BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS
Posgrado en Ciencias Matemáticas

APROXIMACIÓN VÍA Q-LEARNING EN PROBLEMAS DE CONSUMO-INVERSIÓN

TESIS

QUE PARA OBTENER EL GRADO DE
DOCTOR EN CIENCIAS MATEMÁTICAS

PRESENTA

M. en C. Ruy Alberto López Ríos

DIRECTOR DE TESIS:

Dr. Hugo Adán Cruz Suárez

Puebla, Pue. A 25 de agosto del 2021

DR. SEVERINO MUÑOZ AGUIRRE
SECRETARIO DE INVESTIGACIÓN Y
ESTUDIOS DE POSGRADO, FCFM-BUAP
P R E S E N T E:

Por este medio le informo que el C:

RUY ALBERTO LÓPEZ RÍOS

estudiante del Doctorado en Ciencias (Matemáticas), ha cumplido con las indicaciones que el Jurado le señaló en el Coloquio que se realizó el día 2 de julio de 2021, con la tesis titulada:

"APROXIMACIÓN VÍA Q-LEARNING EN PROBLEMAS DE CONSUMO-INVERSIÓN"

Por lo que se le autoriza a proceder con los trámites y realizar el examen de grado en la fecha que se le asigne.

A T E N T A M E N T E.
H. Puebla de Z, a 11 de agosto de 2021

DRA. PATRICIA DOMÍNGUEZ SOTO
COORDINADORA DEL POSGRADO
EN MATEMÁTICAS.

D*PDS/mtrv

Facultad
de Ciencias
Físico Matemáticas

Av. San Claudio y 18 Sur, edif. FM1
Ciudad Universitaria, Col. San
Manuel, Puebla, Pue. C.P. 72570
01 (222) 229 55 00 Ext. 7550 y 7552

AGRADECIMIENTO

Al Consejo Nacional de Ciencia y Tecnología (CONACYT-México) agradezco el apoyo brindado durante los estudios de doctorado en Ciencias Matemáticas, sin duda, ayuda indispensable en mi formación.

Agradecimiento personal también a mi director de tesis, Dr. Hugo Adán Cruz Suárez por el seguimiento en todos estos años.

También, a quienes participaron como jurado haciendo revisión y comentarios sobre esta tesis:
Dra. María Selene Georgina Chávez Rodríguez, Dr. Víctor Hugo Vázquez Guevara, Dr. Fernando Velasco Luna, Dr. Francisco Solano Tajonar Sanabria, Dra. Hortensia Reyes Cervantes. Muchas gracias.

Al Dr. Miguel Antonio Jiménez Pozo y al Dr. David Villa Hernández.

A María Teresa Rivera Velázquez, la Dra. Patricia Domínguez Soto y personal administrativo de la FCFM-BUAP.

Ruy Alberto López Ríos

DEDICATORIA

Para Inés, Inés e Inés.

A mi familia y compañeros del laboratorio.

Dedicatoria para el Ing. Gerónimo Lover Martínez y el Dr. José Dionicio Zacarías Flores, quienes inesperadamente adelantaron su camino. Que en paz descansen.

Ruy Alberto López Ríos

Índice general

Introducción

El control óptimo estocástico es un área de las matemáticas dedicada a resolver problemas de optimización cuya evolución en el tiempo es susceptible a ser influenciado por variables aleatorias.

Los procesos de control de Markov (PCM) son problemas de control estocástico, también conocidos como procesos de decisión de Markov, procesos de Markov controlados. Los PCM aparecen en muchos campos, por ejemplo, ingeniería, economía, investigación de operaciones, estadística, administración de recursos, control de epidemias, etc.

La técnica básica para resolver problemas de control de Markov es la programación dinámica, técnica creada por Richard Bellman en 1953, con el propósito de optimizar problemas complejos que pueden ser discretizados y secuencializados. Sin embargo, se complica su utilidad al trabajar con espacios de interés de grandes dimensiones, en la literatura esto se conoce como *maldición de la dimensionalidad* [18], [22].

Los métodos numéricos para resolver problemas de control óptimo presentan un limitante al momento de discretizar el tiempo o los espacios de estados o acciones. Con la finalidad de aumentar la eficiencia y precisión de tales métodos se han creado versiones que combinan técnicas iterativas en adición con otras herramientas que eviten o que minimicen el efecto de tales complicaciones.

Machine learning es el estudio de algoritmos computacionales que automáticamente mejoran a través de la experiencia. Los algoritmos machine learning construyen un modelo matemático sobre datos muestrales, conocido como training data, para hacer predicciones o tomar decisiones sin ser explícitamente programado para ello. Dentro de las herramientas de machine learning se encuentran los algoritmos del *Refuerzo por Aprendizaje*, que se ocupa de cómo un agente debe realizar acciones en un entorno para maximizar alguna recompensa o minimizar algún costo asociado.

La técnica Q-Learning pertenece a una clase de algoritmos machine learning, y es un algoritmo de refuerzo por aprendizaje el cual fue introducido por Watkins en 1989, [29]. Q-learning es un método asíncrono de programación dinámica que permite al controlador aprender a actuar óptimamente en dominios Markovianos, en [30] demuestran que la solución en los algoritmos de Q-learning convergen a los valores de acciones óptimos con probabilidad uno. Por lo tanto, Q-learning es una técnica adecuada para implemen-

tar en la solución, en particular, del problema de consumo-inversión, objeto de estudio de esta tesis. En el método *Q-Learning* se aplica el conocido método de iteración de valores (o de políticas) sobre funciones particulares llamadas *Q-Factores*. Este método iterativo realiza el cálculo aproximado de algún valor óptimo actualizándolo de manera iterada.

Este escrito, hace énfasis particular en el *problema de consumo-inversión*, que surgió inicialmente de los trabajos sobre problemas de portafolios por Samuelson en 1969 [23]. Samuelson consideró un inversor quien quiere incrementar su riqueza en cada punto de tiempo y, desea asignar esta riqueza entre inversión y consumo. El objetivo es maximizar la utilidad de consumo del inversor, donde el inversor elige sobre la mejor estrategia de consumo-inversión para colocar la inversión total entre varias acciones [17], [32], [8] y [28].

La principal aportación de este estudio es proponer procedimientos de aproximación basados en técnicas machine learning para encontrar la estrategia de inversión en un problema de consumo-inversión. Se realiza mediante el uso de Q-Learning combinado con algoritmos clásicos de iteración de valores para procesos de decisión de Markov en tiempo discreto con el método de aproximación estocástica de Robbins-Monro [20], definiendo y optimizando una función auxiliar, y, discretizando los elementos de interés, logrando una aproximación de la política óptima.

El trabajo está organizado como sigue. Capítulo 1, se introducen preliminares del tema. El Capítulo 2 presenta el método Q-Learning en contexto general. El Capítulo 3 presenta los algoritmos de aproximación para problemas de consumo inversión. consideramos espacios de estados y espacios de acciones continuos, y dedicamos una sección al proceso de discretización de tales espacios, así que ellos pueden ser adaptados a Q-Learning. La riqueza del inversor se supone que es afectada por una función de producción. En este trabajo, la inversión se asume que es perturbada por un ruido aleatorio, que asegure la reinversión de capital. Se supone una función logarítmica y exponencial como utilidad del consumo, [15]. Además, Las condiciones que garantizan la existencia de una solución son cubiertos, tanto para el problema original como para la convergencia de la aproximación Q-Learning. Finalmente, la teoría es implementada en ejemplos numéricos, [8].

Algunas notas sobre abreviaturas y notación utilizadas a lo largo del escrito pueden ser encontradas en el *Apéndice A*. También, algunos resultados y definiciones se encuentran en el *Apéndice B*.

Capítulo 1

Procesos de decisión de Markov

En un problema de control óptimo se tiene un sistema dinámico cuyo comportamiento puede ser influenciado por algunas variables, que son llamadas *variables de control (o acción, decisión)*. Los controles pueden ser aplicados en cualquier momento dado y son elegidas de acuerdo a reglas conocidas como *políticas de control*.

También se cuenta con una función llamada *criterio de rendimiento* definida sobre el conjunto de políticas de control, que mide o evalúa en cierto sentido la respuesta o la calidad del sistema a las políticas de control utilizadas. Entonces el Problema de Control Óptimo es determinar una política de control que optimice un criterio de rendimiento.

En este primer capítulo se detallan los generales de los procesos de control de Markov, se definen y clasifican a las políticas y se establece el Teorema de Programación Dinámica.

1.1. Procesos de control de Markov

Definición 1.1.1 *Un modelo de control de Markov es una quíntupla*

$$\big(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x)|x \in \mathbb{X}\}, \mathcal{Q}, R\big)$$

la cual consiste de

a) un espacio de Borel $\mathbb{X}$*, llamado* espacio de estados*;*

b) un espacio de Borel $\mathbb{A}$*, llamado* espacio de control *(o acción);*

c) una familia $\{\mathbb{A}(x)|x \in \mathbb{X}\}$ *de subconjuntos medibles* $\mathbb{A}(x)$ *de* $\mathbb{A}$*, donde* $\mathbb{A}(x)$ *denota el* conjunto de controles *(o acciones)* admisibles *cuando el sistema está en el estado* $x \in \mathbb{X}$*, y con la propiedad de que el conjunto*

$$\mathbb{K} = \{(x, a)|x \in \mathbb{X}, a \in \mathbb{A}(x)\}$$

de parejas estado-acción admisibles es un subconjunto medible de $\mathbb{X} \times \mathbb{A}$*;*

d) un kernel estocástico $\mathcal{Q}$ *sobre* $\mathbb{X}$ *dado* $\mathbb{K}$ *llamada* ley de transición *(ver Apéndice B);*

e) una función medible $R : \mathbb{K} \to \mathbb{R}$ *llamada* función de recompensa (o costo).

Denotamos x_t y a_t el estado del sistema y el control (o acción) aplicado en el tiempo t, respectivamente, la evolución del sistema puede ser descrita como sigue. Si el sistema está en el estado $x_t = x \in \mathbb{X}$ en el tiempo t y el control $a_t = a \in \mathbb{A}(x)$ es aplicado, entonces dos cosas ocurren:

i) una recompensa $R(x, a)$ es incurrida, y

ii) el sistema se mueve al siguiente estado x_{t+1}, que es una variable aleatoria con valores en $\mathbb{X}$ con distribución $\mathcal{Q}(\cdot|x, a)$.

Una vez que ocurre la transición al nuevo estado, un nuevo control es elegido y el proceso es repetido.

En muchas aplicaciones, la evolución de un Proceso de Control de Markov es especificada por una ecuación de diferencias de la forma

$$x_{t+1} = F(x_t, a_t, \xi_t) \;\; t = 0, 1, ... \quad \textit{con } x_0 \textit{ conocido.}$$

donde $\{\xi_t\}$ es una sucesión de variables aleatorias independientes e idénticamente distribuidas con valores en algún espacio S y distribución común μ e independiente del estado inicial x_0.

En este caso, la ley de transición $\mathcal{Q}$ resulta ser, para $B \in \mathcal{B}(\mathbb{X})$:

$$\begin{aligned} \mathcal{Q}(B|x, a) &= \mu(\{s \in S | F(x, a, s) \in B\}) \\ &= \int_S I_B[F(x, a, s)]\mu(ds) \\ &= \mathbb{E}\Big[\mathbb{I}_B[F(x, a, \xi)]\Big], \end{aligned}$$

donde $\mathbb{I}_B(\cdot)$ es la función indicadora del conjunto B. $\mathbb{E}$ denota el valor esperado con respecto a μ y ξ es una variable aletoria genérica con distribución μ.

Para especificar un problema de control óptimo se requiere de un criterio de rendimiento a optimizar. Un criterio de rendimiento típico es el de recompensa total esperada hasta un cierto tiempo N, digamos

$$J_N(\pi, x) := \mathbb{E}_x^\pi \left\{ \sum_{t=0}^{N-1} R(x_t, a_t) \right\},$$

donde $\mathbb{E}_x^\pi$ indica el valor esperado (ver Apéndice B) cuando se aplica la política $\pi = \{a_t\}$, dado el estado inicial $x_0 = x$. Entonces el problema de control óptimo consiste en maximizar la función $\pi \to J_N(\pi, x)$ sobre Π, para todo x. Una política π^* tal que

$$J_N(\pi^*, x) = \sup_{\Pi} J_N(\pi, x), \qquad \forall x \in \mathbb{X},$$

se dice ser una *política óptima*, y la utilidad total máxima, es decir,

$$J_N^*(x) = \sup_{\Pi} J_N(\pi, x), \qquad x \in \mathbb{X},$$

se refiere a la *utilidad óptima* o *función de valor* del problema de control.

El número N es llamado el *horizonte de planificación*, representa el número de etapas en que el sistema es operado. Puede ser finito o infinito. En el primer caso el problema de control se dice ser problema de horizonte finito, en el segundo caso, problema de horizonte infinito. Si $N = +\infty$, entonces la sumatoria expresada anteriormente puede no converger para algunas políticas π. Por razones técnicas o interpretación física o económica es conveniente considerar otros criterios de rendimiento. Uno de ellos es la recompensa total descontada esperada,

$$V(\pi, x) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \alpha^t \, R(x_t, a_t) \right], \qquad \forall x \in \mathbb{X},$$

donde α $(0 < \alpha < 1)$ es llamado *factor de descuento*. El problema de control óptimo correspondiente a maximizar es llamada *recompensa α-descontada*.

1.2. Políticas

Considere el modelo de control de Markov de la Definición 1.1.1, y para cada $t = 0, 1, ...$, se tienen las siguientes definiciones:

Definición 1.2.1 *Se define el espacio H_t de* historias admisibles *hasta el tiempo t como*

$$\begin{aligned} H_0 &:= \mathbb{X} \\ H_t &= \mathbb{K}^t \times \mathbb{X} = \mathbb{K} \times H_{t-1} \quad \textit{para } t = 1, 2, ... \end{aligned}$$

Una política de control aleatorizada *o* política de control *es una sucesión $\pi = \{\pi_t\}_{t=0}^{\infty}$ de kerneles estocásticos π_t sobre $\mathbb{A}$ dada la* t-historia admisible

$$h_t = (x, a_0, x_1, a_1 ..., x_{n-1}, a_{t-1}, x_t),$$

para cada $t = 0, 1, ...$, con $(x_t, a_t) \in \mathbb{K}$. Esto es, para $t \geq 0$, π_t es un kernel estocástico si satisface las siguientes propiedades:

a) $\pi_t(\cdot \,|\, h_t)$ *es una medida de probabilidad sobre* $\mathbb{X}$*, para cada* $h_t \in \mathbb{K}^t \times \mathbb{X}$*.*

b) $\pi_t(B \,|\, \cdot)$ *es una variable aleatoria para cada* $B \in \mathcal{B}(\mathbb{X})$*, donde* $\mathcal{B}(\mathbb{X})$ *denota la* σ*-álgebra de Borel de* $\mathbb{X}$*.*

Es decir, se satisface la restricción

$$\pi_t\big(A(x_t)|h_t\big) = 1 \quad \forall h_t \in H_t, \;\; t = 0, 1, \ldots .$$

El conjunto de todas las políticas es denotado por Π*.*

Una política $\pi = \{\pi_t\}$ puede ser interpretada definiendo una sucesión $\{a_t\}$ de variables aleatorias con valores en $\mathbb{A}$, llamadas acciones (o controles), tal que, para toda historia h_t y $t = 0, 1, ...$, la distribución de a_t es $\pi(\cdot|h_t)$, y que está concentrada en $\mathbb{A}(x_t)$, el conjunto de acciones admisibles en el estado x_t.

