k\to\infty} \varphi(x_k) = \varphi(x_0)$, lo cual demuestra que φ es continua en x_0. □

Sea $CA(X)$ el espacio de todas las funciones $w : X \to \mathbb{R}$ continuas y acotadas. Para cada $w \in CA(X)$, definimos el *operador de Bellman*

$$T[w](x) := \inf_{a\in A(x)} \{c(x, a) + \beta w(f(x, a))\}.$$

Para cada $w \in CA(X)$ consideramos la norma $\|w\| := \sup_{x\in X} w(x)$.

Lema 3.2. *Sea f continua y sea c continua y acotada. Supongamos que $\mathbb{K}$ satisface la hipótesis 3.1. Entonces el operador de Bellman T satisface lo siguiente*

(i) *$T[w] \in CA(X)$ para cada $w \in CA(X)$,*
(ii) *T es una contracción, es decir,*

$$\|T[w_1] - T[w_2]\| \leq \beta\|w_1 - w_2\| \quad \forall w_1, w_2 \in CA(X).$$

Demostración. Sea $w \in CA(X)$. Entonces $\phi(x, a) := c(x, a)+\beta w(f(x, a))$ es continua y acotada en $\mathbb{K}$. Puesto que cada $A(x)$ es compacto, el operador de Bellman se puede escribir como $T[w](x) = \min_{a\in A(x)}\{\phi(x, a)\}$. Por el lema 3.1, $T[w]$ es continua y, además, acotada lo que demuestra (i).

Para demostrar (ii), consideremos $w_1, w_2 \in CA(X)$ arbitrarias. Entonces, para cada $x \in X$,

$$\begin{aligned} T[w_1](x) &= \min_{a\in A(x)} \{c(x,a) + \beta w_2(f(x,a)) + \beta[w_1(f(x,a)) - w_2(f(x,a))]\} \\ &\leq \min_{a\in A(x)} \{c(x,a) + \beta w_2(f(x,a)) + \beta\|w_1 - w_2\|\} \\ &= T[w_2](x) + \beta\|w_1 - w_2\|. \end{aligned}$$

De forma análoga podemos verificar que $T[w_2](x) \leq T[w_1](x) + \beta\|w_1 - w_2\|$ para cada $x \in X$. Por lo tanto

$$\|T[w_1] - T[w_2]\| = \sup_{x\in X} |T[w_1](x) - T[w_2](x)| \leq \beta\|w_1 - w_2\|.$$

Esto demuestra (ii) ya que w_1 y w_2 son arbitrarias. □

Teorema 3.3 (Iteración de valores)**.** *Sea f continua y sea c continua y acotada. Supongamos que $\mathbb{K}$ satisface la hipótesis 3.1. Si $w_0 \in CA(X)$, entonces la sucesión de funciones*

$$w_k(x) := \min_{a\in A(x)} \{c(x,a) + w_{k-1}(f(x,a))\}, \qquad k \in \mathbb{N},\ x \in X, \quad (24)$$

converge a la función de valor v. Más aún, si $h : X \to A$ es una función tal que $h(x) \in A(x)$ y

$$\begin{aligned} v(x) &= \min_{a\in A(x)} \{c(x,a) + v(f(x,a))\} \\ &= c(x,h(x)) + v(f(x,h(x))) \end{aligned} \quad (25)$$

para todo $x \in X$, entonces $\hat{\pi} := \{\hat{a}_k = h(x_k) \mid k = 0, 1, \ldots\}$ minimiza el costo total descontado, es decir,

$$C(x_0, \hat{\pi}) = v(x_0) \qquad x_0 \in X.$$

Demostración. Notemos que $(CA(X), \|\cdot\|)$ es un espacio métrico completo y que, por el lema 3.2, $T : CA(X) \to CA(X)$ es una contracción. Entonces, por el teorema de punto fijo de Banach, T tiene un único punto fijo $w \in CA(X)$ que es el límite de la sucesión $\{T[w_{k-1}]\}$. Dicho punto fijo w satisface (20), por ser una función acotada, entonces w es igual a la función de valor v, por el teorema 3.2.

