Aprendizaje Reforzado

Lección 27

Dr. Pablo Alvarado Moya

CE5506 Introducción al reconocimiento de patrones
Área de Ingeniería en Computadores
Tecnológico de Costa Rica

II Semestre, 2019

Contenido

Repaso de regulación cuadrática lineal (LQR) (1)

- El objetivo de un MDP de horizonte finito es maximizar

$$\max \mathrm{E}\left[R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + R(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)}) + \ldots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})\right]$$

- Se utiliza para eso un algoritmo de programación dinámica:

$$V^*_{(T)}(\underline{\mathbf{s}}) = \max_{\underline{\mathbf{a}}_{(T)}} R(\underline{\mathbf{s}},\underline{\mathbf{a}}_{(T)})$$

$$V^*_{(t)}(\underline{\mathbf{s}}) = \max_{\underline{\mathbf{a}}} \left(R(\underline{\mathbf{s}},\underline{\mathbf{a}}) + \sum_{\underline{\mathbf{s}}'} P_{\underline{\mathbf{sa}}}(\underline{\mathbf{s}}') V^*_{(t+1)}(\underline{\mathbf{s}}') \right)$$

$$\pi^*_{(t)}(\underline{\mathbf{s}}) = \arg\max_{\underline{\mathbf{a}}} \left(R(\underline{\mathbf{s}},\underline{\mathbf{a}}) + \sum_{\underline{\mathbf{s}}'} P_{\underline{\mathbf{sa}}}(\underline{\mathbf{s}}') V^*_{(t+1)}(\underline{\mathbf{s}}') \right)$$

Repaso de regulación cuadrática lineal (LQR) (2)

- Un ejemplo particular de MDP de horizonte finito es el LQR.
- En LQR usamos espacios de estado $\mathcal{S} = \mathbb{R}^n$ y de acción $\mathcal{A} = \mathbb{R}^d$ continuos
- En **L**QR la dinámica del sistema se asume **lineal**, descrita por:

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}_{(t)}\underline{\mathbf{s}}_{(t)} + \mathbf{B}_{(t)}\underline{\mathbf{a}}_{(t)} + \underline{\mathbf{w}}_{(t)}$$

  con $\underline{\mathbf{w}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{0}}, \mathbf{\Sigma}_{\underline{\mathbf{w}}})$.
- Este modelo se puede obtener por linealización (aproximación por serie de Taylor de primer orden alrededor de punto de operación).

Repaso de regulación cuadrática lineal (LQR) (3)

- En L**Q**R suponemos la recompensa cuadrática:

$$R_{(t)}\left(\underline{\mathbf{s}}_{(t)},\underline{\mathbf{a}}_{(t)}\right) = -\left(\underline{\mathbf{s}}_{(t)}^{T}\mathbf{U}_{(t)}\underline{\mathbf{s}}_{(t)} + \underline{\mathbf{a}}_{(t)}^{T}\mathbf{V}_{(t)}\underline{\mathbf{a}}_{(t)}\right)$$

- Con lo anterior, en LQR la función de valor para cada instante de tiempo es:

$$V_{(t)}^{*}(\underline{\mathbf{s}}_{(t)}) = \underline{\mathbf{s}}_{(t)}^{T}\mathbf{\Phi}_{(t)}\underline{\mathbf{s}}_{(t)} + \psi_{(t)}$$

Repaso de regulación cuadrática lineal (LQR) (4)

- La programación dinámica produce:

$$\mathbf{\Phi}_{(T)} = -\mathbf{U}_{(T)} \qquad \psi_{(T)} = 0$$

y la recursión con la ecuación de Riccati:

$$\begin{aligned}\mathbf{\Phi}_{(t)} =& \mathbf{A}_{(t)}^T \left(\mathbf{\Phi}_{(t+1)} - \mathbf{\Phi}_{(t+1)} \mathbf{B}_{(t)} \left(\mathbf{B}_t^T \mathbf{\Phi}_{(t+1)} \mathbf{B}_t - \mathbf{V}_{(t)} \right)^{-1} \mathbf{B}_{(t)}^T \mathbf{\Phi}_{(t+1)} \right) \mathbf{A}_{(t)} \\ & - \mathbf{U}_{(t)} \\ \psi_{(t)} =& - \operatorname{tr} \left[\mathbf{\Sigma}_{\underline{\mathbf{w}}} \mathbf{\Phi}_{(t+1)} \right] + \psi_{(t+1)}\end{aligned}$$

Repaso de regulación cuadrática lineal (LQR) (5)

- Finalmente, la **política óptima** estará dada por:

$$\pi^*(\underline{\mathbf{s}}_{(t)}) = \mathbf{L}_{(t)}\underline{\mathbf{s}}_{(t)}$$

$$\mathbf{L}_{(t)} = \left(\mathbf{B}_{(t)}^T \mathbf{\Phi}_{(t+1)} \mathbf{B}_{(t)} - \mathbf{V}_{(t)}\right)^{-1} \mathbf{B}_{(t)}^T \mathbf{\Phi}_{(t+1)} \mathbf{A}_{(t)}$$

  que ¡**no** depende de $\psi_{(t)}$ ni $\psi_{(t+1)}$!
- Como no depende de $\psi_{(t)}$, ¡el ruido no tiene efecto en la política óptima! pues $\mathbf{\Sigma}_{\underline{\mathbf{w}}}$ solo aparece en los $\psi_{(t)}$.
- Esta es una propiedad única para los sistemas LQR.
- Note que la función de valor **sí** depende de $\psi_{(t)}$ y por tanto sí es afectada por el ruido.

LQR: Programación dinámica diferencial (DDP) (1)

- Vamos a presentar una forma específica de aplicar LQR, denominada **programación dinámica diferencial** (DDP).
- Partamos de un simulador de la planta

$$\underline{\mathbf{s}}_{(t+1)} = f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)})$$

que suponemos ser no-lineal y determinista.

- Supongamos además que hay alguna "trayectoria" que queremos que la planta siga.
- DDP es una estrategia de linealización no-estacionaria para poder seguir esa trayectoria.

