

# **MAT-468: Sesión 12, Métodos MCMC**

**Felipe Osorio**

<http://fosorios.mat.utfsm.cl>

Departamento de Matemática, UTFSM



## Métodos Markov Chain Monte Carlo (MCMC)

Anteriormente se indicó que nuestro interés era, por ejemplo, aproximar

$$\theta = \int h(\mathbf{x}) f(\mathbf{x}) d\mathbf{x},$$

usando una muestra (independiente) generadas desde  $f$ .

Una estrategia diferente es obtener una muestra  $\mathbf{x}_1, \dots, \mathbf{x}_n$  distribuida **aproximadamente** desde  $f$  sin necesidad de simular **directamente** desde  $f$ .

### Objetivo:

Usar una **cadena de Markov** ergódica con distribución estacionaria  $f$ .



# Métodos Markov Chain Monte Carlo (MCMC)

## Definición 1:

Una cadena de Markov  $\{X_t\}_{t \in \mathbb{N}}$  es una secuencia de variables aleatorias

$$X_0, X_1, X_2, \dots, X_t, \dots$$

tal que la distribución de probabilidades de  $X_t$  dado el pasado, depende sólo de  $X_{t-1}$ .

Esta distribución condicional es llamada **kernel de transición**,

$$X_{t+1}|X_0, X_1, \dots, X_t \sim K(X_t, X_{t+1}).$$

## Ejemplo:

Una secuencia de variables aleatorias  $\{X_t\}$  es una **caminata aleatoria** si satisface

$$X_{t+1} = X_t + \epsilon_t,$$

donde  $\epsilon_t$  es generado independientemente de  $X_t, X_{t-1}, \dots$ .<sup>1</sup> Por ejemplo, si  $\epsilon_t \sim N(0, 1)$ , independiente de  $X_t$ . De este modo,

$$K(X_t, X_{t+1}) \stackrel{d}{=} N(X_t, 1).$$

---

<sup>1</sup>Si la distribución de  $\epsilon_t$  es simétrica en torno de cero, decimos que la secuencia es llamada **caminata aleatoria simétrica**.



# Métodos Markov Chain Monte Carlo (MCMC)

## Definición 2:

Un **método Monte Carlo de cadena de Markov (MCMC)** para simular desde una distribución  $f$  es cualquier procedimiento para producir una cadena de Markov ergódica  $\{X_t\}$  cuya **distribución estacionaria**<sup>2</sup> es  $f$ , y el kernel debe satisfacer

$$\int_{\mathcal{X}} K(x, y) f(x) \, dx = f(y)$$

## Observación:

Usar una cadena  $\{X_t\}$  producida por un método MCMC con distribución estacionaria  $f$  es esencialmente equivalente a usar una muestra IID desde  $f$ . Debido a la propiedad de **ergodicidad** asegura

$$\frac{1}{M} \sum_{t=1}^M h(X_t) \xrightarrow{\text{a.s.}} \mathbf{E}_f[h(X)].$$

---

<sup>2</sup>En efecto, si  $X_t \sim f$  entonces  $X_{t+1} \sim f$



# Métodos Markov Chain Monte Carlo (MCMC)

*Ejemplo:*

Considere la cadena de Markov

$$X_{t+1} = \rho X_t + \epsilon_t, \quad \epsilon_t \sim N(0, 1),$$

para  $t = 0, 1, \dots, n$ , con  $X_0 \sim N(0, 1)$  y  $\rho = 0.9$ .

Note que la **distribución estacionaria** es dada por

$$N\left(0, \frac{1}{1 - \rho^2}\right).$$



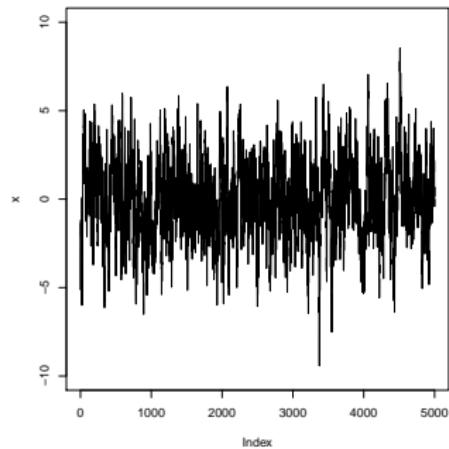
# Métodos Markov Chain Monte Carlo (MCMC)

```
# simula una cadena de Markov
> Nsim <- 5000
> rho <- 0.9
> x <- double(Nsim)
> x[1] <- rnorm(1)
> for (i in 2:Nsim) {
>   eps <- rnorm(1)
>   x[i] <- rho * x[i-1] + eps
> }

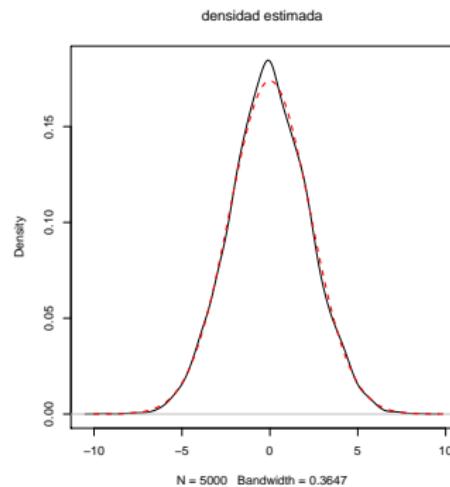
# explora la cadena y su distribución
> plot(x, type = "l", ylim = c(-10,10))
> z <- density(x)
> x0 <- seq(-10, 10, length = 500)
> y0 <- dnorm(x0, mean = 0, sd = 1 / sqrt(1 - rho^2))
> plot(z, main = "densidad estimada", font.main = 1)
> lines(x0, y0, col = "red", lwd = 2, lty = 2)
```