Si h satisface (25), entonces

$$\begin{aligned} v(x_0) &= c(x_0, h(x_0)) + \beta v(f(x_0, h(x_0))) \\ &= c(x_0, \hat{a}_0) + \beta v(x_1) \\ &= c(x_0, \hat{a}_0) + \beta[c(x_1, h(x_1)) + \beta v(x_2)]. \end{aligned}$$

Por inducción tenemos $v(x_0) = \sum_{k=0}^{K} \beta^k c(x_k, \hat{a}_k) + \beta^{K+1} v(x_{K+1})$. Como v es acotada, al hacer $K \to \infty$ obtenemos

$$v(x_0) = \lim_{K\to\infty} \sum_{k=0}^{K} \beta^k c(x_k, \hat{a}_k) = C(x_0, \hat{\pi}). \qquad \square$$

4. Solución numérica de la ecuación de Bellman

Aunque la descripción de la ecuación de Bellman es transparente y breve, usualmente no existe forma explicita de solución. El lector interesado en profundizar sobre los aspectos numéricos y computacionales para resolver la ecuación de Bellman, puede consultar las referencias [4,13,22].

De acuerdo con Powell [16, p.4], la razón más citada para evitar el uso de programación dinámica es la dimensión de los espacios involucrados. Por ejemplo, supóngase que se requiere optimizar un inventario de N diferentes productos con $0, 1, \ldots, M$ unidades. Tomando un vector de N entradas para describir el inventario de acuerdo a la cantidad de cada producto, existirán M^N estados posibles. La dimensión del espacio de estados crece de forma exponencial, dificultad conocida como la maldición dimensional—en palabras de Bellman (véase el prefacio de [1]), «curse of dimensionality». Powell señala al menos tres fuentes de maldición dimensional: el espacio de estados, el espacio de acciones y, en el caso estocástico, el espacio de realizaciones. Estas fuentes determinarán el tipo de problema y método de aproximación a usar. Por ejemplo, si se tiene un problema con espacio de estados y acciones continuos, se deberá utilizar un mecanismo para aproximar con problemas de naturaleza discreta, es decir, con espacios de estados y acciones a lo más numerables. Y si la dimensión de algunas de estas fuentes es grande, entonces es recomendable considerar un problema aproximado de menor dimensión, usar optimización heurística u otra técnica.

El esquema de aproximación más simple de usar en PD es el algoritmo de iteración de valores (véase el algoritmo 2) el cual se puede implementar de forma muy intuitiva a partir de la demostración del teorema 3.3.

Algoritmo 2: Iteración de Valores

Entrada: Función de costo c, factor de descuento β, dinámica del sistema f, restricciones $A(x)$, función continua y acotada w_0 inicial y criterio de paro $\epsilon > 0$.

Salida: Función de valor v y política estacionaria óptima h aproximadas.

repetir

  para cada $x \in X$ **hacer**

    $w_{k+1}(x) \leftarrow \min_{a \in A(x)} \{c(x,a) + \beta w_k(f(x,a))\}$

    $v(x) \leftarrow w_{k+1}(x)$

    $h(x) \leftarrow \arg\min_{a \in A(x)} \{c(x,a) + \beta v(f(x,a))\}$

  fin

hasta que $\|w_{k+1} - w_k\| < \epsilon$

En el siguiente ejemplo se aplica este esquema de solución. El código MATLAB y otros detalles se pueden consultar en [9].

Ejemplo 4.1 (El problema del lago de Dechert y O'Donnell [5])**.** En términos económicos, los lagos son de vital importancia en una comunidad agrícola. En parte, proveen de agua, peces, zonas recreativas y aumentan la plusvalía de complejos residenciales. Por otro lado, son parte medular de la actividad agrícola pues sirven como vertedero de desechos.

Los desechos agrícolas son ricos en fósforo: nutriente principal para hierbas y algas marinas. En consecuencia, arrojar cantidades significativas de fósforo crea condiciones favorables para el crecimiento y reproducción de algas. Las algas consumen oxígeno y secretan toxinas, por ello, su sobrepoblación afecta el desarrollo de peces, haciendo el lago inseguro para usarlo como zona recreativa y al mismo tiempo, deteriora la plusvalía de zonas residenciales. En resumen, la descarga de fósforo en un lago, afecta su beneficio económico.

