



# El algoritmo de Metropolis-Hastings

Simulación mediante Monte Carlo con Cadenas de Márkov

---

Juan Esaul González Rangel

9 de diciembre de 2022

Centro de Investigación en Matemáticas

# Motivación

En el contexto del método de Monte Carlo, se busca calcular integrales de la forma

$$\mathcal{I}(h) = \int_A h(x) d\pi(x),$$

donde  $\pi(x)$  es una medida de probabilidad.

Este problema se reduce a ser capaces de tomar muestras aleatorias de  $\pi(x)$  mediante algún método, lo que **no siempre es posible o factible**.

# Motivación

En el contexto del método de Monte Carlo, se busca calcular integrales de la forma

$$\mathcal{I}(h) = \int_A h(x) d\pi(x),$$

donde  $\pi(x)$  es una medida de probabilidad.

Este problema se reduce a ser capaces de tomar muestras aleatorias de  $\pi(x)$  mediante algún método, lo que **no siempre es posible o factible**.

La alternativa es crear una cadena de Márkov  $(X_t)_t$  cuya distribución estacionaria sea  $\pi(x)$ .

$$\underbrace{\pi \mathbf{P} = \pi}_{\text{Caso discreto}}, \quad \underbrace{\pi(dy) = \int P(x, dy) \pi(x) dx}_{\text{Caso continuo}}.$$

# Un teorema útil de cadenas de Márkov

## Teorema (Convergencia al equilibrio) [Norris, 1998]

Sea  $\mathbf{P}$  la matriz de transición de una cadena de Márkov irreducible y aperiódica  $(X_n)_n$  y supongase que  $\mathbf{P}$  tiene distribución invariante  $\pi$ . Sea  $\lambda$  cualquier distribución inicial de  $(X_n)_n$ . Entonces

$$\mathbb{P}(X_n = j) \rightarrow \pi_j \quad \text{cuando } n \rightarrow \infty \quad \text{para todo } j.$$

Dada una medida de probabilidad  $\pi(x)$ , nuestro objetivo es construir una cadena de Márkov  $(X_n)_n$  que

- Sea irreducible.
- Sea aperiódica.
- $\pi(x)$  sea su distribución estacionaria.

Lograr esto significa que, eventualmente<sup>1</sup>, tomar muestras de  $(X_n)_n$  es aproximadamente igual a tomar muestras de  $\pi(x)$ .

---

<sup>1</sup>Cuando la cadena está *bien mezclada*

# Supuestos del algoritmo de Metropolis-Hastings

- Una densidad objetivo  $\pi(x)$  que podemos evaluar, o que es proporcional a una función  $\hat{\pi}(x)$  que podemos evaluar.
- Cualquier matriz de transición  $\mathbf{R}$  con entradas  $r_{ij}$  de una cadena de Márkov irreducible.

# Construcción del algoritmo de Metropolis-Hastings

Necesitamos definir una matriz de transición  $\mathbf{P}$  tal que  $\pi$  sea una distribución invariante.

Una condición suficiente es que  $\mathbf{P}$  y  $\pi$  cumplan las ecuaciones de balance detallado:

$$\pi(i)p_{ij} = \pi(j)p_{ji}.$$

Dada  $\mathbf{R}$ , definamos

$$\pi(i)p_{ij} = \min \{ \pi(i)r_{ij}, \pi(j)r_{ji} \}, \quad \forall i \neq j, \quad (1)$$

$$p_{jj} = 1 - \sum_{j \neq i} p_{ij} \geq 0. \quad (2)$$

Si  $(X_n)_n$  tiene matriz de transición  $\mathbf{P}$  con entradas  $p_{ij}$  definidas como en (1) y (2), entonces  $\pi$  es la distribución límite de  $(X_n)_n$ .

# Construcción del algoritmo de Metropolis-Hastings (continúa)

Notamos que  $p_{ij}$  tiene dos posibilidades:

1. Si  $\pi(i)r_{ij} \leq \pi(j)r_{ji}$ , entonces  $p_{ij} = r_{ij}$ .
2. Si  $\pi(i)r_{ij} > \pi(j)r_{ji}$ , entonces  $p_{ij} = \frac{\pi(j)}{\pi(i)} r_{ji} = \frac{\pi(j)r_{ji}}{\pi(i)r_{ij}} r_{ij}$ .

Es decir,

$$p_{ij} = r_{ij} \min \left\{ 1, \frac{\pi(j)r_{ji}}{\pi(i)r_{ij}} \right\}.$$

Por lo tanto, si simulamos (a partir de  $\mathbf{R}$ ) a  $Y_{n+1} = j | X_n = i$  con probabilidad  $r_{ij}$ , entonces  $X_{n+1} = Y_{n+1}$  con probabilidad  $\min \left\{ 1, \frac{\pi(j)r_{ji}}{\pi(i)r_{ij}} \right\}$ , y se mantiene igual con el complemento de esta probabilidad.



