Monte Carlo (continuación): Cadenas de Markov

¿Cómo elegimos exactamente los estados tal que cada uno aparezca con una probabilidad tipo distribución de Boltzmann?

Una solución usual: Cadenas de Markov

Proceso de Markov

* objetivo: generar un conjunto de estados al azar de acuerdo con la probabilidad de Boltzmann
* Casi todas las simulaciones de Monte Carlo usan procesos de Markov como motor de la generación de estados →

→ Dado un estado μ, se genera un nuevo estado ν en forma aleatoria.
→ No genera necesariamente el mismo estado cada vez que el estado inicial es μ:

Probabilidad de transición de μ a ν

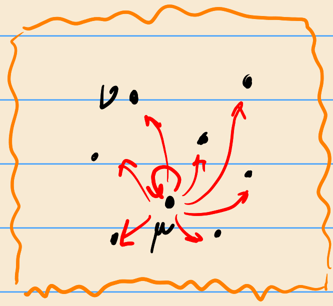
$$\left\{ \begin{array}{l} \text{Probabilidad de generar } \nu \\ \text{dado que el sistema está en } \mu \end{array} \right\} = P(\mu \to \nu)$$

Para definir un proceso de Markov todas las $P(\mu \to \nu)$ deben satisfacer:

1. No cambian con el tiempo.
2. Deben depender sólo de las propiedades de los estados μ y ν y de ningún otro estado del sistema por el cuál hayan pasado

Esto implica:

* La probabilidad de que el proceso de Markov genere el estado ν cuando el sistema está en μ debe ser la misma toda vez que el sistema visite el estado μ. Esto debe ser así, independientemente del historial de estados por los que se haya pasado.



* La $P(\mu \to \nu)$ debe satisfacer:

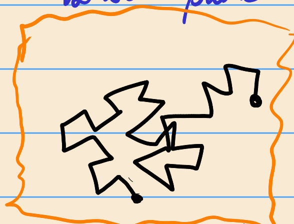
$$\boxed{\sum_{\nu} P(\mu \to \nu) = 1} \qquad (1)$$

porque los procesos de Markov deben poder generar algún nuevo estado una vez que el sistema está en μ.

* La probabilidad $p(\mu \to \mu)$ no tiene porqué ser $\emptyset$ $\Rightarrow$ Existe probabilidad no nula de que el sistema se quede en μ

* En una simulación de MC, usamos el proceso de Markov para generar una cadena de Markov de estados:

$$\boxed{\mu \to \nu \to \dots \to \lambda}$$



* El proceso de Markov se elige de manera tal que si se corre lo suficiente, arrancando de cualquier estado, se producirá una sucesión de estados que "aparecen" con la distribución de Boltzmann (Finalmente el sistema llega a equilibrio ó "termaliza").

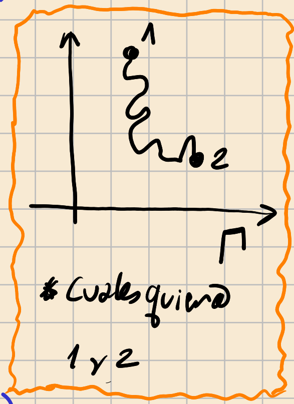
* Es el "mismo" proceso que hace el sistema real con su "computadora analógica" para llegar al equilibrio (Por ejemplo llegar a T ambiente desde un T menor).

El proceso de Markov debe cumplir además:

1_ Ergodicidad
2_ Balance Detallado (detailed balance)

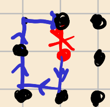
1. Ergodicidad

* Para un proceso de Markov, debe ser posible alcanzar cualquier estado del sistema desde cualquier otro estado, si esperamos lo suficiente (si "corremos" lo suficiente)



* Cada estado ν tiene probabilidad de Boltzmann P_ν no nula. Si el estado fuera inaccesible desde cualquier estado μ $\Rightarrow$ La probabilidad de encontrar a ν desde nuestra cadena de Markov sería 0 y no P_ν (Boltzmann) (como necesitamos que sea!).

