Aprendizaje por refuerzo

Aproximación de políticas

Antonio Manjavacas

manjavacas@ugr.es

Contenidos

Recapitulando...

En el tema anterior vimos cómo aproximar funciones de valor para, posteriormente, obtener una política óptima.

El proceso seguido era el siguiente:

1. A partir de la interacción con el entorno, ajustábamos un **modelo** (ej. lineal, red neuronal...) para aproximar la **función de valor**, ya sea $\hat{v}(s, \boldsymbol{w}) \simeq v(s)$ o $\hat{q}(s, a, \boldsymbol{w}) \simeq q(s, a)$.
   - Empleábamos el **descenso del gradiente** para minimizar el error en las predicciones.
   - Asegurábamos la exploración mediante ε-*greedy*.
2. Una vez ajustado nuestro modelo, lo empleábamos para elegir aquellas acciones que maximizasen el retorno esperado.

💡 *Do not solve a more general problem as an intermediate step.*

∼ Vladimir Vapnik

Políticas parametrizadas

Estamos aprendiendo una función de valor parametrizada para posteriormente derivar la política óptima...

> 💡 ¿Por qué no emplear directamente una **política parametrizada**?

- Vamos a estudiar **algoritmos basados en una política parametrizada**, que seleccionan acciones sin consultar la función de valor.
- La función de valor puede seguir siendo utilizada, pero no se requiere para seleccionar acciones.

Políticas parametrizadas

Representamos el vector de parámetros de la política como $\boldsymbol{\theta} \in \mathbb{R}^{d'}$.

De esta forma, definimos una política parametrizada de la siguiente forma:

$$\pi(a \mid s, \boldsymbol{\theta}) = \Pr\{A_t = a \mid S_t = s, \boldsymbol{\theta}_t = \boldsymbol{\theta}\}$$

Gradiente de la política

Ajustaremos $\boldsymbol{\theta}$ basándonos en el **gradiente** de una medida escalar $J(\boldsymbol{\theta})$ que representa el rendimiento.

- Como estos métodos buscan MAXIMIZAR el rendimiento, las actualizaciones se basan en el **gradiente ascendente** de J:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \widehat{\nabla J(\boldsymbol{\theta}_t)}$$

donde $\widehat{\nabla J(\boldsymbol{\theta}_t)} \in \mathbb{R}^{d'}$ es una estimación estocástica cuya expectación aproxima el gradiente de $J(\boldsymbol{\theta}_t)$.

Todos los métodos que siguen este esquema se denominan ***policy gradient*** (métodos basados en el **gradiente de la política**).

Aproximación de políticas

Aproximación de políticas

Los métodos basados en **gradiente de la política** se basan en el uso de una **política parametrizada**, tal que:

$$\pi(a|s, \boldsymbol{\theta})$$

⚠️ Es necesario que $\pi_{\boldsymbol{\theta}}$ sea diferenciable con respecto a $\boldsymbol{\theta}$.

En la práctica, para garantizar la **exploración**, se necesita que la política nunca sea determinista, es decir:

$$\pi(a|s, \boldsymbol{\theta}) \in (0, 1) \quad \forall s, a, \boldsymbol{\theta}$$

Políticas parametrizadas *soft-max*

Si el espacio de acciones es discreto y no demasiado grande, una forma común de parametrización es establecer **preferencias numéricas** $h(s, a, \boldsymbol{\theta}) \in \mathbb{R}$ para cada par acción-estado.

- A mayor preferencia, mayor probabilidad de elegir una acción.
- Por ejemplo, podemos utilizar **soft-max** sobre estas preferencias (*soft-max action preferences*):

$$\pi(a|s, \boldsymbol{\theta}) = \frac{e^{h(s,a,\boldsymbol{\theta})}}{\sum_b e^{h(s,b,\boldsymbol{\theta})}}$$

- La parametrización de las preferencias pueden venir dadas por los pesos de una **red neuronal** (ej. AlphaGo) o por características lineales (*linear features*):

$$h(s, a, \boldsymbol{\theta}) = \boldsymbol{\theta}^T \boldsymbol{x}(s, a) \quad \boldsymbol{x}(s, a) \in \mathbb{R}^{d'}$$

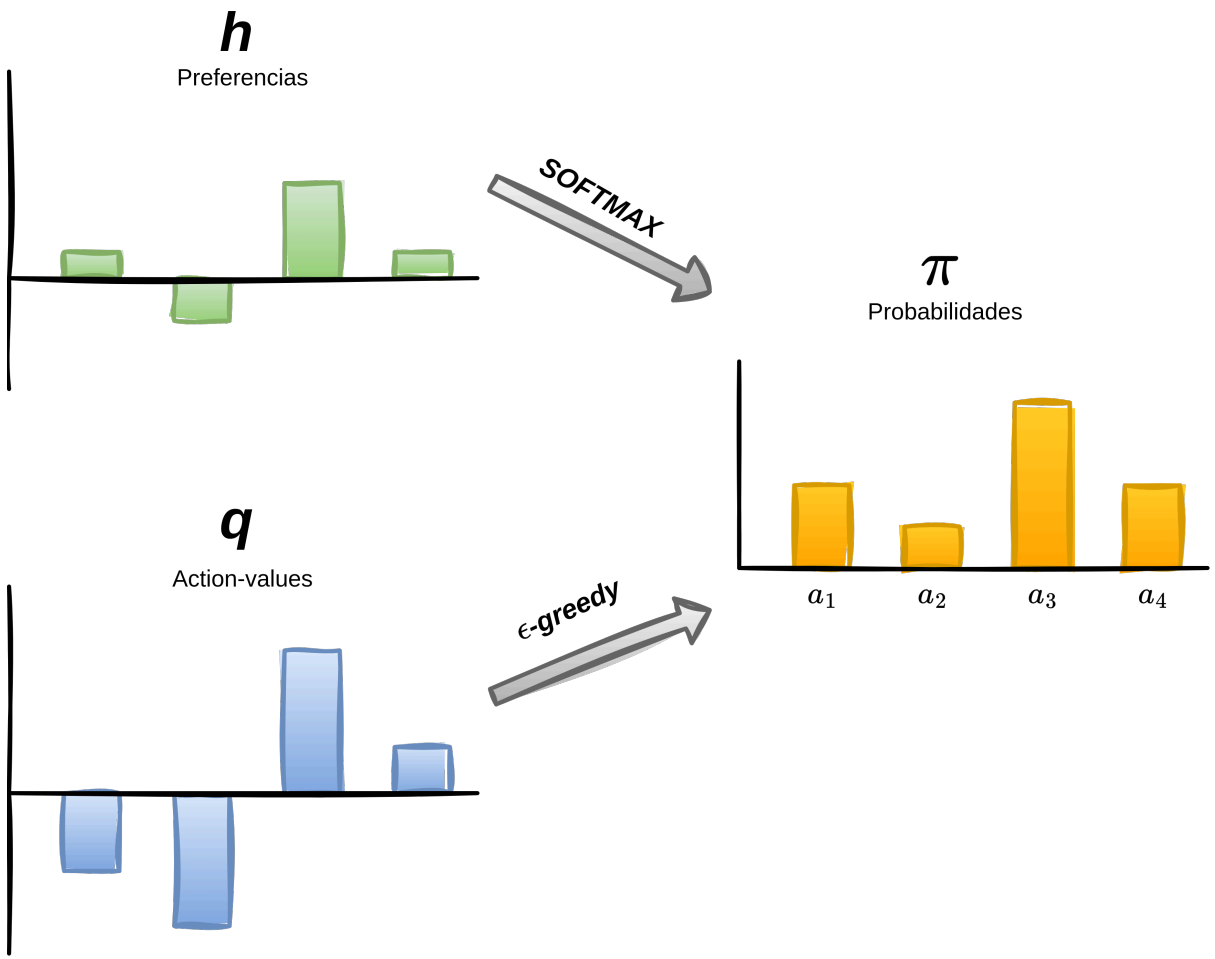
Políticas parametrizadas *soft-max*

$$\pi(a|s,\boldsymbol{\theta}) = \frac{\overbrace{e^{h(s,a,\boldsymbol{\theta})}}^{\text{Preferencia por acción}}}{\underbrace{\sum_{b\in\mathcal{A}} e^{h(s,b,\boldsymbol{\theta})}}_{\text{Preferencia por el resto de acciones}}}$$

- Se emplea e^x para que siempre se obtenga un valor positivo.
- El denominador normaliza el resultado para que la suma de probabilidades sea 1.

Debemos distinguir entre **preferencias de acción** y **valores de acción**:

Valor *vs.* preferencia



Valor *vs.* preferencia

Valor

Estimación de cuán bueno es un estado/acción en términos de la recompensa esperada a largo plazo.

Preferencia

Medida de cuánto prefiere el agente una acción sobre otra bajo su política actual.

Políticas parametrizadas *soft-max* vs. ε-*greedy*

Ventaja de emplear políticas parametrizadas + *soft-max* frente a ε-*greedy*:

- Con ε-*greedy* siempre existe cierta capacidad de explorar, a menos que el valor de ε se reduzca con el tiempo.
- Si empleamos *soft-max* con preferencias de acción, las probabilidades se asignan de modo que, si una acción es mucho mejor, su preferencia crece sin límite.

> Esto permite que, si realmente hay una mejor acción, la política sea **prácticamente determinista**.

Políticas estocásticas

Otra ventaja de las políticas parametrizadas basadas en preferencias *soft-max* es que **permiten la selección de acciones con probabilidades arbitrarias**. Es decir, permiten aprender **políticas estocásticas óptimas.**

- Útil en problemas donde, en un mismo estado, dos acciones pueden ser viables con diferente probabilidad.
  - Por ejemplo, juegos de cartas parcialmente observables donde jugar de forma óptima requiere en ocasiones actuar de forma diferente con probabilidades específicas (ej. jugar de farol en Poker).
- Los métodos basados en *action values* no tienen una forma natural de converger en políticas estocásticas óptimas.

Políticas parametrizadas *soft-max*

Quizá la mayor y más simple ventaja de las políticas parametrizadas es que **generalmente son funciones más simples de aproximar que las funciones de valor**.