Definición 1.2.2 Φ *denota el conjunto de todos los kérneles estocásticos* $\phi \in \mathcal{P}(\mathbb{A}|\mathbb{X})$ *tal que* $\phi(A(x)|x) = 1$*, para todo* $x \in \mathbb{X}$*.* $\mathbb{F}$ *denota el conjunto de todas las funciones medibles* $f : \mathbb{X} \to \mathbb{A}$ *satisfaciendo que* $f(x) \in \mathbb{A}(x), \;\forall x \in \mathbb{X}$*.*

Suposición 1.2.3 $\mathbb{K} = \{(x, a)|x \in \mathbb{X}, a \in \mathbb{A}(x)\}$ *contiene al grafo de una función medible de* $\mathbb{X}$ *a* $\mathbb{A}$*; esto es, existe una función medible* $f : \mathbb{X} \to \mathbb{A}$ *tal que* $f(x) \in \mathbb{A}(x), \;\forall x \in \mathbb{X}$*.*

La medibilidad es útil para conservar estructura o correspondencia entre los espacios de estados y acciones.

Definición 1.2.4 *Una política* $\pi = \{\pi_t\} \in \Pi$ *se dice ser una*

a) política aleatorizada de Markov *si existe una sucesión* $\{\phi_t\}$ *de kérneles estocásticos* $\pi_t \in \Phi$ *tal que*

$$\pi_t(\cdot|h_t) = \phi_t(\cdot|x_t), \quad \forall h_t \in H_t, \quad t = 0, 1, ...;$$

b) política aleatorizada estacionaria *si existe una sucesión* $\phi \in \Phi$ *tal que*

$$\pi_t(\cdot|h_t) = \phi(\cdot|x_t), \quad \forall h_t \in H_t, \quad t = 0, 1, ...$$

El conjunto de todas las políticas aleatorizadas de Markov (aleatorizadas estacionarias) es denotado por Π_{RM}*, (*Π_{RS}*). Notar que* $\Pi_{RS} \subset \Pi_{RM} \subset \Pi$*.*

Además, $\pi = \{\pi_t\}$ *se dice que es una*

c) política determinista *si existe una sucesión* $\{g_t\}$ *de funciones medibles* $g_t : H_t \to \mathbb{A}$ *tal que, para todo* $h_t \in H_t$ *y* $t = 0, 1, \ldots$, $g_t(h_t) \in \mathbb{A}(x_t)$ *y* $\pi_t(\cdot|h_t)$ *está concentrada en* $g_t(h_t)$*, es decir,*

$$\pi_t(C \mid h_t) = \mathbb{I}_C\big[g_t(h_t)\big] \text{ para todo } C \in \mathcal{B}(\mathbb{A}).$$

d) política determinista de Markov *si existe una sucesión* $\{f_t\}$ *de funciones* $f_t \in \mathbb{F}$ *tal que* $\pi_t(\cdot|h_t)$ *está concentrado en* $f_t(x_t) \in \mathbb{A}(x_t)$ *para todo* $h_t \in H_t$ *y* $t = 0, 1, \ldots$ *;*

e) política determinista estacionaria *si existe una función* $f \in \mathbb{F}$ *tal que* $\pi_t(\cdot|h_t)$ *está concentrada en* $f(x_t) \in \mathbb{A}(x_t)$ *para todo* $h_t \in H_t$ *y* $t = 0, 1, \ldots$.

Π_D*,* Π_{DM} *y* Π_{DS} *denotan respectivamente el conjunto de todas las políticas deterministas, deterministas de Markov y deterministas estacionarias. Notar que* $\Pi_{DS} \subset \Pi_{DM} \subset \Pi_D \subset \Pi$.

1.3. Programación dinámica

Una opción para resolver problemas de control óptimo es hacer uso del *Teorema de la programación dinámica*, el cual provee un algoritmo para encontrar la función objetivo J^* y una política óptima π^*. Ver [2], [12], [21].

Presentamos tal teorema en casos, para un criterio de rendimiento α-descontado con horizonte finito e infinito.

Horizonte finito. Consideramos el modelo de control de Markov

$$\big(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x)|x \in \mathbb{X}\}, \mathcal{Q}, R, \alpha\big)$$

y el problema de control de interés es maximizar el criterio de rendimiento con horizonte finito

$$J(\pi, x) := \mathbb{E}_x^\pi\left[\sum_{t=0}^{N} \alpha^t\, R(x_t, a_t)\right], \qquad \forall x \in \mathbb{X}, \tag{1}$$

donde $\alpha \in (0, 1)$ es un factor de descuento.

J^* denota la función de valor (o función objetivo), es decir,

$$J^*(x) := \sup_{\Pi} J(\pi, x) \qquad \forall x \in \mathbb{X}, \tag{2}$$

el problema es encontrar una política $\pi \in \Pi$ tal que

$$J(\pi^*, x) = J^*(x) \qquad \forall x \in \mathbb{X}. \tag{3}$$

Teorema 1.3.1 *Sean $J_0, J_1, ..., J_N$ funciones sobre $\mathbb{X}$ definidas por*

$$J_N(x) := 0, \tag{4a}$$

y para $t = N-1, N-2, ..., 0$,

$$J_t(x) := \max_{a \in \mathbb{A}(x)} \left\{ R(x,a) + \alpha \int_{y \in \mathbb{X}} J_{t+1}(y)\, \mathcal{Q}(dy|x,a) \right\}. \tag{4b}$$

Suponga que estas funciones son medibles y que, para cada $t = 0, ..., N-1$, existe un selector $f_t \in \mathbb{F}$ tal que $f_t(x) \in \mathbb{A}(x)$ alcance el máximo en (4b) para toda $x \in \mathbb{X}$; esto es, $\forall x \in \mathbb{X}$ y $t = 0, ..., N-1$,

$$J_t(x) = R(x, f_t) + \alpha \int_{y \in \mathbb{X}} J_{t+1}(dy)\, \mathcal{Q}(y|x, f_t). \tag{5}$$

Entonces la política determinista de Markov $\pi^* = \{f_0, f_1, ..., f_{N-1}\}$ es óptima, y la función de valor J^* es

$$J^*(x) = J_0(x) = J(\pi^*, x), \qquad \forall x \in \mathbb{X}. \tag{6}$$

La hipótesis de existencia de tal selector en el Teorema 1.3.1 se encuentra resumida en la *condición de selección medible* sobre los elementos del modelo de control.

Suposición 1.3.2 ***Condición de selección medible.*** *El modelo de control de Markov y una función medible dada $\vartheta : \mathbb{X} \to \mathbb{R}$ son tales que*

$$\vartheta^*(x) := \sup_{\mathbb{A}(x)} \left\{ R(x,a) + \alpha \int_{y \in \mathbb{X}} \vartheta(y)\, \mathcal{Q}(dy|x,a) \right\}.$$

es medible y existe un selector $f \in \mathbb{F}$ tal que la función alcance su máximo en $f(x) \in \mathbb{A}(x)$ para todo x, es decir,

$$\vartheta^*(x) := R(x,f) + \alpha \int_{y \in \mathbb{X}} \vartheta(y)\, \mathcal{Q}(dy|x,f) \qquad \forall x \in \mathbb{X}.$$

Existen múltiples condiciones para que la *condición de selección medible* sea satisfecha, ver [11].

Horizonte infinito. Por otro lado, consideramos ahora el problema de control de interés que consiste en maximizar una utilidad total esperada con horizonte infinito. Sea

$$\big(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x)|x \in \mathbb{X}\}, \mathcal{Q}, R, \alpha\big)$$

el modelo de control de Markov y el criterio de rendimiento con horizonte infinito

$$V(\pi, x) := \mathbb{E}_x^{\pi}\left[\sum_{t=0}^{\infty} \alpha^t \, R(x_t, a_t)\right], \qquad \forall x \in \mathbb{X}, \pi \in \Pi, \tag{7}$$

donde $\alpha \in (0, 1)$ es un factor de descuento.

Definición 1.3.3 *Una política π^* satisfaciendo*

$$V(\pi^*, x) = \sup_{\Pi} V(\pi, x) =: V^*(x), \qquad \forall x \in \mathbb{X},$$

se dice ser política óptima $\alpha-$descontada*, y V^* es llamada* función de valor (función objetivo o función de utilidad óptima) $\alpha-$descontada*.*

Además, si usamos V_n para denotar la utilidad en la $n-$ésima etapa

$$V_n(\pi, x) := \mathbb{E}_x^{\pi}\left[\sum_{t=0}^{n-1} \alpha^t R(x_t, a_t)\right],$$

podemos escribir, por el Teorema de Convergencia Monótona, que

$$V(\pi, x) := \lim_{n\to\infty} V_n(\pi, x),$$

Definición 1.3.4 *Una función medible $\vartheta : \mathbb{X} \to \mathbb{R}$ se dice ser una solución de la ecuación de optimalidad con utilidad $\alpha-$descontada si satisface*

$$\vartheta(x) = \max_{a\in\mathbb{A}(x)} \left\{ R(x, a) + \alpha \int_{\mathbb{X}} \vartheta(y) \, \mathcal{Q}(dy|x, a) \right\}, \tag{8}$$

donde Q es la correspondiente ley de transición inducida por una ecuación en diferencias de acuerdo a la Sección 1.2 en [12].

Definición 1.3.5 *Las funciones de iteración de valor están definidas como*

$$\vartheta_n(x) = \max_{a\in\mathbb{A}(x)} \left\{ R(x, a) + \alpha \int_{\mathbb{X}} \vartheta_{n-1}(y) \mathcal{Q}(dy|x, a) \right\} \tag{9}$$

para todo $x \in \mathbb{X}$ y $n = 1, 2, ...,$ con $\vartheta_0(x) = 0$.

Se puede demostrar que V^* es una solución de la ecuación de optimalidad con recompensa $\alpha-$descontada (8), es decir,

$$V^*(x) = \max_{a\in\mathbb{A}(x)} \left\{ R(x, a) + \alpha \int_{\mathbb{X}} V^*(y) \, \mathcal{Q}(dy|x, a) \right\}, \tag{10}$$

y que V^* satisface:

$$V^*(x) = \lim_{n\to\infty} \vartheta_n(x), \quad \forall x \in \mathbb{X}. \tag{11}$$

Antes, requerimos de algunas suposiciones para garantizar la existencia de solución para la optimización de (7).

Suposición 1.3.6 *Para todo estado $x \in \mathbb{X}$:*

a) El conjunto de controles restringidos $\mathbb{A}(x)$ es compacto;

b) La recompensa $R(x,a)$ es semicontinua inferior en $a \in \mathbb{A}(x)$.

c) La función $R'(x,a) := \int R(y)\mathcal{Q}(dy|x,a)$ es continua en $a \in \mathbb{A}(x)$ para toda función $R \in \mathbb{B}(\mathbb{X})$, donde $\mathbb{B}(\mathbb{X})$ denota el espacio de Banach de las funciones reales, acotadas y medibles sobre $\mathbb{X}$, con respecto a la norma $||R|| := sup_{\mathbb{X}}|R(x)|$.

Suposición 1.3.7 *Existen constantes no-negativas $\bar{c}$ y β, con $1 \leq \beta \leq 1/\alpha$, y una función de peso $w \geq 1$ sobre $\mathbb{X}$ tal que para todo estado $x \in \mathbb{X}$:*

a) $\sup_{\mathbb{A}(x)} |R(x,a)| \leq \bar{c}\, w(x)$; y

b) $\sup_{\mathbb{A}(x)} \int w(y)\, \mathcal{Q}(dy|x,a) \leq \beta\, w(x)$.

Suposición 1.3.8 *Para todo estado $x \in \mathbb{X}$, la función $w'(x,a) := \int w(y)\mathcal{Q}(dy|x,a)$ es continua en $a \in \mathbb{A}(x)$.*

Para todo $n = 1, 2, ...$, ϑ_n es la recompensa óptima en la $n-$etapa, es decir,

$$\vartheta_n(x) = \sup_{\Pi} V_n(\pi, x) \quad \forall x \in \mathbb{X}.$$

El siguiente teorema establece que la sucesión $\{\vartheta_n\}$ converge geométricamente en la w-norma a V^*.

Teorema 1.3.9 *Suponga que las Suposiciones 1.3.6, 1.3.7 y 1.3.8 son satisfechas. Sea β la constante en la Suposición 1.3.7 b), y definimos $\gamma := \alpha\beta$. Entonces:*

a) La función de valor α-descontada V^* es la única solución de la ecuación de optimalidad con recompensa α-descontada, es decir,

$$V^*(x) = \max_{a\in\mathbb{A}(x)} \left\{ R(x,a) + \alpha \int_{\mathbb{X}} V^*(y)\, \mathcal{Q}(dy|x,a) \right\}, \quad \forall x \in \mathbb{X},$$

en el espacio $\mathbb{B}_w(\mathbb{X})$, y

$$\|\vartheta_n - V^*\|_w \leq \bar{c}\,\gamma^n/(1-\gamma), \quad n = 1, 2, ..., \tag{12}$$

donde $\bar{c}$ es la constante en la Suposición 1.3.7 a).

b) Existe un selector $f_* \in \mathbb{F}$ tal que $f_*(x) \in \mathbb{A}(x)$ alcanza el máximo en (10) para todo estado x, es decir,

$$V^*(x) = R(x, f_*) + \alpha \int_{\mathbb{X}} V^*(y)\, \mathcal{Q}(dy|x, f_*), \quad \forall x \in \mathbb{X}, \tag{13}$$

y la política determinista estacionaria f_*^∞ es óptima α-descontada; recíprocamente, si $f_*^\infty \in \Pi_{DS}$ es óptima α-descontada, entonces satisface (13).

c) Una política π^* es óptima $\alpha-$descontada si y sólo si la función de recompensa correspondiente $V(\pi^*, \cdot)$ satisface la ecuación de optimalidad con recompensa α-descontada (10).

d) Si una política óptima α-descontada existe, entonces existe una política determinista estacionaria que es óptima $\alpha-$descontada.

Definición 1.3.10 *Dada una función medible $\nu : \mathbb{X} \to \mathbb{R}$, y $0 \leq \alpha \leq 1$ definimos el operador de programación dinámica, $\mathcal{T}_\alpha(\nu)$ a la función dada por*

$$\mathcal{T}_\alpha \nu(x) := \sup_{a \in \mathbb{A}(x)} \left\{ R(x, a) + \alpha \int_{y \in \mathbb{X}} \nu(y)\, \mathcal{Q}(dy|x, a) \right\}, \quad x \in \mathbb{X}. \tag{14}$$

Si el supremo en (14) es alcanzado en alguna acción $a \in \mathbb{A}(x)$ para todo $x \in \mathbb{X}$, podemos sustituir *sup* por *max*. Para $0 \leq \alpha \leq 1$, se puede demostrar que $\mathcal{T}_\alpha$ es un operador de contracción sobre el espacio $\mathbb{B}(\mathbb{X})$ como lo indica la Proposición 1.3.12.

Requerimos del siguiente lema, que es equivalente a la Proposición D.5 en [12]. Ver en *Apéndice B* las definiciones requeridas.