* Ergodicidad $\longrightarrow$ permite hacer 0 alguna probabilidad de transición $P(\mu \to \nu)$ del proceso de Markov pero debe haber al menos un camino de $P(\mu \to \nu)$'s no nulo entre cualesquiera dos estados



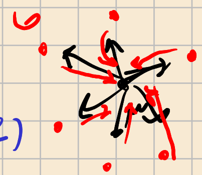
$\Downarrow$

En la práctica: los algoritmos de Monte Carlo fijan casi todas las probabilidades de transición en 0 y hay que tener cuidado que, al hacer, eso no viole el principio de ergodicidad.

2. Balance detallado

* Asegura que generamos la distribución de Boltzmann una vez que el sistema llega a equilibrio y no cualquier otra distribución.

* Importante: la tasa a la cual el sistema hace transiciones hacia y desde cualquier estado μ debe ser igual:



$$\sum_{\nu} P_{\mu} P(\mu \to \nu) = \sum_{\nu} P_{\nu} P(\nu \to \mu) \qquad (2)$$

Usando la "regla de suma", ecuación (1): $\sum_{\nu} P(\mu \to \nu) = 1$, reemplazamos en (2):

$$P_{\mu} = \sum_{\nu} P_{\nu} P(\nu \to \mu) \qquad (3)$$

Ejemplo $P_{\mu} = \frac{e^{-\beta E_{\mu}}}{Z_c}$

* Para cualquier conjunto de probabilidades de transición que satisfacen (3), la distribución de probabilidad P_{μ} será un equilibrio de la dinámica del proceso de Markov.

> ⊃ En rigor, (3) no garantiza que la distribución de probabilidades tenderá a P_{μ} desde cualquier estado del sistema. La evolución podría quedar atrapada en un "ciclo límite"
>
> ⇝ Para más detalles ver Newman y Barkema

⇓

Se asegura la convergencia a P_{μ} pidiendo una condición algo más fuerte:

$$\boxed{P_{\mu} P(\mu \to \nu) = P_{\nu} P(\nu \to \mu)} \qquad (4) \quad \text{Balance Detallado}$$

⊛ Si se satisface, seguro se satisface (2) y elimina la posibilidad de ciclos límite.

* Puede hacerse que la cadena de Markov tienda a cualquier distribución p_μ, eligiendo un conjunto de probabilidades de transición que cumplan la Ecuación de balance detallado (4)!

⇓

Queremos que la distribución de equilibrio sea la de Boltzmann ⇒

$$p_\mu = \frac{e^{-\beta E_\mu}}{Z_c}, \quad p_\nu = \frac{e^{-\beta E_\nu}}{Z_c}$$

⇒ Reemplazando y reescribiendo (4)

A. $$\boxed{\frac{P(\mu\to\nu)}{P(\nu\to\mu)} = \frac{p_\nu}{p_\mu} = e^{-\beta(E_\nu - E_\mu)}} \qquad (5)$$

B. $$\sum_\nu P(\mu\to\nu) = 1 \qquad \text{(ecuación (1))}$$

* A. y B. son las condiciones que le imponemos a $P(\mu\to\nu)$

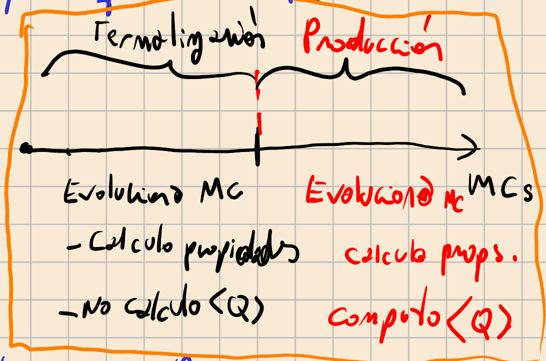
* Satisfaciendo A., B. y ergodicidad ⇒ La distribución de equilibrio en el proceso de Markov será la de Boltzmann

<u>Idea de la simulación de Monte Carlo</u>: dado un conjunto de probabilidades de transición, hacemos un programa que ejecute el proceso de Markov y genere una cadena de estados.

