UNIVERSIDAD DE SEVILLA

FACULTAD DE MATEMÁTICAS

DOBLE MÁSTER EN MATEMÁTICAS Y PROFESORADO EN ENSEÑANZA SECUNDARIA OBLIGATORIA Y BACHILLERATO, FORMACIÓN PROFESIONAL Y ENSEÑANZA DE IDIOMAS

MÉTODO DE MONTE CARLO. USO Y APLICACIONES EN INFERENCIA BAYESIANA.

Trabajo de fin de máster presentado por

Ana Rosa Chaves López

Tutora: Inmaculada Barranco Chamorro

Firma de la alumna

Firma de la tutora

Sevilla, noviembre de 2021

Abstract

It's sure that, nowadays, Statistics is one of the most applicable areas in Mathematics. Because of that, it's so important to study new methods to allow as to improve this important discipline. So this research work borns from the neccesity of predicting future events, under the knowledge of past events.

First, it will be introduced some definitions and results about Bayesian Statistics and Markov chains. This will be shown in a discrete way, althought we will use them in continuous form.

Talking about the algorithm, it will be presented how it works, and what properties it follows, seeing that it generates a Markov chain, and some variants. We'll see that this algorithm is the one to be used to simulate samples from unknown distributions.

Later, it'll be introduced the Gibbs sampler, that will be used to simulate sample from that functions that, ones you fixed their values, we'll obtain easy simple forms.

Then, we will learn something about other methods like slice sampling or Hamiltonian dynamics.

At last, we'll be into a real-life example. It's about studying the number of electric failure happened in a set of trains from 1992 to 1998, in order to calculate predictions about what number of failures will take place in a future. It well be seen, Bayesian Statistics will be used to estimate futures failures.

A todas las personas que me hacen aprender.

Resumen

Sin lugar a dudas, hoy en día la estadística es una de las ramas de las matemáticas que más aplicaciones tiene. De ahí, el porqué investigar en métodos y procedimientos que permitan mejorar la disciplina es tan importante. Así, este trabajo comienza con la necesidad de predecir fenómenos futuros, bajo el conocimiento de fenómenos pasados.

En primer lugar se desarrollarán algunas definiciones y resultados sobre estadística bayesiana y cadenas de Markov. Se desarrollarán en su versión discreta para su mejor compresión, aunque serán utilizados en su forma continua.

Sobre el algoritmo de Metropolis-Hasting, se conocerán tanto su funcionamiento como algunas propiedades, teniendo en cuenta que genera una cadena de Markov y algunas de las variantes que presenta el algoritmo.

A continuación, se tratará el muestreo de Gibbs, que se utilizará para simular muestras para aquellas funciones que, una vez fijados los distintos parámetros, se obtienen funciones de distribución condicionadas conocidas y con formas sencillas.

Casi en última instancia, se puntualizarán otros métodos como son el *slice sampling* y la dinámica hamiltoniana.

Por último, se abarcará un ejemplo de la vida real. Se trata de estudiar el número de fallos de tipo eléctrico que se producen en una serie de trenes estudiados de 1992 a 1998, con el fin de calcular predicciones sobre qué cantidad de fallos podrían ocasionarse en un futuro cercano. Como se verá, se utilizará la inferencia bayesiana, ya que, bajo el conocimiento de los fallos de años anteriores, se estimarán los posibles fallos que se cometerán en los años siguientes.

Agradecimientos

Agradecer a Inmaculada Barranco por permitirme desarrollar este trabajo con su ayuda y asesoramiento, sin los cuales no hubiera sido posible. Además, agradecer a Alfonso Suárez y Marta Sánchez de la Universidad de Cádiz por su colaboración. Gracias a mi familia por apoyarme siempre.

Ana Rosa

noviembre 2021

Índice general

CAPÍTULO

1

Conceptos previos

1.1 Inferencia Bayesiana

Hasta ahora, los procedimientos utilizados en la estadística (de aquí en adelante, denominada como clásica), llevan a conocer el comportamiento de alguna variable aleatoria, o bien a trabajar datos muestrales en función de uno o varios parámetros fijos. Ahora bien, si lo que se quiere es determinar el parámetro, no como valor fijo, sino como variable con distribución propia, se está hablando de estadística bayesiana.

De manera algo más formal, en estadística clásica se tiene como objetivo aproximar un parámetro θ fijo. Dada una población descrita por una variable aleatoria X dependiente del parámetro θ, se extrae una muestra aleatoria $X_1, \ldots, X_n$ y se hace inferencia del parámetro en función de la misma. En contrapartida, en el enfoque bayesiano θ es una variable aleatoria que puede ser modelada con su propia distribución de probabilidad. Esta distribución, que se denomina **distribución a priori**, se estima antes de observar la muestra. Después, una vez observada, la distribución a priori se actualiza con esa nueva información y se obtiene, mediante el Teorema de Bayes, la denominada **distribución a posteriori**.

A continuació, se indagará de forma más detallada en los conceptos básicos de la estadística bayesiana, como pueden ser la distribución a priori, la noción de función de verosimilitud y la distribuación a posteriori.

1.1.1 Distribución a priori y función de verosimilitud

Sea X una variable aleatoria, con $X \sim f(x|\theta)$, siendo f la función de densidad con $\theta \in \Theta \subseteq \mathbb{R}^m, m \geq 1$. Así, $\theta = (\theta_1, \ldots, \theta_m)$ es una variable aleatoria en sí misma, sobre la cual se propone la llamada **distribución a priori** $\pi(\theta)$, con soporte Θ. Esta contiene el conocimiento previo a la observación de los datos de la muestra.

Si $X_1, \ldots, X_n$ es una muestra tomada de X, se dice que la **función de verosimilitud** es aquella función de densidad conjunta de esa muestra, denotada por $f(\underline{x}|\theta) = f(x_1, \ldots, x_n|\theta)$. Si la muestra es una muestra aleatoria simple, $x_1, /dots, x_n$ son independientes e identicamente distribuidas, por lo que

$$f(x_1, \ldots, x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

.

Como transición entre la distribución a priori junto con la función de verosimilitud y la distribución a posteriori, se aplica el Teorema de Bayes.

Teorema 1.1. ***Teorema de Bayes*** *(Bayes, 1763)*

Sea $A_1, \ldots, A_m$ un conjunto de sucesos exhaustivo y completo, es decir, que cumple lo siguiente:

- $\bigcup_{i=1}^{m} A_i = \Omega$
- $A_i \cap A_j =$, *con* $i \neq j$,

y sea B un suceso tal que $P(B) > 0$, se cumple que

$$P(A_i|B) = \frac{P(A_i)P(B|A_i)}{\sum_{j=1}^{m} P(A_j)P(B|A_j)}.$$

1.1.2 Distribución a posteriori

Ahora se procede a definir la distribución a posteriori, que es una distribución del parámetro θ dada la muestra, denotándose por

$$\pi(\theta|\underline{x}) = \pi(\theta|x_1, \ldots, x_n).$$

Aplicando el Teorema de Bayes para el caso continuo, se tiene que

$$\pi(\theta|\underline{x}) = \frac{f(\underline{x};\theta)}{m(\underline{x})} = \frac{\pi(\theta)f(\underline{x}|\theta)}{m(\underline{x})},$$

donde

- $\pi(\theta)$ es la distribución a priori,
- $f(\underline{x};\theta)$ es la función de densidad conjunta de $(\underline{x},\theta)$
- y $m(\underline{x})$ la densidad marginal de la muestra, que se define por

$$m(\underline{x}) = \int f(\underline{x};\theta)d\theta = \int \pi(\theta)f(\underline{x}|\theta)d\theta.$$

Al no depender $m(\underline{x})$ del parámetro θ, se puede concluir que

$$\pi(\theta|\underline{x}) \propto \pi(\theta)f(\underline{x}|\theta).$$

Además, se podrán hacer predicciones de futuro con lo que se define como densidad predictiva. A partir de p observaciones $\underline{y} = (y_1, \ldots, y_p)$ independientes de $\underline{x} = (x_1, \ldots, x_n)$, se calcula como

$$m(\underline{y}|\underline{x}) = \int_\Theta \pi(\theta|\underline{x})f(\underline{y}|\theta)d\theta.$$

Se clarifican a continuación las nociones de este capítulo con un ejemplo.

Ejemplo 1.1. Sea X una variable aleatoria tal que $X \sim Ber(p)$, con $0 < p < 1$ siendo p la proporción de éxitos. Por tanto, la función de probabilidad correspondiente será

$$f(x|p) = p^x(1-p)^{1-x} \quad \text{para } x = 0, 1.$$

Se supone una muestra aleatoria simple $X_1, \ldots, X_n$ de X. Así, se denota Y como el número de éxitos en n pruebas Bernouilli. Por tanto la función de verosimilitud será

$$P[Y = k|p] = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, \ldots, n,$$

donde k es el número de éxitos observados e $Y = \sum_{i=1}^{n} X_i$.

Ahora, el caso concreto será siguiente conjunto que contiene los valores de éxito, es decir, $(0.01, 0.02, 0.03, 0.05, 0.08, 0.10) \ni p$.

Como distribución a priori, se toman los valores $(0.15, 0.10, 0.01, 0.27, 0.34, 0.13)$.

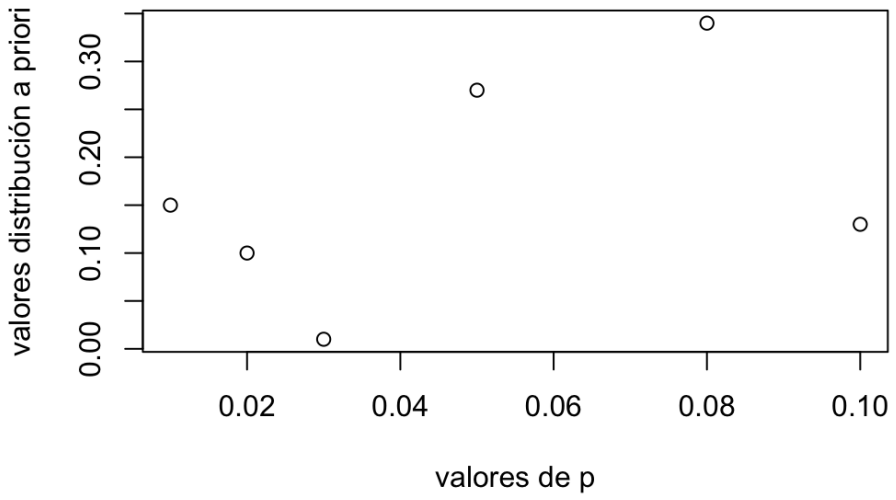

Figura 1: Distribución a priori

Se genera una muestra de cardinal 7 de X, teniéndose que $Y \sim B(7, p)$, por lo que la

función de verosimilitud quedaría

$$P[Y = k|p] = \binom{7}{k} p^k (1-p)^{7-k}, \quad k = 0, 1, \ldots, 7.$$

Se calcula para cada uno de los valores de p su imagen mediante la función de verosimilitud. Con eso, se calcula la distribución a posteriori de cada uno de los valores.

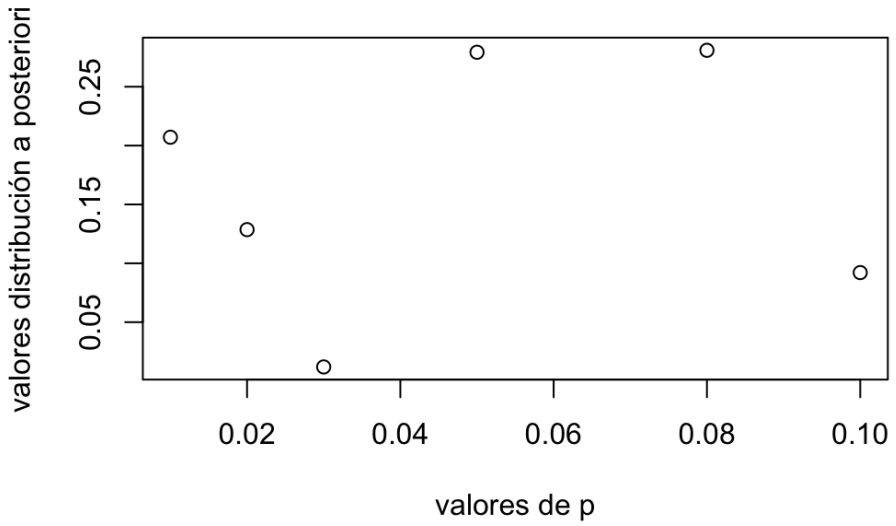

Figura 2: Distribución a posteriori

Se pueden ver todos los valores recogidos en la siguiente tabla:

p	priori	verosimilitud	posteriori
0.01	0.15	0.9320653	0.20709505
0.02	0.10	0.8681255	0.12859220
0.03	0.01	0.8079828	0.01196835
0.05	0.27	0.6983373	0.27929367
0.08	0.34	0.5578466	0.28094790
0.10	0.13	0.4782969	0.09210284

Con esto, si se quisiera calcular la densidad predictiva, se procedería de la siguiente manera:

Sea Z una observación de la muestra a calcular. Al ser una observación $Z \sim Ber(p)$, esto es $Z \in \{0, 1\}$, luego $P[Z = 0|p] = 1 - p$ y $P[Z = 1|p] = p$. Por tanto

$$P[Z = 0|\underline{x}] = \sum \pi(p_i|\underline{x})P[Z = 0|p_i] = \sum \pi(p_i|\underline{x})(1-p_i) = 0.20709505 \cdot (1-0.01) + \cdots = 0.9493474$$

$$P[Z = 1|\underline{x}] = \sum \pi(p_i|\underline{x})P[Z = 1|p_i] = \sum \pi(p_i|\underline{x})(p_i) = 0.20709505 \cdot 01 + \cdots = 0.05065264$$

Se puede ver la codificación de este ejemplo en el Apéndice A.1.

1.2 Resultados básicos de cadenas de Markov

A la hora de implementar inferencia bayesiana, a menudo se necesita el uso de métodos computacionales de simulación. En concreto, son de especial interés aquellos basados en el muestreo de valores de Monte Carlo, que se generan de una distribución a posteriori normal multivariante $h(\theta|x)$, $\theta \in \Theta \subseteq \mathbb{R}^k$. El uso de los métodos basados en simulaciones es una de las maneras de tratar las formas analíticas, a veces complejas, de la inferencia bayesiana. Dependiendo de la complejidad de la distribución a posteriori $h(\cdot)$, la evaluación de los sumatorios a posteriori como $E[g(\theta)|x]$ puede realizarse por métodos de Monte Carlo (MC) clásicos, generando muestras independientes e idénticamente distribuidas (i.i.d.) de la propia distribución objetivo, o de alguna distribución apropiada utilizando muestreo de importancia, cuya construcción involucre a la distribución objetivo.