Lema 1.3.11 *Sea $\mathbb{K} = \{(x, a) | x \in \mathbb{X}, a \in \mathbb{A}(x)\}$ y $\mathbb{A}(x)$ compacto, y sea $v : \mathbb{K} \to \mathbb{R}$ una función medible dada. Definimos*

$$v^*(x) := \sup_{\mathbb{A}(x)} v(x, a), \quad x \in \mathbb{X}. \tag{15}$$

a) Si $v(x, \cdot)$ es l.s.c sobre $\mathbb{A}(x)$ para todo $x \in \mathbb{X}$, entonces existe un selector $f \in \mathbb{F}$ tal que $f(x) \in \mathbb{A}(x)$ alcanza el máximo en (15) para todo $x \in \mathbb{X}$, esto es,

$$v^*(x) = v(x, f) \quad \forall x \in \mathbb{X},$$

y v^* es una función medible.

b) Si el mapeo $x \mapsto \mathbb{A}(x)$ es u.s.c. y v es l.s.c. y acotada por abajo sobre $\mathbb{X}$, entonces existe un selector $f \in \mathbb{F}$ que satisface (15), y además, v^* es l.s.c. y acotada por abajo sobre $\mathbb{X}$.

c) Suponga que $x \mapsto \mathbb{A}(x)$ *es u.s.c.,* v *es l.s.c., y además,*

$$\sup_{\mathbb{A}(x)} |v(x,a)| \leq k\, w(x) \quad \forall x \in \mathbb{X},$$

donde k *es una constante y* $w(\cdot) \geq 1$ *es una función continua sobre* $\mathbb{X}$*. Entonces existe un selector* $f \in \mathbb{F}$ *que satisface (15),* v^* *es l.s.c., y*

$$|v^*(x)| \leq k\, w(x) \quad \forall x \in \mathbb{X};$$

esto es, v^* *es una función l.s.c. en el espacio* $\mathbb{B}_w(\mathbb{X})$ *y su* $w-$*norma satisface* $\|v^*\|_w \leq k$*.*

Proposición 1.3.12 *Para* $0 < \alpha < 1$*, si las Suposiciones 1.3.6, 1.3.7 y 1.3.8 se satisfacen, y sea* $\mathcal{T}_\alpha$ *el mapeo de la Definición 1.3.10. Entonces:*

a) $\mathcal{T}_\alpha$ *es un operador contracción sobre* $\mathbb{B}_w(\mathbb{X})$*, con módulo* $\gamma := \alpha\beta < 1$*; esto es* $\mathcal{T}_\alpha$ *mapea* $\mathbb{B}_w(\mathbb{X})$ *a sí mismo y*

$$\|\mathcal{T}_\alpha u - \mathcal{T}_\alpha u'\| \leq \gamma\, \|u - u'\|_w \quad \forall u, u' \in \mathbb{B}_w(\mathbb{X}). \tag{16a}$$

b) Para toda función $u \in \mathbb{B}_w(\mathbb{X})$ *existe un selector* $f \equiv f_u \in \mathbb{F}$ *tal que*

$$\mathcal{T}_\alpha u(x) = R(x,f) + \alpha \int_{\mathbb{X}} u(y)\, \mathcal{Q}(dy|x,f) \quad \forall x \in \mathbb{X},\ \ 0 \leq \alpha \leq 1. \tag{16b}$$

Una de las grandes complicaciones de la programación dinámica es combatir la *maldición de la dimensionalidad* originada al manejar espacios de estados o acciones demasiado grandes, por lo cual se suele optar por métodos de refuerzo de aprendizaje (Reinforcement Learning) [26], [24], [25]. El siguiente Capítulo trata específicamente del método Q-Learning.

Capítulo 2

Q-Learning

Los algoritmos de Refuerzo de Aprendizaje (RL) utilizan la función objetivo (o función de valor) de la programación dinámica, ver Sección 1.3. En RL la función objetivo es almacenada en funciones llamadas *Q-Factores* [4], [10], [31]. En este capítulo se definen los Q-Factores. El camino que se sigue para la obtención del algoritmo final que se ha de utilizar en el Capítulo 3 es el siguiente:

Introducción de los Q-Factores al método iteración de valor

$\downarrow$

Aplicación del método de Robbins-Monro

$\downarrow$

Algoritmo de iteración de valor para Q-Factores para costos descontados

Finalmente, se cierra este capítulo con un ejemplo del funcionamiento de Q-Learning trabajando sobre un espacio de 5 estados y 5 acciones.

2.1. Q-Factores

Consideremos el modelo de control de Markov

$$\big(\tilde{\mathbb{X}}, \tilde{\mathbb{A}}, \{\tilde{\mathbb{A}}(x)|x \in \mathbb{X}\}, \hat{\mathcal{Q}}, R, \alpha\big).$$

Supondremos que la función de recompensa está dada por $R(x,a) = u(x-a)$, donde $u : \mathbb{R} \to \mathbb{R}$ es una función de utilidad y que el problema de control de interés sea maximizar el criterio de rendimiento con horizonte finito

$$J(\pi, x) := \mathbb{E}_x^{\pi}\left[\sum_{t=0}^{N} \alpha^t\, u(x_t - a_t)\right], \qquad \forall x \in \mathbb{X},$$

donde $\alpha \in (0,1)$ es un factor de descuento. Por razones de conveniencia que se especificarán en el Capítulo 3, se realiza un cambio de variable que llamaremos *consumo*

$\tilde{a}_t := x_t - a_t$.

Redefiniendo la *Ecuación de Optimalidad de utilidad α-descontada*, [10]

$$J^*(\tilde{x}_i) := \max_{\tilde{a}\in\tilde{\mathbb{A}}(\tilde{x}_i)} \left\{ u(\tilde{a}) + \alpha \sum_{j=1}^{|\tilde{\mathbb{X}}|} J^*(\tilde{x}_j)\, \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) \right\} \tag{17}$$

para todo $\tilde{x}_i \in \tilde{\mathbb{X}}$, siendo $\tilde{\mathbb{X}}$ el conjunto finito discretizado de $\mathbb{X}$, y $|\tilde{\mathbb{X}}|$ denotando su cardinalidad, donde $J^*(\tilde{x}_i)$ denota el $i-$ésimo elemento del vector de la función objetivo asociado con la política óptima.

Definición 2.1.1 *Para $\tilde{x}_i \in \tilde{\mathbb{X}}$, $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$ se define la función de* Q-factores *como sigue:*

$$Q(\tilde{x}_i, \tilde{a}) := u(\tilde{a}) + \alpha \sum_{j=1}^{|\tilde{\mathbb{X}}|} J^*(\tilde{x}_j)\, \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}). \tag{18}$$

Así que,

$$J^*(\tilde{x}_i) = \max_{\tilde{a}\in\tilde{\mathbb{A}}(\tilde{x}_i)} Q(\tilde{x}_i, \tilde{a}).$$

Por lo tanto, podemos escribir la siguiente ecuación que es conocida como la versión de la *Ecuación de Optimalidad de Bellman para Q-Factores*:

$$Q(\tilde{x}_i, \tilde{a}) = u(\tilde{a}) + \alpha \sum_{j=1}^{|\tilde{\mathbb{X}}|} \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q(\tilde{x}_j, \tilde{b}) \tag{19}$$

para $\tilde{x}_i \in \tilde{\mathbb{X}}$, $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$.

Es momento de definir un algoritmo de iteración de valor a los Q-Factores definidos en esta sección.

2.2. Iteración de valor para Q-Factores

Un algoritmo clásico en aproximación es el de iteración de valores [1], [19], que se aplica a los Q-Factores de la siguiente manera.

Paso 1 Hacer $k = 0$, especificar un $\epsilon > 0$ y seleccionar valores arbitrarios para el vector Q_0, e.d., para todo $\tilde{x}_i \in \tilde{\mathbb{X}}$ y $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$, pongamos $Q_0(\tilde{x}_i, \tilde{a}) = 0$.

Paso 2 Para cada $\tilde{x}_i \in \tilde{\mathbb{X}}$ y $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$, calcular:

$$Q_{k+1}(\tilde{x}_i, \tilde{a}) = u(\tilde{a}) + \alpha \sum_{j=1}^{|\tilde{\mathbb{X}}|} \hat{\mathcal{Q}}(\tilde{x}_j | \tilde{x}_i, \tilde{a}) \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b}).$$

Paso 3 Calcular para cada $\tilde{x}_i \in \tilde{\mathbb{X}}$:

$$J^{k+1}(\tilde{x}_i) = \max_{\tilde{a}\in\tilde{\mathbb{A}}(i)} Q_{k+1}(\tilde{x}_i, \tilde{a})$$

y

$$J^{k}(\tilde{x}_i) = \max_{\tilde{a}\in\tilde{\mathbb{A}}(\tilde{x}_i)} Q_{k}(\tilde{x}_i, \tilde{a}).$$

Entonces, si $||\overrightarrow{J}^{k+1} - \overrightarrow{J}^{k}|| < \epsilon(1-\alpha)/2\alpha$ (o si $k \geq kmax$, el número máximo de iteraciones), ir al *Paso 4*.
En otro caso, incrementar el valor de k en 1, y regresar al *Paso 2*.

Paso 4 Para cada $\tilde{x}_i \in \tilde{\mathbb{X}}$, elegir $d(\tilde{x}_i) \in \underset{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_i)}{\operatorname{argmax}} Q(\tilde{x}_i, \tilde{b})$, donde d denota la $\epsilon-$política óptima, y detener.

En la siguiente sección, se introduce el algoritmo de aproximación estocástica conocido como *Robbins-Monro* [20], dentro del algoritmo de iteración de valor de esta sección.

2.3. Algoritmo de Robbins-Monro en Q-Factores

La definición de función de Q-Factores en la forma de la ecuación de Bellman, se presenta de la siguiente manera:

$$\begin{aligned} Q(\tilde{x}_i, \tilde{a}) &= u(\tilde{a}) + \alpha \sum_{j=1}^{|\tilde{\mathbb{X}}|} \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q(\tilde{x}_j, \tilde{b}) \\ &= \sum_{j=1}^{|\tilde{\mathbb{X}}|} \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) \left[u(\tilde{a}) + \alpha \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q(\tilde{x}_j, \tilde{b}) \right] \\ &= \mathbb{E}\left[u(\tilde{a}) + \alpha \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q(\tilde{x}_j, \tilde{b}) \right]. \end{aligned}$$

La esperanza puede ser estimada usando el método de Monte Carlo, por ejemplo. Esto es, es posible estimar los Q-Factores usando el esquema del algoritmo de Robbins-Monro [20]:

$$\begin{aligned} Q_{k+1}(\tilde{x}_i, \tilde{a}) &= (1-\lambda_{k+1}) Q_k(\tilde{x}_i, \tilde{a}) + \lambda_{k+1}\left[u(\tilde{a}) + \alpha \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b}) \right] \\ &= Q_k(\tilde{x}_i, \tilde{a}) \; + \; \lambda_{k+1}\left[u(\tilde{a}) + \alpha \max_{\tilde{b}\in\tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b}) - Q_k(\tilde{x}_i, \tilde{a}) \right], \end{aligned}$$

donde la función λ_k es denominada *tasa de refuerzo o aprendizaje*. Derivado del Algoritmo de Robbins-Monro se obtiene $\lambda_k = 1/(k+1)$. Otras tasas de refuerzo distintas a la usual pueden ser consideradas, siempre y cuando satisfagan las condiciones:

$$\sum_{k=1}^{\infty} \lambda_k = \infty \quad y \quad \sum_{k=1}^{\infty} {\lambda_k}^2 < \infty.$$

El siguiente paso ahora, es el de modificar el algoritmo previo, para resolver problemas de decisión de Markov, con recompensa (o utilidad) descontada.

2.4. Iteración de valor para Q-Factores para procesos de decisión de Markov con recompensa descontada

El algoritmo para la aplicación de Q-Learning mediante iteración de valor es:

Paso 1 Inicializar los Q-Factores. Para todo $(\tilde{x}, \tilde{a})$ donde $\tilde{x} \in \tilde{\mathbb{X}}$ y $\tilde{a} \in \tilde{\mathbb{A}}(x)$,

$$Q(\tilde{x}, \tilde{a}) = 0.$$

Hacer $k = 0$, el número de transiciones de estado.
El algoritmo se ejecutará un número k_{max} de iteraciones. Iniciar la simulación del sistema en un estado arbitrario $\tilde{x}_i \in \tilde{\mathbb{X}}$.

Paso 2 En el estado actual $\tilde{x}_i$, seleccionar una acción $\tilde{a}$ con probabilidad $\frac{1}{|\tilde{\mathbb{A}}(\tilde{x}_i)|}$.

Paso 3 Simular la acción $\tilde{a}$. Producir el siguiente estado $\tilde{x}_j$. Sea $u(\tilde{a})$ la utilidad inmediata ganada en la transición al estado $\tilde{x}_j$ desde al estado actual $\tilde{x}_i$ bajo la influencia de la acción $\tilde{a}$. Incrementar el valor de k en una unidad. Actualizar el valor de la función λ.

Paso 4 Actualizar $Q(\tilde{x}_i, \tilde{a})$ usando la ecuación:

$$Q(\tilde{x}_i, \tilde{a}) = (1 - \lambda_k) Q(\tilde{x}_i, \tilde{a}) + \lambda_k \left\{ u(\tilde{a}) + \alpha \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q(\tilde{x}_j, \tilde{b}) \right\}.$$

Paso 5 Si $k < k_{max}$, hacer $\tilde{x}_i \leftarrow \tilde{x}_j$ e ir al *Paso 2*. De otra manera, ir al *Paso 6*.

Paso 6 Para cada $\tilde{x}_i \in \tilde{\mathbb{X}}$, seleccionar

$$d(\tilde{x}_i) \in \operatorname*{argmax}_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_i)} Q(\tilde{x}_i, \tilde{b}).$$

En la siguiente Sección, se presenta un ejemplo haciendo uso de tal algoritmo para resolver un problema de control para un sistema de 5 estados, para un criterio de rendimiento con recompensa descontada. Como se verá a detalle en la Sección 2.6, este algoritmo converge a la política óptima.

2.5. Algoritmo Q-Learning

Es momento de reunir los algoritmos de iteración de valor dados en la sección 2.2 y 2.3, proponiendo el método de discretización, para ejecutar Q-Learning.

Notación Las aproximaciones iteradas de la política óptima $\pi^* \in \Pi$ generadas por el método Q-Learning para cada $\tilde{x} \in \tilde{\mathbb{X}}$ son denotados por $\tilde{\pi}_k(\tilde{x})$, $k = 0, 1, 2, \ldots$.