En contraste, de forma indirecta, dichas descargas forman parte de la actividad agropecuaria y por consiguiente también se relacionan con la utilidad económica. Dechert y O'Donnell [5] plantean un problema de optimización estocástica, para calcular la cantidad de fósforo óptima a descargar en un lago, que maximice la utilidad generada por la actividad agrícola. En esta sección se considera la versión determinista.

Sea x_k el nivel de fósforo en la etapa k del lago y a_k la correspondiente descarga de fósforo. Empleando una función de utilidad usada por ecologistas, Dechert y O'Donell proponen el siguiente problema de

optimización descontado

$$\max_{a_k} \sum_{k=0}^{\infty} \beta^k \left(\log(a_k + \text{eps}) - \kappa x_k^2\right) \tag{26}$$

$$\text{sujeto a} \qquad x_{k+1} = b\, x_k + \frac{x_k^q}{1 + x_k^q} + a_k, \tag{27}$$

donde el parámetro b representa la fracción de fósforo que queda en el lago partiendo de un tiempo k hasta un tiempo $k+1$, el parámetro q está asociado a un punto en el que ya no es posible revertir un lago saturado con fósforo y eps es un parámetro de regularización relacionado con la precisión aritmética de punto flotante.

Como se ha mencionado, para un problema de maximización, la ecuación (17) toma la forma

$$\begin{aligned} v(x) &= \sup_{a \in A(x)} \{r(x,a) + \beta v(f(x,a))\}, \\ f(x,a) &= b\, x + x^q/(1+x^q) + a, \\ r(x,a) &= \log(a + \text{eps}) - \kappa x^2, \end{aligned} \tag{28}$$

con $x \in [0, 2.5]$. Dechert y O'Donnell prueban la existencia de al menos dos estados estacionarios óptimos. El primero de ellos, caracteriza la utilidad óptima relacionada con el beneficio económico generado por el agua al mantener un lago limpio. El segundo estado representa un lago contaminado e irrecuperable con altos niveles de fósforo, para el cual el beneficio económico proviene de la actividad agrícola. En otras palabras, la evolución del nivel de fósforo descrita por (26) funciona como un «switch» biológico. Sorprendentemente, aunque esta dinámica produce una función de valor continua, la correspondiente política óptima no lo es para ciertos parámetros.

Consideremos el conjunto de parámetros $b = 0.52$, $q = 2$, $\kappa = 0.33$. Discretizando el intervalo $[0, 2.5]$ con $N = 500$ puntos igualmente espaciados y resolviendo iterativamente la ecuación (28), con $\beta = 0.997$, se obtiene la gráfica mostrada en la figura 3. Usando la discretización mencionada, se construye, para la k-ésima iteración, el vector

$$\mathbf{v}_k = [v(x_1), \ldots, v(x_{100})]_k. \tag{29}$$

Con este vector, se escogió como criterio de paro un valor de 10^{-5} para el error cuadrático medio:

$$\text{Error} = \frac{\|\mathbf{v}_{k+1} - \mathbf{v}_k\|^2}{N}. \tag{30}$$

Las componentes del vector inicial $\mathbf{v}_0$, se tomaron como números aleatorios en [0,1]. La optimización se realizó discretizando con 5000 puntos el espacio de búsqueda $A(x)$, evaluando la función objetivo en dichos

puntos y escogiendo el valor óptimo. Dado que, en general, el punto $f(x,a)$ no corresponderá a un punto de la discretización, el valor $v(f(x,a))$ se obtuvo haciendo una interpolación lineal para $v(x)$.

La figura 3 muestra en la gráfica superior con línea punteada, la política de estados estacionarios, es decir, la política que conserva el nivel de fósforo actual. Etiquetamos con x^L, x^C y x^U los estados estacionarios óptimos: los puntos de intersección con la política óptima. En línea gris continua, representamos la política de descarga óptima que corresponde a el lago limpio. Como podemos ver, a medida que el nivel de fósforo actual en el sistema aumenta, la política óptima de descarga decrece hasta que el nivel de fósforo en el lago alcanza una concentración $x^U = 0.75$. A partir de este nivel umbral, el lago pierde su capacidad natural de autoregeneración, en consecuencia la política óptima «salta» a niveles más altos de descarga, niveles relacionados con el beneficio de la actividad agrícola. Observando la función de valor, la utilidad neta correspondiente al intervalo con niveles de fósforo por debajo de 0.75 es mayor.

