Ejercicio 1

[Programación] Para el proceso de Markov de recompensas (MRP) de la figura 1, calcular los valores de los estados de forma iterativa con los siguientes algoritmos y compare sus convergencias. Considere factor de descuento $\gamma < 0.9$.

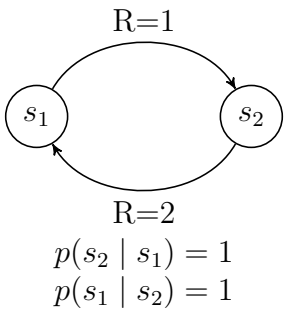


Figura 1: Grafo de transición de estados

- Actualizar todos los valores a la vez por iteración: $v_{k+1} = r + \gamma P v_k$, con $v_k\, r$ siendo los vectores de valores y recompensas, respectivamente; y P la matriz de probabilidades de transición.
- Actualizar los valores de un estado por vez *(in place)*: $v_{k+1}(s') = r(s') + \gamma v_k(s)$, con $v_k(s)$ y $r(s')$ siendo los valores y recompensas correspondientes a los estados s y s', respectivamente.

Solución

La implementación de los algoritmos se realizó en Python, que se puede encontrar en el repositorio de GitHub.
Sea, P la matriz de probabilidades de transición:

$$P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

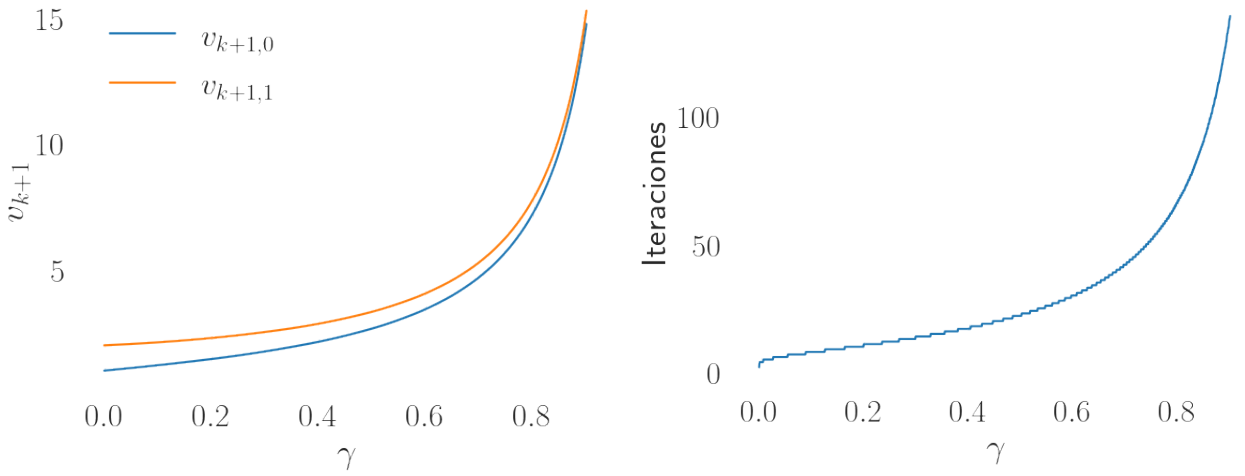
Sea r el vector de recompensas:

$$r = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

A continuación los resultados de los valores de los estados para $\gamma \in [0, 0.9)$ usando los métodos *iterativo* e *in place*.

Método *iterativo* Se calculó el vector de valor v_{k+1} usando el cálculo iterativo con la librería `numpy` de Python[1]. La figura 2a muestra los valores de v_{k+1} y la figura 2b muestra el número de iteraciones necesarias para converger, ambas para $\gamma \in [0, 0.9)$.
La figura 2 muestra los resultados para el método *iterativo*.

[1] Se una tolerancia de 10^{-6} y un máximo de 1000 iteraciones.