Para problemas complejos se ha vuelto común, especialmente desde el año 1990, usar los métodos MC generales basados en la simulación de una cadena de Markov (homogénea) construida para tener una distribución $\pi(\theta)$ ergódica similar a la distribución objetivo, $\pi(\theta) \equiv h(\theta|x)$. Estos métodos, conocidos como **Métodos de Monte Carlo basados en Cadenas de Markov** (MCMC), utilizan muestras dependientes de θ e implican también resultados asintóticos complejos y la necesidad de una simulación de muestras de tamaños más grandes en comparación a los métodos MC clásicos.

El redescubrimiento de los métodos de MCMC por los estadísticos durante la década de 1990 llevó a progresos considerables en la simulación basada en los métodos inferenciales y,

en particular, análisis bayesianos para modelos que eran demasiado complejos para los métodos anteriores.

Dada la naturaleza de los métodos MCMC, es esencial entender el conocimiento de resultados básicos para cadenas de Markov, que se resumen en el siguiente apartado. Por simplicidad, se utilizará notación genérica para los estados de una cadena. Se describirán algunos métodos a lo largo del escrito, incluyendo las cadenas junto a Metropolis-Hasting, muestreo de Gibbs, *slice sampling*, y el método hamiltoniano de Monte Carlo.

En primer lugar, antes de desarrollar los diferentes algoritmos, se exponen diferentes definiciones y resultados indispensables para la construcción de las sucesivas secciones. Este capítulo se centrará en el estudio de los procesos estocásticos, cadenas de Markov y propiedades de estas.

1.2.1 Definiciones y resultados

Para comenzar, se define la noción más básica necesaria para el desarrollo del tema: el concepto de proceso estocástico.

Definición 1.1. Un **proceso estocástico** es un conjunto de variables aleatorias que están definidas sobre un mismo espacio de probabilidad $\{U(t), t \in T\}$, donde T es un subconjunto de $\mathbb{R}$ que, sin pérdida de generalidad, puede ser considerado como un conjunto de índices temporales.

Cuando este conjunto T es de la forma $T = \{0, 1, 2, \dots\}$, el proceso estocástico suele escribirse como $\{U_n, n \geq 0\}$. El conjunto $\mathcal{U}$ de valores de las variables U_n es conocido como el **espacio de estados**.

Tener conocimiento sobre los estados pasados y presentes de un proceso normalmente informa sobre el comportamiento de los estados futuros. Cuando se condiciona sobre un estado presente dado y el comportamiento de los estados futuros no depende del pasado, se dice que ese proceso posee una **dependencia de Markov**. Un proceso $\{U_t, t \geq 0\}$ con la propiedad de condicionalidad independiente es conocida como cadena de Markov, y puede definirse de la siguiente forma:

Definición 1.2. Una **cadena de Markov** es un proceso $\{U_t, t \geq 0\}$ con la propiedad de condicionalidad independiente, cumpliéndose que la probabilidad de paso del estado anterior al estado actual, sólo depende de este último y no de los anteriores. Esto es,

$$P(U_{t+1} = u_j \in A | U_0 = u_0, \ldots, U_t = u_i) = P(U_{t+1} = u_j \in A | U_t = u_i) \equiv P_t(u_i, u_j),$$

para todos los sucesos A y $n \geq 0$.

La probabilidad $P_t(u_i, u_j)$ es conocida como **probabilidad de transición** o **función de transición** del estado u_i al estado u_j en el tiempo t, y una cadena particular puede ser definida enteramente por ella. Esta será la probabilidad de que el proceso pase de un estado a otro en un único paso.

Cuando la función de transición es invariante con respecto a t, se escribe $P(u_i, u_j)$ y la cadena de Markov es llamada **homogénea**. En lo siguiente, unicamente se tratarán cadenas de Markov homogéneas, por lo que se supondrán de esta manera.

Para un espacio de estados discreto, la cadena de Markov está enteramente definida por las probabilidades condicionales $P(u_j|u_i)$, i.e.

$$P(U_{n+1} = u_j | U_0 = u_0, \ldots, U_n = u_i) = P(U_{n+1} = u_j | U_n = u_i) \equiv p(u_i, u_j),$$

para todo $n \geq 0, u_i, u_j \in \mathfrak{U}$.

Además, se denotará la probabilidad de que una cadena esté en la posición i en el momento $t + 1$ como

$$\pi_i(t + 1) = P(U_{t+1} = u_i).$$

Esta probabilidad viene dada por la **ecuación de Chapman-Kolmogorov**, la cual suma la probailidad de estar en una posición concreta en el momento actual y la probabilidad de paso desde dicha posición hasta la posición u_i. Como consecuencia directa del teorema de la probabilidad

total se tiene la siguiente expresión:

$$\begin{aligned}\pi_i(t+1) = &\quad P(U_{t+1} = u_i) \\ &= \sum_k P(U_{t+1} = s_i | U_t = s_k) \\ &= \sum_k P(u_i, u_k)\pi_k(t).\end{aligned}$$

Compactificando la ecuación de Chapman-Kolmogorov se puede expresar la misma en forma matricial.

Definición 1.3. Se define la **matríz de probabilidad de transición o de paso** como la matriz cuyo elemento ij-ésimo viene dado por $P(u_i, u_j)$.

Por tanto, la ecuación de Chapman Kolmogorov se escribe como

$$\pi(t+1) = \pi(t)P,$$

siendo P la ya nombrada matriz de probabilidad de paso, la cual existe siempre y cuando el espacio de estados sea finito.

Y gracias a la iteratividad de la misma, es inmediato que

$$\pi(t) = \pi(t-1)P = (\pi(t-2)P)P = \pi(t-2)P^2.$$

De esta forma, finalmente

$$\pi(t) = \pi(0)P^t.$$

Concretamente, denotando $p_{ij}^{(n)}$ como la probabilidad de pasar del estado u_i al estdo u_j en n pasos, es decir,

$$p_{ij}^{(n)} = P(U_{t+n} = u_j | U_t = u_i),$$

se tiene que es el elemento ij-ésimo de la matriz P^n.

Por otro lado, cuando $\mathcal{U}$ es infinito no numerable y $F(u, v)$ es absolutamente continua, la función de transición puede definirse por la densidad $p(u, v) = \dfrac{\partial F(u, v)}{\partial v}$.

De momento, se supone una cadena de Markov discreta. Se tiene que

$$P(U_{n+1} = v) = \sum_u P(U_n = u)p(u, v) = \sum_u P(U_0 = u)p^n(u, v),$$

donde $p^n(u, v) = P(U_n = v|U_0 = u) = \sum_u p^{n-1}(u, z)p(z, v), n \geq 1$ define la función de transición de n pasos (en forma matricial del producto P^n). La construcción de una cadena de Markov está por tanto completamente determinada por la función de transición, siempre y cuando se dé una distribución inicial.

Definición 1.4. Se dice que una distribución de probabilidad $\pi(u), u \in \mathcal{U}$ es una **distribución estacionaria** si $\pi(v) = \sum_u \pi(u)p(u, v)$. En particular, una distribución inicial $P(U_0 = u) = \pi(u)$ es estacionaria si y solo si la distribución marginal de U_n es invariante sobre n, i.e. $P(U_n = u) = \pi(u), \forall n \geq 0$.

La existencia y unicidad de las distribuciones estacionarias dependen de ciertas características de la cadena, conocidas como irreducibilidad y recurrencia.

Definición 1.5. Una cadena es **irreducible** si puede alcanzar cualquier estado dentro de un número finito de transiciones, comenzando en el estado inicial.

Definición 1.6. Una cadena se dice **recurrente** si vuelve infinitas veces a cualquier estado inicial. Se dice que es **recurrente positiva** si el tiempo esperado del primer regreso a cualquier estado u es finito para todos los estados u.

Proposición 1.1. *La irreducibilidad implica recurrencia positiva si $\mathcal{U}$ es finito.*

Se ilustra a continuación, estas últimas nociones con un ejemplo.

Ejemplo 1.2. Se supone el siguiente espacio de estados (Lluvioso, Soleado, Nublado) y que el tiempo sigue un proceso de Markov. Así, la probabilidad del tiempo que va a hacer mañana únicamente depende del tiempo de hoy, y no del tiempo de días anteriores. En este caso, la observación de que llueva durante tres días seguidos no altera la probabilidad del tiempo de mañana en comparación con la situación de que llueva hoy, pero estuviera soleado la semana

pasada. Se supone que la probabilidad de transición de que hoy llueva a los siguientes estados viene dada por:

P(Llueva mañana | Llueva hoy)=0.5,

P(Soleado mañana | Llueva hoy)=0.25,

P(Nublado mañana | Llueva hoy)=0.25.

Este primer vector de la matriz de probabilidad de transición será $(0.5, 0.25, 0.25)$. Suponiendo que el resto de la matriz de transición es la dada a continuación, donde el elemento ij-ésimo denota la probabilidad de que mañana haga el correspondiente tiempo

$$P = \begin{pmatrix} 0.5 & 0.25 & 0.25 \\ 0.5 & 0 & 0.5 \\ 0.25 & 0.25 & 0.5 \end{pmatrix}$$

Notesé que esta cadena de Markov es irreducible, ya que todas las posiciones se comunican entre ellas.

Suponiendo que hoy está soleado, ¿cómo se espera el tiempo dentro de dos días? ¿Y siete días? Al ser en este caso $\pi(0) = (0\ 1\ 0)$ se tiene que

$$\pi(2) = \pi(0)P^2 = (0.375\ \ 0.25\ \ 0.375) \quad \text{y} \quad \pi(7) = \pi(0)P^7 = (0.4\ \ 0.2\ \ 0.4).$$

Por el contrario si hoy llueve, es decir $\pi(0) = (1\ 0\ 0)$, el tiempo esperado será

$$\pi(2) = (0.4375\ \ 0.1875\ \ 0.375) \quad \text{y} \quad \pi(7) = (0.4\ \ 0.2\ \ 0.4).$$

Se observa así en este ejemplo que después de una cantidad de tiempo suficiente, el tiempo esperado (7 días en este caso) es independiente del valor inicial. En otras palabras, la cadena ha alcanzado una distribución estacionaria, donde el valor de la probabilidad es independiente del valor inicial actual.

□

Proposición 1.2. *Una cadena irreducible y recurrente (con $\mathcal{U}$ finito) tiene una única distribución estacionaria. Por otro lado, si hay una distribución estacionaria $\pi(v)$ tal que $\lim_{n\to\infty} p^n(u, v) = \pi(v)$, entonces la distribución estacionaria es única y $\lim_{n\to\infty} P(U_n = v) = \pi(v)$.*

Por tanto, la convergencia a la distribución estacionaria π no está garantizada para una cadena irreducible y recurrente positiva. Sin embargo, existe una condición adicional que sí que lo garantiza:

Definición 1.7. Se dirá que un estado de una cadena de Markov es **aperiódico** si $\min\{n \geq 1 : p^n(u,u) > 0\} = 1$. Si todos los estados de dicha cadena cumplen esta propiedad, la cadena se dirá **aperiódica**.

Definición 1.8. Se dirá que un estado de una cadena de Markov es un estado **ergódico** si es recurrente, no nulo y aperiódico. Una cadena cuyos estados cumplan la citada condición, se denominará **cadena ergódica**.

Las cadenas ergódicas tienen un comportamiento límite tal que

$$p^n(u,v) \longrightarrow \pi(v), \quad \forall u, v \in \mathcal{U},$$

lo cual asegura la convergencia de $P(U_n = u)$ a $\pi(u), \forall u$.

Además, si $g(U)$ es una función definida en el espacio de estados de una cadena de Markov ergódica con una esperanza finita menor que $\pi(u)\ \forall u$ entonces se tiene que

$$\frac{1}{n}\sum_{t=1}^{n} g(U_t) \underset{n\to\infty}{\longrightarrow} E_\pi[g(U)] \text{ casi seguro.}$$

Esto es comúnmente conocido como **teorema ergódico**, que generaliza la ley fuerte de los grandes números para las cadenas de Markov con las características indicadas.

Cuando los estados de una cadena son variables aleatorias absolutamente continuas, la definición de las propiedades descritas necesitan modificarse para referirnos a los sucesos $A \subseteq \mathcal{U}$ en lugar de estados individuales, similares a la definición de función de transición, y están sujetos a algunos detalles técnicos de la teoría de la medida. Se exponen algunos de ellos a continuación.

Definición 1.9. Una medida probabilística Π se dice que es **estacionaria** si para cualquier

suceso A,

$$\Pi(A) = \int_{\mathcal{U}} P(u, A)\Pi(du),$$

que en términos de densidad corresponde a

$$\pi(v) = \int_{\mathcal{U}} p(u, v)\pi(u)du.$$

Los resultados de convergencia para cadenas con espacio de estados infinito no numerable son análogos a los enunciados anteriores, con la diferencia de que estos necesitan condiciones más fuertes, como se puede ver en (Paulino, Amaral Turkman, Murteira y Silva, 2018).

Otra propiedad de las cadenas de Markov que es de gran importancia en el análisis de los comportamientos límite es la reversibilidad de las dinámicas probabilísticas.

Definición 1.10. Una cadena se dice **reversible** si para algún suceso A y estado u en el espacio de estados $\mathcal{U}$ (discreto o no),

$$P(U_{n+1} = v \in A|U_n = u) = P(U_{n+1} = v \in A|U_{n+2} = u).$$

En concreto, la reversibilidad de una cadena con función de transición $p(\cdot, \cdot)$ y distribución estacionaria $\pi(\cdot)$ es equivalente a

$$\pi(u)p(u, v) = \pi(v)p(v, u), \quad \forall u, v \in \mathcal{U}.$$

Esta última condición es conocida como **ecuación de balance**. Puede interpretarse como un equilibrio implícito en la cadena en el sentido de que estar en u y pasar por v es igualmente plausible que estando en v se haya pasado por u, para cualquier par (u, v) de estados. En particular, una cadena que satisface esta condición para una densidad de probabilidad π no es solo reversible, sino también tiene la misma distribución estacionaria π.

1.3 Procesos de Poisson

Como ejemplo de proceso estocástico concreto, se va a introducir la definición del proceso de Poisson, así como un caso particular del mismo, que será de gran utilidad para algunos de los ejemplos posteriores.

Definición 1.11. Un **proceso de Poisson** es un proceso estocástico de tiempo que consiste en el conteo de "sucesos raros" que ocurren a lo largo del tiempo. Se denotará por N_t al número de eventos que tienen lugar en el intervalo $[0, t]$.

Concretamente, los ejemplos venideros, utilizarán el conocido como Proceso de Poisson no homogéneo.