- Aunque siempre dependerá del problema abordado, es común que esto se cumpla.

También es posible incluir cierto **conocimiento experto** en el sistema a partir de la elección de los parámetros de la política.

- Esto no ocurre si el ajuste de dichos parámetros es automático (ej. red neuronal).

Actualización de la política

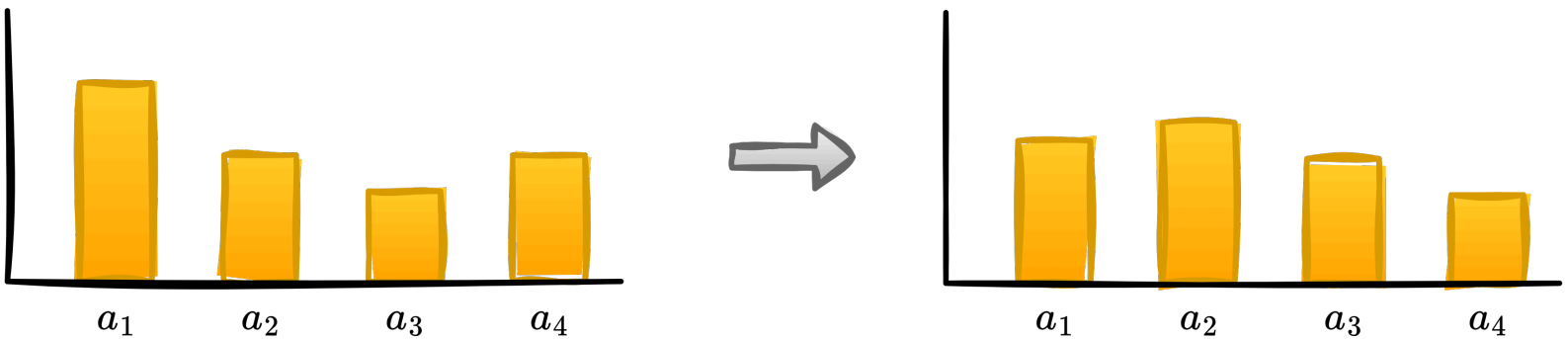
Finalmente, las políticas parametrizadas ofrecen una importante **ventaja teórica**:

- La **parametrización de la política** supone **cambios suaves** en las probabilidades de las acciones durante el aprendizaje.
- Por el contrario, la **aproximación de funciones de valor + ε-*greedy***, hace que las probabilidades de las acciones cambien de forma **drástica** dada una variación menor en el valor estimado de pares acciones-estado.

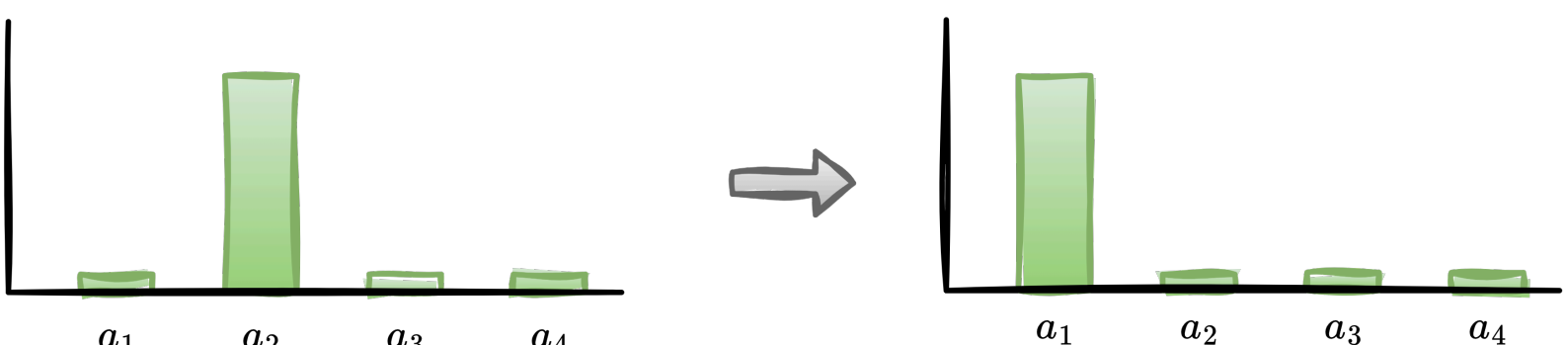
Esto será especialmente relevante a la hora de aplicar **ascenso del gradiente** para optimizar θ.

Actualización de la política

Gradiente de la política



ϵ-greedy



Teorema del gradiente de la política

Rendimiento de una política

Al contrario que en los métodos basados en funciones de valor, donde buscábamos minimizar un error de estimación, en los métodos basados en políticas buscamos **maximizar el rendimiento**.

- Una **política óptima** será aquella que maximice $v_{\pi_{\boldsymbol{\theta}}}$, es decir, la función estado--valor real de $\pi_{\boldsymbol{\theta}}$.
- En el caso **episódico**, nuestra métrica de **rendimiento** puede expresarse como:

$$\begin{aligned} J(\boldsymbol{\theta}) &= v_{\pi_{\boldsymbol{\theta}}}(s_0) \\ &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[G_t \mid S_t = s_0] \\ &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{k=0}^{\infty} R_{t+k+1} \mid S_t = s_0\right] \end{aligned}$$

Rendimiento de una política

$$J(\boldsymbol{\theta}) = v_{\pi_{\boldsymbol{\theta}}}(s_0)$$

Esta formulación representa el rendimiento como la expectación de la recompensa total obtenida a partir de un estado inicial s_0.

- También puede expresarse como la recompensa esperada dada una trayectoria τ:

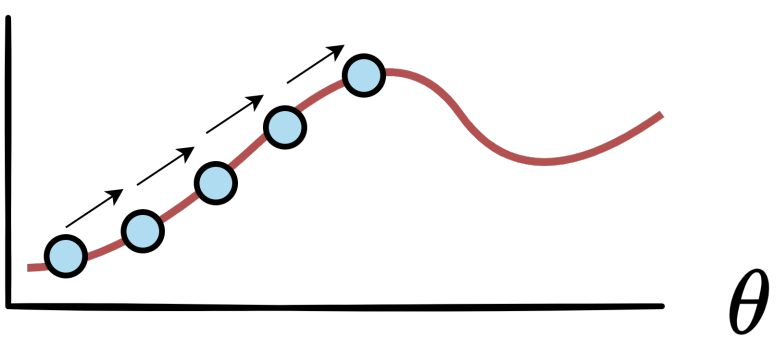
$$J(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\tau)]$$

Rendimiento de una política

La política parametrizada por θ se optimiza a medida que los valores de θ dan lugar a un mejor rendimiento.

- Actualización basada en **ascenso del gradiente**.

$$\pi_\theta \Rightarrow J(\theta)$$



$$\theta_{t+1} = \theta_t + \alpha \widehat{\nabla J(\theta_t)}$$

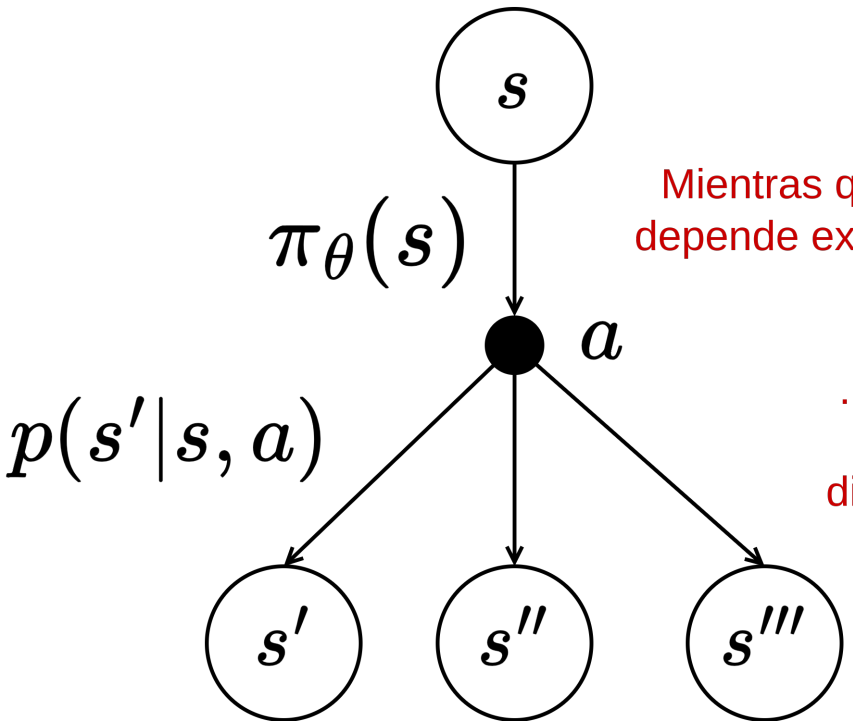
Optimizar de $J(\boldsymbol{\theta})$ requiere calcular el gradiente $\nabla_{\theta} J(\boldsymbol{\theta})$

Debemos tener en cuenta que el rendimiento depende de:

1. Las **acciones elegidas** $\rightarrow$ directamente determinadas por $\pi_{\boldsymbol{\theta}}$
2. La **secuencia de estados visitados** $\rightarrow$ influenciada tanto por $\pi_{\boldsymbol{\theta}}$ como por las **dinámicas del entorno** $p(s'|s,a)$.

- Nuestro problema es que las dinámicas del entorno **están fuera de la capacidad de control del agente**.

Gradiente $J(\theta)$



Mientras que la acción elegida depende exclusivamente de π_θ...

...el siguiente estado depende de las dinámicas del entorno

Gradiente de $J(\theta)$

- Dado un estado, es relativamente sencillo calcular cómo afectan los parámetros de la política a la **elección de la acción** y, por tanto, a la recompensa obtenida.
- Sin embargo, el efecto de la política sobre la **distribución de estados** (es decir, qué estados se visitarán) es algo que también depende del entorno y es, generalmente, desconocido.[1]

> ❓ ¿Cómo podemos estimar la dirección del gradiente y **mejorar la política** cuando parte de ese gradiente depende de un **efecto desconocido** sobre la distribución de estados?