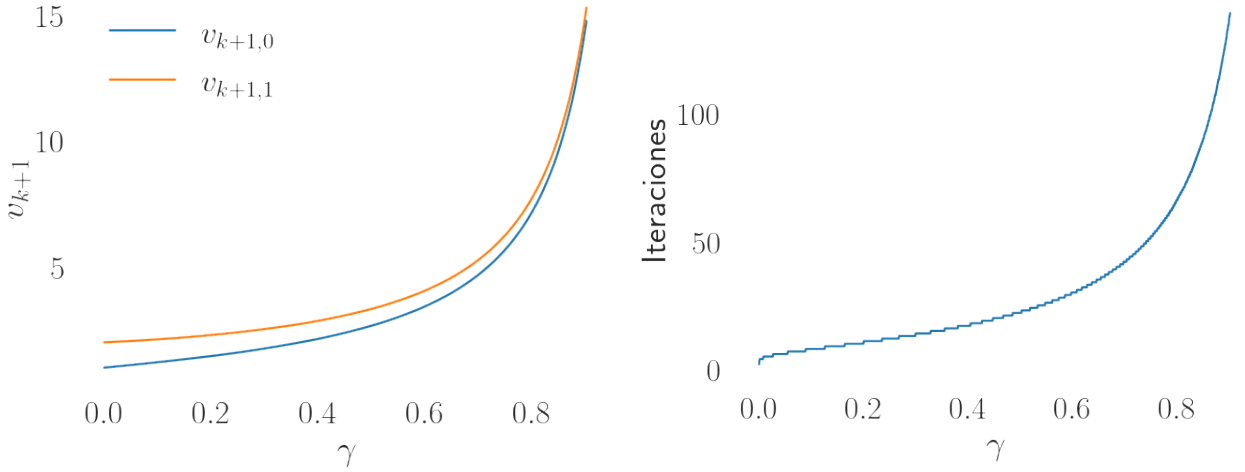


(a) Vector de valores

(b) Número de iteraciones para converger

Figura 2: Resultados del método *iterativo*

Método *in place* Se calculó el vector de valor v_{k+1} usando el cálculo iterativo con la librería `numpy` de Python[2]. La figura 3a muestra los valores de v_{k+1} y la figura 3b muestra el número de iteraciones necesarias para converger, ambas para $\gamma \in [0, 0.9)$.
La figura 3 muestra los resultados para el método *in place*.



(a) Vector de valores

(b) Número de iteraciones para converger

Figura 3: Resultados del método *inplace*

Conclusiones

No existe una diferencia significativa entre los métodos *iterativo* e *in place* para valores de $\gamma \in [0, 0.9)$. A continuación se presentan las conclusiones de los resultados obtenidos.

Vector de valores v_{k+1} Las figuras 2a y 3a muestran los valores de v_{k+1} para $\gamma \in [0, 0.9)$ para los métodos *iterativo* e *in place*, respectivamente. De las figuras se concluye,

[2]Se fijó una tolerancia de 10^{-6} y un máximo de 1000 iteraciones.

- Ambos métodos mostraron una rápida convergencia cuando $\gamma \approx 0$, dando como resultado $v_{k+1} = r = \begin{bmatrix}1\\2\end{bmatrix}$.
- Los valores de v_{k+1} aumentan de forma exponencial a medida que γ aumenta.
- Conforme $\gamma \to 0.9$, los métodos convergen a los valores teóricos $v = \begin{bmatrix}14.74\\15.26\end{bmatrix}$.

Iteraciones Las figuras 2b y 3b muestran el número de iteraciones necesarias para converger para $\gamma \in [0, 0.9)$ para los métodos *iterativo* e *in place*, respectivamente. De las figuras se concluye,

- Las iteraciones necesarias para converger aumentan de forma exponencial a medida que γ aumenta.
- Conforme $\gamma \to 0.9$, se requieren más iteraciones para converger, lo que se traduce en un mayor tiempo de cómputo.

Implementación En cuanto a la implementación de ambos métodos, la diferencia entre el método *iterativo* e *in place* radica en que el segundo no requiere de una matriz de transición P. Desde una perspectiva computacional el método *in place* es más eficiente, ya que no requiere de una matriz de transición. También es útil en escenarios en los que la matriz de transición no es conocida, como en el caso de un agente que interactúa con un entorno desconocido.