LQR: Programación dinámica diferencial (DDP) (2)

- Necesitamos entonces
  1. Proponer una **trayectoria nominal** $\bar{\underline{\mathbf{s}}}_{(0)}, \bar{\underline{\mathbf{a}}}_{(0)}, \bar{\underline{\mathbf{s}}}_{(1)}, \bar{\underline{\mathbf{a}}}_{(1)}, \ldots, \bar{\underline{\mathbf{s}}}_{(T)}, \bar{\underline{\mathbf{a}}}_{(T)}$ obtenida con algún controlador disponible, no necesariamente bueno.
  2. Linealizamos f alrededor de la trayectoria nominal.

$$\begin{aligned}\underline{\mathbf{s}}_{(t+1)} &\approx f(\bar{\underline{\mathbf{s}}}_{(t)}, \bar{\underline{\mathbf{a}}}_{(t)}) + \mathbf{J}_f(\bar{\underline{\mathbf{s}}}_{(t)}, \bar{\underline{\mathbf{a}}}_{(t)}) \begin{bmatrix} \underline{\mathbf{s}}_{(t)} - \bar{\underline{\mathbf{s}}}_{(t)} \\ \underline{\mathbf{a}}_{(t)} - \bar{\underline{\mathbf{a}}}_{(t)} \end{bmatrix} \\ &= \mathbf{A}_{(t)} \underline{\mathbf{s}}_{(t)} + \mathbf{B}_{(t)} \underline{\mathbf{a}}_{(t)}\end{aligned}$$

  Estamos suponiendo que $(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)}) \approx (\bar{\underline{\mathbf{s}}}_{(t)}, \bar{\underline{\mathbf{a}}}_{(t)})$
  3. Ejecutamos LQR para obtener $\pi_{(t)}$

LQR: Programación dinámica diferencial (DDP) (3)

4. Utilizamos un simulador/modelo para obtener una trayectoria nominal nueva.

$$\begin{aligned} \bar{\underline{\mathbf{s}}}_{(0)} &= \text{estado incial} \\ \bar{\underline{\mathbf{a}}}_{(t)} &= \pi_{(t)}(\bar{\underline{\mathbf{s}}}_{(t)}) \\ \bar{\underline{\mathbf{s}}}_{(t+1)} &= f(\bar{\underline{\mathbf{s}}}_{(t)}, \bar{\underline{\mathbf{a}}}_{(t)}) \end{aligned}$$

   y luego linealizamos sobre esta nueva trayectoria y repetimos desde paso 2.

- Estos pasos convergen relativamente bien al controlador adecuado.

Estados observables

- Hasta ahora hemos supuesto que los estados son observables en todo instante de tiempo.
- Queremos ahora atacar un nuevo problema donde no se puede observar el estado directamente.
- Describiremos ahora algunos sistemas de este tipo, primero sin ningún tipo de control

Ejemplo de estados y variables observables (1)

- Supongamos un sistema simplificado, con dinámica dada por

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\underline{\mathbf{s}}_{(t)} + \underline{\mathbf{w}}_{(t)}$$

- Por ejemplo, podríamos tener un sistema con

$$\underline{\mathbf{s}}_{(t)} = \begin{bmatrix} x_{(t)} \\ \dot{x}_{(t)} \\ y_{(t)} \\ \dot{y}_{(t)} \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0{,}9 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0{,}9 \end{bmatrix}$$

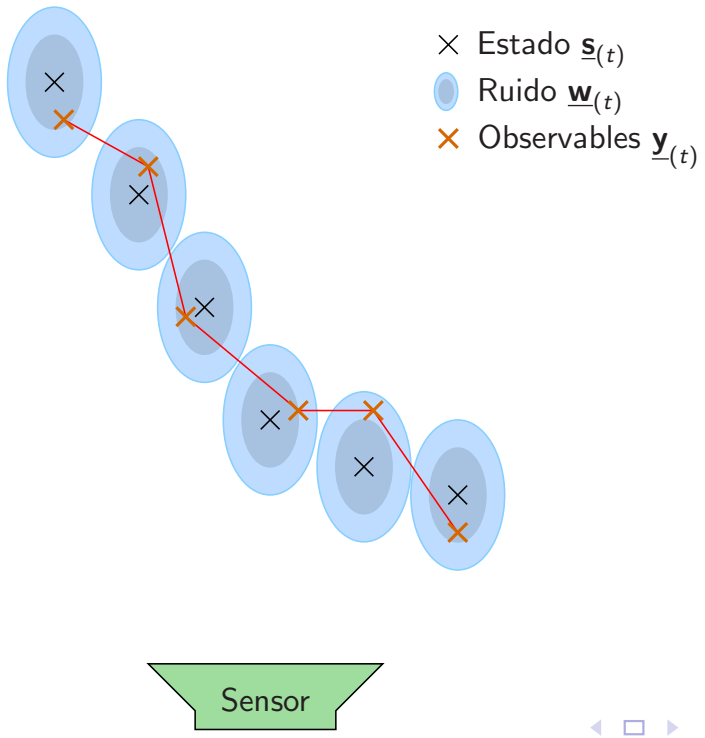
- Supongamos que lo que podemos observar es

$$\underline{\mathbf{y}}_{(t)} = \mathbf{C}\underline{\mathbf{s}}_{(t)} + \underline{\mathbf{v}}_{(t)}, \qquad \underline{\mathbf{v}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{0}}, \underline{\mathbf{\Sigma_v}})$$

Ejemplo de estados y variables observables (2)

- Por ejemplo: supongamos que podemos medir para planta:

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{C}\underline{\mathbf{s}}_{(t)} = \begin{bmatrix} x \\ y \end{bmatrix}$$



Ejemplo de estados y variables observables (3)

- Queremos estimar los estados completos $\underline{\mathbf{s}}_{(t)}$ a partir de las observaciones ruidosas $\underline{\mathbf{y}}_{(t)}$.
- Para eso necesitamos estimar la distribución del estado en términos de los estados anteriores $P(\underline{\mathbf{s}}_{(t)}|\underline{\mathbf{y}}_{(1)},\ldots,\underline{\mathbf{y}}_{(t)})$
- Supondremos que las variables aleatorias $\underline{\mathbf{s}}_{(0)},\underline{\mathbf{s}}_{(1)},\ldots,\underline{\mathbf{s}}_{(t)}$, $\underline{\mathbf{y}}_{(1)},\ldots,\underline{\mathbf{y}}_{(t)}$ tienen una probabilidad conjunta gaussiana:

$$\underline{\mathbf{z}} = \begin{bmatrix} \underline{\mathbf{s}}_{(0)} & \underline{\mathbf{s}}_{(1)} & \cdots & \underline{\mathbf{s}}_{(t)} & \underline{\mathbf{y}}_{(1)} & \cdots & \underline{\mathbf{y}}_{(t)} \end{bmatrix}^T \sim \mathcal{N}(\underline{\boldsymbol{\mu}}, \boldsymbol{\Sigma})$$

- Por medio de marginalización y condicionalización podríamos calcular $P(\underline{\mathbf{s}}_{(t)}|\underline{\mathbf{y}}_{(1)},\ldots,\underline{\mathbf{y}}_{(t)})$, aunque solo conceptualmente, pues computacionalmente no es práctico.
- El **filtro de Kalman** ofrece otra forma de calcular lo anterior.