# Métodos Markov Chain Monte Carlo (MCMC)



(a)



(b)



## Algoritmo Metropolis-Hastings

El **algoritmo Metropolis-Hastings (MH)** permite construir una cadena de Markov  $\{X_t\}$  con distribución estacionaria  $f$  imponiendo un **mínimo de requerimientos** sobre la densidad objetivo.

Considere una función objetivo  $f$ . El procedimiento escoge una densidad condicional  $q(y|x)$ , conocida como **distribucion propuesta** (o instrumental). Este algoritmo es de utilidad cuando  $q(\cdot|x)$  es fácil de simular o bien es simétrica, esto es  $q(x|y) = q(y|x)$ .

Típicamente se requiere que la razón

$$f(y)/q(y|x),$$

sea conocida salvo una constante independiente de  $x$ .



# Algoritmo Metropolis-Hastings

---

## Algoritmo 1: Metrópolis-Hastings.

---

**Entrada:**  $x_t$ .

**begin**

1    Generar  $Y_t \sim q(y|x_t)$ .

2    Hacer

$$X_{t+1} = \begin{cases} Y_t & \text{con probabilidad } \rho(x_t, Y_t), \\ x_t & \text{con probabilidad } 1 - \rho(x_t, Y_t), \end{cases}$$

donde

$$\rho(x, y) = \min \left\{ \frac{f(y)}{f(x)} \frac{q(x|y)}{q(y|x)}, 1 \right\}.$$

4 **end**

---



# Algoritmo Metropolis-Hastings

---

## Algoritmo 2: Metrópolis-Hastings.

---

**Entrada:**  $x_t$ .

```
1 begin
2   Generar  $Y_t \sim q(y|x_t)$ .
3   Simular  $U \sim U(0,1)$ .
4   if  $U \leq \rho(x_t, Y_t)$  then
5     return  $X_{t+1} = Y_t$ 
6   else
7     return  $X_{t+1} = x_t$ 
8   end
9 end
```

---

La probabilidad  $\rho(x, y)$ , dada por:

$$\rho(x, y) = \min \left\{ \frac{f(y)}{f(x)} \frac{q(x|y)}{q(y|x)}, 1 \right\},$$

es conocida como **probabilidad de aceptación Metropolis-Hastings**.



# Algoritmo Metropolis-Hastings

El algoritmo MH siempre acepta valores  $y_t$  tal que

$$\frac{f(y_t)}{q(y_t|x_t)} > \frac{f(x_t)}{q(x_t|y_t)},$$

y sólo en el caso simétrico es que la aceptación depende de  $f(y_t)/f(x_t)$ .

Una característica importante del algoritmo es que puede aceptar valores tal que la razón decrece.

## *Observación:*

Cualquier distribución propuesta permitirá obtener muestras aleatorias desde  $f$ . Sin embargo, la razón de convergencia a la distribución estacionaria depende de la relación entre  $q(\cdot|\cdot)$  y  $f(\cdot)$ .



# Algoritmo Metropolis-Hastings

*Ejemplo:*

Considere un algoritmo Metropolis-Hastings con distribución estacionaria  $N(0, 1)$  y las siguientes propuestas para  $q(\cdot|x)$ :

a.  $q(\cdot|x) \stackrel{d}{=} N(x, 0.5)$

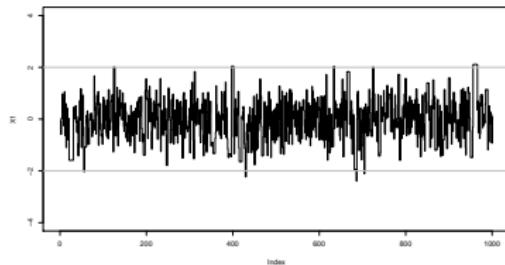
b.  $q(\cdot|x) \stackrel{d}{=} N(x, 0.1)$