Tal como se vio en la demostración del teorema 3.3, la función de valor resulta continua en todo punto, sin embargo, no es diferenciable en x^U. Notemos también que en x^U, la política óptima tiene una discontinuidad. Este comportamiento fue estudiado por Skiba [20] y Sethi [18, 19], de manera independiente. Posteriormente retomado por Dechert y Nishimura [6], Dechert y O'Donnell [5], entre otros.

5. Procesos Markovianos de decisión

Hasta ahora hemos considerados problemas deterministas. En esta sección veremos cómo se puede extender el algoritmo de PD a problemas estocásticos. Específicamente, vamos a concentrarnos en problemas descontados estacionarios como los de la sección 3, pero ahora el sistema evoluciona de acuerdo con la dinámica

$$x_{k+1} = f(x_k, a_k, \xi_k), \qquad k = 0, 1, \ldots, \tag{31}$$

donde $\{\xi_k\}$ es una sucesión de variables aleatorias independientes e idénticamente distribuidas. Para simplificar la exposición, denotemos por ξ a un elemento cualquiera de la sucesión $\{\xi_k\}$ y supongamos que ξ toma valores en algún espacio euclidiano.

Supongamos que las acciones a_k se toman en función del estado x_k, digamos $a_k = h(x_k)$, entonces la sucesión de variables aleatorias

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

forman un *proceso de Markov*, siempre que f y h sean medibles. Es decir, para cualquier conjunto medible $B \subseteq X$, $\{x_k\}$ satisface la propiedad

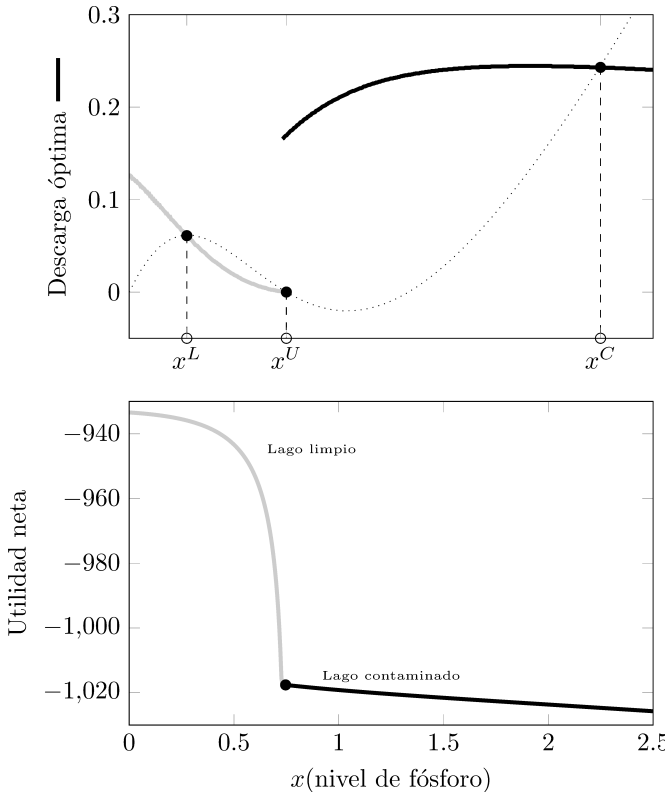

Figura 3. Política óptima y función de valor para el problema del lago (Ejemplo 4.1). Arriba denotamos con linea punteada la política de estados estacionarios. En color gris la descarga óptima que corresponde a niveles de fósforo en un lago limpio y en negro la descarga óptima para un lago contaminado. Respectivamente x^L y x^C, etiquetan los estados óptimos estacionarios para un lago limpio y uno contaminado; con x^U denotamos el valor umbral. Abajo la función de valor correspondiente.

de Markov

$$\mathbb{P}(x_{k+1} \in B \mid x_0, \ldots, x_k) = \mathbb{P}(x_{k+1} \in B \mid x_k). \tag{32}$$

En tal caso, se dice que el proceso $\{x_k\}$ *no tiene memoria* porque la distribución de probabilidad del estado x_{k+1}, condicionada a la historia pasada $x_0, \ldots, x_{k-1}, x_k$ depende únicamente del estado actual x_k. Véase, por ejemplo, [12].