Definición 1.12. Un proceso de Poisson será **no homogéneo** para N_t con $t \geq 0$ e intensidad $\lambda(t)$ si y sólo si

1. $N_0 = 0$.
2. Es un proceso con incrementos independientes, es decir, para cualquier $k \geq 2$ y $0 \leq t_0 < t_1 \ldots < t_k$ las variables aleatorias $N_{t_1} - N_{t_0}, N_{t_2} - N_{t_1}, \ldots, N_{t_k} - N_{t_{k-1}}$ son independientes.
3. $P(\text{número de sucesos en } (t, t+h) \leq 2) = o(h)$, donde $o(h)$ representa un infitésimo en h.
4. $P(\text{número de sucesos en } (t, t+h) = 1) = \lambda(t)h + o(h)$.

Se dirá entonces, que un proceso de Poisson es **homogéneo** si $\lambda(t) = t$ para todo $t \in T$. Además, se denominará ***power law process*** a aquel proceso de Poisson no homogéneo que cumpla que λ es de la forma

$$\lambda(t|\theta) = M\beta t^{\beta-1}, \quad \text{para } \theta = (M, \beta) \in \mathbb{R}^+ \times \mathbb{R}^+.$$

CAPÍTULO

2

Métodos de Monte Carlo: Algoritmo de Metropolis-Hasting

2.1 Motivación histórica

Los métodos de Monte Carlo se basan en el uso del método mediante las cadenas de Markov (MCMC). Estos métodos estadísticos son no deterministas, es decir, no llevan a una única solución, sino que un mismo valor inicial finaliza con varios resultados posibles. Se utilizan para aproximar expresiones complejas y difícilmente evaluables de forma exacta.

Este método se desarrolló formalmente en la década de 1940 y debe su nombre al Casino de Montecarlo situado en Mónaco. Fue Stanislaw Marcin Ulam el que comenzó su desarrollo. Tal y como Ulam (Ulam, 1991) puntualiza en su libro autobiográfico: "*La idea de lo que llamaría luego el Método de Monte Carlo se me ocurrió mientras estaba haciendo un solitario durante mi enfermedad. Me di cuenta de que, para tener una idea de la probabilidad de que salga un solitario (. . .), era mucho más práctico echar las cartas, o experimentando con el proceso y observar cuantas veces sale, que trata de calcular todas las combinaciones, que crecen exponencialmente de tal manera, que, salvo casos muy elementales, no pueden estimarse*".

Como menciona (Higuera de Frutos, 2017), Ulam trabajó durante Segunda Guerra Mundial en el Proyecto Manhattan en Los Álamos, Nuevo Mexico. Este proyecto, desarrollado por el

Gobierno de Estados Unidos en colaboración con Reino Unido y Canadá, tenía como objetivo la creación de la primera bomba atómica, con intención de que saliera a la luz antes que Rusia y Alemania. Aunque Ulam consiguió aplicar su teoría a las ecuaciones diferenciales que modelaban la difusión de neutrones, no se tuvo en cuenta su idea. Fue años más tarde cuando se aplicó el método para investigaciones en el campo de la física.

Ulam trabajó en su método junto al matemático John Von Neumann, sin embargo, fue posteriormente con Nicholas Constantine Metropolis con quien alcanzó mejores estimaciones, como se puede ver en (Mackay, 1986). La primera publicación que se realiza sobre el método se publica en 1949 junto a Metropolis, que es visible en (Ulam y Metropolis, 1949).

2.1.1 Integración de Monte Carlo

Como se ha nombrado, el método de Monte Carlo encontró su aplicación en la resolución de ecuaciones diferenciales difícilmente resolubles, es decir, en cálculo de integrales complejas.

Supóngase que se quiere calcular la siguiente integral:

$$\int_{x_1}^{x_n} q(x)dx.$$

Si se descompone en el producto de una función f y una funcion de densidad p en el intervalo (x_1, x_n) en cuestión, se tiene que

$$\int_{x_1}^{x_n} q(x)dx = \int_{x_1}^{x_n} f(x)p(x)dx = E_{p(x)}(f(x)),$$

así se expresa la integral como la esperanza de $f(x)$. Tomando una muestra independiente e idénticamente distribuida $x_1 < x_2 < \ldots < x_n$, se puede expresar lo que se denomina **integración de Monte Carlo**:

$$\int_{x_1}^{x_n} q(x)dx \simeq \frac{1}{n}\sum_{i=1}^{n} f(x_i).$$

Se expone un ejemplo a continuación que ilustra el concepto.

Ejemplo 2.1. Se quiere calcular el valor de la siguiente integral:

$$\int_{-\infty}^{\infty} \frac{x+3}{3x^4+x^2+1} dx.$$

Es decir, el área bajo la curva que aparece en la siguiente figura:

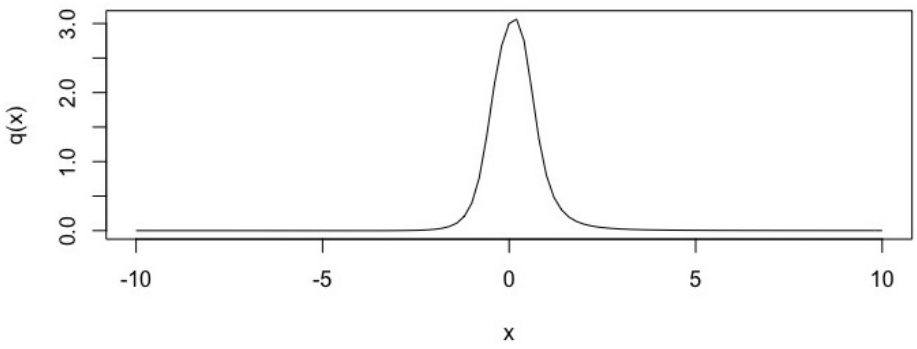

Figura 3: Área a calcular.

Mediante la integración de Monte Carlo, cuya codificación se puede observar en el Apéndice A.2, se generan 1000 y 10000 simulaciones que aproximan el valor de la integral. Estos valores se muestran en la siguiente figura en rojo y morado respectivamente, donde la línea horizontal negra representa el valor real de la integral.

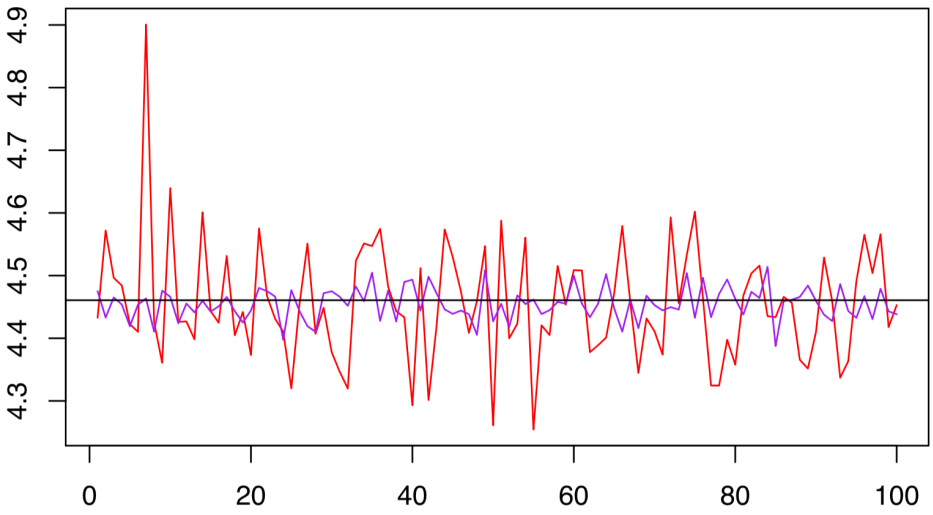

Figura 4: Rrepresentación de las simulaciones para 1000 y 10000 elementos.

Así, se puede observar que para un número suficientemente grande de simulaciones, la muestra se acerca en mayor medida al valor real de la integral.

Se pueden encontrar más ejemplos en (Main Yaque, Navarro Veguillas y Morales Fernández, 2019), y el código que genera el ejemplo en el Apéndice A.2.

2.2 Algoritmo de Metropolis-Hasting

El algoritmo de Metropolis-Hasting nace de la necesidad de los físicos-matemáticos de integrar funciones muy complejas mediante un muestreo aleatorio.

En esta ocasión, se tratarán cadenas de estados multivariantes, por lo que se cambiará el índice temporal por un superíndice para los vectores aleatorios, esto es $U^{(t)}$. Se reserva entonces el subíndice para los elementos del vector, es decir, $U_j^{(t)}$.

Supóngase que el objetivo es obtener muestras de alguna distribución $p(\theta)$ donde $p(\theta) = \pi(\theta)/K$, siendo K una constante que puede no ser conocida, y a menudo muy díficil de calcular. El elemento fundamental del algoritmo es trabajar con una distribución condicional $q(\tilde{u}|u) \equiv q(u|\tilde{u})$ que juega el papel de generador de valores simulados. El requerimiento básico necesario para $q(\cdot|\cdot)$, referida como la **distribución propuesta, de salto o candidato-generadora**, es una generación de variables aleatorias manejables. Los valores $\tilde{u}$ que son generados de la distribución están sujetos a inspección estocástica, basada en $q(\cdot|\cdot)$ y otra función $\pi(\cdot)$, que determina si $\tilde{u}$ es aceptado o rechazado, refiriéndose el rechazo como el reemplazo del valor por el aceptado más reciente.

El algoritmo de Metropolis genera una secuencia de muestras de la distribución objetivo siguiendo el siguiente esquema.

Algoritmo 1: Algoritmo de Metropolis

Sea $u^{(t)}$, con $t = 0, 1, 2, \ldots$ los estados de la cadena que genera el algoritmo.

1. Se toma $u^{(0)}$ tal que $\pi(u^{(0)}) > 0$.

2. Se escoge como candidato $\tilde{u}$ de la llamada distribución de salto $q(u|\tilde{u})$, la cual es la probabilidad de obtener $\tilde{u}$ dado u previamente.

3. Dado el candidato $\tilde{u}$, se calcula el siguiente cociente:

$$\alpha = R(u, \tilde{u}) = \frac{p(\tilde{u})}{p(u)} = \frac{\pi(\tilde{u})}{\pi(u)},$$

siendo evidente que la constante de normalización puede obviarse.

4. Si el cociente es mayor que 1, se acepta el candidato en cuestión por lo que $u^{(t+1)} = \tilde{u}$ y se retorna al paso 2. En el caso de que el salto disminuya la densidad, es decir, que el ratio sea menor que 1, se rechazaría el candidato con probabilidad α y se vuelve al paso 2.

Este algoritmo se puede resumir en el cálculo de

$$\alpha = \text{mín}\left\{\frac{\pi(\tilde{u})}{\pi(u^{(t-1)})}, 1\right\},$$

aceptando al candidato $\tilde{u}$ con probabilidad α. Este proceso genera una cadena de Markov donde las probabilidades de transición de $u^{(t)}$ a $u^{(t+1)}$ tan sólo dependerán de $u^{(t)}$, y no del resto de estados anteriores.

Hastings (Hastings, 1970) generalizó este algoritmo utilizando la siguiente probabilidad de aceptación, o lo que se denominará a partir de ahora como Cociente de Metropolis-Hasting

$$\alpha = R(u, \tilde{u}) = \frac{\pi(\tilde{u})q(u|\tilde{u})}{\pi(u)q(\tilde{u}|u)}.$$

Por tanto, este será llamado el **algoritmo de Metropolis-Hasting**. Cabe destacar que si la distribución propuesta $q(\cdot|\cdot)$ fuera simétrica, se volvería al algoritmo de Metropolis.

Algunas notas sobre el algoritmo:

1. **Soporte de** π. La probabilidad de aceptación de $\tilde{u}$ en la iteración $t+1$ requiere que $\pi(u^{(t)}) > 0$. Esto está garantizado $\forall t \in \mathbb{N}$ si el valor inicial $u^{(0)}$ de la cadena lo satisface, ya que todos los valores simulados con $\pi(\tilde{u}) = 0$ serán rechazados por ser $\alpha(u^{(t)}, \tilde{u}) = 0$. Se establece $R(u, \tilde{u}) = 0$ cuando $\pi(\tilde{u}) = 0 = \pi(u)$. Luego, una vez dentro del soporte

de π, la cadena no sale del mismo casi con total seguridad.

2. **Constantes de normalización**. La naturaleza del ratio M-H muestra que el algoritmo puede implementarse cuando $\pi(\cdot)$ y $q(\cdot|u)$ son conocidas salvo las constantes de normalización, esto es, los factores que no involucran u en el caso de $q(\cdot|u)$. Por otro lado, los valores de $\tilde{u}$ con $\pi(\tilde{u})/q(\tilde{u}|u^{(t)})$ mayores que el mismo cociente que para el valor anterior, $\pi(u^{(t)})/q(u^{(t)}|\tilde{u})$, son siempre aceptados, ya que $\alpha(u^{(t)}, \tilde{u}) = 1$.

3. **Repeticiones**. Una cadena $\{u^{(t)}\}$ generada por este algoritmo puede incluir repeticiones, y es un caso especial de una cadena de Markov, ya que la distribución de $U^{(t+1)}$ condicionada a todos los valores anteriores depende solo de $U^{(t)}$. La convergencia de esta cadena a la distribución objetivo $\pi(u)$ depende, como se puede esperar, de la distribución propuesta.

4. **Función de transición**. Como este es el caso mas común en las aplicaciones, se considera únicamente el caso absolutamente continuo (con respecto a la medida Lebesgue) con un número infinito no numerable de estados, en cuyo caso $\pi(u)$ es la densidad de la distribución estacionaria. Se denota $Q(\cdot, \cdot)$ a la función de transición de una cadena de Markov, con densidad $q(\cdot, \cdot)$, esto es, $Q(u, d\tilde{u}) = q(\tilde{u}|u)d\tilde{u}$. En este caso, el paso 3 del algoritmo M-H define una función de transición

$$P(u, d\tilde{u}) \equiv P\left[U^{(t+1)} \in d\tilde{u}|U^{(t)} = u\right] = \alpha(u, \tilde{u})q(\tilde{u}|u)d\tilde{u} + r(u)\delta_u(d\tilde{u}),$$

donde $\delta_u(d\tilde{u})$ denota la medida de Dirac en $d\tilde{u}$ y $r(u) = 1 - \int \alpha(u, \tilde{u})q(\tilde{u}|u)d\tilde{u}$ es la probabilidad de que la cadena permanezca en u. La función de transición anterior está caracterizada por la densidad de transición

$$p(u, \tilde{u}) = \alpha(u, \tilde{u})q(\tilde{u}|u) + r(u)\delta_u(\tilde{u}),$$

donde, por la definición de α y δ_u, se satisface la condición de balance detallada con π, $\pi(u)p(u, \tilde{u}) = \pi(\tilde{u}, u)$. En consecuencia, una cadena M-H es reversible con una distri-

bución estacionaria precisamente igual a la distribución objetivo deseada π.