¿Cuántos? → Se ve un poco con la experiencia.

- Esperamos un "tiempo" (<u>varios pasos de</u> simulación) para que la distribución se acerque lo suficiente a la de Boltzmann.
- Luego promediamos las variables físicas como:

$$\boxed{Q_M = \frac{1}{M}\sum_{i=1}^{M} Q_{\mu_i}}$$



* <u>Tenemos libertad para elegir las probas</u> de transición. Por ejemplo:
  - $P(\mu\to\nu) \propto e^{-\frac{1}{2}\beta(E_\mu - E_\nu)}$ (es muy mala elección)
  - Algoritmo de Metrópolis.

* Los algoritmos típicos frecuentemente no son los mejores para resolver problemas nuevos.
* Se van a necesitar refinamientos pero el descripto, es el algoritmo básico de Monte Carlo

Tasas de aceptación (acceptance ratio)

* Teniendo en cuenta que $P(\mu \to \mu)$ puede ser no nula (proba de quedarse en casa) $\Rightarrow$ hacemos $\mu = \nu$ en (5):

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{p_\nu}{p_\mu} = \cancel{\frac{p_\mu}{p_\mu}}^{\,1} = e^{-\beta(E_\mu - E_\mu)} = 1 \quad ///.$$

$\Rightarrow$ Balance detallado se cumple para cualquier $P(\mu \to \mu)$

* Puede elegirse un $P(\mu \to \nu)$ y ajustar $P(\mu \to \mu)$ para compensar los cambios y así lograr que $\sum_\nu P(\mu \to \nu) = 1$ siga siendo válida.
* Hay que confirmar que $P(\mu \to \mu)$ se mantenga
* Es útil separar la probabilidad de transición en dos partes:

$$P(\mu \to \nu) = \underbrace{g(\mu \to \nu)}_{\substack{\text{Probabilidad} \\ \text{de selección}}} \; \underbrace{A(\mu \to \nu)}_{\substack{\text{Probabilidad} \\ \text{de aceptación}}}$$

$g(\mu \to \nu) \equiv$ {Probabilidad de que dado un estado inicial μ nuestro algoritmo genere un estado objetivo ν}

$A(\mu \to \nu) \equiv$ {Si el sistema está en μ, y el algoritmo genera un estado ν, deberíamos aceptar el estado y cambiar el sistema a ese estado con probabilidad $A(\mu \to \nu)$}

Terminamos Aquí clase 8

* Si no aceptamos ν, el sistema se queda en μ

* Hay libertad para elegir $A(\mu \to \nu)$ entre 0 y 1, como queramos.

* Eligiendo $A(\mu \to \nu) = 0 \Rightarrow$ Es elegir $P(\mu \to \mu) = 1$
(por supuesto, no sirve para una simulación real)

El balance detallado queda entonces:

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{g(\mu \to \nu)\, A(\mu \to \nu)}{g(\nu \to \mu)\, A(\nu \to \mu)} \qquad (6)$$

* $\dfrac{A(\mu \to \nu)}{A(\nu \to \mu)}$ puede tomar cualquier valor en $[0, \infty]$

* $g(\mu \to \nu)$ y $g(\nu \to \mu)$ pueden tomar cualquier valor

* $\sum_\nu P(\mu \to \nu) = 1$ se satisface: el sistema debe terminar en algún estado, pero puede ser también el estado de partida μ

<u>Hacer Monte Carlo</u> $\to$ Implementar un algoritmo que:

- genere nuevos estado ν a partir de los viejos estados μ con probas $g(\mu \to \nu)$
- Aceptar o rechazar los nuevos estados ν con proba $A(\mu \to \nu)$, que elegimos de manera tal, que se satisfaga (6).

$\Rightarrow$ Este procedimiento satisface todas las condiciones de las cadenas de Markov y produce una cadena de estados que cuando llega al equilibrio satisface Boltzmann

* Esto funciona pero si las probas $A(\mu \to \nu)$ de aceptación son muy bajas, el algoritmo es ineficiente.