Supongamos en una etapa cualquiera el sistema está en el estado x y se toma la acción $a \in A(x)$, entonces la medida de probabilidad del estado siguiente depende del par (x, a) y de la medida de probabilidad μ de la variable aleatoria ξ. Concretamente, definimos la *ley de transición*

$$Q(B \mid x, a) := \mu(\{s \in S | f(x, a, s)\})$$

donde $B \subseteq X$ es cualquier conjunto medible y (S, Σ_S, μ) es el espacio de probabilidad donde está definida la variable aleatoria ξ.

Un *procesos Markoviano de decisión*, estacionario con criterio de costo descontado, puede escribirse de forma simplificada como

$$(X, A, \{A(x) \mid x \in X\}, Q, c, \beta). \tag{33}$$

5.1 El modelo en espacios de Borel

En el caso general, X y A son *espacios de Borel*, en lugar de ser subconjuntos de algún espacio euclidiano. Un espacio de Borel es cualquier subconjunto medible de espacio métrico (X, d) completo y separable dotado con la $\sigma-$álgebra de Borel; es decir, la mínima $\sigma-$álgebra que contiene a todos los conjuntos abiertos de (X, d).

Como ejemplos de espacios de Borel tenemos: (i) cualquier conjunto finito o numerable, con la métrica $d(x, y) = 1$ si $x \neq y$; (ii) el intervalo $[0, 1]$ con $d(x, y) = |x - y|$. De manera sorprendente, cualquier espacio de Borel es *isomorfo* a un espacio del tipo (i) o del tipo (ii). Una demostración de este resultado puede consultarse en [3] o en [7]. El término *isomorfo* significa que existe una biyección f entre dos espacios medibles tal que tanto f como f^{-1} son medibles.

Las acciones en cada etapa pueden escogerse, de manera determinista o aleatoria, de acuerdo con ciertas reglas que pueden depender de la *historia admisible*

$$i_k := (x_0, a_0, \ldots, x_{k-1}, a_{k-1}, x_k) \quad a_l \in A(x_l),\ 0 \leq l \leq k-1.$$

Una *política* $\pi = \{\pi_k\}$ es una sucesión de medidas de probabilidad de la forma $\pi_k(\cdot \mid i_k)$ concentradas en $A(x_k)$, es decir, para cada historia admisible i_k

$$\pi_k(A(x_k) \mid i_k) = 1 \quad k \geq 0.$$

Pedimos, además, que $\pi_k(D \mid \cdot)$ sea una función medible para cada k y cualquier conjunto medible $D \subseteq A$. Al conjunto de todas las políticas lo denotamos por Π.

Dada una política $\pi \in \Pi$ y un estado inicial x, definimos el *costo esperado*

$$C(x, \pi) := \mathbb{E}_x^\pi \sum_{k=0}^{\infty} \beta^k c(x_k, a_k), \qquad x \in X. \tag{34}$$

Queremos encontrar una política $\hat{\pi}$ tal que

$$v(x) := \inf_{\pi \in \Pi} C(x, \pi) = C(x, \hat{\pi}), \quad \forall x \in X.$$

5.1.1. Políticas estacionarias óptimas

Un subconjunto de políticas en Π que son más sencillas de implementar son las *estacionarias deterministas*, es decir, cuando existe una función

$h : X \to A$ tal que

$$\pi_k(D \mid i_k) = I_D(h(x_k)),$$

donde $i_k = (x_0, a_0, \ldots, x_k)$, $D \subseteq X$ es cualquier conjunto medible e I es la *función indicadora* ($I_D(a) = 1$ si $a \in D$, $I_D(a) = 0$ en otro caso). Notemos que cualquier política estacionaria determinista es Markoviana, es decir, las acción $a_k = h(x_k)$ depende sólo del estado actual y no de toda la historia i_k. En este caso denotamos esta política estacionaria como $\pi = \{h^\infty\}$.