Filtros de Kalman (1)

- Los filtros de Kalman realizan la estimación de la distribución de forma recursiva

$$P(\underline{\mathbf{s}}_{(t)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t)})$$
$$P(\underline{\mathbf{s}}_{(t+1)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t)}) \quad \text{Predicción}$$
$$P(\underline{\mathbf{s}}_{(t+1)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t+1)}) \quad \text{Actualización}$$

- Son eficientes, puesto que cada observación nueva se usa para actualizar estimadores, sin tener que recalcular las probabilidades conjuntas.

Idea general del filtro de Kalman

$$\begin{array}{ccccccc} \underline{\mathbf{y}}_{(1)} & & \underline{\mathbf{y}}_{(2)} & & \underline{\mathbf{y}}_{(3)} & & \cdots \\ \downarrow & & \downarrow & & \downarrow & & \vdots \\ P(\underline{\mathbf{s}}_{(1)}|\underline{\mathbf{y}}_{(1)}) & \rightarrow & P(\underline{\mathbf{s}}_{(2)}|\underline{\mathbf{y}}_{(1)},\underline{\mathbf{y}}_{(2)}) & \rightarrow & P(\underline{\mathbf{s}}_{(3)}|\underline{\mathbf{y}}_{(1)},\underline{\mathbf{y}}_{(2)},\underline{\mathbf{y}}_{(3)}) & \rightarrow & \cdots \end{array}$$

Filtro de Kalman: predicción

- En el paso de predicción:

$$\underline{\mathbf{s}}_{(t)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{s}}_{t|t},\mathbf{\Sigma}_{t|t})$$

Luego

$$\underline{\mathbf{s}}_{(t+1)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{s}}_{t+1|t},\mathbf{\Sigma}_{t+1|t})$$

donde

$$\underline{\mathbf{s}}_{t+1|t} = \mathbf{A}\underline{\mathbf{s}}_{t|t} \qquad \mathbf{\Sigma}_{t+1|t} = \mathbf{A}\mathbf{\Sigma}_{t|t}\mathbf{A}^T + \mathbf{\Sigma}_{\underline{\mathbf{v}}}$$

- $\underline{\mathbf{s}}_{(t)}$ y $\underline{\mathbf{y}}_{(t)}$ corresponden al estado y observables verdaderos
- Lo que calculamos es $\underline{\mathbf{s}}_{t|t}$, $\underline{\mathbf{s}}_{t+1|t}$, $\mathbf{\Sigma}_{t|t}$, $\mathbf{\Sigma}_{t+1|t}$

Filtro de Kalman: actualización

- El paso de actualización obtiene

$$\underline{\mathbf{s}}_{(t+1)}|\underline{\mathbf{y}}_{(1)},\cdots,\underline{\mathbf{y}}_{(t+1)} \sim \mathcal{N}(\underline{\mathbf{s}}_{t+1|t+1},\boldsymbol{\Sigma}_{t+1|t+1})$$

donde

$$\begin{aligned}
\underline{\mathbf{s}}_{t+1|t+1} &= \underline{\mathbf{s}}_{t+1|t} + \mathbf{K}_{(t+1)}(\underline{\mathbf{y}}_{(t+1)} - \mathbf{C}\underline{\mathbf{s}}_{t+1|t}) \\
\mathbf{K}_{(t+1)} &= \boldsymbol{\Sigma}_{t+1|t}\mathbf{C}^T(\mathbf{C}\boldsymbol{\Sigma}_{t+1|t}\mathbf{C}^T + \boldsymbol{\Sigma}_{\underline{\mathbf{v}}})^{-1} \\
\boldsymbol{\Sigma}_{t+1|t+1} &= \boldsymbol{\Sigma}_{t+1|t} - \boldsymbol{\Sigma}_{t+1|t}\mathbf{C}^T(\mathbf{C}\boldsymbol{\Sigma}_{t+1|t}\mathbf{C}^T + \boldsymbol{\Sigma}_{\underline{\mathbf{v}}})^{-1}\mathbf{C}\boldsymbol{\Sigma}_{t+1|t}
\end{aligned}$$

- $\underline{\mathbf{s}}_{t+1|t+1}$ es nuestro mejor estimado de $\underline{\mathbf{s}}_{(t+1)}$, dadas todas las observaciones que tenemos hasta ese momento.

Control gaussiano cuadrático lineal (LQG) (1)

- Control LQG es la fusión de lo anterior: control LQR + filtros de Kalman
- Tenemos un sistema dinámico lineal:

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\underline{\mathbf{s}}_{(t)} + \mathbf{B}\underline{\mathbf{a}}_{(t)} + \underline{\mathbf{w}}_{(t)} \qquad \underline{\mathbf{w}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{0}}, \boldsymbol{\Sigma}_{\underline{\mathbf{w}}})$$

donde solo podemos observar con ruido

$$\underline{\mathbf{y}}_{(t)} = \mathbf{C}\underline{\mathbf{s}}_{(t)} + \underline{\mathbf{v}}_{(t)} \qquad \underline{\mathbf{v}}_{(t)} \sim \mathcal{N}(\underline{\mathbf{0}}, \boldsymbol{\Sigma}_{\underline{\mathbf{v}}})$$

- Usamos un filtro de Kalman para estimar el estado:

$$\underline{\mathbf{s}}_{0|0} = \underline{\mathbf{s}}_{(0)}, \boldsymbol{\Sigma}_{0|0} = \mathbf{0} \text{ con } \underline{\mathbf{s}}_{(0)} \sim \mathcal{N}(\underline{\mathbf{s}}_{0|0}, \boldsymbol{\Sigma}_{0|0})$$

Control gaussiano cuadrático lineal (LQG) (2)