Ejercicio 2

En el Ejemplo 4.1 (*GridWorld, Sutton&Barto, 2018*), donde la política π es aleatoria y equiprobable:

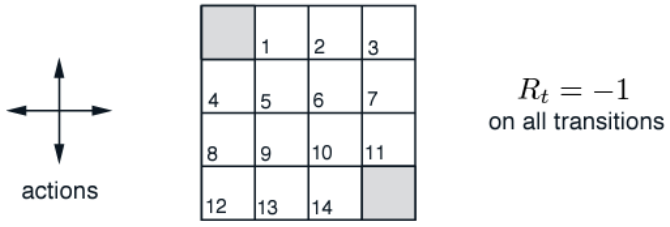


Figura 4: *GridWorld* [1]

- ¿Cuánto vale $q_\pi(11, down)$?
- ¿Cuánto vale $q_\pi(7, down)$?

Nota: utilice $v(11) = -14$ (figura 4).
Justifique sus respuestas.

Solución

Sea,
$S = \{1, 2, \ldots, 14\}$: conjunto de estados no terminales[3]
$A = \{up, down, left, right\}$: conjunto de acciones
$R_t = -1$: recompensa por transición
π: política aleatoria y equiprobable, es decir, $\pi(a|s) = 0.25,\ \forall s \in S,\ \forall a \in A$
q_π: función de valor de acción para la política π

[3]Los estados terminales son aquellos que se encuentran sombreados en la figura 4.

Suposición Se supone que $\gamma = 1$, entonces MDP es episódico y no hay descuento en las recompensas. La ecuación de Bellman para $q_\pi(s, a)$ es:

$$q_\pi(s, a) = \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

Para calcular $q_\pi(11, down)$, se parte del estado 11 y se realiza la acción *down*. Bajo el estado 11 se encuentra el estado terminal, el estado terminal tiene una recompensa de 0, ya que no se obtiene ninguna recompensa por llegar a un estado terminal. Por lo tanto,

$$\begin{aligned} q_\pi(11, down) &= -1 + 0 \\ &= -1 \end{aligned}$$

Para calcular $q_\pi(7, down)$, se tiene que bajo el estado 7 se encuentra el estado 11 y $v(11) = -14$. Por lo tanto,

$$\begin{aligned} q_\pi(7, down) &= -1 + (-14) \\ &= -15 \end{aligned}$$

Ejercicio 3

En el Ejemplo 4.1 (*GridWorld*, Sutton&Barto, 2018) [1], suponga que se agrega un nuevo estado 15 debajo del estado 13 y sus acciones: *left, up, right* y *down*, lleva al agente a los estados 12, 13, 14 y 15, respectivamente.

- Considere que las transiciones desde los estados originales no se cambian. ¿Cuánto vale $v_\pi(15)$ para la política π aleatoria y equiprobable? Utilice $v(12) = -22$, $v(13) = -20$, $v(14) = -14$ (figura 4).

Justifique su respuesta.

Solución

Sea,
$p(s', r|s, a) = 0.25, \ \forall s \in S, \forall a \in A$

La función de valor de estado $v_\pi(s)$ para la política π es:

$$v_{pi} = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

Para calcular $v_\pi(15)$,

$$\begin{aligned} v_\pi(15) &= \sum_a \pi(a|15) \sum_{s',r} p(s', r|15, a) \left[r + \gamma v_\pi(s')\right] \\ &= 0.25 \left[r + \gamma v_\pi(s')\right] \\ &= 0.25 \left[(-1 - 20) + (-1 - 22) + (-1 - 14) + (-1 + v_\pi(15))\right] \\ &= 0.25 \left[-21 - 23 - 15 - 1 + v_\pi(15)\right] \\ &= 0.25 \left[-60 + v_\pi(15)\right] \\ &= -15 + 0.25 \cdot v_\pi(15) \end{aligned}$$

Al considerar que las transiciones desde los estados originales no se cambian, es posible la transición del estado 13 al 15. Pasar al estado 15 tiene un valor igual al del estado 13. Por lo tanto,

$$\begin{aligned} v_\pi(15) &= v_\pi(13) \\ &= -20 \end{aligned}$$

Al reemplazar $v_\pi(15)$ en la ecuación anterior, se obtiene:

$$\begin{aligned} v_\pi(15) &= -15 + 0.25 \cdot v_\pi(15) \\ &= -15 + 0.25 \cdot (-20) \\ &= -15 - 5 \\ &= -20 \end{aligned}$$

Ejercicio 4

En el Ejemplo 4.3 (*Gambler's problem, Sutton&Barto, 2018*) [1], la política óptima tiene una forma particular (ver figura 5) con máximo en 50. Es decir, cuando el jugador tiene \$50, le conviene apostarlo todo; sin embargo, cuando tiene \$51, le conviene apostar \$1. ¿Por qué sucede esto?