Además de las hipótesis requeridas en el teorema 3.3 (ahora con la topología métrica tanto en el espacio de estados como en el espacio de acciones), pedimos que se cumpla la siguiente condición. Para cualquier función medible y acotada $w : X \to \mathbb{R}$, la función

$$W(x, a) := \int_X w(y)Q(dy \mid x, a) = \mathbb{E}[w(f(x, a, \xi))] \tag{35}$$

es continua en $\mathbb{K} = \{(x, a) \mid a \in A(x)\}$. Bajo estas hipótesis, se puede demostrar que la sucesión de funciones $w_0 \equiv 0$,

$$w_{k+1}(x) = \min_{a \in A(x)} \{c(x, a) + \beta\mathbb{E}[w_k(f(x, a, \xi))]\}$$

es tal que $v(x) = \lim_{k\to\infty} w_k(x)$. Más aún, existe una función medible $h : X \to A$ tal que, para cada $x \in X$, $h(x) \in A(x)$,

$$v(x) = c(x, h(x)) + \beta \int_X v(y)Q(dy \mid x, h(x))$$

y la política $\hat{\pi} := \{h^\infty\}$ es óptima

$$C(x, h^\infty) = \inf_{\pi \in \Pi} C(x, \pi).$$

Observación 5.1. En la discusión previa supusimos que la ley de transición Q está determinada por la función f. Sin embargo, en ocasiones sólo se tiene la distribución del estado siguiente sin que haya una relación explícita como en (31). Si éste es el caso, tenemos el algoritmo 3.

Algoritmo 3: Iteración de Valores, caso estocástico.

Entrada: Función de costo c, factor de descuento β, restricciones $A(x)$, ley de transcisión Q, función continua y acotada inicial w_0 y criterio de paro $\epsilon > 0$.

Salida: Función de valor v y política estacionaria óptima h aproximadas.

repetir

  para cada $x \in X$ **hacer**

$$w_{k+1}(x) \leftarrow \min_{a \in A(x)} \left\{ c(x,a) + \beta \int w_k(y) Q(dy|x,a) \right\}$$

$$v(x) \leftarrow w_{k+1}(x)$$

$$h(x) \leftarrow \arg\min_{a \in A(x)} \{c(x,a) + \beta v(f(x,a))\}$$

  fin

hasta que $\|w_{k+1} - w_k\| < \epsilon$

6. Comentarios finales

En el artículo se discutieron algunas clases de problemas de optimización multietapa. Se presentaron aspectos teóricos y computacionales, principalmente en problemas deterministas, y se finalizó con una introducción a los procesos Markovianos de decisión. Como se observa, el APD permite resolver problemas en tiempo discreto (o multietapa) como los ejemplos 2.1 y 4.1. También permite aproximar soluciones de problemas planteados en tiempo continuo como problemas de cálculo de variaciones como el ejemplo 2.2.

Existen otras técnicas para resolver problemas de optimización multietapa, por ejemplo el método de los multiplicadores de Lagrange o métodos variacionales. Este tipo de métodos usualmente requieren de la diferenciabilidad de las funciones c y f, mientras que en PD esto no es necesario. También generan políticas óptimas llamadas de *lazo abierto*, es decir, que sólo dependen del tiempo (como las soluciones de una ecuación diferencial ordinaria). Por esta razón no es natural extender estas técnicas a problemas estocásticos, ya que en un sistema estocástico no se sabe con certeza cuál será el estado en cada instante de tiempo.

El enfoque seguido en el artículo es similar a la estructura del libro clásico de Bellman [1], uno de los principales desarrolladores de la programación dinámica. Sin embargo, las demostraciones de la sección 3 fueron adaptadas de Stokey y Lucas [23] donde se formula un modelo ligeramente distinto. Resultados similares pueden encontrarse también en Sundaram [24, Cap. 12]. Para quienes empiezan a estudiar

estos temas se sugiere, por ejemplo, los libros de Bertsekas [2] y de Puterman [17]. En cuanto a los aspectos computacionales, se recomienda Powell [16] y Sniedovich [21]; adicionalmente, se pueden consultar algunos capítulos de Brandimarte [4], Judd [13], Miranda y Fackler [14] y Stachurski [22]. Para procesos Markovianos de decisión en espacios de Borel pueden consultarse Bertsekas y Shreve [3], Dynkin y Yushkevich [7] o Hernández-Lerma y Lasserre [11].

Bibliografía

[1] R. Bellman, *Dynamic programming*, Dover, Princeton, New Jersey, 2003.