Algoritmo 1: Iteración de valor para Q-factores

Inicialización $k = 0$, especificar $\epsilon > 0$ (o definir k_{max} número máximo de iteraciones), $Q_0(\tilde{x}_i, \tilde{a}) = 0$ para cada $\tilde{x}_i \in \tilde{\mathbb{X}}$;

repeat

  for $\tilde{x}_i \in \tilde{\mathbb{X}}$ *y* $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$ **do**

$$Q_{k+1}(\tilde{x}_i, \tilde{a}) \longleftarrow u(\tilde{a}) + \alpha \sum_{j=1}^{\eta} \hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b});$$

$$J^{k+1}(\tilde{x}_i) \longleftarrow \max_{\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)} Q_{k+1}(\tilde{x}_i, \tilde{a});$$

$$J^{k}(\tilde{x}_i) \longleftarrow \max_{\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)} Q_{k}(\tilde{x}_i, \tilde{a}) \ ;$$

  end

  $k \longleftarrow k + 1$;

until $||J^{k+1} - J^k|| < \epsilon(1-\alpha)/2\alpha$, *(o si* $k \geq k_{max}$*)*;

return $\tilde{\pi}^*(\tilde{x}_i) \in \underset{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_i)}{argmax}\, Q_\epsilon(\tilde{x}_i, \tilde{b})$, *(o* $Q_{k_{max}}(\tilde{x}_i, \tilde{b})$*).*

2.6. Convergencia Q-Learning

Definición 2.6.1 *El esquema de aproximación estocástica asíncrona es dado por la* fórmula recursiva del método Q-learning*:*

$$Q_{k+1}(\tilde{x}_i, \tilde{a}) = Q_k(\tilde{x}_i, \tilde{a}) + \lambda_k \left[u(\tilde{a}) + \alpha \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b}) - Q_k(\tilde{x}_i, \tilde{a}) \right], \tag{20}$$

para $k = 1, 2, ..., k_{max}$*, donde* k_{max} *es el número máximo de iteraciones.*

Primero, notemos que los valores iterados de $Q_k(\tilde{x}_i, \tilde{a})$ están acotados, según el siguiente lema:

Lema 2.6.2 *En Q-Learning (para Procesos de Decisión de Markov con recompensa descontada) bajo actualización síncrona y asincrónica, la iteración $Q_k(\tilde{x}_i, \tilde{a})$ para cualquier par estado-acción $(\tilde{x}_i, \tilde{a})$ en su k-ésima actualización es acotada, siempre que $Q_0(\tilde{x}_i, \tilde{a}) < \infty$, donde $\alpha \in [0, 1)$.*

Demostración. Deseamos afirmar que, para cada par estado-acción $(\tilde{x}_i, \tilde{a})$ se satisface:

$$Q_k(\tilde{x}_i, \tilde{a}) \leq M\Big(1 + \alpha + \alpha^2 + \cdots + \alpha^k\Big) \tag{21}$$

donde α es el factor de descuento y M es un número finito positivo definido como sigue:

$$M = max\left\{u_{max}, \max_{\tilde{x}_i \in \tilde{\mathbb{X}}, \tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)} Q_1(\tilde{x}_i, \tilde{a})\right\}$$

donde

$$u_{max} = \max_{\tilde{x}_i, \tilde{x}_j \in \tilde{\mathbb{X}}, \tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)} |u(\tilde{a})|.$$

Como las recompensas inmediatas son finitas por definición, u_{max} debe ser un escalar finito. Dado que, por elección Q_0 es finito, también M debe serlo. Entonces, de (21) se sigue, cuando $k \to \infty$,

$$\lim sup_{x \to \infty} |Q_k(\tilde{x}_i, \tilde{a})| \leq M \frac{1}{1 - \alpha}$$

para todo $\tilde{x}_i \in \tilde{\mathbb{X}}$ y $\tilde{a} \in \tilde{\mathbb{A}}$. Así que se requiere probar (21). Procedemos por inducción, considerando dos casos, el primero cuando el Q-Factor es actualizado para un único par estado-acción, el segundo cuando el resto de los Q-Factores permanecen sin cambios.

Caso 1. El par estado-acción se actualiza en la k-ésima iteración:

$$Q_{k+1}(\tilde{x}_i, \tilde{a}) = (1 - \lambda) Q_k(\tilde{x}_i, \tilde{a}) + \lambda \left[u(\tilde{a}) + \alpha \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b})\right],$$

$$\begin{aligned} Q_2(\tilde{x}_i, \tilde{a}) &\leq (1 - \lambda) Q_1(\tilde{x}_i, \tilde{a}) + \lambda \Big| u(\tilde{a}) + \alpha \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_1(\tilde{x}_j, \tilde{b}) \Big| \\ &\leq (1 - \lambda) M + \lambda M + \lambda \alpha M \\ &\leq (1 - \lambda) M + \lambda M + \alpha M \\ &= M(1 + \alpha). \end{aligned}$$

Caso 2. El par estado-acción no se actualiza en la k-ésima iteración:

$$Q_{k+1}(\tilde{x}_i, \tilde{a}) = Q_k(\tilde{x}_i, \tilde{a}),$$

Procedemos por inducción, para $k = 1$,

$$\begin{aligned} Q_2(\tilde{x}_i, \tilde{a}) &= Q_1(\tilde{x}_i, \tilde{a}) \\ &\leq M \\ &\leq M(1+\alpha). \end{aligned}$$

Asumimos que la hipótesis de inducción

$$|Q_m(\tilde{x}_i, \tilde{a})| \leq M(1 + \alpha + \alpha^2 + \cdots + \alpha^m) \tag{22}$$

es cierta para $k = m$, así que finalmente se verifica que:

$$\begin{aligned} Q_{m+1} &\leq (1-\lambda)Q_m(\tilde{x}_i, \tilde{a}) + \lambda\Big|u(\tilde{a}) + \alpha \max_{\tilde{x}_j \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_m(\tilde{x}_j, \tilde{b})\Big| \\ &\leq (1-\lambda)M(1+\alpha+\alpha^2\cdots+\alpha^m) + \lambda M + \lambda\alpha M(1+\alpha+\alpha^2\cdots+\alpha^m) \\ &= M(1+\alpha+\cdots+\alpha^m) - \lambda M(1+\alpha+\cdots+\alpha^m) + \lambda M + \lambda\alpha(1+\alpha+\cdots+\alpha^m) \\ &= M(1+\alpha+\cdots+\alpha^m) - \lambda M(1+\alpha+\cdots+\alpha^m) + \lambda M + \lambda M(\alpha+\cdots+\alpha^{m+1}) \\ &= M(1+\alpha+\cdots+\alpha^m) - \lambda M(1+\alpha+\cdots+\alpha^m) + \lambda M + \lambda M(\alpha+\cdots+\alpha^m) + \lambda M\alpha^{m+1} \\ &= M(1+\alpha+\cdots+\alpha^m) - \lambda M(1+\alpha+\cdots+\alpha^m) + \lambda M(1+\alpha+\cdots+\alpha^m) + \lambda M\alpha^{m+1} \\ &= M(1+\alpha+\alpha^2\cdots+\alpha^m) + \lambda M\alpha^{m+1} \\ &\leq M(1+\alpha+\alpha^2\cdots+\alpha^m) + M\alpha^{m+1} \\ &= M(1+\alpha+\alpha^2\cdots+\alpha^m+\alpha^{m+1}). \end{aligned}$$

Ahora bien, si la actualización se realiza como en el Caso 2:

$$\begin{aligned} \Big|Q_{m+1}(\tilde{x}_i, \tilde{a})\Big| &= \Big|Q_m(\tilde{x}_i, \tilde{a})\Big| \\ &\leq M(1+\alpha+\alpha^2+\cdots+\alpha^m) \\ &\leq M(1+\alpha+\alpha^2+\cdots+\alpha^m+\alpha^{m+1}). \end{aligned}$$

Así que (21) es probado para $k = m+1$. ■

Proposición 2.6.3 *Cuando los intervalos de paso y la acción usada en el algoritmo satisfacen las condiciones a), b) y c) del Teorema 11.21 en [10], con probabilidad 1, la sucesión de políticas generadas por el algoritmo Q-learning, $\{\tilde{\pi}_k\}_{k=1}^{\infty}$, converge a π^*.*

Demostración. Definimos las transformaciones $F(\cdot)$ y $F'(\cdot)$:

$$\begin{aligned} F(Q_k)(i,a) &= \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)\Big[u(a) + \alpha \max_{b\in\mathbb{A}(j)} Q_k(j,b)\Big] - Q_k(i,a) \\ F'(Q_k)(i,a) &= \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)\Big[u(a) + \alpha \max_{b\in\mathbb{A}(j)} Q_k(j,b)\Big] \end{aligned}$$

implicando que

$$F(Q_k)(i,a) = F'(Q_k)(i,a) - Q_k(i,a).$$

Se define la transformación $f'(\cdot)$ como:

$$f'(Q_k)(i,a) := \Big[u(a) + \alpha \max_{b\in\mathbb{A}(j)} Q_k(j,b)\Big]$$

y el ruido como:

$$w_k(i,a) = f'(Q_k)(i,a) - F'(Q_k)(i,a).$$

Sustituyendo en la ecuación de Q-Learning:

$$Q_{k+1}(i,a) = Q_k(i,a) + \lambda_k\Big[f'(Q_k)(i,a) - Q_k(i,a)\Big]$$

así, podemos reescribirla como:

$$Q_{k+1}(i,a) = Q_k(i,a) + \lambda_k\Big[F(Q_k)(i,a) + w_k(i,a)\Big]$$

que tiene la misma forma del esquema presentado en el Teorema 11.21 de [10]. Requerimos ahora verificar las condiciones de él para determinar si el algoritmo de Q-Learning converge.

a) La continuidad Lipschitz de $F(\cdot)$ se sigue ya que las derivadas parciales de $F(Q_k)$ son acotadas.

b) El ruido es una martingala, cuya esperanza condicional es 0. Y su segundo momento condicional puede ser acotado por una función del cuadrado de las iteraciones.

c) Estas condiciones son probadas para $\lambda_k(\tilde{x}_i, \tilde{a}) = \frac{Log(k)}{k}$ en la Proposición 3.4.3.

d) Consideremos dos vectores $Q_1^k, Q_2^k \in \mathbb{R}^n$, de la definición

$$F'(Q_k)(i,a) = \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)\Big[u(a) + \alpha \max_{b\in\mathbb{A}(j)} Q_k(j,b)\Big]$$

se tiene que

$$\begin{aligned} F'(Q_1^k)(i,a) - F'(Q_2^k)(i,a) &= \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)u(a) + \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) \max_{b\in\mathbb{A}(j)} Q_1^k(j,b) \\ &\quad - \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)u(a) - \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) \max_{b\in\mathbb{A}(j)} Q_2^k(j,b) \\ &= \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)\Big[\max_{b\in\mathbb{A}(j)} Q_1^k(j,b) - \max_{b\in\mathbb{A}(j)} Q_2^k(j,b)\Big]. \end{aligned}$$

De esto podemos escribir, para cualquier par (i, a)

$$\left|F'(Q_1^k)(i,a) - F'(Q_2^k)(i,a)\right| \leq \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) \left| \max_{b \in \mathbb{A}(j)} Q_1^k(j,b) - \max_{b \in \mathbb{A}(j)} Q_2^k(j,b) \right|$$

$$\leq \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) \max_{b \in \mathbb{A}(j)} \left| Q_1^k(j,b) - Q_2^k(j,b) \right|$$

$$\leq \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) \max_{j \in \mathbb{X}, b \in \mathbb{A}(j)} \left| Q_1^k(j,b) - Q_2^k(j,b) \right|$$

$$= \alpha \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j) ||Q_1^k - Q_2^k||_\infty$$

$$= \alpha \, ||Q_1^k - Q_2^k||_\infty \sum_{j=1}^{|\mathbb{X}|} \mathcal{Q}(i,a,j)$$

$$= \alpha \, ||Q_1^k - Q_2^k||_\infty.$$

Por lo tanto, para cualquier (i, a):

$$\left|F'(Q_1^k) - F'(Q_2^k)(i,a)\right| \leq \alpha \, ||Q_1^k - Q_2^k||_\infty.$$

Dado que eso se satisface para cualesquier (i, a), se satisface para los valores que maximizan el lado izquierdo de la desigualdad anterior. Por lo tanto

$$\left|\left|F'Q_1^k - F'Q_2^k\right|\right|_\infty \leq \alpha \left|\left|Q_1^k - Q_2^k\right|\right|_\infty.$$

Dado que $0 < \alpha < 1$, $F'(\cdot)$ es contractivo con respecto a la norma del máximo. Por Teorema 11.21 en [10], $\frac{dq}{dt} = F(q)$ debe tener un único punto de equilibrio estable, globalmente asintótico.

e) Ver condición de acotamiento en [10].

■

2.7. Aplicación Q-Learning

Se presenta un ejemplo de aplicación del *Algoritmo de Iteración de Valor para Q-Factores*, para tres tasas de refuerzo (aprendizaje) distintas.
Se considera la optimización del criterio de recompensa descontada con factor de descuento $\alpha = 0.8$ y con horizonte infinito.
Consideramos la cadena de Markov con espacio de estados $\mathbb{X} = \{1, 2, 3, 4, 5\}$ y espacio de acciones $\mathbb{A} = \{1, 2, 3, 4, 5\}$. Todas las acciones son admisibles en cada estado, así que $\mathbb{A}(x) = \mathbb{A},\ x \in \mathbb{X}$.
Además, consideremos las matrices de transición respectivas asociadas a cada una de las acciones $\mathcal{Q}[i, a, j]$, que indica la probabilidad de que, estando en el estado i, y eligiéndose la acción a se traslade al nuevo estado j:

$$\mathcal{Q}[i,1,j] = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.25 & 0.25 & 0.25 & 0.25 \\ 0.5 & 0.0 & 0.0 & 0.0 & 0.5 \end{bmatrix}$$

$$\mathcal{Q}[i,2,j] = \begin{bmatrix} 0.5 & 0.5 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix}$$

$$\mathcal{Q}[i,3,j] = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.0 & 0.25 \\ 0.0 & 0.5 & 0.5 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$\mathcal{Q}[i,4,j] = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.25 & 0.25 & 0.25 & 0.25 & 0.0 \\ 0.0 & 0.0 & 0.5 & 0.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$\mathcal{Q}[i,5,j] = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.25 & 0.25 & 0.25 & 0.25 \\ 0.0 & 0.0 & 0.0 & 0.5 & 0.5 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

Al estar en el estado i, elegirse la acción a y trasladarse al nuevo estado j, se obtiene una utilidad inmediata ganada. Escribimos estas utilidades en forma matricial, $u[a,i] := u(i-a)$ asociada a cada acción a.

$$u[1,i] = \begin{bmatrix} 1 & 8 & 0 & 1 & 9 \\ 0 & 2 & 0 & 1 & 0 \\ 4 & 0 & 3 & 0 & 9 \\ 5 & 6 & 0 & 4 & 6 \\ 2 & 8 & 0 & 1 & 5 \end{bmatrix}$$

$$u[2,i] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 4 & 0 \\ 5 & 5 & 5 & 5 & 5 \end{bmatrix}$$

$$u[3,i] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 \\ 1 & 2 & 3 & 0 & 0 \\ 1 & 2 & 3 & 4 & 0 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

$$u[4,i] = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$u[5,i] = \begin{bmatrix} 0 & 1 & 2 & 0 & 10 \\ 0 & 5 & 7 & 0 & 5 \\ 4 & 0 & 2 & 0 & 6 \\ 0 & 3 & 9 & 0 & 4 \\ 0 & 0 & 1 & 0 & 3 \end{bmatrix}$$

La política óptima tiene la forma $\boldsymbol{\pi}^* = \{a_1^*, a_2^*, a_3^*, a_4^*, a_5^*\}$. Cada entrada a_i^* es la acción que debe ser tomada al elegirse el estado inicial $x_i \in \mathbb{X}$.