# El algoritmo de Metropolis-Hastings

## Algoritmo de Metropolis-Hastings

Sea  $X_0 = x_0$  un valor inicial

Para  $n = 1, 2, 3, \dots, n$  hágase lo siguiente:

1. Sea  $i = X_{n-1}$ ,
2. Generar  $Y_n$  de acuerdo con  $r_{ij}$ ,
3. Definir

$$X_{n+1} = \begin{cases} Y_n & \text{con probabilidad } \min \left\{ 1, \frac{\pi(j)r_{ji}}{\pi(i)r_{ij}} \right\}, \\ i & \text{con probabilidad } 1 - \min \left\{ 1, \frac{\pi(j)r_{ji}}{\pi(i)r_{ij}} \right\}. \end{cases}$$

## Ejemplo: La distribución de Rayleigh

La función de densidad de Rayleigh está dada por:

$$f(x) = \frac{x}{\sigma^2} e^{-x^2/(2\sigma^2)}, \quad x \geq 0, \sigma > 0.$$

El algoritmo de M-H para obtener una muestra de v.a. Rayleigh a partir de una  $\chi^2$  es

1. Definir  $p_{ij} \sim \chi^2(i)$ ,
2. Generar  $X_0$  desde  $\chi^2(1)$ ,
3. Para  $i = 1, 2, \dots, n$ :

Generar  $Y$  desde  $\chi^2(X_{i-1})$ ,

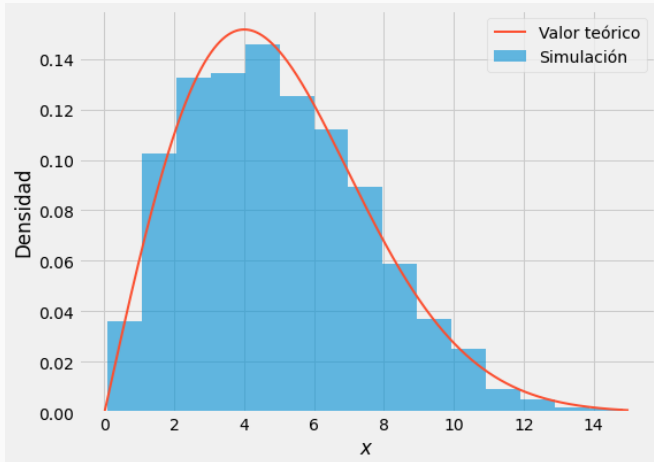
Generar  $U \sim \text{Uniforme}(0,1)$ ,

Calcular  $\rho(X_{i-1}, Y) = \frac{f(Y)p_{X_{i-1}Y}}{f(X_{i-1})p_{YX_{i-1}}}$ ,

Si  $U \leq \rho(X_{i-1}, Y)$ , aceptar  $X_i = Y$ , en caso contrario,  $X_i = X_{i-1}$ .

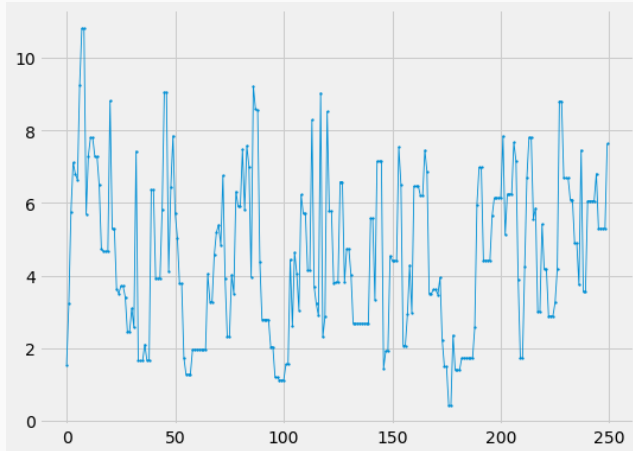
## Ejemplo: La distribución de Rayleigh

Se realizó la simulación con  $\sigma = 4$  y 10,000 iteraciones. Histograma de los datos simulados comparados con la densidad teórica



## Ejemplo: La distribución de Rayleigh

Con  $\sigma = 4$  y 10,000 iteraciones, una porción de la gráfica de la realización de la cadena es la siguiente



## Ejemplo: La densidad de Rosenbrock

La densidad de Rosenbrock es

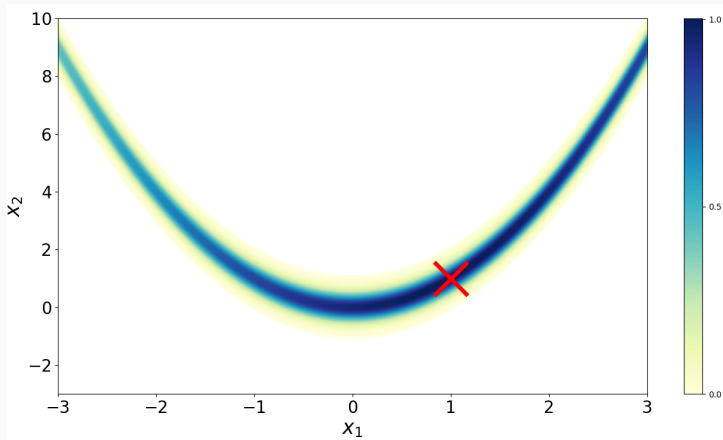
$$\pi^*(x_1, x_2; a, b) \propto \exp \left\{ -\frac{(a - x_1)^2 + b(x_2 - x_1^2)^2}{20} \right\}.$$

Podemos crear una muestra de esta distribución usando la densidad de transición Normal,  $p_{ij} = j|i \sim N(i, \sigma^2)$ , es decir  $j = i + N(0, \sigma^2)$ .