5. **Convergencia**. La convergencia en una cadena de Markov aplicando el algoritmo de M-H a la distribución estacionaria π depende de las condiciones de regularidad que se han expuesto en el apartado anterior. Sea $\mathcal{S} = \{u : \pi(u) > 0\}$ la notación que se va a utilizar para el soporte de π. El uso de la distribución propuesta $q(\cdot, \cdot)$ con $q(\tilde{u}, u) > 0, \forall (u, \tilde{u}) \in \mathcal{S} \times \mathcal{S}$ garantiza que la cadena $\{U^{(t)}\}$ es irreducible con respecto a π. Al ser π una distribución estacionaria para la cadena M-H, esta es recurrente positiva, y se aplica el teorema ergódico, como se puede ver en (Robert y Casella, 2004).

   En resumen, para una cadena M-H que converge a la distribución objetivo π, los estados de la cadena hasta un tiempo concreto pueden considerarse como simulaciones aproximadas de π, incluso si en la implementación estuvieran generados de la distribución propuesta. Esto implica que los balances de π pueden ser determinados empíricamente desde una muestra (generada computacionalmente) por dicha cadena.

A continuación, se muestra un ejemplo que ilustra el funcionamiento del algoritmo de M-H.

Ejemplo 2.2. Sea $\theta > 0$. Se considera

$$p(\theta) = C \cdot \theta^{-\frac{n}{2}} \cdot exp\left(\frac{-a}{2\theta}\right),$$

que se distribuye según una distribución χ^2 invertida con parámetros $a > 0$ y $n > 2$ grados de libertad para $n \in \mathbb{Z}^+$.

Se quieren simular muestras de la distribución con $n = 5$ y $a = 4$ utilizando el algoritmo de M-H.

Tomando como distribución propuesta una distribución uniforme $(0, 100)$, se inicia el algoritmo.

1. Se toma $\theta_0 = 50$ como valor inicial, que verifica $p(1) = 0.1353353 > 0$.

2. Se genera un candidato de forma aleatoria de la distribución propuesta, en este caso, supóngase que devuelve $\theta^* = 55.5852$.

3. Se calcula el ratio como sigue

$$\frac{p(\theta^*)}{p(\theta_0)} = \frac{(55.5852)^{-2.5} \cdot \exp(-2/55.5852)}{(1)^{-2.5} \cdot \exp(-2/2)} = 0.0003094$$

4. Se calcula

$$\alpha = \text{mín}\left(\frac{p(\theta^*)}{p(\theta_0)}, 1\right) = 0.0003094$$

5. Se genera un valor aleatorio u de una distribución uniforme $(0, 1)$, en este caso se obtiene $u = 0.7253844$, y se compara con α. Al ser

$$U = 0.7253844 > \alpha = 0.0003094,$$

el candidato no se acepta y se vuelve al punto 2.

Siguiendo el algoritmo con una segunda iteracion, se obtiene que $\theta^* = 18.6883394$ y

$$\alpha = \text{mín}\left(\frac{p(18.6883394)}{p(1)}\right) = \text{mín}\,(0.75292, 1) = 0.004397285.$$

Como u aleatorio se obtiene 0.003093487, porque $u \leq \alpha$ y el candidato θ^* se aceptaría.

Se puede ver en el Apéndice A.3 una generación del algoritmo con 5000 iteraciones en la que se aceptan 372 valores.

□

La característica quizás más atractiva del algoritmo de M-H es su versatilidad, considerando los pocos y débiles requerimientos que se necesitan imponer sobre π y q para garantizar la convergencia de la cadena a π. Nótese, sin embargo, que la convergencia no implica que el algoritmo sea eficiente logrando una convergencia práctica en un número relativamente pequeño de iteraciones. Es decir, no necesariamente describe una rápida convergencia de la cadena de Markov. Una distribución propuesta bien elegida debería generar valores que cubran el soporte de la distribución objetivo en un número razonable de iteraciones. Estas características están relacionadas con la dispersión de la distribución propuesta que genera los valores simulados. Específicamente, si q está demasiado dispersa en relación con π, los valores propuestos se

rechazarán frecuentemente, y el soporte de π solo puede ser representativamente muestreado después de muchas iteraciones, implicando una convergencia más lenta. En el caso opuesto de la dispersión reducida, solo un pequeño subconjunto $\mathcal{S}$ es visitado a través de muchas iteraciones, con un ratio alto de aceptación que puede ser falsamente interpretado como convergencia rápida, cuando de hecho es necesario un gran número de iteraciones adicionales para explorar otras partes de $\mathcal{S}$. Por esto, es conveniente comenzar con un análisis preliminar de π, tal que q se elija para aproximar la distribución objetivo de la mejor manera posible.

Ejemplo 2.3. Supóngase que se quiere utilizar una distribución χ^2 como densidad candidata mediante una muestra simple de una distribución χ^2 independiente de la posición actual. Renombrando $x \sim \chi^2_n$, se tiene

$$g(x) \propto x^{n/2-1}e^{-x/2}, \quad x > 0.$$

mirar Así, $q(x, y) = g(y) = C \cdot y^{n/2-1}e^{-y/2}$. Nótese que $q(x, y)$ no es simétrica, ya que $q(y, x) = g(x) \neq g(y) = q(x, y)$. Por tanto, se debe utilizar el muestreo de Metropolis-Hasting, con la probabilidad de aceptación

$$\alpha(x, y) = \text{mín}\left\{\frac{p(y)q(y, x)}{p(x)q(x, y)}, 1\right\} = \text{mín}\left[\frac{p(y)x^{n/2-1}e^{-x/2}}{p(x)y^{n/2-1}e^{-y/2}}, 1\right].$$

Recurriendo al Ejemplo 2.2, donde se simula $p(x) = C \cdot x^{-2.5}e^{-2/x}$, la probabilidad de rechazo será

$$\alpha(x, y) = \text{min}\left[\frac{(y^{-2.5}e^{-2/y})(x^{n/2-1}e^{-x/2})}{(x^{-2.5}e^{-2/x})(y^{n/2-1}e^{-y/2})}, 1\right].$$

Si se genera una muestra con dos distribuciones propuestas, en este ejemplo χ^2_2 y χ^2_{10}. Se puede ver en las representaciones como la generada por χ^2_2 tiene una varianza menor y así una probabilidad de aceptación mayor, es decir, habrá más valores aceptados con respecto a la generada por χ^2_{10}. En el Apéndice A.4 se puede ver el código que genera estas gráficas.

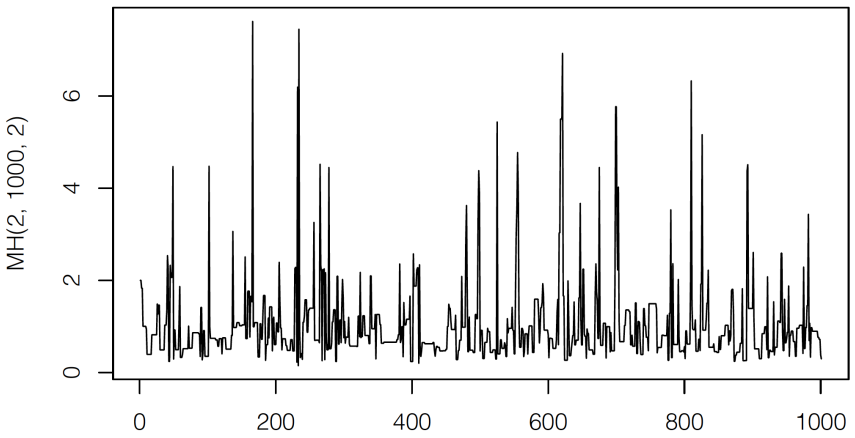

Figura 5: Muestra generada con distribución propuesta χ^2_2 con 1000 iteraciones. Elaboración propia.

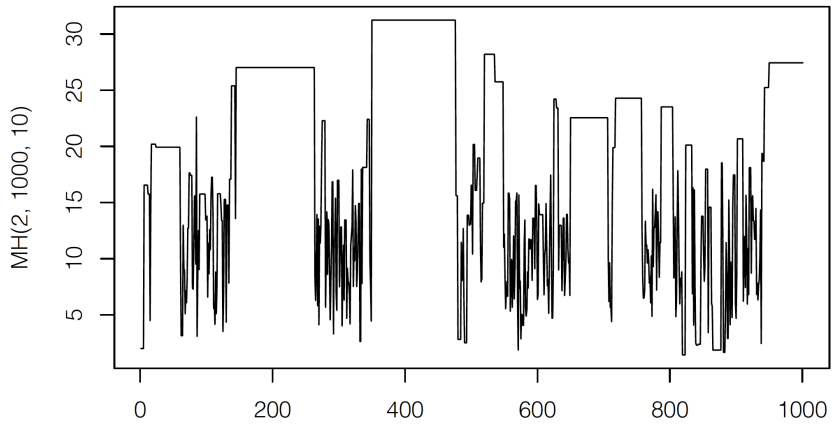

Figura 6: Muestra generada con distribución propuesta χ^2_{10} con 1000 iteraciones. Elaboración propia.

□

2.3 Técnica de *burn-in* en el muestreo

El problema principal en una buena implementación del método de Metropolis-Hasting o cualquier otro muestreo de MCMC, es el número de iteraciones necesarias hasta que la cadena alcance estacionariedad. Normalmente, los primeros 1000 a 5000 elementos son eliminados, y después se utiliza algún test de convergencia para evaluar si se ha alcanzado.

Una mala elección de los valores inicializadores y/o de la distribución propuesta pueden incrementar gravemente el tiempo requerido de *burn-in*, lo que provoca que actualmente se

estudie si es posible encontrar un valor inicial y una distribución propuesta óptimos. Por ahora, sólo se pueden dar algunas reglas básicas. Una sugerencia para el valor inicial es empezar la cadena con un valor lo más cercano posible al centro de la distribución, por ejemplo, tomando un valor cercano a la moda.

Una cadena se dirá que es ***poorly mixing*** si está en pequeñas regiones del parámetro durante largos periodos de tiempo, en oposición a estar ***well mixing***, la cuál parece explorar el espacio. Una cadena *poorly mixed* puede originarse ya que la distribución objetivo es multimodal y nuestra elección de valores iniciales estará cerca de una de esas modas. Lo más sencillo es usar valores iniciales muy dispersos para empezar diferentes cadenas, como se observa en Geman(Geman y Geman, 1984).

2.4 Muestreo de Metropolis-Hasting como cadena de Markov

Para demostrar que el muestreo de Metropolis-Hasting genera una cadena de Markov cuya densidad de equilibrio es la candidata $p(x)$, es suficiente con probar que el *kernel* de transición de Metropolis-Hasting satisface la ecuación de balance con $p(x)$.

Bajo el algoritmo de Metropolis-Hasting, se muetrea desde $q(x, y) = P(x \to y|q)$ y se acepta el movimiento con probabilidad $\alpha(x, y)$, así que el *kernel* de la probabilidad de transición está dado por

$$P(x \to y) = q(x, y)\alpha(x, y) = q(x, y) \cdot \min\left[\frac{p(y)q(y, x)}{p(x)q(x, y)}, 1\right]. \tag{2.1}$$

Así, si el *kernel* de Metropolis-Hasting satisface $P(x \to y)p(x) = P(y \to x)p(y)$, o bien

$$q(x, y)\alpha(x, y)p(x) = q(y, x)\alpha(y, x)p(y), \quad \text{para todo } x, y,$$

entonces la distribución estacionaria de dicho *kernel* corresponde a muestras de la distribución objetivo. Se puede observar que, en efecto, la ecuación de balance se satisface con este *kernel* considerando tres posibles casos para cualquier par x, y.

1. $q(x, y)p(x) = q(y, x)p(y)$. Así, $\alpha(x, y) = \alpha(y, x) = 1$, lo cual implica que

$$P(x,y)p(x) = q(x,y)p(x) \quad \text{y} \quad P(y,x)p(y) = q(y,x)p(y),$$

y por tanto, $P(x,y)p(x) = P(y,x)p(y)$, demostrando que, para este caso, se cumple la ecuación de equilibrio.

2. $q(x,y)p(x) > q(y,x)p(y)$, en cuyo caso se tendría que

$$\alpha(x,y) = \frac{p(y)q(y,x)}{p(x)q(x,y)} \quad \text{y} \quad \alpha(y,x) = 1.$$

Por tanto,

$$\begin{aligned} P(x,y)p(x) &= q(x,y)\alpha(x,y)p(x), \\ &= q(x,y)\frac{p(y)q(y,x)}{p(x)q(x,y)}p(x), \\ &= q(y,x)p(y) = q(y,x)\alpha(y,x)p(y), \\ &= P(y,x)p(y). \end{aligned}$$

3. $q(x,y)p(x) < q(y,x)p(y)$. Así,

$$\alpha(y,x) = \frac{p(x)q(x,y)}{p(y)q(y,x)} \quad \text{y} \quad \alpha(x,y) = 1.$$

Luego,

$$\begin{aligned} P(y,x)p(y) &= q(y,x)\alpha(y,x)p(y), \\ &= q(y,x)\frac{p(x)q(x,y)}{p(y)q(x,y)}p(x), \\ &= q(x,y)p(x) = q(x,y)\alpha(x,y)p(x), \\ &= P(x,y)p(x). \end{aligned}$$

Dada la naturaleza general del algoritmo M-H, se describen a continuación dos de los casos más usados. Para otros casos, se puede consultar (Givens y Hoeting, 2005).

2.5 Algoritmo Metropolis-Hasting independiente

Como indica el nombre del algoritmo, la distribución propuesta es independiente del estado actual, esto es, $q(\tilde{u}|u) = q(\tilde{u})$. Esto implica que la probabilidad de aceptación es

$$\alpha(u^{(t)}, \tilde{u}) = \text{mín}\left\{\frac{\pi(\tilde{u})q(u^{(t)})}{\pi(u^{(t)})q(\tilde{u})}, 1\right\}, \quad t \geq 0.$$

De forma similar a lo que se especificó sobre el algoritmo M-H general, la ergodicidad de la cadena $\{U^{(t)}\}$ requiere que el soporte de la distribución propuesta q, ahora sin condicionar, contenga al soporte de π.

Por ejemplo, considerando la simulación de una distribución a posteriori, esto es,

$$\pi(\theta) = h(\theta|x) \propto L(\theta|x)h(\theta),$$

donde $h(\theta)$ es la distribución a priori, $L(\theta|x)$la función de verosimilitud y los estados $\{U^{(t)} \equiv \theta^{(t)}\}$. Una particularización de una cadena M-H independiente en este contexto es el caso especial en el que se tome $q(\theta) = h(\theta)$. En ese caso, el soporte de q cubre el soporte de π, incluso si las dos distribuciones fueran muy diferentes. También, en este caso, el ratio M-H se reduce al ratio de verosimilitud $R(\theta^{(t)}, \tilde{u}) = \dfrac{L(\tilde{u}|x)}{L(\theta^{(t)}|x)}$.