- En el paso de predicción obtenemos:

$$\underline{\mathbf{s}}_{t+1|t} = \mathbf{A}\underline{\mathbf{s}}_{t|t} + \mathbf{B}\underline{\mathbf{a}}_{(t)}$$
$$\mathbf{\Sigma}_{t+1|t} = \mathbf{A}\mathbf{\Sigma}_{t|t}\mathbf{A}^T + \mathbf{\Sigma}_{\underline{\mathbf{v}}}$$

- Calculamos entonces la matriz $\mathbf{L}_{(t)}$ usando LQR.
- Puesto que la acción a seguir en LQR es

$$\underline{\mathbf{a}}_{(t)} = \mathbf{L}_{(t)}\underline{\mathbf{s}}_{(t)}$$

pero no podemos observar $\underline{\mathbf{s}}_{(t)}$ directamente, entonces usamos

$$\underline{\mathbf{a}}_{(t)} = \mathbf{L}_{(t)}\underline{\mathbf{s}}_{t|t}$$

- En este tipo de problemas, este es el procedimiento óptimo en cuanto al tratamiento del ruido.

MDP observables parcialmente (POMDP)

MDP observables parcialmente (1)

- Lo anterior se formaliza en POMDP, definido como la tupla $\langle \mathcal{S}, \mathcal{A}, \mathcal{Y}, \{P_{\underline{\mathbf{sa}}}\}, \{O_{\underline{\mathbf{s}}}\}, T, R \rangle$ con
  - $\mathcal{S}$, $\mathcal{A}$, $P_{\underline{\mathbf{sa}}}$, T y R como antes
  - $\mathcal{Y}$ es el conjunto de observaciones posibles
  - $O_{\underline{\mathbf{s}}}$ son las distribuciones de observaciones
- En cada paso se observa $\underline{\mathbf{y}}_{(t)} \sim O_{\underline{\mathbf{s}}_{(t)}}$ estando en $\underline{\mathbf{s}}_{(t)}$
- El cálculo de la política óptima de un POMDP es en general un problema NP-hard.

MDP observables parcialmente (2)

- En el caso particular de dinámica lineal usamos los filtros de Kalman y seguimos la estrategia para calcular la política, que resulta ser óptima
  1. Estimamos primero los estados a partir de las observaciones
  2. Usamos los estados estimados para calcular las acciones
- En cualquier otro caso, esa estrategia **no** produce la política óptima.
- Vamos entonces a presentar otra clase de algoritmos de aprendizaje reforzado

Algoritmos de búsqueda de políticas

Algoritmos de búsqueda de políticas
Policy search algorithms

- Los algoritmos de búsqueda de políticas se aplican tanto a MDP (estados observables) como a POMDP.
- Vamos a concentrarnos en el caso de MDP
- Luego esbozaremos el caso de POMDP, que en general no se asegura que converja al óptimo global (por ser NP-hard)

Búsqueda directa de políticas

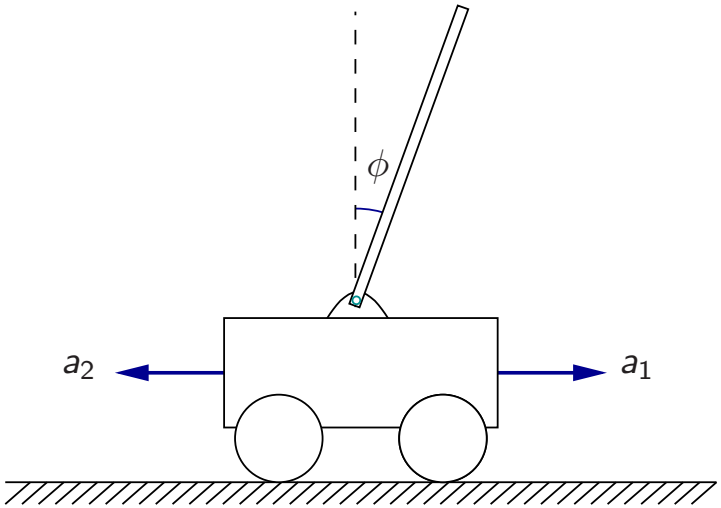
- Vamos a definir la clase Π de políticas
- La estrategia es buscar una buena política $\pi \in \Pi$
- La estrategia es análoga a cuando definimos la clase de hipótesis $\mathcal{H}$ y buscábamos un $h \in \mathcal{H}$
- En todos los algoritmos hasta ahora primero determinamos la función de valor óptima $V^*(\underline{\mathbf{s}})$ y con eso luego estimamos la política óptima $\pi^*(\underline{\mathbf{s}})$.
- Ahora queremos calcular **directamente** la política $\pi^*(\underline{\mathbf{s}})$ (sin usar $V^*(\underline{\mathbf{s}})$).

Definición de política estocástica

- Una **política estocástica** es una función $\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ donde $\pi(\underline{\mathbf{s}},\underline{\mathbf{a}})$ es la probabilidad de tomar la acción $\underline{\mathbf{a}}$ estando en el estado $\underline{\mathbf{s}}$.
- Por tanto $\sum_{\underline{\mathbf{a}}} \pi(\underline{\mathbf{s}},\underline{\mathbf{a}}) = 1$, $\pi(\underline{\mathbf{s}},\underline{\mathbf{a}}) \geq 0$

Ejemplo: péndulo invertido (1)

- Supongamos que tenemos un péndulo invertido



- La política estocástica puede ser

$$\pi_{\underline{\theta}}(\underline{\mathbf{s}}, a_1) = \frac{1}{1 + e^{-\underline{\theta}^T \underline{\mathbf{s}}}} \qquad \pi_{\underline{\theta}}(\underline{\mathbf{s}}, a_2) = 1 - \frac{1}{1 + e^{-\underline{\theta}^T \underline{\mathbf{s}}}}$$

Ejemplo: péndulo invertido (2)

- Con $\underline{\mathbf{s}} = \begin{bmatrix} 1 & x & \dot{x} & \phi & \dot{\phi} \end{bmatrix}^T$ y $\underline{\boldsymbol{\theta}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T$ si el péndulo está cayendo a la derecha aceleramos de forma proporcional al ángulo.
- Obviamente esta política $\underline{\boldsymbol{\theta}}$ no es buena porque ignoramos todo excepto el ángulo ϕ.
- Si tuviéramos d acciones, podríamos usar softmax

$$\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}, \underline{\mathbf{a}}_i) = \frac{e^{\underline{\boldsymbol{\theta}}_i^T \underline{\mathbf{s}}}}{\sum_{j=1}^{d} e^{\underline{\boldsymbol{\theta}}_j^T \underline{\mathbf{s}}}} \qquad \underline{\boldsymbol{\theta}} = \{\underline{\boldsymbol{\theta}}_1, \ldots, \underline{\boldsymbol{\theta}}_d\}$$

- En general, tenemos libertad para elegir las políticas adecuadas para la aplicación.