Resolvemos el problema de maximizar la recompensa total descontada aplicando el método de *Q-Learning* visto en la sección anterior a través de un programa en MATHEMATICA 10.0, y considerando tres casos distintos por separado según la tasa

de refuerzo:
A)

$$\lambda(i,a) = \frac{1}{M(i,a)}$$

con $M(i,a)$ la matriz de visitas entre el estado i y acción a, se obtienen después de $k_{max} = 2500$ iteraciones las siguientes matrices:

$$M(i,a) = \begin{bmatrix} 67 & 65 & 66 & 63 & 77 \\ 97 & 81 & 90 & 82 & 69 \\ 90 & 95 & 87 & 96 & 99 \\ 97 & 96 & 92 & 93 & 111 \\ 158 & 162 & 169 & 154 & 144 \end{bmatrix}$$

$$Q(i,a) = \begin{bmatrix} 9.6376 & 11.3077 & 12.4271 & 9.6775 & 8.8557 \\ 15.1670 & 15.4122 & 13.5747 & 14.7368 & 18.5710 \\ 15.1256 & 15.2315 & 15.0210 & 15.0134 & 16.3046 \\ 21.0819 & 19.7826 & 19.9463 & 16.9824 & 16.8820 \\ 15.7364 & 19.1164 & 19.7313 & 15.5809 & 17.8582 \end{bmatrix}$$

$Q(i,a)$ es la matriz de Q-Factores. De la matriz Q se recogen los valores máximos por fila, una fila por cada estado inicial:

$$\begin{aligned} Q(1,3) &= 12.4271 \\ Q(2,5) &= 18.5710 \\ Q(3,5) &= 16.3046 \\ Q(4,1) &= 21.0819 \\ Q(5,3) &= 19.7313 \end{aligned}$$

y la respectiva política óptima resulta:

$$\boldsymbol{\pi}^* = \{\mathbf{3}, \mathbf{5}, \mathbf{5}, \mathbf{1}, \mathbf{3}\}.$$

B)

$$\lambda(k) = \frac{150}{300 + k}$$

donde se obtienen después de $k_{max} = 2500$ iteraciones las siguientes matrices:

$$M(i,a) = \begin{bmatrix} 49 & 62 & 42 & 42 & 42 \\ 42 & 49 & 58 & 42 & 44 \\ 75 & 75 & 75 & 88 & 79 \\ 91 & 113 & 101 & 103 & 94 \\ 223 & 204 & 253 & 227 & 227 \end{bmatrix}$$

$$Q(i,a) = \begin{bmatrix} 14.4907 & 15.2424 & 18.1705 & 14.3187 & 13.4125 \\ 17.3779 & 17.5472 & 17.3906 & 19.5947 & 20.3770 \\ 20.3148 & 20.0197 & 20.1102 & 20.1903 & 20.9256 \\ 25.5579 & 24.4020 & 24.4096 & 21.4232 & 22.7558 \\ 21.8252 & 24.0549 & 24.9836 & 20.9837 & 22.9826 \end{bmatrix}$$

de donde se recogen los máximos por fila:

$$\begin{aligned} Q(1,3) &= 18.1705 \\ Q(2,5) &= 20.3770 \\ Q(3,5) &= 20.9256 \\ Q(4,1) &= 25.5579 \\ Q(5,3) &= 24.9836 \end{aligned}$$

donde, de la misma manera que en el caso anterior, se tiene política óptima idéntica:

$$\boldsymbol{\pi}^* = \{\mathbf{3}, \mathbf{5}, \mathbf{5}, \mathbf{1}, \mathbf{3}\}.$$

C)

$$\lambda(k) = \frac{Log(k)}{k}$$

donde se obtienen después de $k_{max} = 2500$ iteraciones las siguientes matrices:

$$M(i,a) = \begin{bmatrix} 45 & 61 & 46 & 68 & 56 \\ 57 & 54 & 57 & 61 & 51 \\ 84 & 75 & 83 & 72 & 95 \\ 112 & 95 & 98 & 79 & 82 \\ 223 & 215 & 217 & 207 & 207 \end{bmatrix}$$

$$Q(i,a) = \begin{bmatrix} 0.5659 & 1.2282 & 1.2586 & 0.7887 & 0.2395 \\ 2.7705 & 2.7099 & 1.1791 & 2.8851 & 4.7350 \\ 3.0223 & 3.2916 & 3.6127 & 2.8486 & 4.2566 \\ 7.6795 & 5.5252 & 5.7751 & 2.7830 & 4.0591 \\ 7.2616 & 8.3700 & 9.5490 & 5.7945 & 7.6191 \end{bmatrix}$$

de donde se recogen los máximos por fila:

$$\begin{aligned} Q(1,3) &= 1.2586 \\ Q(2,5) &= 4.7350 \\ Q(3,5) &= 4.2566 \\ Q(4,1) &= 7.6795 \\ Q(5,3) &= 9.5490 \end{aligned}$$

donde, de la misma manera que en los casos anteriores, se tiene política óptima idéntica:

$$\boldsymbol{\pi}^* = \{\mathbf{3}, \mathbf{5}, \mathbf{5}, \mathbf{1}, \mathbf{3}\}.$$

Esto indica que, la acción óptima π a ser tomada para optimizar $V(\pi, 1)$ debe ser $a^* = 3$, para $V(\pi, 2)$ debe ser $a^* = 5$, así sucesivamente.
En [5] pueden encontrarse problemas similares a éste, que podrían resolverse por Q-learning.

En el Capítulo 3 será expuesta una extensión y modificación del algoritmo Q-Learning presentada en este capítulo. El replanteo del algoritmo será debido a los espacios de estados y acciones que ahora serán continuos y realizar discretización sobre tales espacios teniendo un número de elementos muy grande.

Capítulo 3

Aproximaciones numéricas a un problema de consumo-inversión

En este capítulo se aborda el problema de consumo-inversión en contexto general. Se construyen e implementan tres distintos algoritmos, cada uno con su respectivo método de discretización de los espacios de interés, a saber, el espacio de estados $\mathbb{X}$ y acciones $\mathbb{A}$.

Sección 3.2 Dedicado a un problema presentado en [12] donde indica solución exacta al problema. Se propone una discretización "natural" considerando intervalos de paso común al particionar $\mathbb{X}$ y $\mathbb{A}$.

Sección 3.3 Se aborda el mismo problema que en la Sección 3.2, pero se utiliza un método de discretización que sugiere [6]. Se utiliza Programación Dinámica y se comparan resultados con la solución exacta y el método en la Sección 3.2.

Sección 3.4 Se propone un algoritmo Q-Learning con su respectivo método de discretización para un problema de consumo-inversión optimizando un criterio de rendimiento para una utilidad de consumo α-descontada.

3.1. Plan de consumo

Presentamos una versión controlada del problema presentado en [8]. Primero, supongamos que la riqueza de un inversor es gobernada por una ley definida por la ecuación en diferencias $x_{t+1} = (h(x_t) - a_t)\,\xi_t$, donde x_t es la riqueza actual al tiempo t, para $t = 0, 1, 2, ...$, $\delta < 1$, y $\{\xi_t\}$ es una sucesión de variables aleatorias independientes e idénticamente distribuidas con función de densidad Δ. Suponga que el inversor quiere administrar de manera óptima su capital actual x_t, dedicando parte de este capital a su consumo a_t y el resto, $h(x_t) - a_t$, a inversión. En particular, considere la función de

producción $h(x)$ con $x \in \mathbb{X} := [0, \infty)$ llamado el *espacio de estados*. La ley de transición es entonces dada por

$$x_{t+1} \;=\; (h(x_t) - a_t)\,\xi_t, \tag{23}$$

y $X_0 = x$ conocida. Los préstamos no son permitidos, por lo tanto $a_t \in [0, h(x_t)]$ y denota el consumo al tiempo t. $\mathbb{A}(x_t) := [0, h(x_t)]$ es el *espacio de acciones admisibles* al tiempo t, y $\mathbb{A} := [0, +\infty)$ es el *conjunto de acciones admisibles*.

La dinámica descrita en este sistema de consumo-inversión es como sigue: si el sistema es observado en el tiempo t, el estado considerado es $x_t = x \in \mathbb{X} = [0, +\infty)$ y la acción $a_t = a \in \mathbb{A}(x)$ es aplicada. Una utilidad $u(a)$ es obtenida y el sistema se mueve al siguiente estado, $x_{t+1} \in \mathbb{X}$, por medio de la ley de transición (23). Este proceso se repite mientras se acumulan las recompensas descontadas en cada tiempo t hasta un horizonte infinito de acuerdo a un criterio de rendimiento. Denotamos $\mathbb{K} := \{(x, a) \,|\, x \in \mathbb{X}, a \in \mathbb{A}(x)\}$ como el conjunto de parejas de estado-acción admisibles.

Dado un capital inicial $X_0 = x \in \mathbb{X}$, un plan $\pi \in \Pi$, y una utilidad de consumo $u : \mathbb{R} \to \mathbb{R}$

El criterio de rendimiento usado para evaluar la calidad del plan $\pi \in \Pi$ es dado por

$$V(\pi, x) := \mathbb{E}_x^{\pi}\left[\sum_{t=0}^{N} \alpha^t\, u(x_t - a_t)\right], \qquad \forall x \in \mathbb{X}, \tag{24}$$

donde $\mathbb{E}_x^{\pi}[\cdot]$ es el operador esperanza cuando $x \in \mathbb{X}$ es la condición inicial y la política $\pi \in \Pi$ es aplicada. El horizonte de planificación N puede ser finito o infinito. Las utilidades de consumo futuras están descontadas de acuerdo a un factor de descuento $\alpha \in (0, 1)$.

El objetivo del inversor es maximizar la utilidad de consumo descontada para todos los planes $\pi \in \Pi$, esto es,

$$V(\pi^*, x) = \sup_{\pi \in \Pi} V(\pi, x) := V^*(x), \qquad \forall x \in \mathbb{X}. \tag{25}$$

En este caso, π^* es la política óptima, y V^* es la función de valor óptima. Entonces, el problema de control consiste en determinar la política óptima $\pi^* \in \Pi$.

3.2. Algoritmo aproximador I: Discretización común para espacios de estados y acciones

El siguiente ejemplo es útil para iniciar con procesos de discretización para un problema de consumo-inversión donde se emplea programación dinámica para encontrar la solución exacta al problema, y con la discretización encontrar una forma alternativa aproximando la solución. Un problema de control óptimo clásico de consumo-inversión trata de un inversor que desea asignar o distribuir su riqueza actual x_t en dos partes:

- una para inversión a_t, y
- otra para un consumo $x_t - a_t$ en cada período $t = 0, 1, \ldots, N-1$.

Suponemos que el préstamo no está permitido, así que la restricción de inversión, o el conjunto de acciones admisibles es $\mathbb{A}(x) = [0, x]$. Consideramos $\mathbb{X} = [0, \infty)$.

La relación entre la inversión y el capital acumulado está dado por

$$x_{t+1} = a_t \xi_t, \qquad t = 0, 1, ...$$

donde las variables aleatorias $\{\xi_t\}$ son independientes e idénticamente distribuidas, independientes de x_0, y además con $m = \mathbb{E}\{\xi_t\} > 1$, esto último para asegurar que la reinversión pueda ser rentable.

Deseamos primero, maximizar la utilidad de consumo descontada total esperada con horizonte finito:

$$V(\pi, x) := \mathbb{E}_x^{\pi} \left\{ \sum_{t=0}^{N-1} \alpha^t u(x_t - a_t) \right\}, \quad x \in \mathbb{X},\ 0 < \alpha < 1.$$

Requerimos de una función de recompensa $r(x, a)$, empleamos una utilidad del consumo $c = x - a$, la llamada utilidad marginal isoelástica

$$r(x, a) = u(x - a) := \frac{b}{\gamma}(x - a)^{\gamma}, \qquad b > 0, \quad 0 < \gamma < 1.$$

Las Ecuaciones de Programación Dinámica para todo $x \in \mathbb{X}$ y $t = N-1, ..., 0$ son:

$$\begin{aligned} V_N(x) &= 0 \\ V_t(x) &= \max_{a \in \mathbb{A}(x)} \{u(x - a) + \alpha\, \mathbb{E}\{V_{t-1}(a\xi_t)\}\}. \end{aligned}$$

Solución Exacta. Cabe mencionar que en [11] se deduce la solución exacta al problema de control óptimo considerado en los dos algoritmos de las Secciones 3.2 y 3.3.

La solución depende de la forma de la función de utilidad u. Consideramos dos casos para la función $u(x) = (b/\gamma)x^{\gamma}$, donde $b > 0$ y $0 < \gamma < 1$.

Caso 1: $u(x - a) = b(x - a), b > 0$ constante. Suponemos que $\alpha m > 1$. De las Ecuaciones de Programación Dinámica,

$$V_{N-1}(x) = \max_{a \in \mathbb{A}(x)} \{b(x - a)\} = bx,$$

donde $a = f_{N-1}(x) = 0$. Similarmente,

$$\begin{aligned} V_{N-2}(x) &= \max_{a\in\mathbb{A}(x)} \left\{ b(x-a) + \alpha\, \mathbb{E}\{V_{N-1}(a\xi_{N-1})\} \right\} \\ &= \max_{a\in\mathbb{A}(x)} \{ b(x-a) + \alpha\, mba \} \\ &= \alpha m b x, \end{aligned}$$

donde $a = f_{N-2}(x) = x$. En general, por inducción se obtiene

$$V_t(x) = (\alpha m)^{N-t-1} bx \quad \forall x \in \mathbb{X},\, t = 0, ..., N-1, \tag{26}$$

con $f_t(x) = x\ \forall t = 0, ..., N-2$, y $f_{N-1}(x) = 0$. Por lo tanto, del Teorema 1.3.1, la función objetivo es

$$J^*(x) = J_0(x) = (\alpha m)^N bx \tag{27}$$

para cualquier riqueza inicial $x_0 = x$, y la política de inversión óptima es

$$\pi^* = \{f_0, f_1, ...f_{N-1}\}. \tag{28}$$

Caso 2: $u(x-a) = \frac{b}{\gamma}(x-a)^\gamma, b > 0, 0 < \gamma < 1$. De nuevo, de las Ecuaciones de Programación Dinámica,

$$V_{N-1}(x) = \max_{a\in\mathbb{A}(x)} \left\{ \frac{b}{\gamma}(x-a)^\gamma \right\} = \frac{b}{\gamma} x^\gamma, \ \ \forall x \in \mathbb{X},$$

con la decisión de inversión óptima $f_{N-1}(x) = 0$.

Definiendo

$$\delta := (\alpha \mathbb{E}\{\xi_0^\gamma\})^{\frac{1}{\gamma-1}},$$

y de manera recursiva a la variable

$$D_{N-1} := 1, \qquad D_t := \frac{\delta^{\gamma-1} D_{t+1}}{\left(1 + \delta (D_{t+1})^{\frac{1}{\gamma-1}}\right)^{\gamma-1}}, \ \ t = N-2, ..., 0 \tag{29}$$

se puede demostrar que, para todo $x \in \mathbb{X}$,

$$\begin{aligned} V_t(x) &= \frac{b}{\gamma} D_t x^\gamma, \qquad t = N-1, N-2, ..., 0 \quad y \\ f_t(x) &= \frac{x}{1 + \delta (D_{t+1})^{\frac{1}{\gamma-1}}}, \qquad t = N-2, ..., 0. \end{aligned}$$

La iteración de (29) produce

$$D_t = \left[\frac{\delta^{N-t-1}}{\left(1 + \delta + \cdots + \delta^{N-t-1}\right)} \right]^{\gamma-1}, \ \ \forall t = 0, ..., N-1. \tag{30}$$

En particular, si $\delta \neq 1$,

$$D_0 = \left(\frac{\delta^{N-1}(1-\delta)}{1-\delta^N} \right)^{\gamma-1}. \tag{31}$$

Por lo que, la utilidad de consumo total descontada esperada óptima en N períodos queda explícitamente como

$$V^*(x) = V_0(x) = \left(\frac{b}{\gamma}\right) D_0 x^\gamma = \left(\frac{b}{\gamma}\right) \left(\frac{\delta^{N-1}(1-\delta)}{1-\delta^N} \right)^{\gamma-1} x^\gamma. \tag{32}$$

Por ejemplo, considere $\xi_t \sim LogNor(0.5, 0.01)$, y los siguientes parámetros

$$b = 1,\ \gamma = 0.1, \qquad \alpha = 0.5, \quad x_0 = 10, \quad N = 15,$$

podemos obtener la solución exacta con $\alpha m = 0.828495 < 1$,

$$\begin{aligned} a^* &= \mathbf{4.894} \\ V^*(a^*, x_0) &= V_0(10) = \frac{b}{\gamma} D_0 (10)^\gamma = \mathbf{23.0563}. \end{aligned}$$

Este resultado será contrastado con la solución aproximada obtenida mediante los métodos siguientes propuestos.