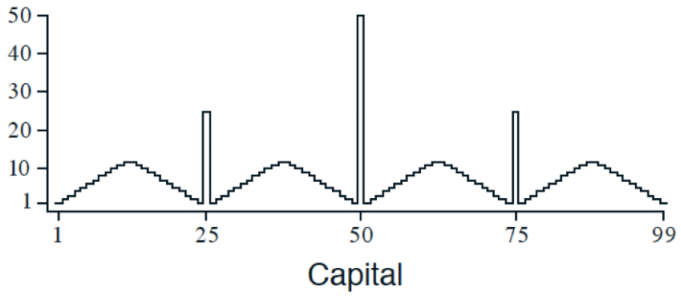


Figura 5: Ejemplo 4.3 [1]

Solución

El ejemplo 4.3 (*Gambler's problem, Sutton&Barto, 2018*) [1] se plantea como un MDP finito, sin descuento, episódico y con un conjunto de estados $s \in \{0, 1, \ldots, 99\}$ y un conjunto de acciones $a \in \{0, 1, \ldots, \text{mín}(s, 100 - s)\}$.
Por otro lado, la política final encontrada es tal que $p_h = 0.4$, es decir, la probabilidad de que la moneda salga cara es 0.4.
Por lo tanto, cuando la moneda está cargada y al jugador le conviene minimizar el número de apuestas realizadas porque en el largo plazo, el jugador perderá dinero. Por lo tanto, el jugador tiene una *ventaja*[4]y le conviene apostar todo cuando tiene \$50, ya que a la izquierda y derecha de este valor las apuestas llevarán al jugador de vuelta a \$50. Por lo que si tiene \$51 le conviene apostar \$1 sabiendo que si pierde, volverá a \$50.

Ejercicio 5

[Programación] Implemente el Algoritmo de Iteración de Valores para el el Ejemplo 4.3 (*Gambler's problem, Sutton&Barto, 2018*) [1] para los siguientes casos:

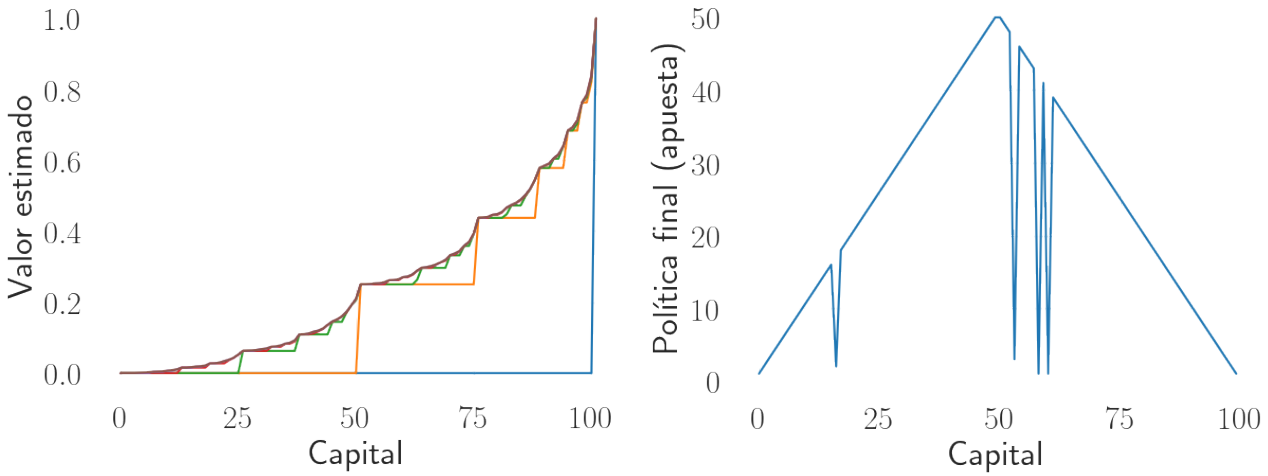
[4]La ventaja se refiere a que si el jugador apuesta los \$50 tendrá el 40 % de probabilidad de ganar.