[2] D. P. Bertsekas, *Dynamic programming and optimal control*, 3.ª ed., vol. 1, Athena Scientific, Nashua, New Heaven, 2005.

[3] D. P. Bertsekas y S. E. Shreve, *Stochastic optimal control: the discrete-time case*, Athenea Scientific, Belmont, Massachusetts, 1996.

[4] P. Brandimarte, *Numerical methods in finance and economics: a matlab-based introduction*, 2.ª ed., John Wiley & Sons, Hoboken, New Jersey, 2013.

[5] W. D. Dechert y S. O'Donnell, «The stochastic lake game: A numerical solution», *Journal of Economic Dynamics and Control*, vol. 30, núm. 9, 2006, 1569–1587.

[6] W. Dechert y K. Nishimura, «A complete characterization of optimal growth paths in an aggregated model with a non-concave production function», *Journal of Economic Theory*, vol. 31, núm. 2, 1983, 332 – 354.

[7] E. Dynkin y A. Yushkevich, *Controlled markov processes*, Grundlehren der mathematischen Wissenschaften, Springer-Verlag, New York, New York, 1979.

[8] P. C. Fishburn, *Utility theory for decision making*, John Wiley & Sons, New York, New York, 1970.

[9] D. Gónzalez-Sánchez, S. Díaz-Infante y F. Peñu nuri, «Código MATLAB Miscelanea Matemática», `https://github.com/SaulDiazInfante/C-digo-MATLAB-Miscelanea-Matematica`, Accessed: 2017-11-05.

[10] M. Held y R. M. Karp, «A dynamic programming approach to sequencing problems», *J. Soc. Indust. Appl. Math.*, vol. 10, 1962, 196–210.

[11] O. Hernández-Lerma y J. B. Lasserre, *Discrete-time markov control processes: basic optimality criteria*, vol. 30, Springer Science & Business Media, New York, New York, 1996.

[12] ———, *Markov chains and invariant probabilities*, vol. 211, Birkhäuser, New York, New York, 2003.

[13] K. L. Judd, *Numerical methods in economics*, MIT Press, Cambridge, Massachusetts, 1998.

[14] M. J. Miranda y P. L. Fackler, *Applied computational economics and finance*, MIT press, Cambridge, Massachusetts, 2002.

[15] J. D. Murray, *Mathematical biology i. an introduction*, 3.ª ed., Interdisciplinary Applied Mathematics, vol. 17, Springer, New York, 2002.

[16] W. B. Powell, *Approximate dynamic programming: Solving the curses of dimensionality*, 2.ª ed., vol. 703, John Wiley & Sons, Hoboken, New Jersey, 2007.

[17] M. L. Puterman, *Markov decision processes. discrete stochastic dynamic programming mvspa*, 1.ª ed., Wiley Series in Probability and Statistics, Wiley-Interscience, Hoboken, New Jersey, 2005.

[18] S. P. Sethi, «Nearest feasible paths in optimal control problems: Theory, examples, and counterexamples», *Journal of Optimization Theory and Applications*, vol. 23, núm. 4, 1977, 563–579.

[19] ———, «Optimal advertising policy with the contagion model», *Journal of Optimization Theory and Applications*, vol. 29, núm. 4, 1979, 615–627.
[20] A. K. Skiba, «Optimal growth with a convex-concave production function», *Econometrica*, vol. 46, núm. 3, 1978, 527–39.
[21] M. Sniedovich, *Dynamic programming: foundations and principles*, 2.ª ed., CRC press, Hoboken, New Jersey, 2011.
[22] J. Stachurski, *Economic dynamics: Theory and computation*, Computational economics, MIT Press, Cambridge, Massachusetts, 2009.
[23] N. L. Stokey y R. E. Lucas, *Recursive methods in economic dynamics*, Harvard University Press, Boston, Massachusetts, 1989.
[24] R. K. Sundaram, *A first course in optimization theory*, Cambridge university press, New York, New York, 1996.
[25] O. Sundstrom y L. Guzzella, «Dpm function», `http://www.idsc.ethz.ch/research-guzzella-onder/downloads.html`, Accessed: 2010-09-30.
[26] ———, «A generic dynamic programming matlab function», en *Control Applications,(CCA) & Intelligent Control, (ISIC), 2009 IEEE*, IEEE, 2009, 1625–1630.