Objetivo de la búsqueda de política

- El objetivo de la búsqueda de política es encontrar

$$\max_{\underline{\boldsymbol{\theta}}} \mathrm{E}\left[R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + \cdots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}) \middle| \pi_{\underline{\boldsymbol{\theta}}},\underline{\mathbf{s}}_{(0)}\right]$$

Algoritmo REINFORCE

- Lo siguiente diverge del algoritmo REINFORCE original de Ronald Williams (1987), pero mantiene el concepto.
- Supongamos que $\underline{\mathbf{s}}_{(0)}$ es un estado inicial fijo.
- El objetivo es maximizar la suma de recompensas esperada:

$$
\begin{aligned}
&\mathrm{E}\left[R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + \cdots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})\right] \\
&= \sum_{\substack{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdots \\ \cdots\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}}} P(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdots\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})\left[R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + \cdots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})\right] \\
&= \sum_{\substack{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdots \\ \cdots\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}}} P(\underline{\mathbf{s}}_{(0)})\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})P_{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}}(\underline{\mathbf{s}}_{(1)})\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})\cdots\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}) \\
&\qquad \cdot \underbrace{\left[R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + \cdots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})\right]}_{\text{saldo}}
\end{aligned}
$$

Algoritmo REINFORCE

1: **repeat**
2: Muestree $\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)},\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)}\cdots\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}$
3: Calcule saldo$= R(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}) + \cdots + R(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})$
4: Actualice $\underline{\boldsymbol{\theta}} \leftarrow$

$$\underline{\boldsymbol{\theta}} + \alpha \left[\frac{\nabla_{\underline{\boldsymbol{\theta}}} \pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}{\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})} + \cdots + \frac{\nabla_{\underline{\boldsymbol{\theta}}} \pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}{\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})} \right] \cdot \text{saldo}$$

5: **until** convergencia

- Este algoritmo realiza un ascenso estocástico.
- Queremos averiguar la tendencia general en los cambios de los parámetros

Valor esperado de cambio de los parámetros

- Queremos encontrar entonces qué es

$$\mathrm{E}\left[\left(\frac{\nabla_{\underline{\boldsymbol{\theta}}}\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}{\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}+\cdots+\frac{\nabla_{\underline{\boldsymbol{\theta}}}\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}{\pi_{\underline{\boldsymbol{\theta}}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}\right)\cdot \text{saldo}\right]$$

  como dirección de actualización de $\underline{\boldsymbol{\theta}}$ cuando intentamos maximizar el valor esperado del saldo.
- El algoritmo REINFORCE asegura el ascenso en esa dirección, pero de forma estocástica.

Derivación de REINFORCE (1)

- Queremos probar que si calculamos el ascenso de gradiente del valor esperado, llegamos al algoritmo anterior.
- Para ello requerimos la regla de la cadena de la derivación:

$$\frac{d}{d\theta} f(\theta)g(\theta)h(\theta) = f'(\theta)g(\theta)h(\theta) + f(\theta)g'(\theta)h(\theta) + f(\theta)g(\theta)h'(\theta)$$

Derivación de REINFORCE (2)

- Tenemos entonces que (paso 1/4)

$$\begin{aligned}\nabla_{\underline{\theta}} \mathrm{E}[\text{saldo}] = \sum_{\substack{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdots \\ \cdots\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}}} \Big[& P(\underline{\mathbf{s}}_{(0)})\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})P_{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}}(\underline{\mathbf{s}}_{(1)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})\cdots\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}) \\ & + P(\underline{\mathbf{s}}_{(0)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})P_{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}}(\underline{\mathbf{s}}_{(1)})\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})\cdots\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}) \\ & \vdots \\ & + P(\underline{\mathbf{s}}_{(0)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})P_{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}}(\underline{\mathbf{s}}_{(1)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})\cdots\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}) \\ = & \quad \Big] \cdot \text{saldo}\end{aligned}$$

Derivación de REINFORCE (3)

- Con algunas manipulaciones algebraicas (Paso 2/4)

$$\nabla_{\underline{\theta}} \mathrm{E}[\text{saldo}] = \sum_{\substack{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdot\cdot \\ \cdot\cdot\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}}} P(\underline{\mathbf{s}}_{(0)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})P_{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}}(\underline{\mathbf{s}}_{(1)})\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})\cdots\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})$$
$$\cdot\left[\frac{\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}+\frac{\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})}+\cdots+\frac{\nabla_{\underline{\theta}}\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}\right]$$
$$\cdot\,\text{saldo}$$

Derivación de REINFORCE (4)

- El primer término anterior ya lo trabajamos inicialmente, y es la probabilidad conjunta de la serie de estados-acciones

$$\nabla_{\underline{\theta}} \mathrm{E}[\text{saldo}] = \sum_{\substack{\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)}\cdots \\ \cdots \underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)}}} P(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)} \cdots \underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})$$

$$\cdot \left[\frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})} + \frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})} + \cdots + \frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})} \right]$$

$$\cdot \text{saldo}$$

$$= \mathrm{E}\left[\left(\frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(0)},\underline{\mathbf{a}}_{(0)})} + \frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(1)},\underline{\mathbf{a}}_{(1)})} + \cdots + \frac{\nabla_{\underline{\theta}} \pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})}{\pi_{\underline{\theta}}(\underline{\mathbf{s}}_{(T)},\underline{\mathbf{a}}_{(T)})} \right) \cdot \text{saldo} \right]$$

Derivación de REINFORCE (5)

- La anterior es la dirección que usamos en REINFORCE
- Por ser estocástica y solo en promedio correcta, el número de iteraciones requeridas por REINFORCE para converger puede fácilmente ascender a millones (¡es un algoritmo muy lento!)
- Por esa razón REINFORCE se usa para sistemas simulados, pues la captura de datos en sistemas reales no es factible.