- $p_h = 0.25$.
- $p_h = 0.55$.

Solución

La implementación de los algoritmos se realizó en Python, que se puede encontrar en el repositorio de GitHub.
Sea,
$N = 100$: capital máximo
$S = \{0, 1, \ldots, 100\}$: conjunto de estados
$A = \{0, 1, \ldots, \text{mín}(s, 100 - s)\}$: conjunto de acciones

El Algoritmo de Iteración de Valores para el Ejemplo 4.3 (*Gambler's problem, Sutton&Barto, 2018*) [1] se implementó en Python[5]. A continuación, los resultados de la implementación, $p_h = 0.25$, se muestran en las figuras 6a y 6b.

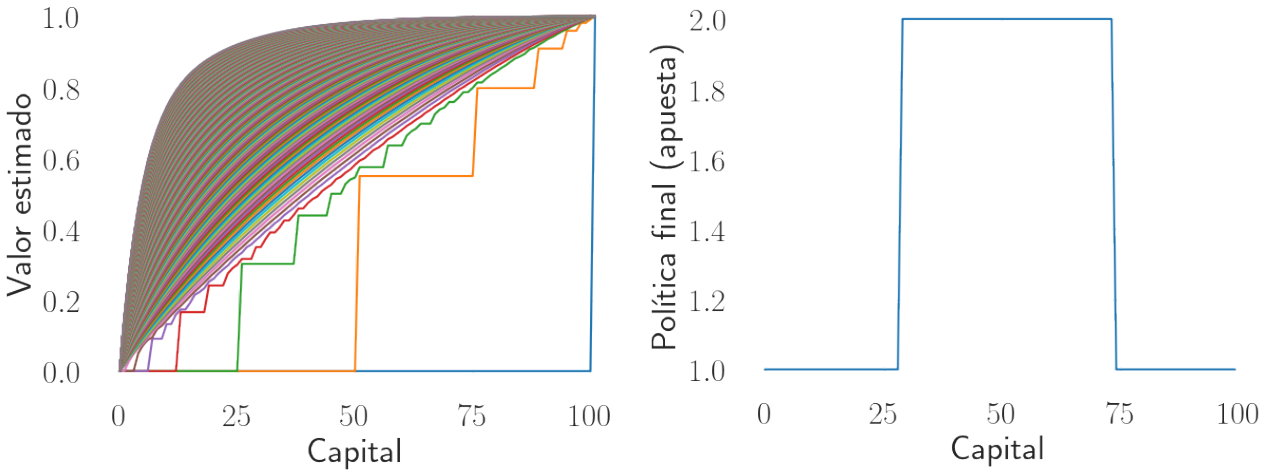


(a) *sweeps* $\in [0, N]$ (b) Política final

Figura 6: $ph = 0.25$

A continuación, los resultados de la implementación, $p_h = 0.55$, se muestran en las figuras 7a y 7b.

[5]Se fijó una tolerancia de 10^{-3} para la convergencia.



(a) *sweeps* $\in [0, N]$

(b) Política final

Figura 7: $ph = 0.55$

Conclusiones

Convergencia En las figuras 6a y 7a se muestra que a medida que se aumentan las iteraciones (sweeps), los valores de vada estado convergen a un valor estable.

Política final Se encuentra influenciada por el valor de p_h. Cuando la probabilidad de ganar es baja la política óptima es apostar poco, especialmente cuando el capital es bajo, hecho que se deba al alto riesgo de perder el capital. Además, para este caso la política final tiene forma de V invertida como se muestra en la figura 6b.
Por otro lado, cuando la probabilidad de ganar es alta, la política óptima es apostar más, especialmente cuando el capital es intermedio, hecho que se debe a la alta probabilidad de ganar y la posibilidad de maximizar la ganancia. Además, para este caso la política final tiene forma de V achatada, como se muestra en la figura 7b.
En ambos casos:

- Se apuesta poco si se tiene poco capital para evitar perderlo.
- Se apuesta más si se tiene un capital intermedio, aprovechando la oportunidad de ganar.
- Se apuesta poco si se tiene mucho capital para asegurar ganancia.

Se hace la salvedad que en el caso de $p_h = 0.25$ la política final tiene un máximo en el que se encuentra poca estabilidad.

Referencias

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, second ed., 2018.