c.  $q(\cdot|x) \stackrel{d}{=} N(x, 10)$

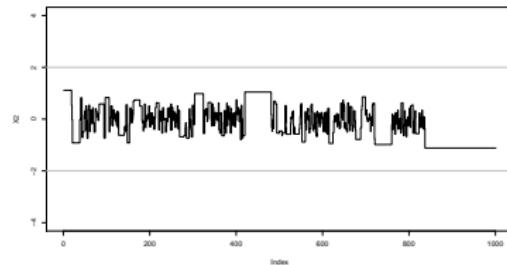
Se simuló cadenas de tamaño  $M = 5000$  para cada caso. En el siguiente gráfico se presenta la porción final de la cadena (últimas 1000 observaciones)



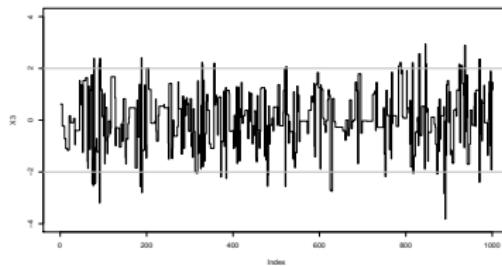
# Algoritmo Metropolis-Hastings



(a)



(b)



(c)



# Algoritmo Metropolis-Hastings

*Ejemplo:*

Sea  $\mathbf{X} = (X_1, X_2)^\top$  es un vector aleatorio bidimensional con densidad

$$f(\mathbf{x}) = k \exp\{-(x_1^2 x_2^2 + x_1^2 + x_2^2 - 8x_1 - 8x_2)/2\},$$

donde  $k \approx 1/20216.335877$  es la constante de normalización.

Suponga que se desea estimar  $h = E(X_1)$ , via

$$\hat{h} = \frac{1}{M} \sum_{t=1}^M X_{1t},$$

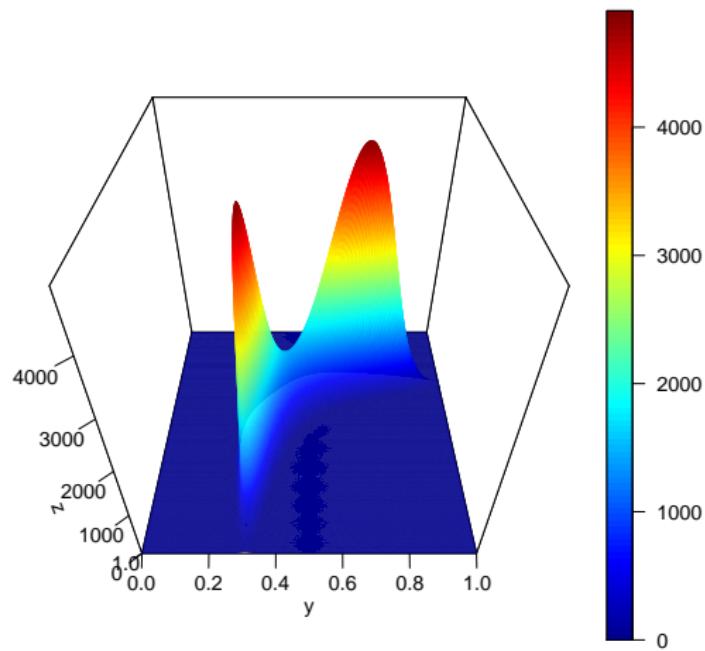
usando una caminata aleatoria para generar una muestra dependiente  $\{\mathbf{X}_t\}$  desde  $f(\mathbf{x})$ .<sup>3</sup>

---

<sup>3</sup>En efecto,  $\mathbf{X}_{t+1} = \mathbf{X}_t + \epsilon_t$ .



# Algoritmo Metropolis-Hastings



# Algoritmo Metropolis-Hastings

---

## Algoritmo 3: Random walk sampler.

---

```
1 begin
2   Iniciar  $\mathbf{x}_1 = (x_{11}, x_{12})^\top$ , hacer  $t = 1$ 
3   Simular  $\mathbf{z} = (z_1, z_2)^\top \sim N_2(\mathbf{0}, \mathbf{I})$  independientemente, y hacer
       $\mathbf{y} = \mathbf{x}_t + 2\mathbf{z}$ .
4   Calcular
      
$$\rho(\mathbf{x}_t, \mathbf{y}) = \min \left\{ \frac{f(\mathbf{y})}{f(\mathbf{x}_t)}, 1 \right\}.$$

5   Generar  $u \sim U(0, 1)$ .
6   if  $U \leq \rho(\mathbf{x}_t, \mathbf{y})$  then
7     | return  $\mathbf{x}_{t+1} = \mathbf{y}$ 
8   else
9     | return  $\mathbf{x}_{t+1} = \mathbf{x}_t$ 
10  end
11  Incrementar  $t$  en 1. Si  $t = M$  detener, sino volver a Paso 2.
12 end
```