[1] Andriy Drozdyuk expresa esto con el ejempo: «las dinámicas de una roca no dependen de quién la levanta».

Teorema del gradiente de la política

La solución se encuentra en el **teorema del gradiente de la política** (*policy gradient theorem*).

- Gradiente de $J(\boldsymbol{\theta})$ en problemas episódicos:

$$\nabla_\theta J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a|s,\boldsymbol{\theta})$$

Éste ofrece una definición del gradiente del rendimiento con respecto a θ **sin necesidad de tener en cuenta la derivada de la distribución de estados visitados** empleando $\pi_{\boldsymbol{\theta}}$.

- Es decir, calcular $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ sin necesidad de calcular $\sum_s \nabla_{\boldsymbol{\theta}} \mu(s)$.

Teorema del gradiente de la política

$$\nabla_\theta J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a|s,\boldsymbol{\theta})$$

- Los gradientes son vectores columna de las derivadas parciales con respecto a los componentes de $\boldsymbol{\theta}$.
- π es la política asociada al vector de parámetros $\boldsymbol{\theta}$.
- $\propto$ denota propocionalidad. En problemas **episódicos**, la constante de proporcionalidad es la longitud media del episodio, mientras que en problemas **continuados** es 1 (la relación pasa a ser de igualdad).
- La distribución μ es la distribución *on-policy* bajo π_θ. Representa la fracción de tiempo invertido en un determinado estado.

Teorema del gradiente de la política

¿Qué nos indica el teorema del gradiente de la política?

$$\nabla_\theta J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a|s,\boldsymbol{\theta})$$

1. Que la actualización de $\boldsymbol{\theta}$ (es decir, el gradiente de $J(\boldsymbol{\theta})$) es proporcional a la suma de los gradientes de la política ponderados por los valores de acción.
2. Que no se requiere calcular ningún gradiente relacionado con las dinámicas del entorno (no se depende de $\nabla \mu(s)$).

Teorema del gradiente de la política

$$\nabla_\theta J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \underbrace{\sum_a q_{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a|s,\boldsymbol{\theta})}_{\begin{array}{c}\text{Mejorar el rendimiento} \\ \text{de la política implica} \\ \text{ajustar los parámetros} \\ \text{de tal forma que las acciones} \\ \text{más valiosas sean más probables}\end{array}}$$

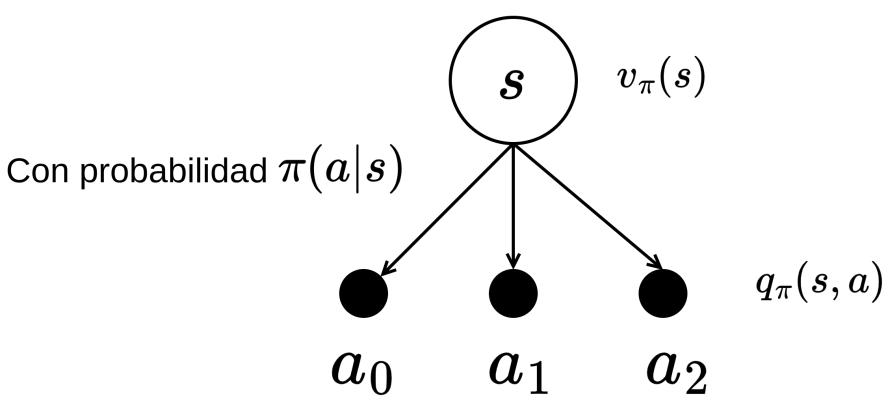
Demostración del teorema

Demostración del teorema del grandiente de la política

⚠️ Para simplificar la notación, de aquí en adelante expresaremos $\nabla_{\boldsymbol{\theta}}$ como ∇, y $\pi_{\boldsymbol{\theta}}$ como π.

Partimos de la relación entre $v_\pi(s)$ y $q_\pi(s,a)$:

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi[q_\pi(s,a)] \\
&= \underbrace{\sum_a \pi(a|s) q_\pi(s,a)}_{\substack{\text{Suma ponderada/} \\ \text{valor esperado}}}
\end{aligned}
$$



Demostración del teorema del grandiente de la política

Si incluimos el gradiente, tenemos:

$$\nabla v_\pi(s) = \nabla \left[\sum_a \pi(a|s) q_\pi(s,a) \right] \quad \forall s \in \mathcal{S}$$

- Estamos expresando el gradiente de la **función de valor de estado** en términos de la **función de valor acción-estado**.

Demostración del teorema del grandiente de la política

Si aplicamos la regla del producto de gradientes, obtenemos:

$$\begin{aligned}\nabla v_\pi(s) &= \nabla \left[\sum_a \pi(a|s) q_\pi(s,a) \right] \\ &= \sum_a [\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla q_\pi(s,a)]\end{aligned}$$

- Regla del producto de gradientes:

$$\nabla(f(x)g(x)) = \nabla f(x) g(x) + f(x) \nabla g(x)$$

- El sumatorio se puede extraer: $\nabla \sum f(x) = \sum \nabla f(x)$.

Demostración del teorema del grandiente de la política

Expandimos el siguiente término:

$$\nabla v_\pi(s) = \sum_a [\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla q_\pi(s,a)]$$

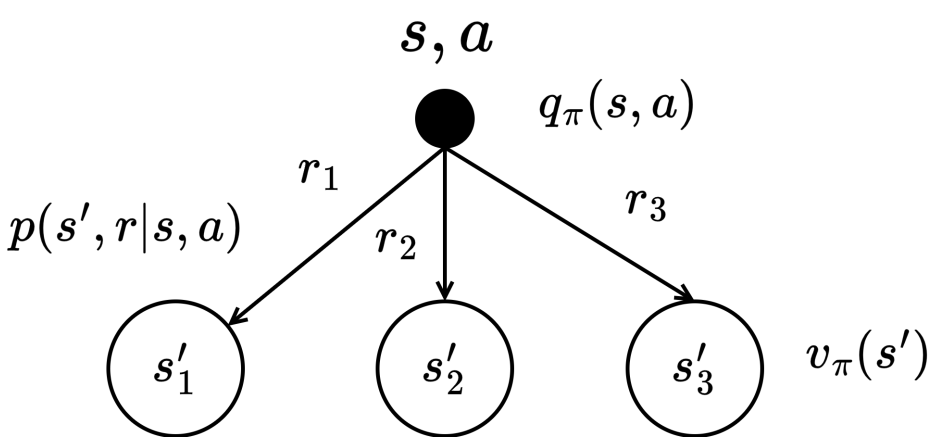
Y obtenemos:

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

Demostración del teorema del grandiente de la política

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

El valor $q(s,a)$ se expresa en términos de recompensa esperada.



Demostración del teorema del grandiente de la política

A continuación, buscamos eliminar las recompensas de $\sum_{s',r}$

Para ello, tratamos de volver a comprimir esta parte de la ecuación:

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

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

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

Demostración del teorema del grandiente de la política

- Simplificamos $p(s', r|s, a)$ como ρ.

Aplicando la regla del producto tenemos:

$$\nabla v_\pi(s) = \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla \sum_{s'} \sum_r \rho r + \rho v_\pi(s') \right]$$

$$= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} \sum_r \nabla \rho r + \rho \nabla r + \nabla \rho v_\pi(s') + \rho \nabla v_\pi(s') \right]$$

Demostración del teorema del grandiente de la política

Los términos con gradiente 0 con respecto a θ se anulan:

$$\nabla v_\pi(s) = \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} \sum_r \cancel{\nabla \rho r} + \cancel{\rho \nabla r} + \cancel{\nabla \rho v_\pi(s')} + \rho \nabla v_\pi(s') \right]$$

- Ya que $\nabla_{\boldsymbol{\theta}} r = 0$ y $\nabla_{\boldsymbol{\theta}} p(s', r|s, a) = 0$

Finalmente obtenemos:

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

Demostración del teorema del grandiente de la política

Si reorganizamos la ecuación:

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

- Para eliminar $\sum_r$, reducimos $\sum_r p(s', r|s, a)$ (expectación en términos de recompensa) a una expresión más general:
- Dado que: $p(s'|s,a) = \Pr\{S_t = s' \mid S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in \mathcal{R}} p(s', r|s, a)$

Demostración del teorema del grandiente de la política

De esta forma, tenemos:

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

Demostración del teorema del grandiente de la política

Ahora, sustituimos $\nabla v_\pi(s')$ por su definición (*unrolling*):

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

$$= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p(s'|s,a) \right.$$

$$\left. \sum_{a'} \nabla \pi(a'|s') q_\pi(s',a') + \pi(a'|s') \sum_{s''} p(s''|s',a') \nabla v_\pi(s'') \right]$$

Podemos aplicar este *unrolling* de forma continuada.

Demostración del teorema del grandiente de la política

Finalmente, resumimos la expresión obtenida como:

$$\nabla v_\pi(s) = \sum_{x \in \mathcal{S}} \sum_{k=0}^{\infty} \Pr(s \to x, k, \pi) \sum_a \nabla \pi(a|x) q_\pi(x, a)$$

- $\Pr(s \to x, k, \pi)$ es la probabilidad de transición desde el estado s al estado x en k *timesteps* bajo la política π.

Demostración del teorema del grandiente de la política

Bajo estas premisas, podemos demostrar el **teorema del gradiente de la política**.