Ocasión para cada algoritmo

- Vimos algoritmos que calculan V^* y con eso π^*
- Ahora presentamos un algoritmo que calcula directamente π^*
- Los algoritmos de búsqueda de política son efectivos cuando se puede elegir una clase de políticas Π.
- Este es el caso cuando se requieren decisiones de bajo nivel inmediatas, casi **reflejos** (como el péndulo invertido)
- Cuando se requieren procesos de varios/muchos pasos, con buen "razonamiento" (toma de decisiones secuenciales de alto nivel, menos instintivos, más estratégicas) se usan los métodos de aproximación de la función de valor.

Búsqueda de política con POMDP

- Si solo contamos con una aproximación $\hat{\underline{\mathbf{s}}}$ del estado $\underline{\mathbf{s}}$ (como por ejemplo $\hat{\underline{\mathbf{s}}} = \underline{\mathbf{s}}_{t|t}$ con un filtro de Kalman), entonces todavía es posible usar algoritmos de búsqueda de políticas.
- Por ejemplo, podemos usar

$$\pi_{\underline{\boldsymbol{\theta}}}(\hat{\underline{\mathbf{s}}}, \underline{\mathbf{a}}_0) = \frac{1}{1 + e^{-\underline{\boldsymbol{\theta}}^T \hat{\underline{\mathbf{s}}}}}$$

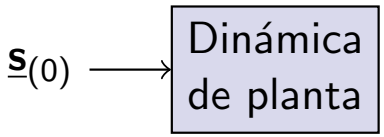
Pegasus

- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$

$\underline{\mathbf{s}}_{(0)}$

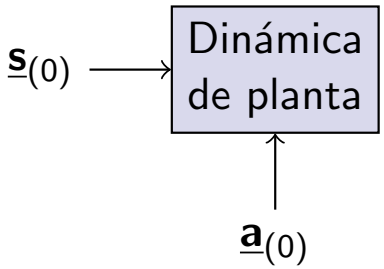
Pegasus

- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



Pegasus

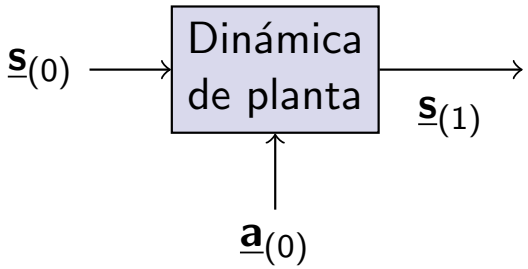
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

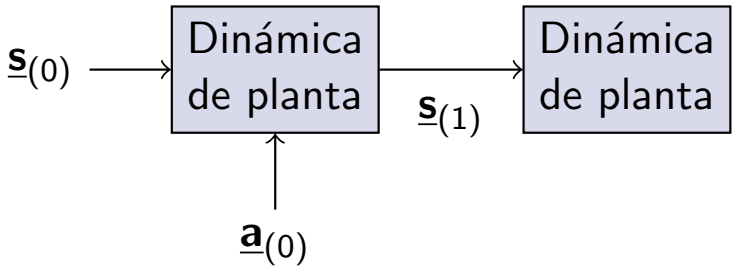
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

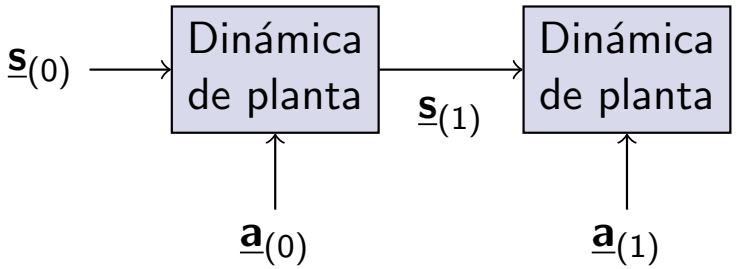
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

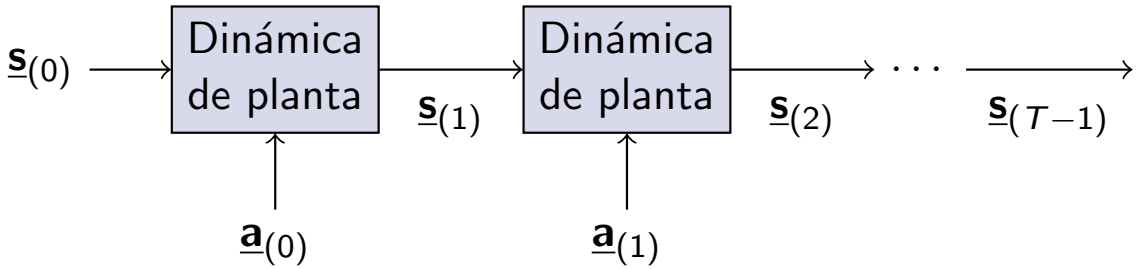
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

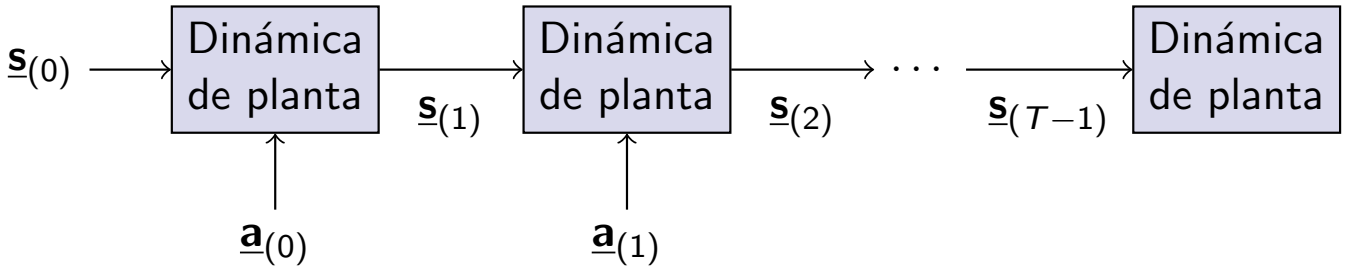
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