---

*Tarea:*

Correr el algoritmo anterior para producir  $M = 10^5$  muestras.<sup>4</sup>

---

<sup>4</sup> El valor verdadero es  $E(X_1) \approx 1.85997$



# Algoritmo Metropolis-Hastings

## *Observaciones:*

Algoritmo MH es un método genérico que puede ser usado para generar virtualmente cualquier distribución objetivo, sin importar su dimensionalidad y complejidad. Sin embargo

- ▶ Las muestras generadas son altamente correlacionadas.
- ▶ Típicamente se requiere bastante tiempo hasta que la cadena alcance su estado estacionario.
- ▶ Los estimadores obtenidos por MCMC tienden a tener varianzas mayores que sus contrapartes obtenidas por muestreo independiente desde la distribución objetivo.



## Algoritmo Gibbs sampler

Este procedimiento es particularmente útil para la generación de vectores aleatorios  $n$ -dimensionales. En este caso la cadena de Markov es construida desde una secuencia de distribuciones condicionales.

Gibbs sampling es ventajoso es más fácil general desde las distribuciones condicionales que desde la distribuciones condicionales que desde la distribución conjunta.

Suponga que se desea simular un vector aleatorio  $\mathbf{X} = (X_1, \dots, X_n)^\top$  de acuerdo con la densidad objetivo  $f(\mathbf{x})$ . Sea

$$f(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n),$$

la densidad condicional del  $i$ -ésimo componente  $X_i$  dado los restantes. Considere el siguiente algoritmo.



# Algoritmo Gibbs Sampler

---

## Algoritmo 4: Gibbs sampler.

---

**Entrada:**  $x_t$  dado.

**Salida :** Genera  $y = (y_1, y_2, \dots, y_n)^\top$  desde  $f(x)$ .

```
1 begin
2   Simular  $y_1$  desde la densidad condicional
3    $f(x_1|x_{t,2}, \dots, x_{t,n})$ 
4   for  $i = 2$  to  $n - 1$  do
5     Generar  $y_i$  desde
6      $f(x_i|y_1, \dots, y_{i-1}, x_{t,i+1}, \dots, x_{t,n})$ 
7   end
8   Simular  $y_n$  desde hacer
9    $f(x_n|y_1, \dots, y_{n-1})$ 
10  return  $x_{t+1} = y$ 
11 end
```

---



# Algoritmo Gibbs Sampler

## *Observaciones:*

- ▶ Todas las muestras generadas por el Gibbs sampler son aceptadas, en contraste al algoritmo Metropolis-Hastings.
- ▶ Bajo condiciones suaves la distribución límite del proceso  $\{\mathbf{X}_t\}_{t \geq 1}$  generado por el algoritmo de Gibbs es precisamente  $f(\mathbf{x})$
- ▶ El algoritmo Gibbs sampler actualiza los elementos de forma determinística (siguiendo el orden  $1, 2, \dots, n$ ). Es posible escoger las coordenadas de forma aleatoria desde  $\mathcal{U}\{1, \dots, n\}$ .



## Algoritmo Gibbs Sampler

*Ejemplo:*

Suponga que se desea muestrear desde

$$f(x, y) = k \exp\{-(x^2y^2 + x^2 + y^2 - 8x - 8y)/2\},$$

donde  $k \approx 1/20216.335877$  usando el [Gibbs sampler](#).

Escribiendo

$$f(x, y) = k_1(y) \exp\left\{-\frac{1+y^2}{2}\left(x - \frac{4}{1+y^2}\right)^2\right\},$$

donde  $k_1(y)$  sólo depende de  $y$ . Note que condicional a  $y$ ,  $X$  tiene distribución normal con media  $4/(1+y^2)$  y varianza  $1/(1+y^2)$ .



# Algoritmo Gibbs Sampler

---

## Algoritmo 5: Gibbs sampler.

---

```
1 begin
2   Iniciar  $x_1$  y  $y_1$ , hacer  $t = 1$ 
3   Generar  $z \sim N(0, 1)$ , tomar  $a = 1/(1 + x_t^2)$  y hacer
      
$$y_t = 4a + z\sqrt{a}$$

4   Simular  $z \sim N(0, 1)$ , tomar  $b = 1/(1 + y_t^2)$  y hacer
      
$$x_t = 4b + z\sqrt{b}$$

5   Incrementar  $t$  en 1. Si  $t = M$  detener, sino volver a Paso 2.
6 end
```

---

### Tarea:

Correr el algoritmo anterior para producir  $M = 10^5$  muestras desde  $f(\mathbf{x})$ .