Algoritmo aproximador I. En los problemas de control óptimo donde no sea posible encontrar solución cerrada, será necesario discretizar los espacios de estados y acciones. En esta sección se muestra un algoritmo que consta de una discretización natural, [13], [14], para los espacios de estados y acciones finitos formando nodos $\tilde{x}_i \in \mathbb{X}$ y $\tilde{a}_i \in \mathbb{A}(x)$ con intervalos de paso común entre ellos.

Creamos una partición común para $\mathbb{X}$ y $\mathbb{A}(x)$, sus elementos $\tilde{x}$ y $\tilde{a}$ conformarán $\widetilde{\mathbb{X}}$ y $\widetilde{\mathbb{A}}(\tilde{x})$, respectivamente. De tal manera que las nuevas ecuaciones de Programación Dinámica a resolver son:

$$\begin{aligned} \widehat{V}_0(\tilde{x}_i) &= 0, \\ \widehat{V}_t(\tilde{x}_i) &= \max_{\tilde{a}_i \in \widetilde{\mathbb{A}}(\tilde{x}_i)} \left\{ u(\tilde{x}_i - \tilde{a}_i) + \alpha\, \mathbb{E}\{V_{t-1}(\tilde{a}_i \xi_t)\} \right\}. \end{aligned}$$

donde $\mathbb{E}\{V_{t-1}(\tilde{a}_i \xi_t)\} \approx \frac{1}{M} \sum_{j=1}^{M} V_{t-1}(\tilde{a}_i \xi_j)$.

Se procede a la generación de los posibles valores de $V_t(\tilde{x}_i)$ para $t = 0, 1, ..., N$, esto se hace para cada acción $\tilde{a}_j$.

Una manera práctica de realizarla es mediante matrices, con las filas representando los estados discretizados, y las columnas acciones discretizadas.

Las matrices en cada etapa son de la forma:
$V[0, \tilde{x}_i, \tilde{a}_j] = [0]$

$$V[1, \tilde{x}_i, \tilde{a}_j] = \begin{bmatrix} u(\tilde{x}_0, \tilde{a}_0) & 0 & 0 & 0 & \dots & 0 \\ u(\tilde{x}_1, \tilde{a}_0) & u(\tilde{x}_1, \tilde{a}_1) & 0 & 0 & \dots & 0 \\ u(\tilde{x}_2, \tilde{a}_0) & u(\tilde{x}_2, \tilde{a}_1) & u(\tilde{x}_2, \tilde{a}_2) & 0 & \dots & 0 \\ u(\tilde{x}_3, \tilde{a}_0) & u(\tilde{x}_3, \tilde{a}_1) & u(\tilde{x}_3, \tilde{a}_2) & u(\tilde{x}_3, \tilde{a}_3) & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ u(\tilde{x}_n, \tilde{a}_0) & u(\tilde{x}_n, \tilde{a}_1) & u(\tilde{x}_n, \tilde{a}_2) & u(\tilde{x}_n, \tilde{a}_3) & \dots & u(\tilde{x}_n, \tilde{a}_n) \end{bmatrix}$$

Observación 3.2.1 *La matriz es triangular inferior, puesto que se tiene la restricción $a_t \leq x_t$. No se puede invertir más de lo que se tiene debido a que no hay préstamos. El espacio de estados-acciones con esta restricción es el* espacio de estados-acciones admisibles.

Para las matrices posteriores, definiremos

$$V_{st} := \alpha \frac{1}{M} \sum_{i=1}^{M} V[s, \tilde{a}_t \xi_i \to \tilde{x}_t, \tilde{a}_t], \quad 0 < \alpha < 1.$$

La notación $\tilde{a}_t \xi_i \to \tilde{x}_t$ se deriva porque la generación simulada del valor estado $\tilde{a}_t \xi_t$ no necesariamente se encontrará en la partición $\widetilde{\mathbb{X}}$, le será asignado entonces el valor $\tilde{x}_t$ más cercano que se encuentre en el espacio de estados discretizado.

Por ejemplo, en la etapa 2, la matriz resultaría ser:

$$V[2, \tilde{x}_i, \tilde{a}_j] = \begin{bmatrix} u(\tilde{x}_0, \tilde{a}_0) + V_{10} & 0 & 0 & \dots & 0 \\ u(\tilde{x}_1, \tilde{a}_0) + V_{10} & u(\tilde{x}_1, \tilde{a}_1) + V_{11} & 0 & \dots & 0 \\ u(\tilde{x}_2, \tilde{a}_0) + V_{10} & u(\tilde{x}_2, \tilde{a}_1) + V_{11} & u(\tilde{x}_2, \tilde{a}_2) + V_{12} & \dots & 0 \\ u(\tilde{x}_3, \tilde{a}_0) + V_{10} & u(\tilde{x}_3, \tilde{a}_1) + V_{11} & u(\tilde{x}_3, \tilde{a}_2) + V_{12} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u(\tilde{x}_n, \tilde{a}_0) + V_{10} & u(\tilde{x}_n, \tilde{a}_1) + V_{11} & u(\tilde{x}_n, \tilde{a}_2) + V_{12} & \dots & u(\tilde{x}_n, \tilde{a}_n) + V_{1n} \end{bmatrix}$$

Se procede de manera similar con el resto de las matrices. Una vez hecho el llenado de matrices, se buscan los máximos por filas (estado), en la columna donde se encuentre, significará el argumento máximo (control elegido) del espacio de acciones discretizadas.

Cuando

$$b = 1,\ \gamma = 0.1, \qquad \alpha = 0.5, \quad x_0 = 10, \qquad \mu = 0.5,\ \sigma = 0.2,$$

El diseño del algoritmo hasta la etapa 10 con $x_0 = 10$ arroja los siguientes resultados:

$\widehat{V}_t[x_0] \ = V[t, x_0, \tilde{a}]$
$\widehat{V}_1[10] = V[1, 10, 0.00] = 12.5893$
$\widehat{V}_2[10] = V[2, 10, 3.33] = 18.0314$
$\widehat{V}_3[10] = V[3, 10, 4.35] = 20.6269$
$\widehat{V}_4[10] = V[4, 10, 4.54] = 21.8882$
$\widehat{V}_5[10] = V[5, 10, 4.68] = 22.5079$
$\widehat{V}_6[10] = V[6, 10, 4.93] = 22.8162$
$\widehat{V}_7[10] = V[7, 10, 5.18] = 22.9562$
$\widehat{V}_8[10] = V[8, 10, 4.91] = 23.0220$
$\widehat{V}_9[10] = V[9, 10, 5.18] = 23.0654$
$\widehat{V}_{10}[10] = V[10, 10, \mathbf{4.80}] = \mathbf{23.0914}$.

3.3. Algoritmo aproximador II: Discretización con kernel normalizado

En esta sección se presenta un replanteamiento de las funciones de utilidad y transición basada en [6], [7] y [27], debida a la discretización de espacios. En la versión finita del Teorema de la Programación Dinámica, será necesario crear una partición de los espacios continuos de estados y de acciones, para la aplicación de algoritmos iterativos que se desarrollen más adelante, y trabajar con los nodos de tales particiones. A continuación, se cita un método clásico de discretización. En la Sección 3.4 se propone un nuevo método alternativo de discretización, [16]. Debemos particionar un intervalo $I \subset \mathbb{X}$ en un número finito η de subintervalos. Definimos $\rho := x/\eta$ y llamamos a ρ el tamaño de malla. Cada subintervalo es un subconjunto $\mathcal{I}_i$ que consiste de intervalos no vacíos de la forma

$$\mathcal{I}_i = (i\rho, (i+1)\rho] \cap I, \quad i = 0, 1, 2, ..., \eta.$$

Elegimos un elemento representativo de cada $\mathcal{I}_i$ y sea $\tilde{I}$ el conjunto de todos los elementos representativos. Para cualquier $x \in \mathbb{X}$, designamos a σ_x como el elemento $\mathcal{I}_i, \ \ (i = 0, 1, 2, ..., \eta)$ para el cual x pertenece a él. También usamos $\tilde{\sigma}_x$ para denotar el elemento representativo del conjunto σ_x. Por conveniencia, tomamos el extremo derecho de cada subintervalo $\mathcal{I}_i$ como sus elementos representativos.

Entonces, el espacio de estados discretizado es dado por

$$\tilde{\mathbb{X}} := \{\tilde{x}_i \in \mathbb{X} : \tilde{x}_i = i\rho \text{ para } i \in \{1, 2, ..., \eta\}\}. \tag{33}$$

En [6], [7] y [27] consideran una reformulación de la función de costo y el kernel debida a la discretización.

$$\tilde{c}(x,\tilde{a}) := c(\check{\sigma}_x,\tilde{a})$$

$$\tilde{f}(y|x,\tilde{a}) := \frac{f(\check{\sigma}_y|\check{\sigma}_x,\tilde{a})}{\int_{\mathbb{X}} f(\check{\sigma}_z|\check{\sigma}_x,\tilde{a})dz}$$

donde, al hacer la partición, $\check{\sigma}_x$ denota el elemento representativo del subintervalo de $\mathbb{X}$ donde pertenece x.

En el contexto del ejemplo inicial de la Sección 3.2, se tiene como ley de transición a

$$x_{t+1} = a_t\xi_t.$$

Como hemos considerado $\{\xi_t\}$ iid con $\mathbb{E}\{\xi_t\} > 1$ eligiendo que $\xi_t \sim$LogNor(μ,σ^2), denotamos

$$f(y;\mu,\sigma^2) = \frac{1}{y\ \sigma\sqrt{2\pi}}\ Exp\left\{-\frac{(Log\{y\}-\mu)^2}{2\sigma^2}\right\}$$

y

$$r(x,a) = u(x-a) := \frac{b}{\gamma}(x-a)^\gamma.$$

Debido a la discretización la función de recompensa y el kernel normalizado resultan ser:

$$\tilde{u}(x,\tilde{a}) := u(\tilde{x},\tilde{a})$$

$$QN[i,j] = \frac{Q[i,j]}{S[j]} = \frac{f\left(\frac{\tilde{x}_i}{\tilde{a}_j}\right)}{\sum_{i=1}^n f\left(\frac{\tilde{x}_i}{\tilde{a}_j}\right)}.$$

El diseño del algoritmo con la modificación del kernel normalizado hasta la etapa 10 con $x_0 = 10$ arroja los siguientes resultados:

$\widehat{V}_t[x_0] \ = V[t,x_0,\tilde{a}]$
$\widehat{V}_1[10] = V[1,10,0.00] = 12.5893$
$\widehat{V}_2[10] = V[2,10,3.28] = 18.0183$
$\widehat{V}_3[10] = V[3,10,4.22] = 20.6063$
$\widehat{V}_4[10] = V[4,10,4.58] = 21.8596$
$\widehat{V}_5[10] = V[5,10,4.74] = 22.4699$
$\widehat{V}_6[10] = V[6,10,4.80] = 22.7679$
$\widehat{V}_7[10] = V[7,10,4.84] = 22.9136$
$\widehat{V}_8[10] = V[8,10,4.86] = 22.9848$
$\widehat{V}_9[10] = V[9,10,4.86] = 23.0197$
$\widehat{V}_{10}[10] = V[10,10,\mathbf{4.86}] = \mathbf{23.0367}$.

Se tiene a manera de resumen de la Sección 3.2 y 3.3, tres vías de solución a un problema de consumo-inversión usando programación dinámica, y mezclando en los dos últimos, dos formas de discretización en los espacios de estados y acciones. La Tabla 1 reune tanto las políticas óptimas como los criterios de rendimiento óptimos considerando horizonte finito que se obtienen por tales métodos.

Tabla 1. Política y rendimiento óptimos aproximados y solución exacta

Método	$\tilde{a}^*(x)$	$V^*(x)$
Solución exacta	4.894	23.0563
Primer algoritmo aproximador (Sección 3.2)	4.80	23.0914
Algoritmo kernel normalizado (Sección 3.3)	4.86	23.0367

El siguiente capítulo da entrada al método Q-Learning y lo cierra un ejemplo de aplicación. Se busca un algoritmo para un problema de control $\alpha-$descontado para una función de utilidad general.

3.4. Algoritmo aproximador III: Q-Learning α-descontado

Bajo el mismo contexto de las Sección 3.1, consideremos ahora la función de producción $h(x) := x^{\delta}$, con ecuación de transición $x_{t+1} = (x_t^{\delta} - a_t)\,\xi_t$, y función de utilidad de consumo

$$u(a) := Log(a), \qquad \forall\, a \in \mathbb{A}. \tag{34}$$

Sea $C^2(X, Y)$ el conjunto de funciones $l : X \to Y$ con segunda derivada continua para espacios euclidianos X y Y.
La función de producción y la función de utilidad de consumo satisfacen ciertas condiciones usuales, [9].

Suposición 3.4.1 *En el contexto del problema previo, las funciones de producción y utilidad satisfacen las siguiente condiciones:*
a) $h \in C^2((0,\infty),(0,\infty))$,
b) h *es una función cóncava sobre* $\mathbb{X}$,
c) $h' > 0$ *y* $h(0) = 0$,
d) $u \in C^2((0,\infty),\mathbb{R})$ *es estrictamente creciente y estrictamente cóncava, y*
e) u' *es una función invertible,* $u'(0) = \infty$, *y* $lim_{a\to\infty} u'(a) = 0$.

La función de producción considerada satisface claramente la Suposición 3.4.1 a), b) y c). Además, en [8] se verifica que la función de utilidad $u(a) := Log(a)$ satisface d) y e). Los incisos en la Suposición 3.4.1 son comunes en análisis, garantizan la existencia de máximos, y por ende, se cumplen las condiciones para asegurar la validez del teorema de la Programación Dinámica.

Observación 3.4.2 *Sea Θ^k el índice del par estado-acción visitado en la iteración k-ésima en el algoritmo, y $\mathbb{I}(\cdot)$ la función indicadora. Definimos*

$$V_k(\tilde{x}_i,\tilde{a}) := \sum_{m=1}^{k} \mathbb{I}\Big[(\tilde{x}_i,\tilde{a}) = \Theta^k\Big], \quad \textit{para } \tilde{x}_i \in \tilde{\mathbb{X}} \textit{ y } \tilde{a} \in [0,{\tilde{x}_i}^{\delta}],$$

esto es, la función indicadora devolverá para un Q-factor que no es visitado en la k-ésima iteración, que implica que los Q-factores que no sean visitados en esta iteración no serán actualizados. Existe un escalar $\chi > 0$ tal que casi seguramente obtendremos para todo $\tilde{x}_i \in C := \tilde{\mathbb{X}}\backslash\{0\}$ y $\tilde{a} \in [0,{\tilde{x}_i}^{\delta}]$,

$$\liminf_{k\to\infty} \frac{V_k(\tilde{x}_i,\tilde{a})}{k} \geq \chi,$$

ya que C es un conjunto cerrado irreducible de estados recurrentes. Además, cuando

$$K^k(z) := min\left\{ m > k \,:\, \sum_{s=k+1}^{m} \frac{Log(s)}{s} > z \right\},$$

obtenemos para cualquier $z > 0$, el límite

$$\lim_{k\to\infty} \frac{\sum_{m=V_k(l)}^{V_{K^k(z)}(l)} \frac{Log(m)}{m}}{\sum_{m=V_k(l')}^{V_{K^k(z)}(l')} \frac{Log(m)}{m}} < +\infty$$

casi seguramente.