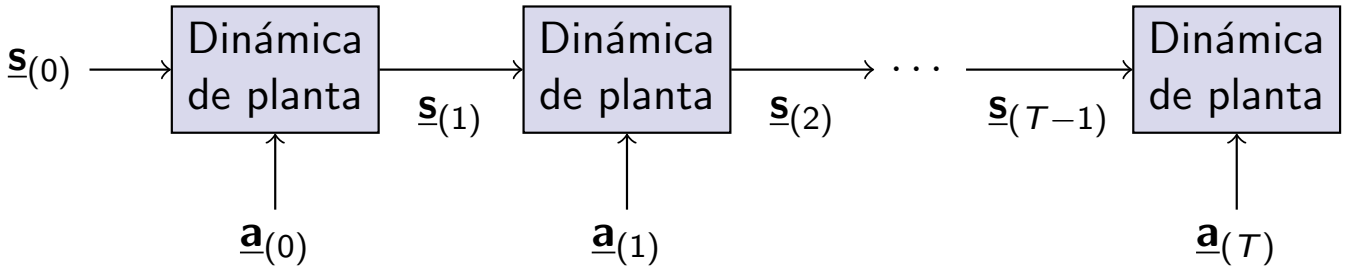
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

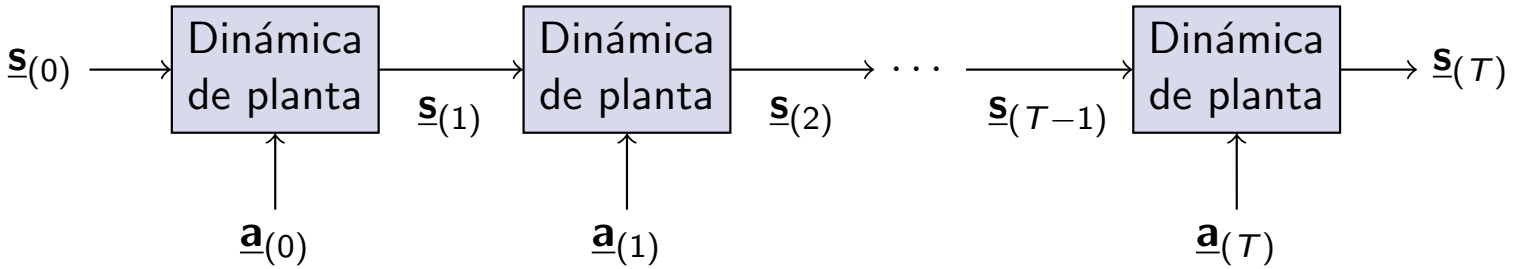
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

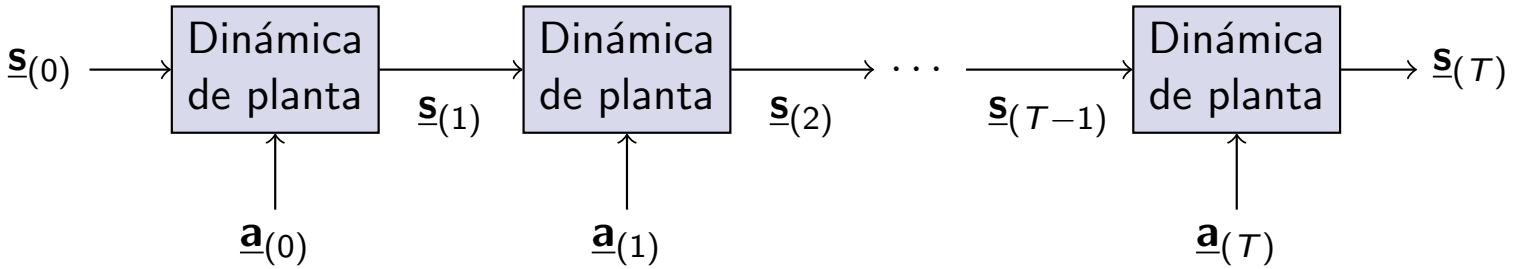
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$

Pegasus

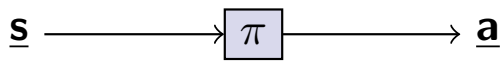
- Este es el algoritmo usado por el grupo de Andrew Ng para volar un helicóptero.
- En cada instante de tiempo, el sistema está en un estado $\underline{\mathbf{s}}_{(t)}$



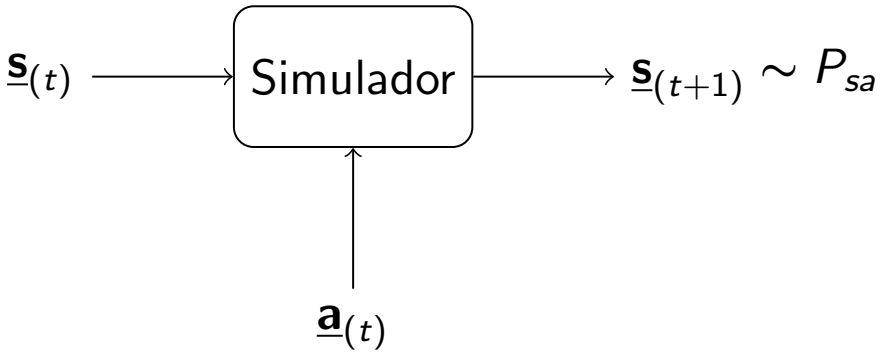
- Entonces aplicamos una acción $\underline{\mathbf{a}}_{(t)}$
- La tarea es maximizar la recompensa esperada:

$$\mathrm{E}\left[R(\underline{\mathbf{s}}_{(0)}) + R(\underline{\mathbf{s}}_{(1)}) + \cdots + R(\underline{\mathbf{s}}_{(T)})\right]$$