Si partimos de:

$$\nabla J(\boldsymbol{\theta}) = \nabla v_\pi(s_0)$$

...y sustituimos por la fórmula obtenida previamente, tenemos que:

$$\nabla J(\boldsymbol{\theta}) = \sum_s \left(\sum_{k=0}^{\infty} \Pr(s_0 \to s, k, \pi) \right) \sum_a \nabla \pi(a|s) q_\pi(s, a)$$

Demostración del teorema del grandiente de la política

Comprimimos la probabilidad de transición en el término $\eta(s)$ y normalizamos para que sea una distribución de probabilidad:

$$\begin{aligned}
\nabla J(\boldsymbol{\theta}) &= \sum_s \left(\sum_{k=0}^{\infty} \Pr(s_0 \to s, k, \pi) \right) \sum_a \nabla \pi(a|s) q_\pi(s,a) \\
&= \sum_s \eta(s) \sum_a \nabla \pi(a|s) q_\pi(s,a) \\
&= \left(\sum_{s'} \eta(s') \right) \sum_s \frac{\eta(s)}{\sum_{s'} \eta(s')} \sum_a \nabla \pi(a|s) q_\pi(s,a)
\end{aligned}$$

Demostración del teorema del grandiente de la política

Dado que $\sum_{s'} \eta(s')$ es una constante:

$$\nabla J(\boldsymbol{\theta}) = \left(\sum_{s'} \eta(s') \right) \sum_{s} \frac{\eta(s)}{\sum_{s'} \eta(s')} \sum_{a} \nabla \pi(a|s) q_{\pi}(s,a)$$

$$\propto \sum_{s} \frac{\eta(s)}{\sum_{s'} \eta(s')} \sum_{a} \nabla \pi(a|s) q_{\pi}(s,a)$$

Demostración del teorema del grandiente de la política

- La distribución $\mu(s) = \frac{\eta(s)}{\sum_{s'} \eta(s')}$ es la distribución *on-policy* bajo π (tiempo invertido en el estado s bajo la política π).

Teniendo esto en cuenta, obtenemos que:

$$\nabla J(\boldsymbol{\theta}) \propto \sum_{s} \mu(s) \sum_{a} \nabla \pi(a|s) q_{\pi}(s, a)$$

Esta expresión permite obtener el gradiente del rendimiento con respecto a los parámetros de la política (es decir, $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$) sin necesidad de calcular la derivada de la distribución de estados.

Demostración del teorema del grandiente de la política

Finalmente, podemos reordenar la expresión y escribirla en términos de **valor esperado**, obteniendo:

$$\nabla J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s)$$

$$= \mathbb{E}_\pi \left[\sum_a q_\pi(S_t, a) \nabla \pi(a|S_t) \right]$$

Demostración del teorema del grandiente de la política

$$\nabla J(\boldsymbol{\theta}) \propto \mathbb{E}_{\pi}\left[\sum_{a} q_{\pi}(S_t, a)\nabla\pi(a|S_t)\right]$$

Teorema del gradiente de la política

El gradiente del rendimiento con respecto a los parámetros de la política $\nabla_{\boldsymbol{\theta}} J(\theta)$ es proporcional al **valor esperado de los gradientes de la política** $\pi_{\boldsymbol{\theta}}$ ponderados por los **valores de acción** $q_{\pi_{\boldsymbol{\theta}}}(s, a)$.

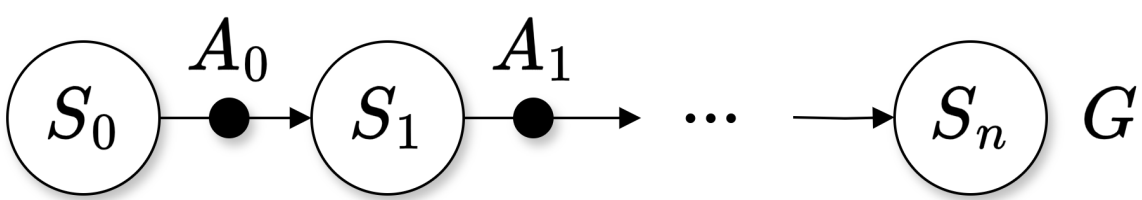
Demostración del teorema del grandiente de la política

Una vez introducido el teorema, veamos cómo aplicarlo para obtener **políticas óptimas**...

REINFORCE

REINFORCE[2], o ***gradiente de la política Monte Carlo***, es un algoritmo clásico basado en el gradiente de la política[3].

- Es de tipo **Monte Carlo**, ya que actualiza los parámetros de la política basándose en el retorno obtenido al final de cada episodio.



[2] ***RE*** *eward* ***I*** *ncrement =* ***N*** *onnegative* ***F*** *actor* $\times$ ***O*** *ffset* ***R*** *einforcement* $\times$ ***C*** *haracteristic* ***E*** *ligibility*

[3] 📚 *Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8, 229-256.*

Partimos de la formulación del **teorema del gradiente de la política**:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}\left[\sum_{a} \nabla \pi(a|S_t, \boldsymbol{\theta}) q_{\pi}(S_t, a)\right]$$

Multiplicamos y dividimos por $\pi(a|S_t, \boldsymbol{\theta})$:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}\left[\sum_{a} \nabla \pi(a|S_t, \boldsymbol{\theta}) q_{\pi}(S_t, a) \frac{\pi(a|S_t, \boldsymbol{\theta})}{\pi(a|S_t, \boldsymbol{\theta})}\right]$$

Si reorganizamos la expresión, tenemos:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}\left[\sum_{a} \pi(a|S_t, \boldsymbol{\theta}) q_{\pi}(S_t, a) \frac{\nabla \pi(a|S_t, \boldsymbol{\theta})}{\pi(a|S_t, \boldsymbol{\theta})}\right]$$

Después, reemplazamos a por $A_t \sim \pi$:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}\left[q_{\pi}(S_t, A_t) \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta})}{\pi(A_t|S_t, \boldsymbol{\theta})}\right]$$

Dado que $q_\pi(S_t, A_t) = \mathbb{E}_\pi[G_t \mid S_t, A_t]$, finalmente obtenemos:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_\pi \left[G_t \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta})}{\pi(A_t|S_t, \boldsymbol{\theta})} \right]$$

Donde:

- G_t es el retorno obtenido[4].
- La expresión $G_t \frac{\nabla \pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta})}$ es una cantidad que puede ser muestreada en cada *timestep*, cuyo valor esperado es igual al gradiente.

[4] **REINFORCE es un algoritmo de tipo Monte Carlo** porque actualiza los parámetros de la política basándose en el retorno obtenido al final de cada episodio.

REINFORCE: regla de actualización

A partir de esta fórmula obtenemos la **regla de actualización** de θ empleada por REINFORCE:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha G_t \frac{\nabla \pi(A_t|S_t, \boldsymbol{\theta}_t)}{\pi(A_t|S_t, \boldsymbol{\theta}_t)}$$

Su interpretación es intuitiva:

- Cada incremento de θ es proporcional a el **retorno acumulado** G_t hasta el instante t, multiplicado por el **gradiente de la probabilidad de realizar** A_t dividido por la **probabilidad de realizar** A_t.
- El resultado es un **vector** que representa la dirección en el espacio de parámetros que más incrementa la probabilidad de repetir A_t en futuras visitas a S_t.

REINFORCE: regla de actualización

👍 Si una acción lleva a una alta recompensa acumulada, su probabilidad aumentará en futuros epsiodios.

👎 Si el retorno obtenido es bajo, la acción se tomará con menor frecuencia.

🦾 Esto **refuerza** acciones buenas y reduce la probabilidad de realizar acciones potencialmente peores.

Además, **la actualización es proporcional a la probabilidad**:

- **Acciones que ya eran probables no cambian tanto**. Esto evita que acciones seleccionadas con más frecuencia no tengan una ventaja injusta, ya que podrían predominar sin generar altos retornos.
- **Acciones que eran improbables** pero resultaron en un alto G_t, se ven reforzadas significativamente.

REINFORCE: regla de actualización compacta

Una **forma alternativa** de expresar la regla de actualización de REINFORCE es la siguiente:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha G_t \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta}_t)$$

La simplificación propuesta se debe a la igualdad:

$$\nabla \ln x = \frac{\nabla x}{x}$$

Esto permite una representación más compacta y facilita el cálculo del gradiente.

- Al vector $\nabla \ln \pi(A_t|S_t, \boldsymbol{\theta}_t)$ se le suele denominar **vector de elegiblidad** (*eligibility vector*).

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

- Se incluye el *factor de descuento* γ para problemas con retorno descontado.

Convergencia y limitaciones de REINFORCE

✅ REINFORCE ofrece garantías de **convergencia** en óptimos locales en condiciones estocásticas de aproximación y descenso continuo de α.

❌ No obstante, al tratarse de un método de tipo Monte Carlo, presenta un **aprendizaje lento** y una **alta varianza**.

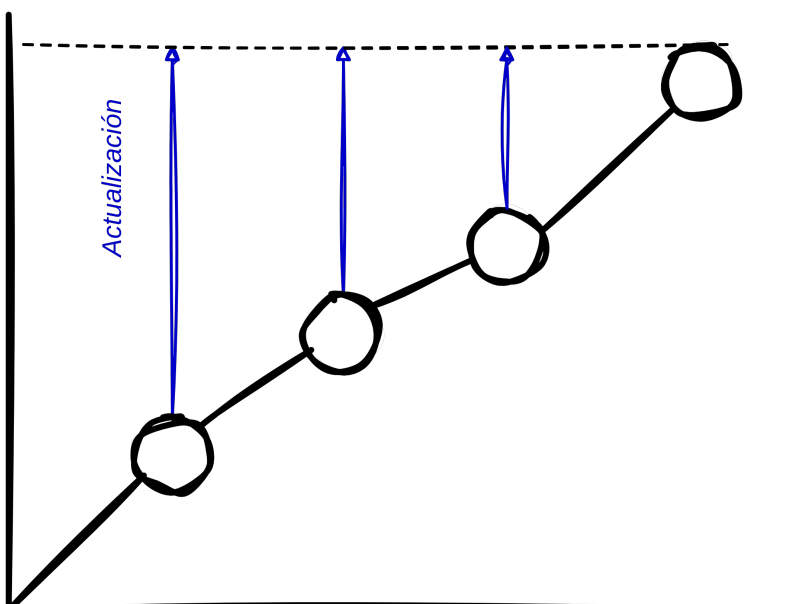
Para abordar estas limitaciones, se proponen las variaciones de REINFORCE que estudiaremos a continuación.