2.6 Algoritmo Metropolis-Hasting de camino aleatorio

Este algoritmo está definido por la distribución propuesta $\tilde{U} = U^{(t)} + \varepsilon_t$, donde ε_t es un error aleatorio con distribución q^* independiente de $U^{(t)}$. Esto define un camino aleatorio con densidad de transición $q(\tilde{u}|u) = q^*(\tilde{u} - u)$. Las elecciones usuales para q^* incluyen distribuciones uniformes en una bola centrada alrededor del origen, distribuciones gaussianas y distribuciones t de Student. Nótese que si la distribución propuesta es simétrica, esto es, $q(\tilde{u}|u) = q(u|\tilde{u})$, el ratio M-H se simplifica a $R(u, \tilde{u}) = \frac{\pi(\tilde{u})}{\pi(u)}$, destacando que la distribución objetivo necesita solo ser conocida, salvo por la constante de normalización. La simetría tiene lugar cuando $q^*(y)$ depende de y sólo mediante $|y|$. Cuando una cadena se basa en un camino aleatorio

$\tilde{U} \sim q^*(|\tilde{u} - u^{(t)}|)$ se considera el algoritmo de Metropolis introducido en (Metropolis, Rosenbluth, Rosenbluth, Teller y Teller, 1953) en el contexto de un problema de física de partículas con un espacio de estados discreto.

2.7 Algoritmo Metropolis-Hasting por bloques

Si en lugar de querer simular valores unidimensionales se pretendiera estimar un vector, por ejemplo bidimensional, la función candidato-generadora tendría que admitir parámetros bidimensionales, así como devolver un vector también bidimensional, tarea que sería bastante costosa. Si en lugar de dos parámetros se tuviera un número mayor, la función de densidad propuesta sería aún mas compleja de encontrar, por lo que el procedimiento puede dividirse por bloques.

Por ejemplo, si $u = (u^{(1)}, u^{(2)}, u^{(3)})$, se definirían tres densidades propuestas q_1, q_2 y q_3. Por ende el algoritmo quedaría de la siguiente manera.

Algoritmo por bloques (Caso n=3)

Inicializando con $u^{(0)} = (u^{(1,0)}, u^{(2,0)}, u^{(3,0)})$,

1. Simular $\tilde{u}_1 \sim q_1(\tilde{u}^{(1)}|u^{(2,t)}, u^{(3,t)})$ y aceptar con probabilidad:

$$\alpha = \text{mín}\left(1, \frac{\pi(\tilde{u}^{(1)}|x, u^{(2,t)}, u^{(3,t)})q_1(u^{(1,t)}|u^{(2,t)}, u^{(3,t)})}{\pi(u^{(1,t)}|x, u^{(2,t)}, u^{(3,t)})q_1(\tilde{u}^{(1)}|u^{(2,t)}, u^{(3,t)})}\right).$$

2. Simular $\tilde{u}_2 \sim q_2(\tilde{u}^{(2)}|u^{(1,t+1)}, u^{(3,t)})$ y aceptar con probabilidad análoga.

3. Simular $\tilde{u}_3 \sim q_3(\tilde{u}^{(3)}|u^{(1,t+1)}, u^{(2,t+1)})$ y aceptar con probabilidad análoga.

Si estas distribuciones condicionadas fueran independientes del resto de parámetros se conseguiría una gran ventaja. Esta condición es la que da lugar al que se conoce como el Método de Gibbs, que se estudiará en el siguiente capítulo.

CAPÍTULO

3

Muestreo de Gibbs

3.1 Fundamentos del Muestreo de Gibbs

En el capítulo anterior era evidente el carácter general del algoritmo de M-H, concretamente el hecho de que no es necesario indicar la dimensión de u en la distribución objetivo $\pi(u)$. En contraste, el algoritmo de muestreo de Gibbs está especialmente diseñado para las distribuciones k-dimensionales, con $k \geq 2$. Este algoritmo es originario de los hermanos Geman (Geman y Geman, 1984), en una aplicación inferencial de los llamados campos aleatorios de Gibbs, en referencia al físico J. W. Gibbs.

El algoritmo construye una cadena de Markov de forma que esta converge a a la distribución objetivo deseada $\pi(u)$, $u = (u_1, u_2, \ldots, u_k) \in \mathcal{U}$. Esto se implementa mediante un muestreo iterativo de las distribuciones condicionales (normalmente univariantes) dados todos los demás elementos, también denominados como las distribuciones condicionadas completas (*Full Conditionals*). El algoritmo reemplaza sucesivamente los elementos del vector de estados u en ciclos de k pasos, reemplazando u_j en el paso j-ésimo por el valor muestreado de la distribución condicional $\pi(v_j|\{u_i, i \notin j\}), j = 1, 2, \ldots, k$. Es decir, se necesitan k distribuciones tales como

$$U_i|U_1, \ldots, U_{i-1}, U_{i+1}, \ldots, U_k,$$

que se denotan como $U_i|U_{-i}$ para $i \in \{1, \ldots, k\}$. Por ejemplo, considérese el caso particular de $k = 3$, un ciclo de tres pasos sustituye el estado actual u por $v = (v_1, v_2, v_3)$ con

$$(u_1, u_2, u_3) \xrightarrow{\text{Paso 1}} (v_1, u_2, u_3) \xrightarrow{\text{Paso 2}} (v_1, v_2, u_3) \xrightarrow{\text{Paso 3}} (v_1, v_2, v_3).$$

En general, dado un estado actualmente atribuido $u^{(t)} = (u_1^{(t)}, \ldots, u_k^{(t)})$, una transición de Gibbs genera iterativamente los valores $u^{(t+1)}$ para el siguiente vector de estados de la cadena usando las distribuciones condicionales completas

$$\begin{gathered}
U_1^{(t+1)} \sim \pi(u_1|u_2^{(t)}, u_3^{(t)}, \ldots, u_k^{(t)}) \\
U_2^{(t+1)} \sim \pi(u_2|u_1^{(t+1)}, \ldots, u_k^{(t)}) \\
\downarrow \\
U_{k-1}^{(t+1)} \sim \pi(u_{k-1}|u_1^{(t+1)}, u_2^{(t+1)}, \ldots, u_{k-2}^{(t)}, u_k^{(t)}) \\
U_k^{(t+1)} \sim \pi(u_k|u_1^{(t+1)}, u_2^{(t+1)}, \ldots, u_{k-1}^{(t+1)})
\end{gathered}$$

La siguiente transición repite el ciclo de k pasos, ahora comenzando en $u^{(t+1)}$. Es decir, a partir de un punto inicial, el algoritmo simula observacionas de condicionadas univariantes, sustituyendo en cada caso el valor de la variable anterior a la simulada por el obtenido en el paso anterior. El esquema del algoritmo es el siguiente:

Algoritmo 2: Muestreador de Gibbs

Dado un estado actual $u^{(t)} = (u_1^{(t)}, \ldots, u_k^{(t)})$, empezando con $t = 0$, se genera cada componente, $u_j^{(t+1)}$, del siguiente vector de estados de la cadena utilizando

$$U_j^{(t+1)} \sim \pi(u_j^{(t+1)}|u_1^{(t+1)}, \ldots, u_{j-1}^{(t+1)}, u_{j+1}^{(t)}, \ldots, u_k^{(t)}),$$

para $j = 1, 2, \ldots, k$.

Denotándose por u generalmente el vector de estados "imputado" actualmente al comienzo del paso j-ésimo, el muestreo de Gibbs procede generando un vector $\tilde{u} = (u_1, \ldots, u_{j-1}, \tilde{u}, u_{j+1}, \ldots, u_k)$

con

$$\tilde{U}|u \sim q_j(\tilde{u}|u) = \begin{cases} \pi(\tilde{u}_j|u_{-j}), & \text{si } \tilde{u}_{-j} = u_{-j} \\ 0, & \text{en otro caso.} \end{cases}$$

Se muestran a continuación varios ejemplos que ilustran el algoritmo descrito.

Ejemplo 3.1. Sea (X_1, X_2) una variable aleatoria Normal Bivariante, cuyo vector de medias es $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ y con matriz de covarianzas $\begin{pmatrix} \sigma_{x_1}^2 & \rho\sigma_{x_1}\sigma_{x_2} \\ \rho\sigma_{x_1}\sigma_{x_2} & \sigma_{x_2}^2 \end{pmatrix}$ donde ρ es el coeficiente de correlación entre ambas variables X_1 y X_2, y $\sigma_{x_1}, \sigma_{x_2}$ son las desviaciones típicas de las distribuciones marginales. Así, las distribuciones condicionadas completas son distribuciones normales univariantes, es decir,

$$X_2|X_1 = x_1 \sim N(\rho\tfrac{\sigma_{x_2}}{\sigma_{x_1}}x_1, \sqrt{\sigma_{x_2}^2(1-\rho^2)}),$$
$$X_1|X_2 = x_2 \sim N(\rho\tfrac{\sigma_{x_1}}{\sigma_{x_2}}x_2, \sqrt{\sigma_{x_1}^2(1-\rho^2)}).$$

En este caso, el algoritmo del muestreo de Gibbs quedaría de la siguiente forma:

1. Tomar un valor inicial (x_1^0, x_2^0).
2. Simular x_1^j de una variable aleatoria $N(\rho\frac{\sigma_{x_1}}{\sigma_{x_2}}x_2, \sqrt{\sigma_{x_1}^2(1-\rho^2)})$.
3. Simular x_2^j de una variable aleatoria $N(\rho\frac{\sigma_{x_2}}{\sigma_{x_1}}x_1, \sqrt{\sigma_{x_2}^2(1-\rho^2)})$.
4. Volver al paso 2.

En este caso, se toma el valor inicial $(x_1^0, x_2^0) = (-2, 2)$, que está en un área de baja probabilidad de la distribución. Se generan valores, mediante el muestreo de Gibbs, de una distribución Normal Bivariante cuyas conponentes se encuentren fuertemente correladas, con $\rho = 0.8$. Para el resto de parámetros se tomará $\sigma_{x_1} = 1$ y $\sigma_{x_2} = 0.5$, obtiéndose la matriz de covarianzas $\begin{pmatrix} 1 & 0.4 \\ 0.4 & 0.25 \end{pmatrix}$.

Se ilustra el proceso representando, en primer lugar, las curvas de nivel de la ditribución considerada.

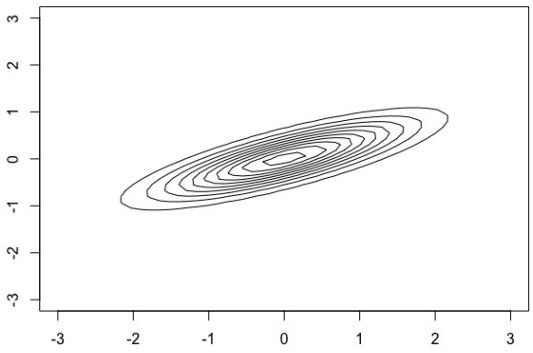

Figura 7: Curvas de nivel de la distribución Normal Bivariante

En la siguiente figura se representan tanto la curva de nivel de los valores generados mediante el método de Gibbs como estos propios utilizando las distribuciones condicionadas mencionadas, donde se han realizado con $n = 1000$ iteraciones y eliminado las 500 primeras como proceso de *burn-in*. Además, se han etiquetado los 5 valores iniciales para ver su comportamiento.

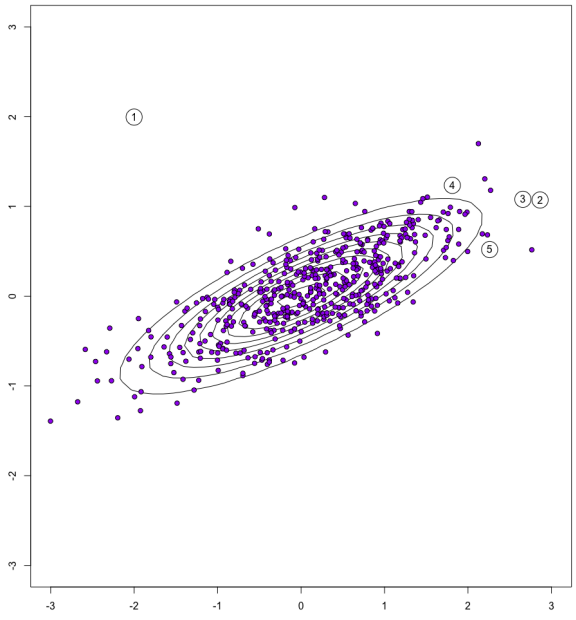

Figura 8: Representación de la muestra generada mediante el Muestreo de Gibbs

Si ahora se realizan $n = 10000$ iteraciones y se eliminan las 2000 primeras en el proceso de *bun-in*, se pueden mostrar los siguientes histogramas, los cuales corresponden a las distribuciones marginales obtenidas de la muestra.

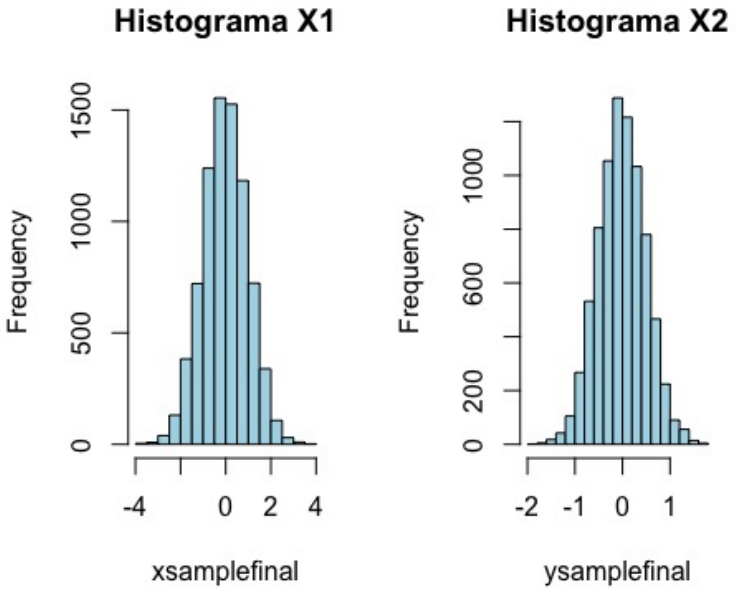

Figura 9: Histogramas de las distribuciones marginales generadas de la muestra para $n = 10000$.

El código correspondiente a la generación de estas figuras se puede consultar en el Apéndice A.5.