Proposición 3.4.3 *El intervalo de paso $\lambda_k(\tilde{x}_i,\tilde{a}) = \frac{Log(k)}{k}$ y la acción seleccionada en el algoritmo Q-learning satisfacen los siguientes postulados:*

a) El intervalo de paso $\lambda_k(\tilde{x}_i,\tilde{a})$ satisface las siguientes condiciones para todo $\tilde{x}_i \in \tilde{\mathbb{X}}$ y $\tilde{a} \in [0,{\tilde{x}_i}^{\delta}]$:

$$\sum_{k=1}^{\infty} \lambda_k(\tilde{x}_i,\tilde{a}) = \infty, \qquad \sum_{k=1}^{\infty} \lambda_k^2(\tilde{x}_i,a) < \infty.$$

b) Para todo $\tilde{x}_i \in \tilde{\mathbb{X}}$ y $\tilde{a} \in [0,{\tilde{x}_i}^{\delta}]$

i) $\lambda_{k+1}(\tilde{x}_i,\tilde{a}) \leq \lambda_k(\tilde{x}_i,\tilde{a})$ a partir de algún valor k.

ii) Para cualquier $z \in (0,1)$, $sup_k\, \lambda_{[zk]}(\tilde{x}_i,\tilde{a})/\lambda_k(\tilde{x}_i,\tilde{a}) < \infty$.

iii) Para cualquier $z \in (0,1)$,

$$\lim_{k\to\infty} \frac{\sum_{m=1}^{[zk]+1} \lambda_m(\tilde{x}_i,\tilde{a})}{\sum_{m=1}^{k} \lambda_m(\tilde{x}_i,\tilde{a})} = 1.$$

Demostración. Considere $\lambda_k = \frac{Log(k)}{k}$, $k = 1, 2, ...$

a) Note que

$$\limsup_{k\to\infty} \frac{k}{k+1}\frac{Log(k+1)}{Log(k)} > \limsup_{k\to\infty} \frac{k}{k+1} = 1, \tag{35}$$

ya que la función logaritmo es creciente, y por lo tanto,

$$\sum_{k=1}^{\infty} \frac{Log(k)}{k} = +\infty.$$

Ahora, aplicando las propiedades de la función zeta ζ, (ver Apéndice), podemos demostrar que

$$\sum_{k=1}^{\infty} \left(\frac{Log(k)}{k}\right)^2 = \zeta''(2) \approx 1.98. \tag{36}$$

Por lo tanto, el resultado se deduce dado (35) y (36).

b) Además, λ_k satisface las siguientes propiedades:

i) λ_k es estrictamente decreciente, $\forall k \geq 1$.

ii) Además, para $z \in (0, 1)$, $\sup_k \lambda_{[zk]}/\lambda_k = \sup_k \frac{kLog([zk])}{[zk]Log(k)} < \infty$.

iii) Adicionalmente, si $z \in (0, 1)$, tenemos

$$\lim_{k\to\infty} \frac{\sum_{m=1}^{[zk]+1} \frac{Log(m)}{m}}{\sum_{m=1}^{k} \frac{Log(m)}{m}} = 1.$$

■

Aproximaremos la solución al problema de control con una técnica de refuerzo por aprendizaje a tiempo discreto, en particular, el método Q-learning, [30], [18] [3]. Para llegar a esto, en la siguiente sección, presentamos un procedimiento de discretización de los espacios de estado-acción.

Discretización de los espacios de estados y acciones.
Las soluciones aproximadas usando el método Q-Learning requiere de la discretización de los espacios de estados y de acciones $\mathbb{X}$ y $\mathbb{A}$, respectivamente, y la discretización requiere de un nuevo conjunto de estados y acciones. Por lo tanto, primero consideramos un espacio truncado finito común del espacio de estados original, digamos, $I := \bar{\mathbb{X}} = [0, x]$.

Debemos particionar el intervalo I en un número finito η de subintervalos. Definimos $\rho := x/\eta$ y llamamos a ρ el tamaño de malla. Cada subintervalo es un subconjunto $\mathcal{I}_i$ que consiste de intervalos no vacíos de la forma

$$\mathcal{I}_i = (i\rho, (i+1)\rho] \cap I, \quad i = 0, 1, 2, ..., \eta.$$

Elegimos un elemento representativo de cada $\mathcal{I}_i$ y sea $\tilde{I}$ el conjunto de todos los elementos representativos. Para cualquier $x \in \mathbb{X}$, designamos a σ_x como el elemento $\mathcal{I}_i$, $(i = 0, 1, 2, ..., \eta)$ para el cual x pertenece a él. También usamos $\tilde{\sigma}_x$ para denotar el elemento representativo del conjunto σ_x. Por conveniencia, tomamos el extremo derecho de cada subintervalo $\mathcal{I}_i$ como sus elementos representativos.

Entonces, el espacio de estados discretizado es dado por

$$\tilde{\mathbb{X}} := \{\tilde{x}_i \in \mathbb{X} : \tilde{x}_i = i\rho \text{ para } i \in \{1, 2, ..., \eta\}\}. \tag{37}$$

Para la discretización del espacio de acciones $\mathbb{A}$, primero simulamos un número finito k_{max} de valores aleatorios de ξ. Entonces, las acciones simuladas a_k son generadas por la ley de transición $x_{t+1} = (x_t^{\delta} - a_t)\,\xi_t$, tal que $a_k \geq 0$ y $a_k < x^{\delta}$, porque el préstamo no es permitido. Por simplicidad, estas realizaciones calculadas por la computadora son redondeadas. Redondeamos estos valores a_k y los asignamos a a_k^r, [16].

Sean $z_1 := Min\{a_k^r\}$ y $z_2 := Max\{a_k^r\}$, para $k \in \{1, 2, ..., k_{max}\}$, y consideramos r_1 y r_2 el redondeo de z_1 y z_2, respectivamente, al valor centésimo o milésimo más cercano.

Finalmente, obtenemos el espacio de acciones discretizado admisible, como sigue:

$$\tilde{\mathbb{A}}(\tilde{x}_i) = \left\{\tilde{a}_j \in \mathbb{A}(\tilde{x}_i) \mid \tilde{a}_j = r_1 + j/100\right\}, \ \ para\ j = 0, 1, ..., (r_2 - r_1)100,\ i \in \{1, 2, ..., \eta\}.$$

Sea $\hat{U} : \tilde{\mathbb{A}} \to \mathbb{R}$, la función de utilidad definida en (34) como

$$\hat{U}(\tilde{a}_i) := Log(\tilde{a}_i), \qquad \forall\, \tilde{a}_i \in \tilde{\mathbb{A}}. \tag{38}$$

La ley de transición $Q(\cdot|\cdot)$ es un kernel estocástico sobre $\mathbb{X}$, dado $\mathbb{K}$.

Definamos la función $\hat{\mathcal{Q}} : \tilde{\mathbb{X}} \times \tilde{\mathbb{A}} \times \tilde{\mathbb{X}} \to [0, 1]$ de acuerdo a [7] poniendo

$$\hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) := \frac{Q(\tilde{x}_j \,|\, \tilde{x}_i, a)}{\sum_{j=1}^{\eta} Q(\tilde{x}_j \,|\, \tilde{x}_i, a)}, \tag{39}$$

donde Q es la función de probabilidad obtenida normalizando una función de densidad lognormal Δ. Por lo tanto, $\{\xi_t\}$ es una sucesión de variables aleatorias independientes

e idénticamente distribuidas.

Entonces,

$$\hat{\mathcal{Q}}(\tilde{x}_j \,|\, \tilde{x}_i, \tilde{a}) = \frac{\delta\, {\tilde{x}_j}^{\delta-1}}{{\tilde{x}_i}^{\delta} - \tilde{a}}\ \Delta\left(\frac{{\tilde{x}_j}^{\delta}}{{\tilde{x}_i}^{\delta} - \tilde{a}}\right)\ \frac{1}{S(\tilde{a}, \tilde{x}_i)} \tag{40}$$

para $\tilde{a} < {\tilde{x}_i}^{\delta}$, donde

$$S(\tilde{a}, \tilde{x}_i) = \sum_{j=1}^{\eta} \frac{\delta\, {\tilde{x}_j}^{\delta-1}}{{\tilde{x}_i}^{\delta} - \tilde{a}}\ \Delta\left(\frac{{\tilde{x}_j}^{\delta}}{{\tilde{x}_i}^{\delta} - \tilde{a}}\right),$$

que es interpretada como la probabilidad de transición al estado $\tilde{x}_j$ cuando el estado actual es $\tilde{x}_i$ y la acción $\tilde{a}$ es elegida.

El algoritmo para aplicar Q-learning al proceso de decisión de Markov con recompensas descontadas por medio de iteración de valores se presenta a continuación como Algoritmo 2.

Algoritmo 2: Algoritmo Q-learning modificado

Inicialización Especificar k_{max} y parámetros de $\{\xi\}$;
for $k = 0$ **to** k_{max} **do**
  Simular ξ_k;
  $a_k \longleftarrow x^{\delta} - \frac{x}{\xi_k}$ tal que $a_k \geq 0$ y $a_k < x^{\delta}$;
  $a_k^r \longleftarrow$ Round$[a_k]$;
end
Poner $r_1 := Round\{Min\{a_k^r\}\}$, $r_2 := Round\{Max\{a_k^r\}\}$;
for $j = 0$ **to** $(r_2 - r_1)100$ **do**
  $\tilde{a}_j \longleftarrow r_1 + j/100 \quad (\tilde{a}_j \in \tilde{\mathbb{A}}(\tilde{x}))$;
  $Q_0(\tilde{x}, \tilde{a}_j) \longleftarrow 0$;
end
repeat
  for $x = \tilde{x}_i \in \tilde{\mathbb{X}}$ *y* $\tilde{a} \in \tilde{\mathbb{A}}(\tilde{x}_i)$ **do**
    $Q_{k+1}(\tilde{x}_i, \tilde{a}_j) \longleftarrow$
    $Q_k(\tilde{x}_i, \tilde{a}_j) + \lambda_k \left[Log(\tilde{a}_j) + \alpha \max_{\tilde{b} \in \tilde{\mathbb{A}}(\tilde{x}_j)} Q_k(\tilde{x}_j, \tilde{b}) - Q_k(\tilde{x}_i, \tilde{a}_j) \right]$;
  end
  Simular el siguiente estado $\tilde{x}_j$, hasta retornar a $x = \tilde{x}_i$;
  $k \longleftarrow k + 1$;
until $k \geq k_{max}$;
return $\tilde{\pi}^*(x) \in \underset{\tilde{b} \in \tilde{\mathbb{A}}(x)}{argmax}\ Q_{k_{max}}(x, \tilde{b})$.

3.4.1. Ejemplo: Utilidad de consumo logarítmica

Consideramos el criterio de rendimiento con horizonte infinito a ser optimizado como sigue:

$$V(\pi, x) \;:=\; \mathbb{E}_x^{\pi}\left[\sum_{t=1}^{\infty} \alpha^t \, Log(\tilde{a}_t)\right], \quad \pi \in \Pi, x \in \mathbb{X},$$

donde el capital inicial es x y $\alpha = 0.5$ es el factor de descuento.

La transición del sistema es gobernado por $x_{t+1} = (x_t^{\delta} - a_t)\,\xi_t$, donde $\delta = 0.75$ y $\{\xi_t\}$ es tal que $Log(\xi_t) \sim N(0, 0.4)$, que surge de la función de probabilidad de masa normalizada sobre el espacio $\tilde{\mathbb{X}} \times \tilde{\mathbb{A}} \times \tilde{\mathbb{X}}$.

Para el método Q-Learning, pongamos el intervalo de paso $\lambda_k(\tilde{x}_i, \tilde{a}) := \frac{Log(k)}{k}$ como tasa de refuerzo o aprendizaje y $k_{max} = 1000$. La Tabla 2 y la Figura 2 concentran algunos resultados, comparados con la solución exacta.

En [8] encuentran que la política óptima es dada por

$$a^*(x) = x^{\delta}(1 - \alpha\delta), \quad x \in \mathbb{X}.$$

Tabla 2. Política óptima aproximada y exacta.

x	$\tilde{a}^*(x)$	$a^*(x)$
0.25	0.26	0.220
0.50	0.40	0.371
0.75	0.52	0.503
1.00	0.64	0.625
1.25	0.70	0.738
1.50	0.84	0.847
1.75	1.07	0.950
2.00	1.12	1.051
2.25	1.12	1.148
2.50	1.24	1.242
2.75	1.33	1.334
3.00	1.44	1.424
3.25	1.49	1.512
3.50	1.69	1.599
3.75	1.65	1.684
4.00	1.79	1.767
4.25	1.79	1.850
4.50	1.84	1.931
4.75	1.94	2.010
5.0	2.10	2.089

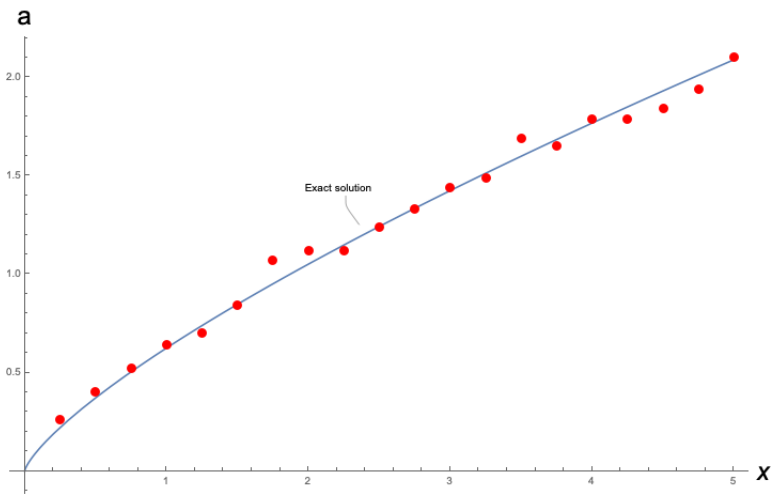

Figura 2. Política óptima aproximada.

La Tabla 2 muestra las acciones óptimas aproximadas para algunos valores de x, obtenidas usando el método descrito en Algoritmo 2 de la Sección 3.5, contrastadas con la política óptima exacta. La Figura 2 muestra otras acciones óptimas aproximadas, denotadas por puntos rojos, que exhiben un comportamiento similar que el de la política óptima exacta, graficada por una línea sólida azul.
El algoritmo modificado propuesto de Q-Learning es una buena aproximación para las acciones óptimas con los parámetros dados, sin necesidad de recurrir a la solución cerrada, si es que existe.

3.4.2. Ejemplo: Utilidad de consumo exponencial

Consideramos ahora el criterio de rendimiento a ser optimizado:

$$V(\pi, x) \;:=\; \mathbb{E}_x^{\pi}\left[\sum_{t=1}^{\infty} \alpha^t \left(1 - Exp(-\tilde{a}_t)\right)\right], \qquad \pi \in \Pi, x \in \mathbb{X},$$

donde el capital inicial es x y $\alpha = 0.5$ es el factor de descuento. La función de utilidad de consumo $U(\tilde{a}_t) = \left(1 - Exp(-\tilde{a}_t)\right) \in C^2((0,\infty)$, es estrictamente creciente, estrictamente cóncava e invertible, además satisface $U'(0) = \infty$, y $lim_{a\to\infty}U'(a) = 0$.