Valores de referencia

Problemas de REINFORCE

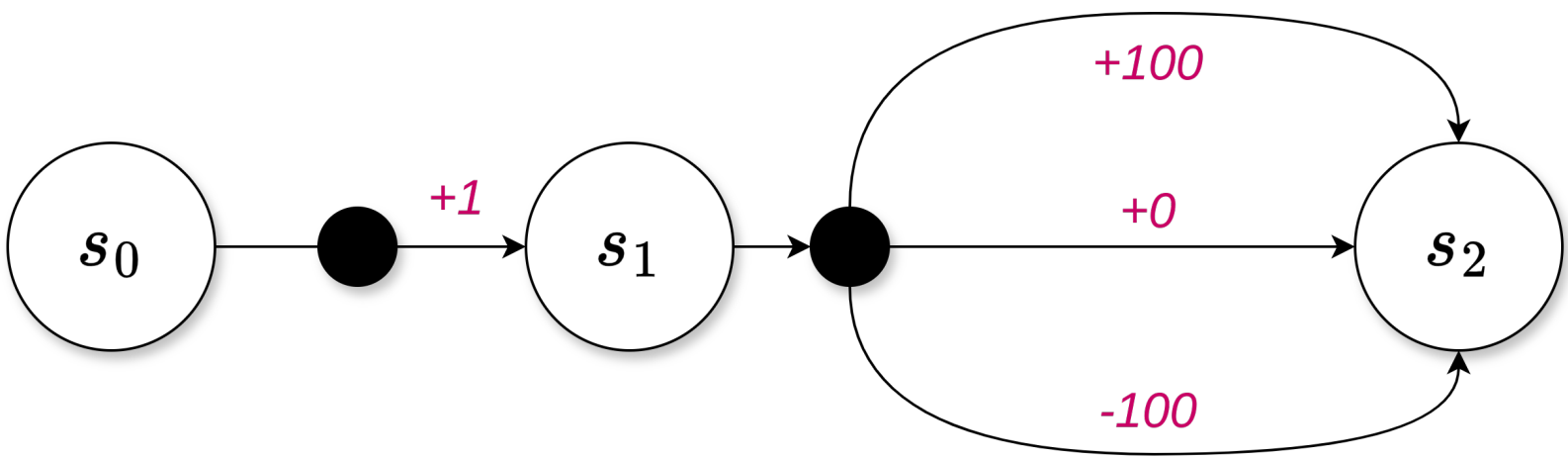
Los algoritmos que emplean el **retorno episódico** G, como **REINFORCE**, tienden a presentar una **alta varianza**.

- En este caso, la varianza se da en la actualización de los parámetros de la política, θ.
- El gradiente se calcula a partir de la recompensa acumulada en un episodio completo, lo que introduce **grandes fluctuaciones** en la actualización de los parámetros.



Problemas de REINFORCE

- Además, la **variabilidad entre episodios** puede ser significativa, dificultando la convergencia del aprendizaje.



➡️ La **acumulación de eventos aleatorios** a lo largo de las trayectorias —incluyendo la **aleatoriedad del estado inicial**— dan lugar a un retorno G que puede variar significativamente de un episodio a otro.

REINFORCE con *baseline*

El teorema del gradiente de la política permite incluir una comparación del valor G con un **valor de referencia** (***baseline***) arbitrario: $b(s)$.

- El **valor de referencia** $b(s)$ puede ser cualquier valor o función, siempre que no dependa de la acción tomada a.

De esta forma, tenemos que:

$$\nabla J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a (G - b(s)) \nabla \pi(a|s, \boldsymbol{\theta})$$

- Dicho valor no afecta al gradiente, simplemente es una expectativa del valor real.
- Si este valor se aproxima mínimamente al valor real de s, la varianza de la actualización de los parámetros se verá reducida.

Regla de actualización de REINFORCE con *baseline*

De esta forma, tenemos el algoritmo llamado **REINFORCE con *baseline***, también denominado ***Vanilla Policy Gradient*** (**VPG**), que emplea la siguiente **regla de actualización**:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(G_t - b(S_t))\nabla \ln \pi(A_t|S_t, \boldsymbol{\theta}_t)$$

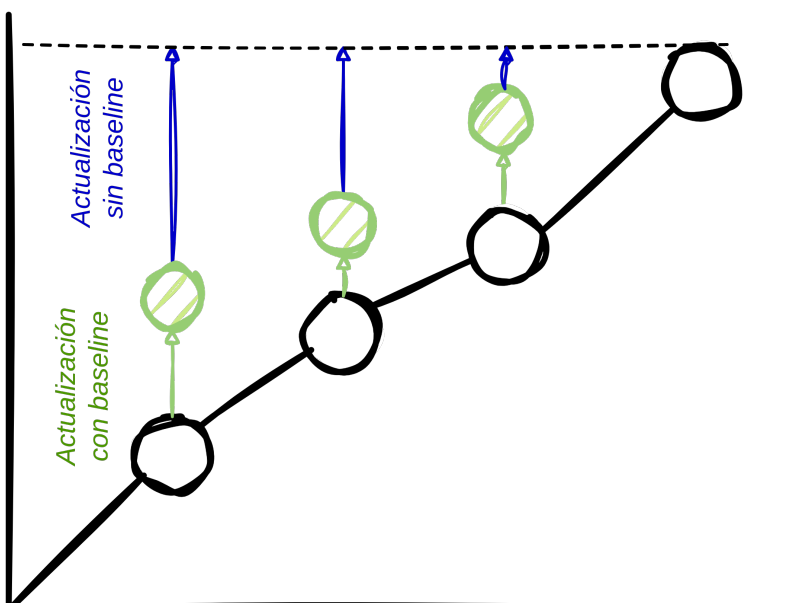
- Se trata de un caso más general que REINFORCE sin *baseline*, ya que si $b(s) = 0$, obtenemos la regla de actualización vista anteriormente.

REINFORCE con *baseline*

Emplear un ***baseline*** permite incorporar una **estimación** del valor real que facilite la actualización de θ.

- La **dirección** del gradiente se mantiene, ya que la resta de $b(s)$ no afecta a la dirección de la actualización.
- La **convergencia** del algoritmo se ve favorecida, ya que la **varianza** de la actualización de los parámetros se reduce.

$$\theta_{t+1} = \theta_t + \alpha(G_t - b(S_t))\nabla \ln \pi(A_t|S_t, \theta_t)$$



¿Qué *baseline* utilizar?

Elección del *baseline*

Es común emplear como *baseline* la estimación del **estado-valor**, $\hat{v}(S_t, \boldsymbol{w})$.

- Dado que que REINFORCE está basado en Monte Carlo, los pesos $\boldsymbol{w}$ pueden estimarse **de la misma manera**, tal y como vimos en el tema anterior.

De esta forma, REINFORCE empleando $\hat{v}$ como *baseline* se resume en:

1. Generar episodio: $S_0, A_0, R_1, ..., S_{T-1}, A_{T-1}, R_T$ siguiendo $\pi_{\boldsymbol{\theta}}$
2. Para cada *timestep* $t = 0, 1, ..., T-1$ del episodio:
   - Calcular $G = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
   - Restar el *baseline*: $\delta = G - \hat{v}(S_t, \boldsymbol{w})$
   - Actualizar $\boldsymbol{w}, \boldsymbol{\theta}$

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
    $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