Ejemplo 3.2. Supóngase que se quiere simular una muestra de una variable aleatoria (X, Y) que tiene la siguiente función de densidad

$$\pi(x,y) = \begin{cases} \frac{1}{\pi} e^{-x(1+y^2)} & \text{para } x > 0, y \in \mathbb{R} \\ 0 & \text{en caso contrario} \end{cases}$$

Si se procede al cálculo de las distribuciones condicionadas completas, se observa que

$$\pi(x|y) = \frac{\pi(x,y)}{\pi_y(y)} \propto \pi(x,y) \propto e^{-x(1+y^2)}, \quad x > 0.$$

Por tanto se puede afirmar que $X|Y = y$ sigue una variable aleatoria $Exp(1 + y^2)$, para $y \in \mathbb{R}$.

De igual forma,

$$\pi(y|x) = \frac{\pi(y,x)}{\pi_x(x)} \propto \pi(x,y) \propto e^{-x(1+y^2)}, \quad y > 0,$$

lo que concluye en este caso que $Y|X = x$ sigue una distribución normal $N(0, \frac{1}{\sqrt{2x}})$, con $x > 0$.

Una realización de este algoritmo se puede ver explicada y codificada en el Apéndice **??**

Ejemplo 3.3. Supóngase una distribución conjunta de dos variables X e Y, siendo X discreta, con $x = 0, 1, ..., n$ e Y continua, con $0 \leq y \leq 1$ que viene dada por

$$p(x, y) = \frac{n!}{(n-x)!x!} y^{x+\alpha-1}(1-y)^{n-x+\beta-1}.$$

Aún siendo la distribución conjunta compleja, se tiene que las densidades condicionales son distribuciones simples. Para ver esto, si consideramos y como una constante fija, se puede ver que

$$p(x|y) \propto \frac{n!}{x!(n-x)!} y^x(1-y)^{n-x},$$

donde $0 < y < 1$ y puede considerarse como parámetro de éxito y n el número de atributos, por lo que $x|y \sim B(n, y)$.

Por otra parte, considerando ahora en la distribución conjunta x como constante fija, se observa que

$$p(y|x) \propto y^{x+\alpha-1}(1-y)^{n-x+\beta-1},$$

por lo que $y|x \sim Beta(x + \alpha, n - x + \beta)$, siendo $x + \alpha, n - x + \beta > 0$.

El poder del muestreo de Gibbs es calcular una secuencia de estas variables aleatorias univariantes condicionadas (una binomial y después una beta). Se puede calcular también cualquier característica de ambas distribuciones marginales. Supóngase $n = 10, \alpha = 1$ y $\beta = 2$. Se comienza con $y_0 = 1/2$, y se toma el muestreo a través de tres iteraciones:

1. x_0 se obtiene generando una variable aleatoria $B(n, y_0) = B(4, 1/2)$, tomando $x_0 = 4$ en esta simulación.

2. y_1 se obtiene de una $Beta(x_0 + \alpha, n - x_0 + \beta) = Beta(4 + 1, 10 - 4 + 2)$, dando $y_1 = 0.35341$.

3. x_1 es una realización de la variable aleatoria $B(n, y_1) = B(10, 0.35)$, dando $x_1 = 6$.

4. y_2 se obtiene de una $Beta(x_1 + \alpha, n - x_1 + \beta) = Beta(6 + 1, 10 - 6 + 2)$, dando $y_2 = 0.7662$.

5. x_2 se obtiene de $B(n, y_2) = B(10, 0.76)$, dando $x_2 = 9$.

Esta realización particular de la secuencia de Gibbs después de tres iteraciones es la siguiente:

$$(4, 0.5), \quad (6, 0.35341), \quad (9, 0.76625).$$

Así se puede ver en las siguientes figuras la representación gráfica de las marginales.

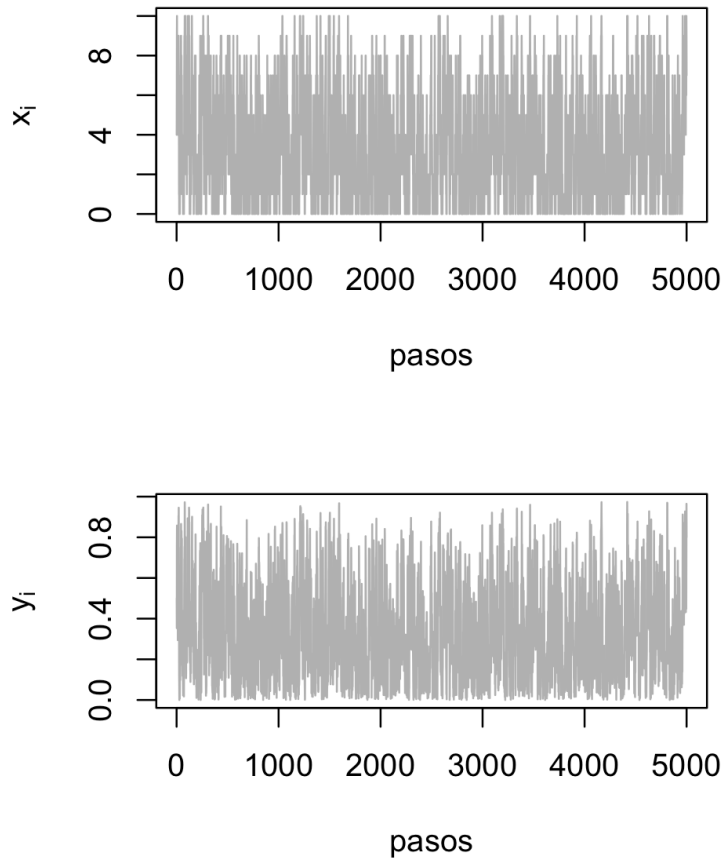

Figura 10: Representación valores marginales de la muestra para $n = 5000$.

Eliminando los primeros 500 valores como proceso de *burn-in*, se calculan algunos valores de interés, recogidos a continuación:

	x_i	y_i
media	3.2766	0.326882
des. típica	2.645485	0.23369
mediana	3	0.2869874

Se podría continuar con el proceso para generar una cadena con la longitud deseada. Se puede observar en el Apéndice **??** cómo se han generado computacionalmente los datos anteriores, así como las comprobaciones que siguen.

Para comprobar si la cadena de Markov generada por el método converge se aplica el test de Heidelberger y Welch (Heidelberger y Welch, 1983). Este sirve para realizar un contraste de hipótesis de que la cadena genera una cadena que es estacionaria. Es decir, como contraste se tiene:

$$\begin{cases} H_0 = \text{la distibución es estacionaria} \\ H_1 = \text{la distibución no es estacionaria} \end{cases}$$

Este test consta de dos fases. En la primera fase se sigue lo siguiente:

1. Una vez generada la cadena se define un nivel de confianza α.

2. Se calcula el test con la cadena al completo.

3. Si la hipótesis nula es rechazada, se descarta el primer $10\,\%$ de la cadena. Con lo restante se vuelve a realizar el test. Se repite así hasta que se acepte la hipótesis o se haya desechado la mitad de la cadena. En este caso, habría que obtener mas iteraciones de la cadena.

En la segunda fase, si la cadena completa con éxito la primera, se aplica el test de *halfwidth*. Si el cociente de esta "semilongitud" y la media es menor que un ϵ suficientemente pequeño, entonces se pasa el test de forma satisfactoria. En otro caso, se necesitan mas iteraciones de la cadena generada.

Para este ejemplo concreto, se plantea la siguiente hipótesis nula: H_0 = "la distribución a posteriori es estacionaria".

Una vez realizado el test, se obtiene un p-valor = $0.214 > 0.05$, por lo que se supera la primera fase, es decir, no se rechaza H_0.

En la segunda fase, se realiza el test de *halfwidth*, obteniéndose que el cociente $\frac{0.199}{3.27} < \epsilon$, por lo que la cadena supera el test.

Se puede ver en el Apéndice A.6 la codificación de este ejemplo.

□

Ejemplo 3.4. Sea $x = \{x_i, i = 1, \ldots, n\}$ una muestra aleatoria de un modelo Weibull con parámetros de escala y forma desconocidos denotados como δ y α, respectivamente. La función de verosimilitud será

$$L(\delta, \alpha|x) = (\delta\alpha)^n \left(\prod_{i=1}^{n} x_i\right)^{\alpha-1} e^{-\delta\sum_i x_i^\alpha}, \quad \delta, \alpha > 0.$$

Supóngase como distribuciones a priori independientes para δ y α las distribuciones gamma, $Ga(a, b)$, y logaritmo-normal $LN(c, d)$ respectivamente, con los hiperparámetros fijados $a, b, d > 0$ y $c \in \mathbb{R}$.

La distribución conjunta a posteriori tiene el núcleo

$$h(\delta, \alpha|x) \propto \alpha^{n+c/(d-1)} \left(\prod_{i=1}^{n} x_i\right)^{\alpha} e^{-(\ln\alpha)^2/2d} \delta^{\alpha+n} e^{-\delta(b+\sum_i x_i^\alpha)}.$$

Esto implica que las distribuciones condicionales completas sean

1. $h(\delta|\alpha, x) \propto \delta^{\alpha+n} e^{-\delta(b+\sum_i x_i^\alpha)}$
2. $h(\alpha|\delta, x) \propto \alpha^{n+c/d-1} (\prod_{i=1}^{n} x_i)^\alpha e^{-\left[\frac{(\ln\alpha)^2}{2d} + \delta\sum_i x_i^\alpha\right]}$

Así, la distribución condicional completa para δ sigue una distribución gamma $Ga(a+n, b+\sum_i x_i^\alpha)$ y la generación de valores para δ dado cualquier α puede ejecutarse con un generador de variables aleatorias gamma con capacidad de cálculo. La distribución condicional completa para α no sigue una forma estándar, requiriendo así el uso de métodos más sofisticados de generación de variables aleatorias.

□

3.2 Variantes del Muestreo de Gibbs

El término muestreador de Gibbs no solo se refiere a la versión descrita. Existen muchas variantes sobre las actualizaciones secuenciales y sobre la simulación. Entre otros, pueden citarse los siguientes.

3.2.1 Muestreador de Gibbs con bloqueo

Aunque la descripción usual del muestreo de Gibbs utiliza distribuciones condicionales univariantes completas, el esquema puede incluir un número flexible de pasos en el ciclo, usando en cada paso la distribución condicional para un subvector de cualquier dimensión. Esta variante tiene la ventaja particular de permitir agrupar las variables más correlacionadas, tal que la generación de las distribuciones condicionales completas para todo el subvector puede acelerar la convergencia del algoritmo.

3.2.2 Muestreador de Gibbs con hibridación

En general, es posible que no se conozcan generadores de variables aleatorias para algunas distribuciones condicionales completas. En estos casos, siempre se puede acudir a otras probabilidades de transición que puedan combinarse con el muestreador de Gibbs para definir un muestreador de Gibbs híbrido. Un ejemplo es el uso de las probilidades de transición de M-H como se discute en (Givens y Hoeting, 2005).

3.3 Aspectos generales del Muestreo de Gibbs

Después de esta breve introducción a algunas variantes, es de utilidad resaltar algunos de los aspectos de los algoritmos de muestreo de Gibbs en general.

1. **Distribución no propuesta**. Como se ha visto, la generación de variables aleatorias en el muestreador de Gibbs está basada en la distribución objetivo propia. Esto permite solucionar el a veces difícil problema de encontrar una buena distribución propuesta,

como es necesario en el algoritmo M-H. Sin embargo, generar una única variable en cada iteración del muestreador de Gibbs no es buena estrategia para una combinación rápida sobre el soporte de la distribución objetivo.

2. **Algoritmos M-H y Gibbs**. A pesar de las diferencias entre los algoritmos de Gibbs y M-H, hay una estrecha conexión entre ambos. Se considera el paso j-ésimo de un ciclo de un muestreador de Gibbs empezando con $u^{(t)}$. La simulación implica la distribución condicional $q_j(\tilde{u}|u) = \pi(\tilde{u}|u_{-j})$ con $\tilde{u}$ siendo el vector que difiere de u únicamente en la componente j-ésima, tal que $\tilde{u}_{-j} = u_{-j}$. La distribución $q_j(\tilde{u}|u)$ juega el papel de distribución propuesta. Por la definición de distribución conjunta y por notación, se escribe $\pi(u) = \pi(u_{-j})\pi(u_j|u_{-j})$, donde el primer y segundo factor se refieren, respectivamente, a las distribuciones marginales para U_{-j} y la distribución condicional para U_j dada U_{-j}. Los factores son también exactamente las distribuciones de $\tilde{U}_{-j}$. Por tanto,

$$\frac{\pi(\tilde{u})}{\pi(u)} = \frac{\pi(\tilde{u}_j|u_{-j})}{\pi(u_j|\tilde{u}_{-j})} \equiv \frac{q_j(\tilde{u}|u)}{q_j(u|\tilde{u})},$$

implicando este paso que el ratio M-H es

$$R_j(u, \tilde{u}) = \frac{\pi(\tilde{u})q_j(u|\tilde{u})}{\pi(u)q_j(\tilde{u}|u)} = 1.$$

Cada ciclo del muestreador de Gibbs puede por tanto verse como una composición de k probabilidades de transición M-H, con probabilidad de aceptación para cada paso igual a la unidad. Nótese que alternativamente un ciclo entero del muestreador de Gibbs puede interpretarse como una única función de transición M-H, con una probabilidad de aceptación global que puede calcularse para la densidad de transición entre el paso inicial y final del ciclo.

3. **Caso bivariante**. La definición del muestreador de Gibbs en el caso bivariante consiste en los pasos $U_1^{(t+1)} \sim \pi_1(\cdot|u_2^{(t)})$ y $U_2^{(t+1)} \sim \pi_2(\cdot|u_1^{(t)})$ para $t \geq 0$, teniendo en cuenta que la secuencia $\{(U_1(t), U_2(t))\}$ define una cadena de Markov. También cada una de estas subsecuencias es una cadena de Markov, por ejemplo, $U_2^{(t)}$ es una cadena de Markov con

densidad de transición

$$P(u_2, \tilde{u}_2) = \int \pi_1(w|u_2)\pi_2(\tilde{u}_2|w)dw, \tag{3.1}$$

que solo depende de lo anterior por el valor previo U_2. En otras palabras, las definiciones de las densidades marginales así como de las densidades de transición para U_2 implican que

$$\pi_2(\tilde{u}_2) = \int \pi_2(\tilde{u}_2|w)\pi_1(w)dw =$$

$$\int \left[\int \pi_2(\tilde{u}_2|w)\pi(w|u_2)dw\right] \pi_2(u_2)du_2 = \int P(u_2, \tilde{u}_2)\pi_2(u_2)du_2,$$

que muestra que π_2 es una distribución estacionaria para la subcadena $U_2^{(t)}$.