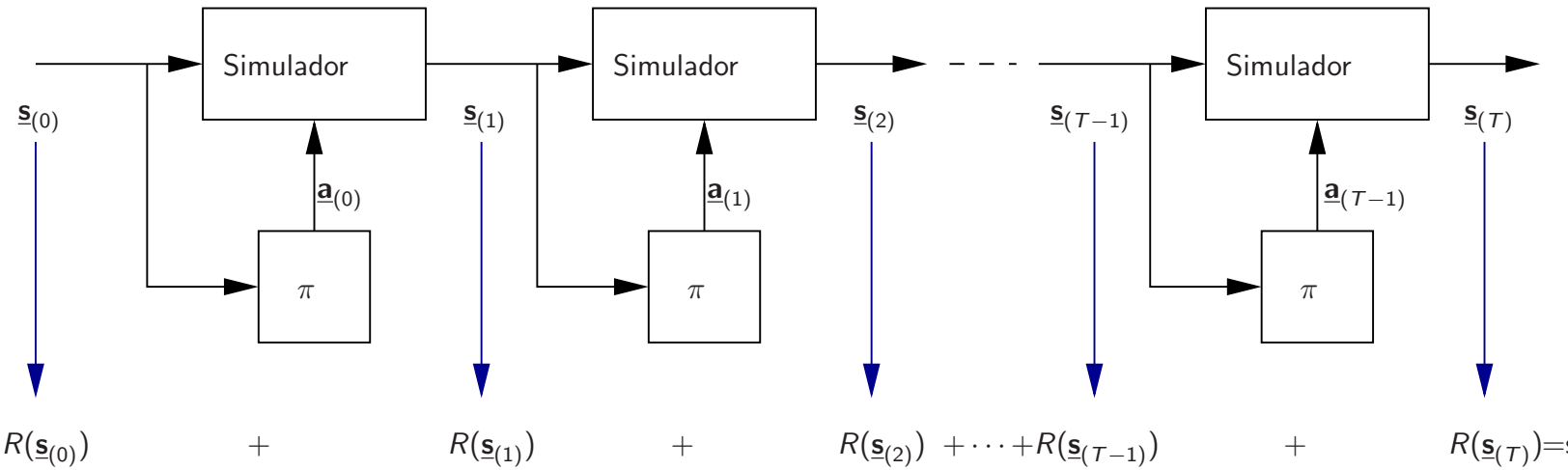
- Queremos aprender una política $\pi : \mathcal{S} \to \mathcal{A}$



- Con el simulador/modelo para el MDP



evaluaremos la política π:



Estrategia en Pegasus

- Una idea es usar el simulador para evaluar la definición parametrizada de $\pi_{\underline{\theta}}(\underline{\mathbf{s}})$, y con ello buscar los parámetros $\underline{\boldsymbol{\theta}}$ que maximizan el saldo.
- El problema es que el simulador es estocástico y por tanto cada vez que evaluamos una política obtenemos una respuesta diferente (esto se denomía muestreo "Monte Carlo").
- El promedio de m muestras no converge de forma uniforme al saldo, y por tanto no funciona para el aprendizaje reforzado.
- Pegasus usa el hecho de que el simulador internamente usa un generador de números aleatorios, que es entonces intercambiado por una secuencia fija predeterminada, lo que convierte al saldo en una función determinista.
- Esto asegura un mejor estimado del saldo, que permite el aprendizaje reforzado.

Depuración de algoritmos de aprendizaje reforzado

Depuración de algoritmos de aprendizaje reforzado

- Para desarrollar un sistema de control con aprendizaje reforzado
  1. Realizamos un simulador de la *planta* (p. ej. helicóptero, péndulo invertido, etc.)
  2. Elegimos entonces la función de recompensa.
     Por ejemplo $R(\underline{\mathbf{s}}) = -\|\underline{\mathbf{s}} - \underline{\mathbf{s}}_{\text{deseado}}\|^2$
  3. Usamos el aprendizaje reforzado para controlar la planta (p. ej. volar el helicóptero) en la simulación y maximizar el saldo $\mathrm{E}[R(\underline{\mathbf{s}}_0) + R(\underline{\mathbf{s}}_1) + \ldots + R(\underline{\mathbf{s}}_T)]$ y así aprender una política $\pi_{RL}(\underline{\mathbf{s}})$

Depuración de algoritmos de aprendizaje reforzado

- Para desarrollar un sistema de control con aprendizaje reforzado
  1. Realizamos un simulador de la *planta* (p. ej. helicóptero, péndulo invertido, etc.)
  2. Elegimos entonces la función de recompensa.
     Por ejemplo $R(\underline{\mathbf{s}}) = -\|\underline{\mathbf{s}} - \underline{\mathbf{s}}_{\text{deseado}}\|^2$
  3. Usamos el aprendizaje reforzado para controlar la planta (p. ej. volar el helicóptero) en la simulación y maximizar el saldo $\mathrm{E}[R(\underline{\mathbf{s}}_0) + R(\underline{\mathbf{s}}_1) + \ldots + R(\underline{\mathbf{s}}_T)]$ y así aprender una política $\pi_{RL}(\underline{\mathbf{s}})$
- Supongamos que la política π_{RL} no cumple las expectativas. ¿Qué hacemos?
  - ¿Mejoramos el simulador?
  - ¿Modificamos la función de recompensa?
  - ¿Modificamos el algoritmo de aprendizaje reforzado?

Premisa de funcionamiento correcto

- Supongamos que:
  1. El simulador es preciso y exacto
  2. El algoritmo de aprendizaje reforzado controla bien la planta en la simulación, pues maximiza el saldo esperado

$$V^{\pi_{RL}}(\underline{\mathbf{s}}_0) = \mathrm{E}[R(\underline{\mathbf{s}}_0) + R(\underline{\mathbf{s}}_1) + \cdots + R(\underline{\mathbf{s}}_T)|\pi_{RL}, \underline{\mathbf{s}}_0].$$

  3. La maximización del saldo esperado corresponde al comportamiento correcto (p. ej. volar el helicóptero).

entonces, el controlador π_{RL} debería funcionar bien en la planta real.

Diagnósitico

Diagnósitico:

1. Si π_{RL} funciona en el simulador pero no en la planta real, entonces el problema es el simulador.
2. Sea π_{human} la política de control humano. Si $V^{\pi_{RL}}(\underline{\mathbf{s}}_0) < V^{\pi_{\text{human}}}(\underline{\mathbf{s}}_0)$ entonces algoritmo de aprendizaje reforzado tiene problemas.
3. Si $V^{\pi_{RL}}(\underline{\mathbf{s}}_0) > V^{\pi_{\text{human}}}(\underline{\mathbf{s}}_0)$ entonces el problema es la función de recompensa.

Resumen

Este documento ha sido elaborado con software libre incluyendo LaTeX, Beamer, GNUPlot, GNU/Octave, XFig, Inkscape, GNU-Make y Subversion en GNU/Linux

Este trabajo se encuentra bajo una Licencia Creative Commons Atribución-NoComercial-LicenciarIgual 3.0 Unported. Para ver una copia de esta Licencia, visite http://creativecommons.org/licenses/by-nc-sa/3.0/ o envíe una carta a Creative Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA.

© 2017–2019 Pablo Alvarado-Moya Área de Ingeniería en Computadores Instituto Tecnológico de Costa Rica