Notamos que  $p_{ij} = p_{ji}$ , por lo tanto, las variables  $Y_n$  se aceptan con probabilidad  $\min \left\{ 1, \frac{\pi^*(j)}{\pi^*(i)} \right\}$ . A este tipo de variante del algoritmo, se le llama *caminata aleatoria Metropolis-Hasting*.

## Ejemplo: La densidad de Rosenbrock

Gráfica de la densidad de Rosenbrock con  $a = 1$  y  $b = 100$ . Las partes más oscuras indican mayor probabilidad.



## Ejemplo: La densidad de Rosenbrock

Simulación mediante M-H de tres muestras de la densidad de Rosenbrock.



## Aspectos importantes a notar

- Aunque el algoritmo de Metropolis Hastings garantiza la construcción de una cadena de Márkov cuya distribución estacionaria es la densidad que buscamos, el tiempo de convergencia es afectado por el valor inicial de la cadena, el tamaño de los pasos, y la condición de rechazo.



## Aspectos importantes a notar

- Aunque el algoritmo de Metropolis Hastings garantiza la construcción de una cadena de Márkov cuya distribución estacionaria es la densidad que buscamos, el tiempo de convergencia es afectado por el valor inicial de la cadena, el tamaño de los pasos, y la condición de rechazo.
- Las observaciones de la cadena eventualmente se distribuyen con la ley buscada, pero esto no necesariamente sucede desde el principio, y es necesario descartar estos primeros valores (período de *burn-in*).

## Aspectos importantes a notar


- Aunque el algoritmo de Metropolis Hastings garantiza la construcción de una cadena de Márkov cuya distribución estacionaria es la densidad que buscamos, el tiempo de convergencia es afectado por el valor inicial de la cadena, el tamaño de los pasos, y la condición de rechazo.
- Las observaciones de la cadena eventualmente se distribuyen con la ley buscada, pero esto no necesariamente sucede desde el principio, y es necesario descartar estos primeros valores (período de *burn-in*).
- El algoritmo de Metropolis-Hastings es un caso general que engloba a varios casos específicos más (Langevin, muestreo de Gibbs, algoritmo de Metropolis, etc.). Además, existen otros algoritmos en MCMC que tienen distintos objetivos.

Dentro de la rama de simulación, los métodos de tipo Monte Carlo tienen un mayor alcance cuando se utilizan cadenas de Márkov para aproximar distribuciones difíciles. El algoritmo de Metropolis-Hastings es el más utilizado y uno de los más generales para crear estas cadenas.

El uso adecuado del algoritmo de Metropolis-Hastings, implica controlar el tiempo de convergencia y disminuir la cantidad de valores no deseados, para estos detalles técnicos consultar [Robert and Casella, 2004] y [Robert, 2015].

¡Gracias!

## Referencias i

-  Chib, S. and Greenberg, E. (1995).  
**Understanding the Metropolis-Hastings algorithm.**  
*The American Statistician*, 49(4):327–335.
-  Grimmett, G. and Stirzaker, D. (2020).  
***Probability and random processes.***  
Oxford university press.
-  Gundersen, G. (2019).  
**Why Metropolis–Hastings Works.**
-  Norris, J. R. (1998).  
***Markov chains.***  
Number 2. Cambridge university press.



Rizzo, M. L. (2019).

***Statistical computing with R.***

Chapman and Hall/CRC.



Robert, C. P. (2015).

**The Metropolis-Hastings algorithm.**



Robert, C. P. and Casella, G. (2004).

***Monte Carlo statistical methods, volume 2.***

Springer.

## Código para la distribución de Rayleigh

```
sigma = 4

for i in range(1, n_iters_r):
    curr_r = samples_r[i-1]
    prop_r = np.random.chisquare(df=curr_r)
    alpha = (rayleigh(prop_r, sigma)*chi2.pdf(curr_r, df=prop_r)) /
    ( rayleigh(curr_r, sigma)*chi2.pdf(prop_r, df=curr_r) )
    if np.random.uniform() < alpha:
        curr_r = prop_r
    samples_r[i] = curr_r
```

Código usado para esta presentación:

*[https://github.com/soy-esaul/Metropolis-Hastings\\_slideshow](https://github.com/soy-esaul/Metropolis-Hastings_slideshow)*