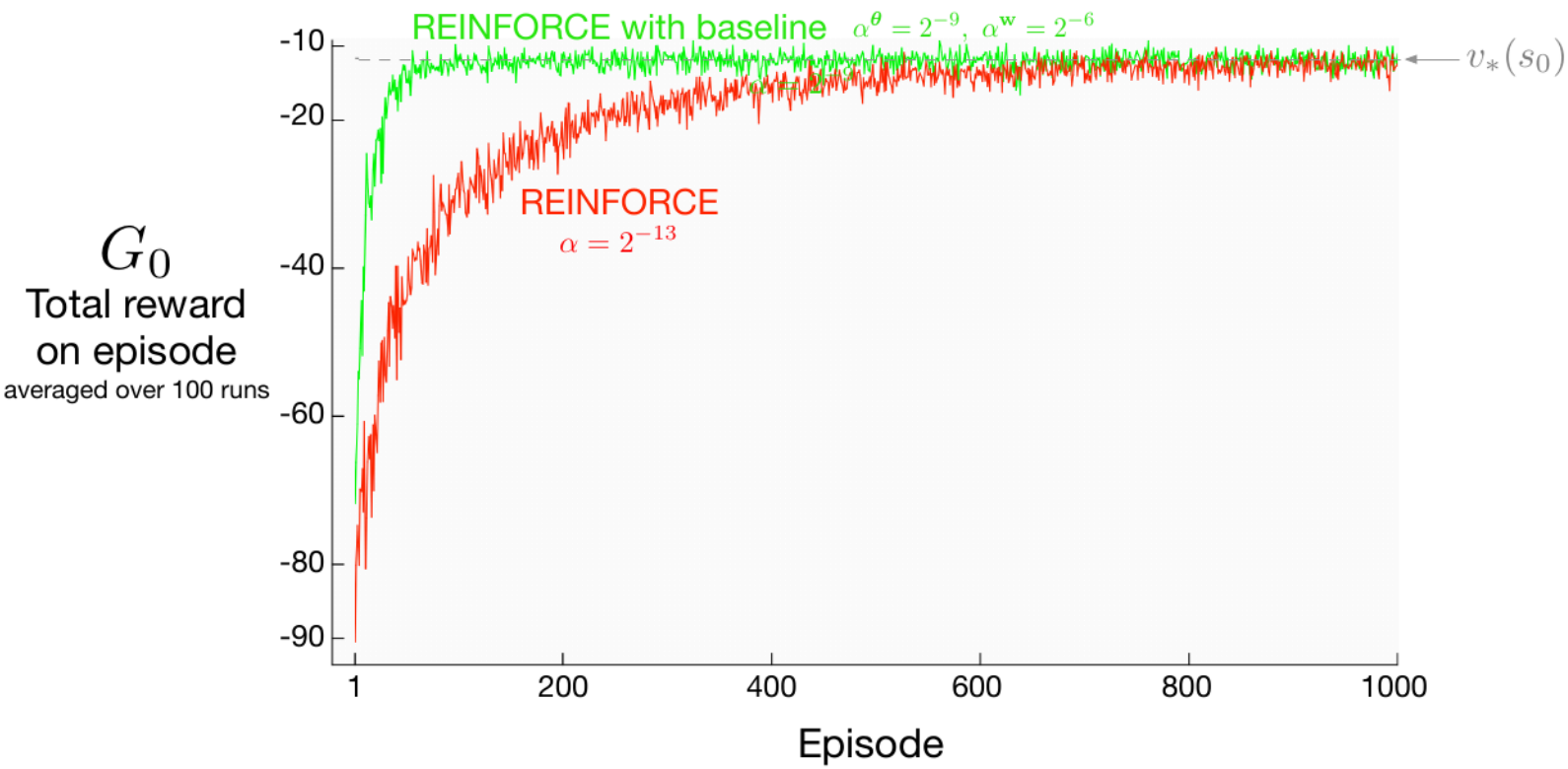


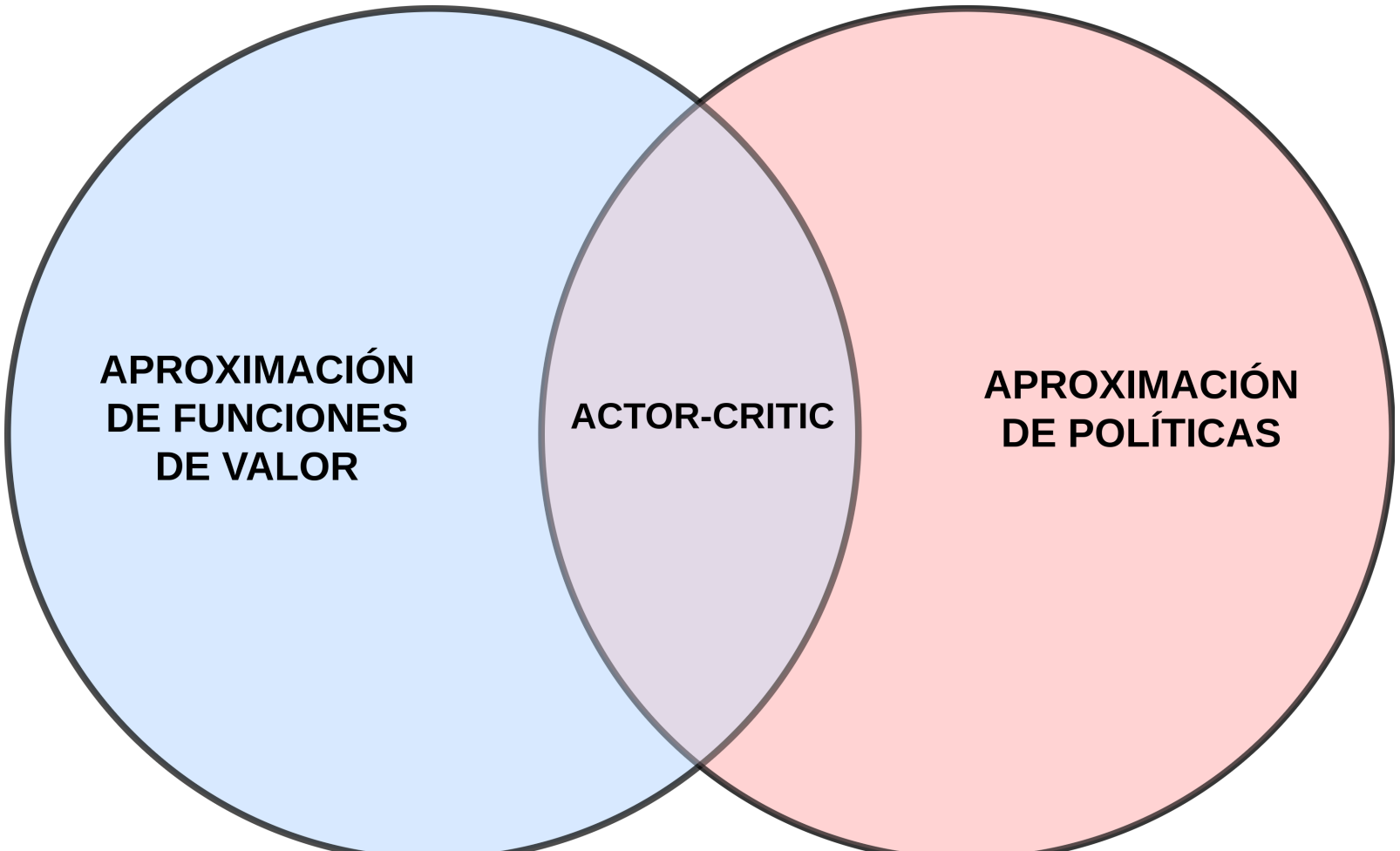
Figure 13.2: Adding a baseline to REINFORCE can make it learn much faster, as illustrated here on the short-corridor gridworld (Example 13.1). The step size used here for plain REINFORCE is that at which it performs best (to the nearest power of two; see Figure 13.1).

Conclusiones

- En resumen, **REINFORCE con *baseline***, o **VPG**, es un método robusto basado en el teorema del gradiente de la política.
- Decimos que es un método ***no-sesgado***, porque la actualización de θ se basa en **retornos episódicos** G, obtenidos directamente a partir de la interacción con el entorno.
- El único **sesgo** que presentan se encuentran la **función de valor aproximada** $\hat{v}$ que empleamos como ***baseline***, pero este sesgo es prácticamente nulo.

Métodos *actor-critic*

Métodos *actor-critic*



Todo algoritmo *actor-critic* está compuesto por dos funciones con sus propios conjuntos de parámetros $(\boldsymbol{\theta}, \boldsymbol{w})$:

Actor

 Aprende la política $\pi(A_t \mid S_t, \boldsymbol{\theta})$ y muestrea las acciones a realizar.

***Critic* (*crítico*)**

 Aprende una función de valor $\hat{v}(S_t, \boldsymbol{w})$, $\hat{q}(S_t, A_t, \boldsymbol{w})$ y estima cómo de buenas son las acciones realizadas por el actor.

Si REINFORCE con *baseline* aprende una **política** π_θ y utiliza una **función de valor aproximada** $\hat{v}(S_t, w)$ como *baseline*... 🤔

¿Por qué no se considera ***actor-critic***?

- **REINFORCE con *baseline*** es un algoritmo de tipo Monte Carlo, ya que requiere esperar **final** del episodio para evaluar el error.

$$\delta = G - \hat{v}(S_t, \boldsymbol{w})$$

- **Actor-critic** permite evaluar el error de forma online, convirtiéndolo en un algoritmo basado en TD, es decir, *bootstrapping*.

$$\delta = R_{t+1} + \gamma \hat{v}(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w})$$

Métodos *actor-critic*

Los métodos ***actor-critic*** son una extensión de REINFORCE con *baseline* que, en lugar de usar el retorno G, emplean ***bootstrapping***.

- Es decir, en vez de esperar al final del episodio y utilizar el retorno real G para actualizar θ, se emplea una **estimación** de este.
- Es análogo a TD, Sarsa o *Q-learning*, donde sustituimos el retorno de episodios completos por **estimaciones *n* pasos hacia delante** desde el estado actual.

One-step actor-critic

El ejemplo más básico de este tipo de algoritmos es ***one-step actor-critic***, donde se sustituye el retorno G_t (*full return*) por el *one-step return*:

$$R_{t+1} + \gamma \hat{v}(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w})$$

- De esta forma, la actualización de $\boldsymbol{\theta}$ es la siguiente:

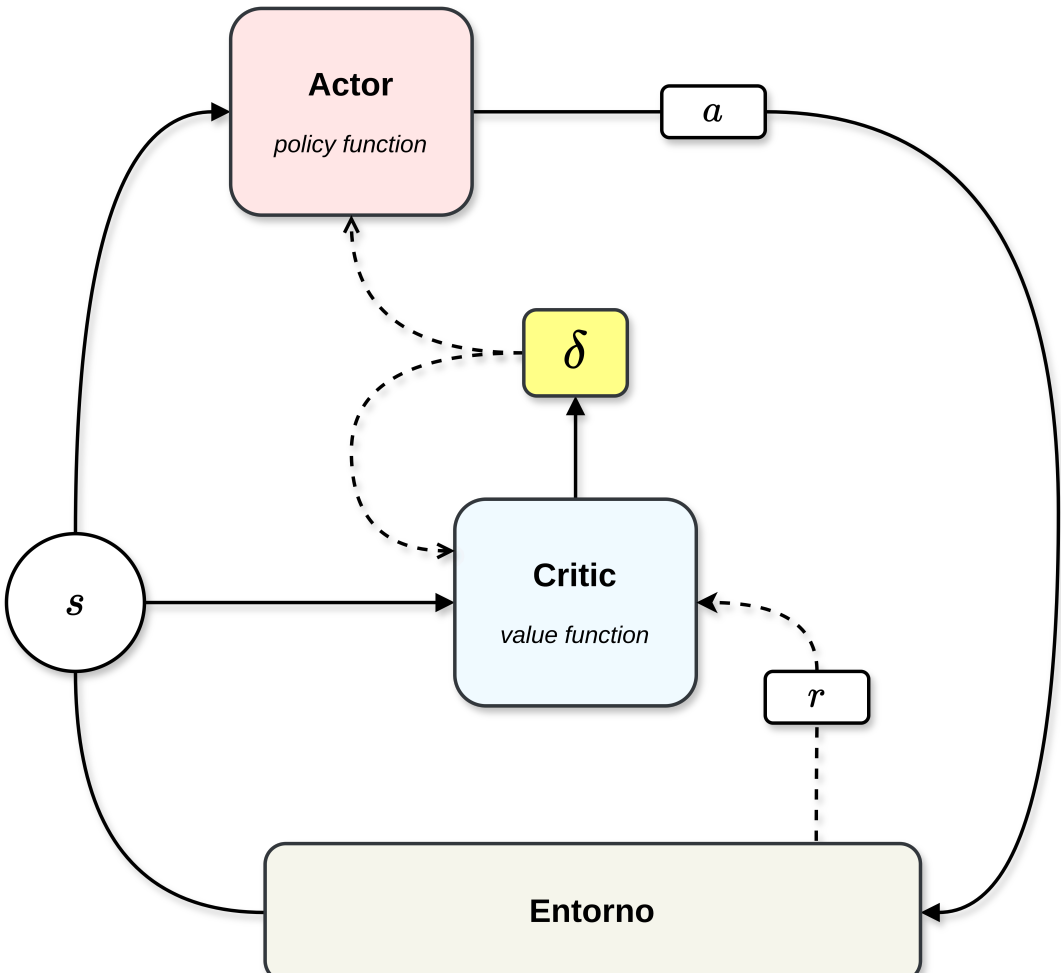
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(R_{t+1} + \gamma \hat{v}(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w}) \nabla \ln \pi(A_t | S_t, \boldsymbol{\theta}_t)$$