Una de las condiciones básicas para la convergencia de un $\{U^{(t)}\}$ multivariante es que el soporte $\mathcal{U}$ de la distribución conjunta $\pi(\cdot)$ sea el producto cartesiano de los soportes $\mathcal{U}_j$ de las distribuciones marginales $\pi_j(\cdot)$. Esto implica que la cadena es irreducible y, en el caso bivariante, se mantienen las mismas para las subcadenas marginales. Si adicionalmente la función de transición es absolutamente continua con respecto a la medida de Lebesgue, tomando la densidad como el producto de densidades condicionales completas,

$$p(u, v) = \pi_1(v_1|u_2, \ldots, u_k)\pi_2(v_2|v_1, u_3, \ldots, u_k) \times \cdots \times \pi_k(v_k|v_1, v_2, \ldots, v_{k-1}),$$

la cadena es recurrente, implicando que π es la distribución estacionaria de la cadena $\{U^{(t)}\}$ y sus marginales son las distribuciones limites de las respectivas subcadenas, con la aplicabilidad resultante del teorema egódico.

En resumen, la estructura y convergencia del muestreador de Gibbs resalta que las distribuciones condicionales completas bastan para caracterizar y generalizar la distribución conjunta. Hay que tener en cuenta, especialmente, la constante de normalización en las distribuciones condicionales completas. En concreto, una constante de normalización

infinita (impropia) hace la existencia de distribución conjunta propia imposible, una condición que no siempre se detecta por inspección de las cadenas de Markov generadas, pero ocurre con algunos modelos de Bayes con distribuciones a priori impropias.

4. La simulación de una distribución condicional completa depende naturalmente de la estructura específica de esas condicionales. En el caso más simple, acudir al método de la función de distribución inversa o métodos eficientes *"ad hoc"* para ciertas distribuciones conocidas puede permitir la generación de variables aleatorias eficientes para los pasos correspondientes del muestreo de Gibbs. En casos más complejos, también es posible el muestreo empleando aproximaciones más sofisticadas. En muchos problemas de inferencia estadística, la distribución objetivo puede ser díficil de evaluar; por ejemplo, si contiene integrales analíticamente intratables. Como consecuencia, no se dispone de un método sencillo para la generación de variantes aleatorias para los pasos del muestreador de Gibbs. Un método que suele resultar existoso es aumentar la distribución objetivo $\pi(u)$ a $f(u, Z)$ introduciendo variables Z implícitas adicionales de forma que $\pi(u)$ permanezca siendo la distribución marginal de la conjunta $f(u, Z)$. En algunos casos, el modelo aumentado $f(u, Z)$ permite una implementación mucho más sencilla del muestreador de Gibbs, involucrando en este caso las distribuciones $f(u|z)$ y $f(z|u)$.

Se muestra en lo siguiente un ejemplo mas sobre el muestreo de Gibbs.

Ejemplo 3.5. Se considera un test de diagnóstico para alguna enfermedad con resultado binario (positivo o negativo). Tomando una muestra aleatoria de N pacientes, sea X el número de resultados positivos, y se asume $X|\phi \sim Bi(N, \phi)$. La mayoría de los tests de diagnóstico están sujetos a una clasificación de errores de forma que la probabilidad de obtener un resultado positivo puede escribirse como $\phi = \alpha\sigma + (1 - \alpha)(1 - \epsilon)$, donde $\theta = (\alpha, \sigma, \epsilon)$, siendo α es la prevalencia de la enfermedad, σ la sensibilidad del test (probabilidad de un verdadero positivo), y ϵ es la especificidad del test (probabilidad de un verdadero negativo).

El vector θ es típicamente desconocido y es el vector de parámetros de interés del problema de inferencia. Sin embargo, dada la sobreparametrización del modelo, no es posible realizar inferencia sobre θ (o función de θ, excepto ϕ, por ejemplo) sin datos adicionales o información

a priori relativa al tipo de test de diagnóstico y la enfermedad en consideración. Supóngase que sólo se tiene acceso a información a priori, representada por distribuciones beta independientes para las componentes de θ, con hiperparámetros fijados. La distribución a posteriori toma la siguiente forma analíticamente intratable:

$$h(\theta|x) \propto f(x|\theta)\alpha^{a_p-1}(1-\alpha)^{b_p-1}\sigma^{c_s-1}(1-\sigma)^{d_s-1}\varepsilon^{c_e-1}(1-\varepsilon)^{d_e-1},$$

$\theta \in (0,1)^3$. Intentando implementar la inferencia a posteriori usando MCMC, el muestreador de Gibbs no es una aproximación particularmente sencilla para este caso ya que las distribuciones a posteriori condicionadas son complejas, requiriendo métodos especializados de generación de variables aleatorias. Sin embargo, la implementación de un muestreador de Gibbs es mucho más simple gracias a los siguientes modelos de aumento de variables con los datos implícitos. Sea $Y = (X, Z_1, Z_2)$, donde Z_1 (resp. Z_2), son datos no observados (latentes) que registran el número de individuos con resultados positivos (negativos) reales. Un modelo para Y consistente con los datos observados X está definido como

$$f(y|\theta) = f(x|\theta)f(z_1|x,\theta)f(z_2|x,\theta),$$

donde ahora $Z_1|x,\theta \sim Bi(x, \alpha\sigma/\phi)$ y $Z_2|x,\theta \sim Bi(N-x, (1-\alpha)\varepsilon(1-\phi))$. Notesé que los parámetros de las distribuciones condicionales para las variables latentes corresponden a los llamados valores predictivos positivos $V_+ = \alpha\sigma/\phi$ y los valores predictivos negativos $V_- = (1-\alpha)\varepsilon(1-\phi)$. La densidad a posteriori, ahora con los datos aumentados y, es entonces

$$\begin{aligned} h(\theta|y) \propto \quad & f(x|\phi)(V_+)^{z_1}(1-V_+)^{x-z_1}(V_-)^{z_2}(1-V_-)^{N-x-z_2}\times \\ & \alpha^{a_p-1}(1-\alpha)^{b_p-1}\sigma^{c_s-1}(1-\sigma)^{d_s-1}\varepsilon^{c_e-1}(1-\varepsilon)^{d_e-1}. \end{aligned}$$

La expresión se ha simplificado considerablemente reduciéndose a un producto de tres densidades beta. De hecho, si se introduce una transformación de los datos desde $y = (x, z_1, z_2)$ a $y^* = (m, z_1, z_2)$, donde $m = z_1 + N - x - z_2$ es el número de individuos en la muestra que padecen la enfermedad, se encuentra para $f(y^*|\theta)$ que

$$f(y^*|\theta) = f(m|\theta)f(z_1|m,\theta)f(z_2|m,\theta),$$

tal que $M|\theta \sim Bi(N, \alpha), Z_1|m, \theta \sim Bi(m, \sigma)$ y $Z_2|m, \theta \sim Bi(N-m, \varepsilon)$.

Esta factorización de la función de verosimilitud, considerando la conjunta a priori y factores de la beta y la binomial, implica que los componentes de θ sean también independientes a posteriori, con las siguientes distribuciones:

$$\begin{aligned}
\alpha|y \sim Be(A_p, B_p), \quad & A_p = a_p + m = a_p + z_1 + N - x - z_2, \\
& B_p = b_p + N - m = b_p + x - z_1 + z_2, \\
\sigma|y \sim Be(C_s, D_s), \quad & C_s = c_s + z_1, D_s = d_s + N - x - z_2, \\
\varepsilon|y \sim Be(C_e, D_e), \quad & C_e = c_e + z_2, D_e = d_e + x - z_1.
\end{aligned}$$

Estas son las distribuciones condicionales completas para los parámetros condicionales con los datos aumentados y. Ya que las partes z_1 y z_2 de los datos aumentados no son observados, se necesita imputar estos en base a los parámetros. Esto último puede hacerse usando las respectivas muestras de distribuciones condicionales en la parte observada x de los datos. Esto conduce a un algoritmo tipo Gibbs para la distribución a posteriori conjunta $h(\theta, z_1, z_2|x)$, definida por los siguientes dos pasos.

Incremento de los datos

1. Paso de asignación: dado $\theta^{(0)} = (\alpha^{(0)}, \sigma^{(0)}, \varepsilon^{(0)})$, calcular $V_+^{(0)} = V_+(\theta^{(0)})$ y $V_-^{(0)} = V_-(\theta^{(0)})$, y generar

$$z_1^{(1)} \sim Bi(x, V_+^{(0)}), \quad z_2^{(1)} \sim Bi(N-x, V_-^{(0)}).$$

2. Paso posterior: basado en $(z_1^{(1)}, z_2^{(1)})$, generar de la distribución a posteriori para θ dados los datos aumentados. Esto es, generar $\theta^{(1)}$ como

$$\alpha_1^{(1)} \sim Be(A_p, B_p)), \sigma^{(1)} \sim Be(C_s, D_s), \varepsilon^{(1)} \sim Be(C_e, D_e).$$

Este algoritmo fue introducido, aún sin ninguna referencia al muestreo de Gibbs, por (Tanner y Wong, 1987), donde prueban que $h(\theta|x, z_1^{(t)}, z_2^{(t)})$ converge cuando $t \to \infty$ a $h(\theta|x)$, bajo algunas condiciones generales.

□

CAPÍTULO

4

Otros métodos

4.1 Slice Sampling

Como se ha visto, las distribuciones objetivo complejas π pueden complicar la generación de variables aleatorias condicionadas aún cuando la evaluación puntual de la función de densidad $\pi(u)$ en algún $u \in \mathcal{U}$ siga siendo posible. En este caso, otra estrategia para implementar MCMC es introducir una variable auxiliar Z con el objetivo de facilitar la simulación de la cadena para $(U, Z) \sim f(u, z) = \pi(u)f(z|u)$. Aquí, Z debería ser elegida tal que la cadena de la distribución ampliada $f(u, z)$ converja y tal que el estudio del soporte $\mathcal{U}$ de la correspondiente subcadena y la evaluación de los sumarios de interes para $\pi(\cdot)$ sean posibles.

Definiendo Z tal que $Z|U = u \sim U(0, \pi(u))$, se toma $(U, Z) \sim U(\mathcal{S})$, donde $\mathcal{S} = \{(u, z) : u \in \mathcal{U}, z \in [0, \pi(u)]\}$ y $U(\mathcal{S})$ se refiere a la distribución uniforme sobre el conjunto $\mathcal{S}$. Así, una forma de obtener un muestreo MC de π es generar una cadena de Markov con una distribución estacionaria que sea precisamente una uniforme multivariante en $\mathcal{S}$.

El "muestreo por porciones" o *slice sampling* es un método alternativo para generar un camino aleatorio en $\mathcal{S}$, moviéndose alternativamente en dos direcciones usando distribuciones uniformes. Usando en el primer paso $Z|U = u \sim U(0, \pi(u))$ sobre la recta real y en el segundo paso sobre $\mathcal{U}$ utilizando $U|Z = z \sim U(\mathcal{S}(z))$, con $\mathcal{S}(z) = \{u \in \mathcal{U} : \pi(u) \geq z\}$, nótese que la densidad marginal $f(z)$ es por tanto proporcional a la medida Lebesgue de $\mathcal{S}(z)$.

Si la cadena converge, este método genera así un muestreo aproximado de π considerando la subcadena correspondiente, requiriendo solo la evaluación de π en las posiciones simuladas por la distribución uniforme. Realmente, los esquemas de muestreo requieren obtener $\pi(\cdot)$ salvo la constante de normalización.

En resumen, el algoritmo *slice sampling* se define como sigue.

Algoritmo 3: Slice Sampler

Dado $(u^{(t)}, z^{(t)})$, comenzando con $t = 0$, simular:

1. $z^{(t+1)} \sim U(0, \pi(u^{(t)}))$;
2. $u^{(t+1)} \sim U(\{u : \pi(u) \geq z^{(t+1)}\})$;

y repetir el ciclo de estos dos pasos incrementando t cada tiempo. Nótese que $\pi(u)$ aquí denota la densidad o su núcleo, el que sea más fácil de evaluar.

Se concluye esta sección con algunos comentarios sobre esta aproximación cuyo nombre se atribuye a Neal (Neal, 1997) y volvió a publicarse en (Neal, 2003) y (Damien, Wakefield y Walker, 1999).

1. **Caso univariante**. Para U univariante el "muestreo por porciones" puede ser fácilmente ilustrado mediante una representación gráfica de la densidad $\pi(u)$ con U y Z marcados en los ejes horizontal y vertical, respectivamente, como se puede ver en la Figura **??**. El punto $(u^{(t)}, \pi(u^{(t)}))$ define en el eje vertical una porción sobre la cuál el valor $z^{(t+1)}$ es generado. La intersección de la línea $Z = z^{(t+1)}$ con $\pi(u)$ define el punto que delimita la porción horizontal $\mathcal{S}(z^{(t+1)})$ (un intervalo o unión de intervalos) sobre el cual $u^{(t+1)}$ se genera. En la práctica, la principal dificultad con este algoritmo está en el segundo paso ya que el soporte $\mathcal{S}(z)$ de la distribución en la porción horizontal puede ser complicada para una $\pi(u)$ multimodal, que necesitaría el uso de otros métodos de simulación en ese paso. En cualquier caso, por la naturaleza del algoritmo este funciona mucho mejor que otros algoritmos con densidades objetivos multimodales, en el sentido de la eficiencia de

exploración del soporte de tal distribución objetivo.

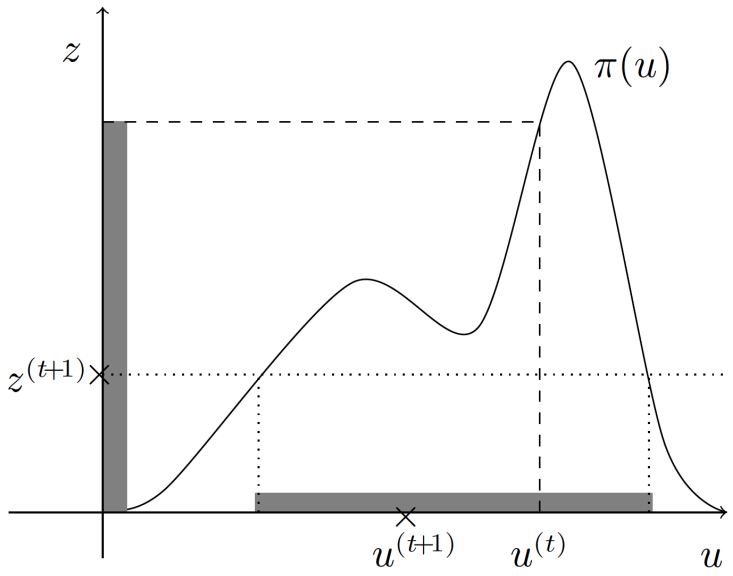

Figura 11: Slice sampler para distribución univariante. Recuperado de (Paulino et al., 2018).