Se toma una función de producción $h(t) = x^{\delta}$ con transición del sistema:

$$x_{t+1} = (x_t^{\delta} - a_t)\xi_t,$$

donde $\{\xi_t\}$ es tal que $Log(\xi_t) \sim N(0, 0.4)$.

Para Q-Learning, consideremos $\lambda_k(\tilde{x}_i, \tilde{a}) := \frac{Log(k)}{k}$ como tasa de refuerzo o aprendizaje y $k_{max} = 1000$. La Tabla 3 y la Figura 3 concentran los resultados.

Tabla 3. Política óptima aproximada.

x	$\tilde{a}^*(x)$
1.0	0.68
2.0	1.01
3.0	1.37
4.0	1.65
5.0	1.79
6.0	1.92
7.0	2.21
8.0	2.25
9.0	2.28
10.0	2.55
20.0	3.62

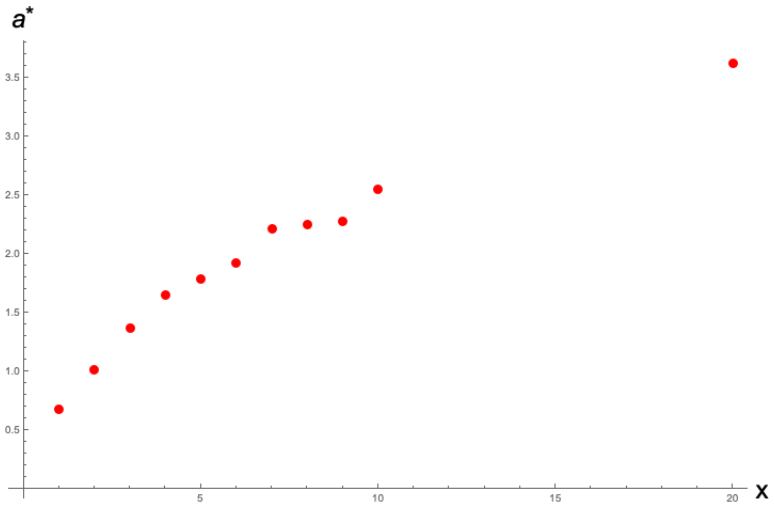

Figura 3. Política óptima aproximada.

La Tabla 3 muestra las acciones óptimas aproximadas para algunos valores de x, obtenidas usando el método descrito en Algoritmo 2 en la Sección 3.4. La Figura 3 representa gráficamente los resultados mostrados en la Tabla 3. Muestran los consumos óptimos, denotados por puntos rojos, que se deberían tomar cuando se tiene inicialmente el capital x, pese a que, para este ejemplo, no se encontró la solución exacta. El seguimiento de las políticas óptimas sugiere que la solución tiene un comportamiento parabólico.

En las Secciones 3.2 y 3.3 se trabajó con el mismo problema de consumo-inversión que tiene solución exacta. Así que primero se presenta ésta, y después dos algoritmos de aproximación con sus respectivos procedimientos de discretización para el caso horizonte finito.
En la Sección 3.4 se trabaja con otro problema de consumo-inversión y función de utilidades de consumo de tipo logarítmica y exponencial, empleamos el método Q-Learning modificado para problemas $\alpha-$descontados con horizonte infinito.

Conclusiones

El aprendizaje por refuerzo (Reinforcement Learning) es un área del aprendizaje automático que se ocupa de cómo los agentes deben tomar acciones en un entorno para maximizar recompensas acumuladas.

Q-learning es un algoritmo de aprendizaje por refuerzo para "aprender" el valor de una acción en un estado particular. Puede manejar problemas con transiciones estocásticas y recompensas sin requerir adaptaciones.

Para cualquier proceso de decisión de Markov finito, *Q-learning* encuentra una política óptima en el sentido de maximizar el valor esperado de la recompensa total en cada iteración en todos los estados por donde transite el sistema, comenzando desde el estado actual.

El entorno se establece típicamente en la forma de un proceso de decisión de Markov, porque muchos algoritmos de aprendizaje por refuerzo para este contexto utilizan técnicas de programación dinámica.

Existe muy poca teoría que conjunte *Q-learning* con procesos de decisión, la literatura se encuentra un poco dispersa.

El método propuesto en este trabajo ofrece una alternativa de solución a la política óptima de un problema de control. Este método también es útil cuando no se dispone de una solución cerrada o cuando las herramientas de resolución clásicas no encuentran la política óptima directamente.

En las estrategias de inversión óptimas que se derivan en forma cerrada, suelen proponerse funciones candidatas con la forma general de la solución. Estas funciones pueden ser muy difíciles de encontrar cuando se utilizan funciones de producción o de utilidad más complejas o desconocidas.

Es en la configuración de parámetros del algoritmo donde se obtiene un plus del algoritmo propuesto, pues por su adaptabilidad para trabajar multiples funciones de utilidad y el ajuste de los parámetros de las distribuciones es donde obtiene su mejor provecho. Además de que, se tienen resultados aceptables ejecutándose desde una computadora de segmento medio.

Los métodos de refuerzo de aprendizaje tienen una ventaja en términos de capacidad de implementación, asignando valores de bondad a los pares estado-acción, poniéndolos a prueba y descartando selecciones de acciones que produzcan resultados adversos.

Como problemas futuros que podrían derivarse de esta tesis, se encuentra la uti-

lización de otros criterios de rendimiento como el costo promedio, inclusive considerar modelos semimarkovianos. También, para obtener mejores resultados con los algoritmos provistos, se requiere implementarlos en equipos de cómputo de alto rendimiento.

Apéndice A

Abreviaturas y Terminología

$v.a.$, $(v.a's)$	Variable(s) aleatoria(s)
$i.i.d.$	independientes e idénticamente distribuidas (v.a's)
$c.s.$	Casi seguramente
$u.s.c.$	Semicontinua superior
$l.s.c.$	Semicontinua inferior

$\mathbb{E}\{f(X)\}$	Valor esperado (esperanza) de la v.a. $f(X)$
$\mathbb{V}ar\{f(X)\}$	Varianza de la v.a $f(X)$
$X \sim Nor(\mu, \sigma^2)$	v.a X con distribución normal y parámetros μ y σ^2
$X \sim LogNor(\mu, \sigma^2)$	v.a X con distribución lognormal y parámetros μ y σ^2

Apéndice B

Resultados

Definición B.0.1 *Un espacio de probabilidad es una tripleta* $(\Omega, \mathcal{F}, P)$ *donde* Ω *es un conjunto arbitrario,* $\mathcal{F}$ *es una* $\sigma-$*álgebra de subconjuntos de* Ω*, y* P *es una medida de probabilidad sobre* $\mathcal{F}$*.*

Definición B.0.2 *La colección* $\{\mathcal{F}_t, t \geq 0\}$ *de* $\sigma-$*álgebras sobre* Ω *es llamada una filtración si*

$$\mathcal{F}_s \subset \mathcal{F}_t \qquad \text{para todo } 0 \leq s \leq t.$$

El proceso estocástico $Y = \{Y_t, t \geq 0\}$ *se dice adaptado a la filtración* $\{\mathcal{F}_t, t \geq 0\}$ *si*

$$\sigma(Y_t) \subset \mathcal{F}_t \qquad \text{para todo } t \geq 0.$$

El proceso estocástico Y *es siempre adaptado a la filtración natural generada por* Y*:*

$$\mathcal{F}_t = \sigma(Y_s, s \leq t).$$

Definición B.0.3 *Un proceso estocástico* $W = \{W_t, t \in [0, \infty)\}$ *es llamado movimiento browniano estándar o un proceso de Wiener si las siguientes condiciones son satisfechas:*

- $W_0 = 0$
- *Tiene incrementos estacionarios independientes.*
- *Para todo* $t > 0$*,* W_t *tiene una distribución normal* $N(0, t)$*.*
- *Tiene trayectorias muestrales continuas.*

Definición B.0.4 *Sean* $\mathbb{X}$ *y* $\mathbb{Y}$ *espacios de Borel. Un kernel estocástico sobre* $\mathbb{X}$ *dado* $\mathbb{Y}$ *es una función* $P(\cdot|\cdot)$ *que satisface las siguientes propiedades:*

a) $P(\cdot \,|\, h_t)$ *es una medida de probabilidad sobre* $\mathbb{X}$*, para cada* $y \in \mathbb{Y}$ *fijo.*

b) $P(B \,|\, \cdot)$ *es una función medible sobre* $\mathbb{Y}$ *para cada* $B \in \mathcal{B}(\mathbb{X})$ *fijo, donde* $\mathcal{B}(\mathbb{X})$ *denota la* σ*-álgebra de Borel de* $\mathbb{X}$*.*

El conjunto de todos los kerneles estocásticos sobre $\mathbb{X}$ *dado* $\mathbb{Y}$ *es denotado por* $P(\mathbb{X} \,|\, \mathbb{Y})$*.*

Teorema B.0.5 *Sea* $\pi = \{\pi_t\}$ *una política de control arbitraria y* ν *una medida de probabilidad arbitraria sobre* $\mathbb{X}$*, referida como distribución inicial. Entonces existe una única medida de probabilidad* P_ν^π *sobre* $(\Omega, \mathcal{F})$ *con* $P_\nu^\pi(H_\infty) = 1$*, y además, para todo* $B \in \mathcal{B}(\mathbb{X})$*,* $C \in \mathcal{B}(\mathbb{A})$*, y* $h_t \in H_t$*,* $t = 0, 1, \cdots$

$$\begin{aligned} P_\nu^\pi(x_0 \in B) &= \nu(B) \\ P_\nu^\pi(a_t \in C \,|\, h_t) &= \pi_t(C \,|\, h_t) \\ P_\nu^\pi(x_{t+1} \in B \,|\, h_t, a_t) &= Q(B \,|\, x_t, a_t). \end{aligned}$$

Definición B.0.6 *La función zeta de Riemann* $\zeta(s)$ *está definida, para valores complejos con parte real mayor que uno, por la serie:*

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}.$$

Definición B.0.7 *El mapeo* $x \mapsto \mathbb{A}(x)$ *de* $\mathbb{X}$ *a* $\mathbb{A}$ *se dice* semicontinua superior (u.s.c.) *si* $\{x \in \mathbb{X} \,|\, \mathbb{A}(x) \cap F \neq \emptyset\}$ *es un conjunto cerrado de* $\mathbb{X}$ *para todo conjunto cerrado* $F \subset \mathbb{A}$*.*
El mapeo $x \mapsto \mathbb{A}(x)$ *de* $\mathbb{X}$ *a* $\mathbb{A}$ *se dice* semicontinua inferior (l.s.c.) *si* $\{x \in \mathbb{X} \,|\, \mathbb{A}(x) \cap G \neq \emptyset\}$ *es un conjunto abierto de* $\mathbb{X}$ *para todo conjunto abierto* $G \subset \mathbb{A}$*.*

Bibliografía

[1] Almudévar, A., *Approximate iterative algorithms*, CRC Press, UK, (2014).

[2] Bellman, R., *Dynamic Programming*, Princeton University Press, USA, (1972).

[3] Bertsekas, D., *Dynamic Programming And Optimal Control*, Vol. I & II, Athena Scientific, USA, (1995).

[4] Busoniu, *Reinforcement learning and dynamic programming using function approximators*, CRC Press, (2010).

[5] Chávez, M. S., *Aleatorización en programación dinámica para el análisis del problema de la dimensionalidad aplicada a procesos de decisión de Markov*, Tesis maestría, (2012).

[6] Chow, C., *Multigrid Algorithms And Complexity Results For Discrete-Time Stochastic Control And Related Fixed-Point Problems*, MIT, USA, (1989).

[7] Chow, C., Tsitsiklis, J., *An Optimal One-Way Multigrid Algorithm For Discrete-Time Stochastic Control*, IEEE Transactions on Automatic Control, Vol. 36, No. 8, (1991).

[8] Cruz-Suárez, H. A., Montes-de-Oca, R., & Zacarías, G., A consumption-investment problem modelled as a discounted Markov decision process, *Kybernetika*, 47(6), pp. 909-929. (2011).

[9] De La Fuente, A. *Mathematical Methods and Models*. Cambridge: Cambridge University Press. (2000).

[10] Gosavi, *Simulation-Based Optimization. Parametric Optimization Techniques and Reinforcement Learning*, Springer, (2015).

[11] Hernández-Lerma, O., *Adaptive Markov Control Processes*, Applied Mathematical Sciences, Vol. 79, Springer-Verlag, USA, (1989).

[12] Hernández-Lerma, O., Laserre, J. B., *Discrete-Time Markov Control Processes. Basic Optimality Criteria*, Applications of Mathematics, Stochastic Modelling and Applied Probability, Vol. 30, Springer, USA, (1996).

[13] Higham, D. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Review*, 43(3), pp. 525-546. (2001).

[14] Higham, D. The numerical solution of stochastic differential equations. *Journal Australia Mathematics Society* No. 20. Australia. (1977).

[15] Liang, Z., & Ma, M., Consumption-investment problem with pathwise ambiguity under logarithmic utility. *Mathematics and Financial Economics*, 13(4), pp. 519-541. (2019).

[16] López-Ríos, R. A., & Cruz-Suárez H. A., Q learning approach to a Consumtion-Investment Problem, *International Journal of Statistics and Probability*, 10(2), (2021).

[17] Mendelssohn, R., & Sobel, M., Capital accumulation and the optimization of renewable resource models, *Journal of Economic Theory*, 23(2), pp. 243-260. (1980).

[18] Powell, W. *Approximate dynamic programming. Solving the curses of dimensionality*, 2ed. Wiley, (2011).

[19] Puterman, M., *Markov Decision Processes. Discrete Stochastic Dynamic Programming*, John Wiley & Sons, USA, (2005).

[20] Robbins, H., & Monro, S. A stochastic approximation method, *Annals of Mathematical Statistics*, University of North Carolina, 22(3), pp. 400-407. (1951).

[21] Ross, S., *Introduction to stochastic dynamic*, Academic Press, (1983).

[22] Rust, *Using randomization to break the curse of dimensionality*, Econometrica, Vol. 65, pp. 487-516, (1997).

[23] Samuelson, P.A., Lifetime portfolio selection by dynamic stochastic programming *The Review of Economic and Statistics*, 51(3), pp. 239-246, (1969).

[24] Si & Barto, *Handbook of learning and approximate dynamic programming*, IEEE Press Wiley Interscience, (2004).

[25] Singh, S., Learning to solve Markovian decision processes. PhD thesis, University of Massachusetts. (1994).

[26] Sutton, R., & Barto, A. (2018) *Reinforcement Learning. An Introduction*, Cambridge, MA: MIT Press.

[27] Tsitsiklis, J., Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16, pp. 185-202. (1994).

[28] Vitoriano, B., & Parlier, G. H. (Eds.). Operations Research and Enterprise Systems, Fifth International Conference, ICORES 2016, Rome, Italy, February 23-25, 2016, Revised Selected Papers, Vol. 695. Rome: Springer (2017).

[29] Watkins, C. (1989). Learning from delayed rewards. PhD thesis. Cambridge, King's College.

[30] Watkins, C., & Dayan, P., Q-Learning. Technical note. *Machine Learning, (8)*, pp. 279-292. Boston, MA: Kluwer Academic Publishers. (1992).

[31] Weissensteiner, A., A Q-Learning Approach to Derive Optimal Consumption and Investment Strategies. IEEE transactions on neural networks, 20(8), pp. 1234-1243. (2009).

[32] White, D. A survey of applications of Markov decision processes. *Journal of the Operational Research Society*, 44, pp. 1073-1096. (1993).