One-step actor-critic

One-step Actor–Critic (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s,\boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)
Loop forever (for each episode):
  Initialize S (first state of episode)
  $I \leftarrow 1$
  Loop while S is not terminal (for each time step):
    $A \sim \pi(\cdot|S,\boldsymbol{\theta})$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} I \delta \nabla \ln \pi(A|S,\boldsymbol{\theta})$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$

Arquitectura *actor-critic*



Como hemos visto, las reglas de actualización para actor ($\boldsymbol{\theta}$) y crítico ($\boldsymbol{w}$) son:

Actualización del actor

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_{\boldsymbol{\theta}}[R_t + \gamma \hat{v}_\pi(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w})] \nabla \ln(\pi(A_t \mid S_t, \boldsymbol{\theta}))$$

Actualización del crítico

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha_{\boldsymbol{w}}[R_t + \gamma \hat{v}_\pi(S_{t+1}, \boldsymbol{w}) - \hat{v}(S_t, \boldsymbol{w})] \nabla \hat{v}_\pi(S_t, \boldsymbol{w})$$

- A diferencia de REINFORCE con *baseline*, se utiliza el *TD target* en lugar del retorno.

⚠ **IMPORTANTE: Estas actualizaciones son similares si empleamos la función** $q(s, a)$.

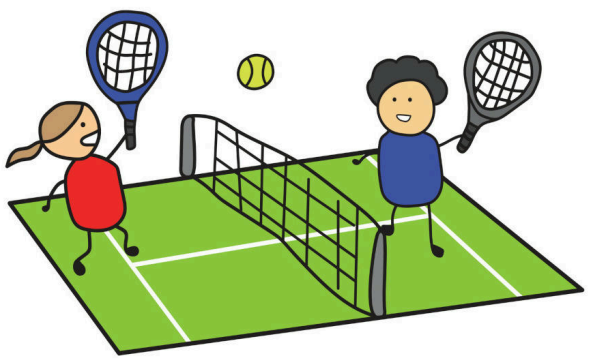
Ventajas de *actor-critic*

- Los métodos *actor-critic* permiten **actualizaciones más frecuentes** de los parámetros de la política, ya que no es necesario esperar al final del episodio para obtener el retorno.
  - De hecho, podemos ajustar al nivel de *bootstrapping* a emplear (número de *steps* hacia delante).
- Mientras que se aumenta el **sesgo** (*bias*), la **varianza** de la actualización de los parámetros se reduce, ya que se emplea una estimación del retorno en lugar del retorno real.
- Esto da lugar a una velocidad **convergencia** mayor, y favorece la aplicación en **problemas continuados**.

Una posible analogía de los métodos vistos...

REINFORCE

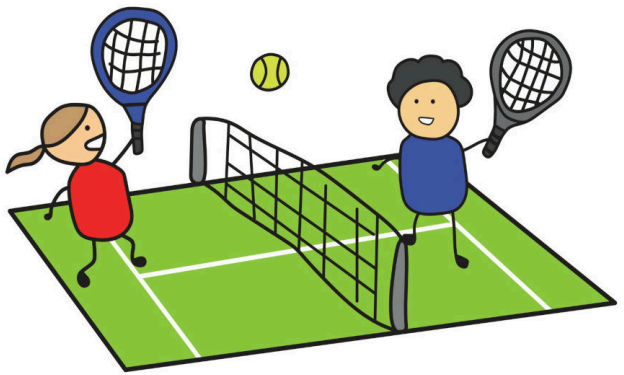
- Juego un partido de tenis contra un amigo.
- **Al final del partido**, analizo cómo he jugado y pienso en qué he de mejorar.



Mejoro lentamente, y mi forma de jugar varía abruptamente hasta dar con la mejor forma de ganar a mi rival.

REINFORCE con *baseline*

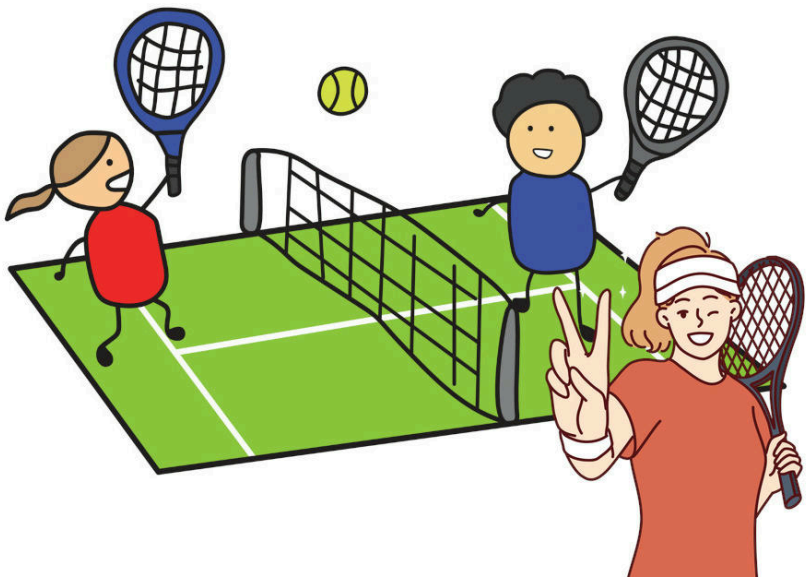
- Juego un partido de tenis contra un amigo.
- **Al final del partido**, analizo cómo he jugado y pienso en qué he de mejorar, teniendo como referencia los consejos de mi **entrenador/a**.



Mi entrenador/a me ofrece buenos consejos sobre cómo poder superar a mi rival.

Actor-critic

- Juego un partido de tenis contra un amigo.
- **Tras cada punto**, analizo cómo he jugado y pienso en qué he de mejorar, teniendo como referencia **los consejos de mi entrenador/a**.



Mi entrenador/a me ofrece consejos sobre cómo poder superar a mi rival con mucha más frecuencia, en base a cómo estamos jugando actualmente.

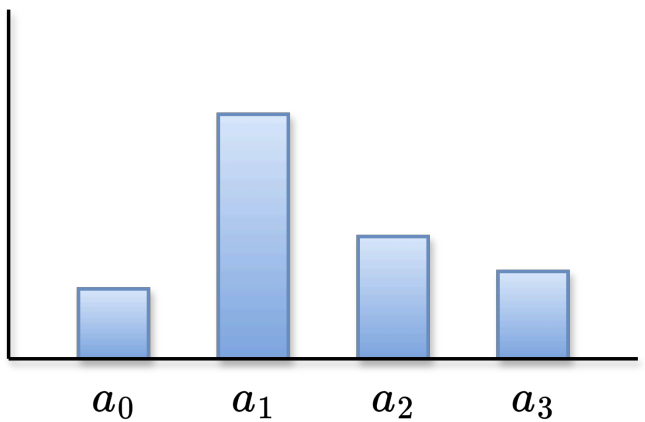
Espacios de acciones continuos

Espacios de acciones continuos

Hasta ahora hemos visto cómo aplicar los métodos basados en *gradient de la política* empleando **espacios de acciones discretos**.

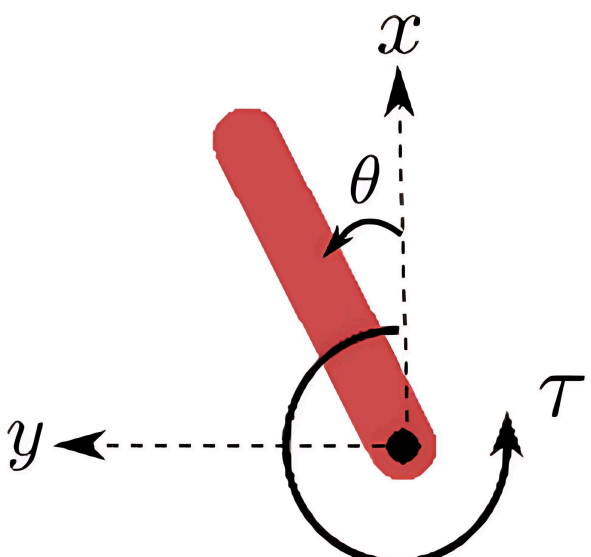
- Es decir, el conjunto de acciones es finito y cada acción tiene una probabilidad asociada.

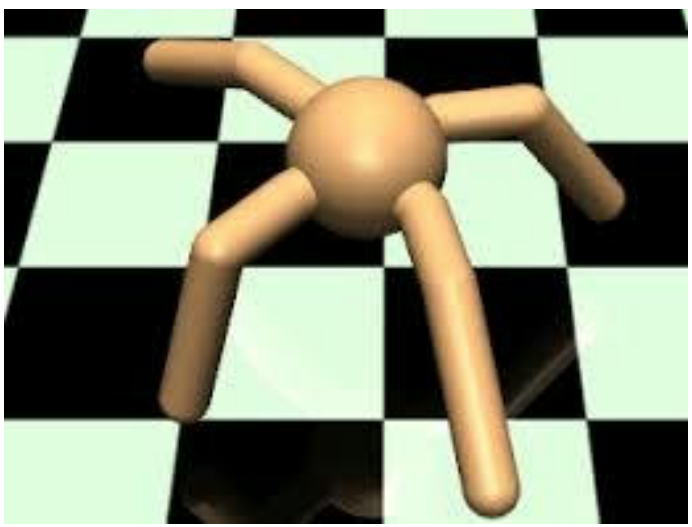
¿Pero qué ocurre si abordamos un problema con **espacio de acciones continuo** / infinito?



Este tipo de espacios de acciones son comunes en problemas de robótica, control de vehículos, etc.

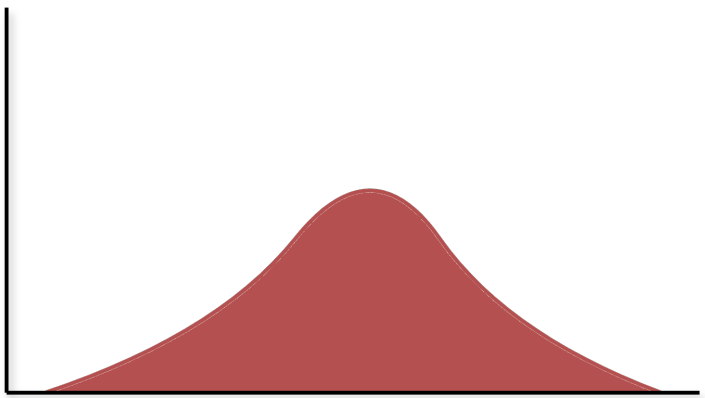
Espacios de acciones continuos





Espacios de acciones continuos

La forma de abordar estos problemas es sustituyendo la distribución categórica de acciones por una **distribución continua** (ej. gaussiana).



- La política se modela como una distribución de densidad de probabilidad. Por ejemplo, una **distribución normal**:

$$\pi(a|s,\boldsymbol{\theta}) = \mathcal{N}(a;\mu(s,\boldsymbol{\theta}),\sigma(s,\boldsymbol{\theta}))$$

Espacios de acciones continuos

$$\pi(a|s, \boldsymbol{\theta}) = \mathcal{N}(a; \mu(s, \boldsymbol{\theta}), \sigma(s, \boldsymbol{\theta}))$$

- a es una acción en el espacio continuo
- s es el estado observado
- $\mu(s, \boldsymbol{\theta})$ es la media de la distribución
- $\sigma(s, \boldsymbol{\theta})$ es la desviación estándar de la distribución

Los espacios de acciones continuas son una generalización de los espacios de acciones discretos. Permiten abordar problemas con mayor precisión a costa de una mayor complejidad.

Así, el agente asigna diferentes distribuciones de acciones para diferentes estados.

$$\pi(a|s,\boldsymbol{\theta}) = \frac{1}{\sigma(s,\boldsymbol{\theta})\sqrt{2\pi}} \exp\left(-\frac{(a-\mu(s,\boldsymbol{\theta}))^2}{2\sigma(s,\boldsymbol{\theta})^2}\right)$$

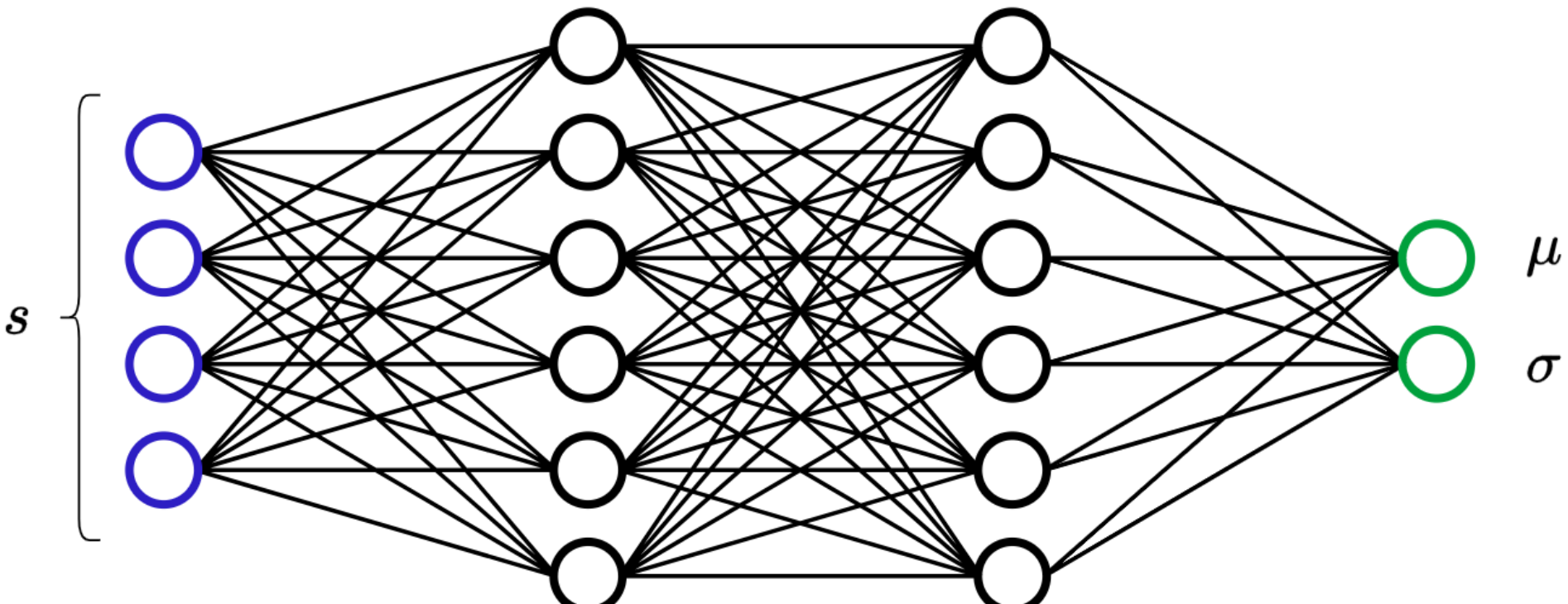
donde $\mu : \mathcal{S} \times \mathbb{R}^{d'} \longrightarrow \mathbb{R}$ y $\sigma : \mathcal{S} \times \mathbb{R}^{d'} \longrightarrow \mathbb{R}^+$ son funciones paramétricas que definen la distribución de acciones para cada estado.

Estimación de μ y σ

El proceso seguido para muestrear una acción en un espacio de acciones continuo es el siguiente:

1. Partimos de un estado arbitrario s
2. Calculamos μ y σ para dicho estado (por ejemplo, pueden venir dados por una red neuronal)
3. Muestreamos la acción de la distribución definida por $\mu(s, \boldsymbol{\theta})$ y $\sigma(s, \boldsymbol{\theta})$.

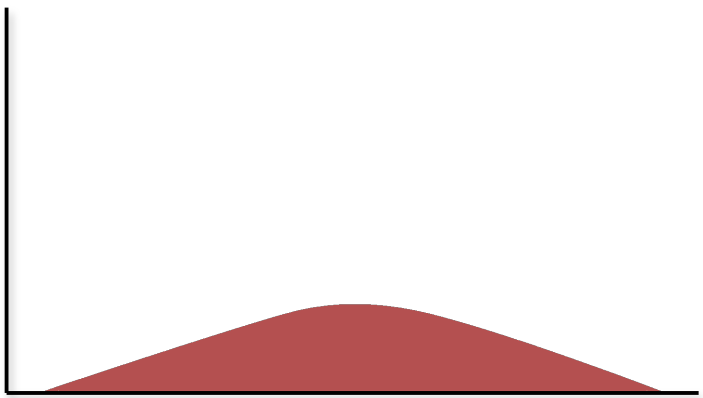
Estimación de μ y σ



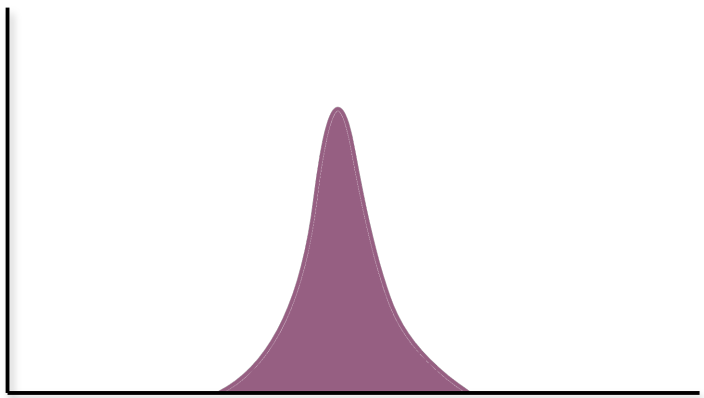
Exploración

El **valor inicial de σ** es importante, ya que determinará la exploración de la política.

- Favorece la exploración de forma natural.



Alta exploración



Baja exploración

A medida que el agente aprende, la desviación estándar se reduce, lo que implica una menor exploración y una mayor tendencia al comportamiento representado por la acción central de la distribución.

Trabajo propuesto

Trabajo propuesto

- **Implementar los algoritmos estudiados** y utilizarlos en un entorno de Gymnasium.
  - REINFORCE, REINFORCE con *baseline* y *One-step actor-critic*.
  - Compara su rendimiento.
- Implementar un agente en un entorno de Gymnasium con un **espacio de acciones continuo**.
- ¿Qué algoritmos basados en gradiente de la política son más empleados en la actualidad?
  - ¿Y *actor-critic*?
- Investiga sobre el algoritmo **A2C** y trata de implementarlo.

Bibliografía y recursos

- **Capítulo 13** de Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction.
- https://github.com/MathFoundationRL/Book-Mathematical-Foundation-of-Reinforcement-Learning
- https://www.youtube.com/watch?v=e20EY4tFC_Q
- https://www.youtube.com/watch?v=AiFM6LZ7Vuo
- https://www.youtube.com/watch?v=ZODHxkjkuv4
- https://lilianweng.github.io/posts/2018-04-08-policy-gradient/
- https://tatika.pythonanywhere.com/post_group/12
- https://www.decisionsanddragons.com/

Aprendizaje por refuerzo

Aproximación de políticas

Antonio Manjavacas

manjavacas@ugr.es