2. **Muestreo por porciones y Gibbs**. La estructura del "muestreo por porciones"para $\mathcal{U}$ recalca que este algoritmo puede verse como un caso especial de un muestreo Gibbs de dos pasos para el modelo ampliado de $\pi(u)$ a $f(u, z) = \pi(u)f(z|u)$, una uniforme en $\mathcal{S}$. La secuencia $\{U^{(t)}\}$ es entonces una cadena de Markov con densidad de transición $P(u, \tilde{u}) = \int f(z|u)f(\tilde{u}|z)dz$ y distribución estacionaria $\pi(u)$.

   Esta interpretación es esencialmente válida también para distribuciones objetivo multivariantes, excepto para ejecuciones con un número grande de pasos. Como consecuencia, las condiciones de convergencia para el "muestreo por porciones" pueden deducirse de aquellas para el muestreo de Gibbs, basadas en la introducción de un vector de variables auxiliares que permita trabajar con las distribuciones objetivo complejas.

4.2 Monte Carlo Hamiltoniano: Dinámica Hamiltoniana

Un problema común en algunos de los esquemas MCMC discutidos anteriormente es, a veces, naturaleza local de las transiciones. Por ejemplo, con una función de transición de camino aleatorio de Metropolis-Hastings, en ocasiones solo se puede recorrer un rango pequeño del

espacio paramétrico. Con un muestreo de Gibbs, solo se actualiza un parámetro en un tiempo. Las altas correlaciones de los valores de la a posteriori generados pueden llevar a cadenas de Markov muy lentamente mezcladas. Una alternativa interesante, que nos puede permitir un movimiento más rápido por el espacio de parámetros es Monte Carlo Hamiltoniano, que puede consultarse en (Neal, 2011). La idea básica es muy simple. Hay tres pasos importantes en la construcción. Se denota $\pi(\theta)$ la distribución objetivo de interes, por ejemplo, la distribución a posteriori $h(\theta|x)$.

Primero, se añade un índice de tiempo (enteramente artificial) a θ, denotándose por $\theta(t)$. Eventualmente, después de la transición, se colocará el índice t de nuevo para obtener nuevos valores paramétricos.

Después, se empieza la construcción con un sistema de ecuaciones diferenciales para $d\theta(t)/dt$ que se conoce que deja invariante la función objetivo dada. Esto es, si se simulan transiciones siguiendo la solución de este sistema, esas transiciones dejarían la función objetivo sin cambios. Si se usara $\ln \pi$ como la función objetivo, se obtendrían transiciones que se muevan igual que las curvas de nivel de $\pi(\theta)$. Tal sistema de ecuaciones diferenciales es, por ejemplo, las ecuaciones Hamiltonianas de mecánicas Hamiltonianas. Lo que se necesita hacer es igualar la energía potencial con la distribución logarítmica a posteriori. Así se simulan las dinámicas Hamiltonianas. Se garantiza que los estados simulados tienen todos la misma densidad a posteriori. Esto es, en todo momento se mueven en el entorno de la distribución conjunta a posteriori.

Denotando $N(x|m, S)$ una función de densidad normal multivariante para el vector aleatorio x con media m y matriz de covarianzas S. Primero se aumenta el modelo de probabilidad a $\pi(\theta, p) = \pi(\theta)N(p|0, I)$, usando una distribución normal multivariante para p (puede ser cualquier otra pero este modelo hace las siguientes demostraciones más sencillas). La notación p para las variables adicionales es elegida con anticipación de la proxima interpretación del modelo ampliado. Nótese que ese modelo ampliado es inusual en el sentido de que θ y la variable implícita p son independientes. Esto resultará una gran simplificación del algoritmo. Denótese por

$$H(\theta, p) = -\ln \pi(\theta) + \frac{1}{2}p'p.$$

$H(\theta, p)$ es como menos el logaritmo neperiano de la función de densidad objetivo aumentada (ignorando los factores constantes). Se usa $H(\theta, p)$ como la función objetivo (potencial) para las ecuaciones Hamiltonianas, interpretando θ como posición y p como impulso. Lo único que quedaría por realizar es establecer las ecuaciones y después implementar una solución numérica aproximada a la ecuación diferencial. Sea $\theta = (\theta_1, \ldots, \theta_d)$ y $p = (p_1, \ldots, p_d)$ las dos posiciones e impulsos d-dimensionales. Las ecuaciones de Hamilton son

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial p_i} \quad \text{y} \quad \frac{dp_i}{dt} = -\frac{\partial H}{\partial \theta_i}.$$

La elección particular de H simplifica las siguientes ecuaciones, teniéndose que

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial p_i} = p_i \text{ y } \frac{dp_i}{dt} = -\frac{\partial H}{\partial \theta_i} = \frac{\partial \ln \pi}{\partial \theta_i}.$$

Hay una bonita interpretación de esta configuración. En la aplicación de las ecuaciones Hamiltonianas a la mecánica, los parámetros θ se convierten en la posición de un objeto y p en el impulso (esto es, velocidad $\times$ masa). Se puede ejemplificar esto con un objeto, en particular, con una bola en una pendiente. Entonces $\ln \pi(\theta)$ es el potencial, es decir, la energía debida a la posición, y $\frac{1}{2}p^2$ es la energía cinética. En ese caso se ha considerado que la bola ignora completamente el rozamiento. La mecánica Hamiltoniana describe cómo se moverá la bola en la posición θ y con un impulso p en el tiempo t . Su movimiento está determinado por la ecuación que establece que la suma de la energía cinética y la energía potencial debe ser constante.

Aproximación *salto de rana*

Para implementar las dinámicas Hamiltonianas se utiliza la discretización del sistema de ecuaciones diferenciales teniéndose el conocido como método del "salto de rana". Empezando en $(\theta(t), \theta(p))$ se genera $(\theta(t+\epsilon), p(t+\epsilon))$ usando una aproximación discreta sobre dos

subintervalos de longitud $\epsilon/2$ cada uno:

$$\begin{aligned} p_i\left(t+\frac{\epsilon}{2}\right) =& p_i(t) + \frac{\epsilon}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i} \\ \theta_i(t+\epsilon) =& \theta_i(t) + \epsilon p_i\left(t+\frac{\epsilon}{2}\right) \\ p_i(t+\epsilon) =& p_i\left(t+\frac{\epsilon}{2}\right) + \frac{\partial \ln \pi(\theta(t+\epsilon))}{\partial \theta_i}. \end{aligned} \tag{4.1}$$

Se denota $T_\epsilon(\theta(t), p(t)) = (\theta(t+\epsilon), p(t+\epsilon))$ la aproximación discreta implementada en la ecuación anterior. Es sencillo verificar que la aproximación es perfectamente reversible, esto es, $T_{-\epsilon}(\theta(t+\epsilon), p(t+\epsilon)) = (\theta(t), p(t))$, o $T_{-\epsilon}(\theta, p) = T_\epsilon^{-1}(\theta, p)$. Aún más fácil que volver atrás en el tiempo, todo lo que hay que hacer para enviar la bola de vuelta por donde ha venido es darle la vuelta. Esto es, $p \equiv -p$, o $T_\epsilon^{-1}(\theta, p) = T_\epsilon(\theta, -p)$. Nótese que el índice temporal en $\theta(t)$ y $p(t)$ sólo se usa para la implementación de $T_\epsilon(\cdot)$. Después del último paso en las ecuaciones anteriores, se vuelve al índice temporal de nuevo.

4.3 Probabilidades de transición en el método de Monte Carlo Hamiltoniano

Se sabe que las dinámicas Hamiltonianas dejan el modelo de probabilidad ampliada $h(\theta, p)$ invariante. Sin embargo, la aproximación $T_\epsilon(\cdot)$ no, ya que el error de aproximación depende del tamaño del paso. Pero este no es un problema de cara a las aplicaciones. Se utiliza $T_\epsilon(\cdot)$ para generar una propuesta en una probabilidad de transición del tipo Metropolis-Hasting. Comenzando por (θ, p), se genera una $(\tilde{\theta}, \tilde{p})$ propuesta con

$$(\tilde{\theta}, \tilde{p}) = \begin{cases} T_\epsilon(\theta, p), & \text{con probabilidad } \frac{1}{2} \\ T_{-\epsilon}(\theta, p), & \text{con probabilidad } \frac{1}{2}. \end{cases}$$

Por lo comentado anteriormente, la distribución propuesta es simétrica, dejando la probabilidad de aceptación $\alpha = \text{mín}(1, R)$ con $\ln(R) = H(\tilde{\theta}, \tilde{p}) - H(\theta, p)$. En esta implementación, un posible error en la ecuación (4.1) es una característica del método, no un problema, ya que

nos permite movernos a través de las curvas de nivel de $H(\cdot)$.

El truco final de Monte Carlo Hamiltoniano es particularmente interesante. Se alterna el paso M-H (4.3) con la probabilidad de transición de Gibbs para generar p de la condicional completa bajo $\pi(\cdot)$,

$$p \sim \pi(p|\theta) = N(0, I).$$

Aquí, se tiene en cuenta el hecho de que la posición y el impulso sean independientes, lo que hace el paso de Gibbs particularmente sencillo.

En resumen, el algoritmo procede con las dos probabilidades de transición siguientes.

Algoritmo 4: Algoritmo MCMC

1. Generar $p_i \sim N(0,1), i = 1, \ldots, d$.
2. Generar $(\tilde{\theta}, \tilde{p}) = \begin{cases} T_\epsilon(\theta, p), & \text{con probabilidad } \frac{1}{2} \\ T_{-\epsilon}(\theta, p), & \text{con probabilidad } \frac{1}{2}. \end{cases}$

   Con probabilidad $\alpha = \text{mín}\{1, \frac{\pi(\tilde{\theta})}{\pi(\theta)}\}$, aceptando la propuesta $\theta \equiv \tilde{\theta}$.

Para justificar la probabilidad de aceptación se plantea una modificación insignificante del Paso 2. Después de generar $(\tilde{\theta}, \tilde{p})$ como se describe, se reemplaza $\tilde{p}$ por $p^\dagger \sim \pi(p|\tilde{\theta})$ y se acepta α como la probabilidad de aceptación M-H para $(\tilde{\theta}, p^\dagger)$. Finalmente, no se necesita dejar constancia de p en el Paso 2 ya que p es inmediatamente sobreescrito en el Paso 1 de la siguiente iteración. El algoritmo resulta inalterado (y el error de aproximación aumenta con múltiples pasos).

El algoritmo puede simplificarse aún mas teniendo en cuenta que $p_i(t + \epsilon)$ en el último paso de la aproximación "salto de rana" (4.1) no se utiliza en el MCMC aquí descrito. Es inmediatamente reemplazado por el nuevo valor, $p_i \sim N(0,1)$ en el Paso 1 de la siguiente iteración. Las primeras dos líneas de (4.1), pueden ser escritas como

$$\theta_i(t + \epsilon) = \theta_i(t) + \epsilon \left\{ p_i(t) + \frac{\epsilon}{2} \frac{\partial \ln \pi(\theta(t))}{\partial \theta_i} \right\}.$$

Y, aún más sencillo, se puede resumir lo anterior con $p_i \sim N(0,1)$ en el Paso 1 para obtener

$$\theta_i(t+\epsilon)|\theta_i(t) \sim N\left\{\theta_i(t) + \frac{\epsilon^2}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i}, \epsilon^2\right\}.$$

Es decir, se reemplazan los Pasos 1 y 2 en uno solo.

Generar una propuesta,

$$\theta_i(t+\epsilon)|\theta_i(t) \sim N\left\{\theta_i(t) + \frac{\epsilon^2}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i}, \epsilon^2\right\}.$$

Aceptar con probabilidad $\alpha = \text{mín}\{1, R\}$ con el ratio de aceptación

$$R = \frac{\pi(\tilde{\theta})}{\pi(\theta)} \prod_{i=1}^{d} \frac{N\left\{\theta_i|\tilde{\theta}_i + \frac{\epsilon^2}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i}, \epsilon^2\right\}}{N\left\{\tilde{\theta}_i|\theta_i + \frac{\epsilon^2}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i}, \epsilon^2\right\}},$$

siendo el primer factor el ratio de las distribuciones objetivo y el segundo factor de las distribuciones propuestas para los movimientos propuesto y recíproco. Esta y otras implicaciones se pueden ver en (Welling y Teh, 2011). En particular, se observa la siguiente interpretación inteligente de la última versión del algoritmo HMC. Primero, siendo $\delta = \epsilon^2$, se reescribe la distribución propuesta como

$$\tilde{\theta}_i = \theta_i + \frac{\delta}{2}\frac{\partial \ln \pi(\theta(t))}{\partial \theta_i} + \sqrt{\delta} Z,$$

donde $Z \sim N(0,1)$ es la variable aleatoria normal estándar. De esta forma, notesé que para un δ grande el término gradiente domina, consierando que para un δ pequeño, el término normal domina.

CAPÍTULO

5

Aplicación

Este capítulo tratará de dar una visión más práctica de lo desarrollado, pues recoge una aplicación a un problema real. Para más información sobre la aplicación en cuestión, se puede consultar (Ruggeri, Sánchez-Sánchez, Sordo y Suárez-Llorens, 2021).

Se van a considerar una serie de datos procedentes de un proyecto de consultoría aportados por Fabrizio Ruggieri, profesor del *Istituto di Matematica Applicata e Tecnologie Informatiche* de Milán. Estos datos reflejan distintos fallos ocurridos en el sistema de puertas de 40 trenes italianos durante 7 años. La construcción de dichos trenes se llevó a cabo entre noviembre de 1989 y marzo de 1991 por una compañía de transporte europea , y fueron puestos en funcionamiento entre el 20 de marzo de 1990 y el 20 de julio de 1992. Los fallos se monotorizaron hasta la fecha del 31 de diciembre de 1998. El procedimiento para la contabilización de fallos fue el siguiente: cuando un error tenía lugar, se registraban la fecha del mismo, el código de la componente fallida y la medida del edómetro[1]. El conjunto de datos se analizó por primera vez en (Pievatolo, Ruggeri y Argiento, 2003), donde no se hizo distinción con respecto al tipo de fallo sucedido, dentro de los 7 que podían tener lugar. Años más tarde, se volvieron a analizar en (Pievatolo y Rugger, 2010), obteniéndose la siguiente clasificación:

[1] Instrumento de medición que calcula la distancia total o parcial recorrida por un objeto.

Código	Subsistema	Nº de partes	Total de fallos
1	sistema de apertura (eléctrico)	14	530
2	cables y pinzas	4	33
3	piezas mecánicas	67	1182
4	protecciones eléctricas	12	9
5	suministro de energía	2	7
6	engranaje neumático	31	295
7	electroválvulas	8	38

Tabla 1: Clasificación de los tipos de fallos y total de fallos por tipo de todos los trenes en 9 años. Traducción de Tabla 1 de (Pievatolo y Rugger, 2010).

Así se determinaron como irrelevantes ciertos fallos, por lo que finalmente solo fueron objeto de estudio dos de ellos, mecánicos y eléctricos. Se observó que las causas que provocaban dichos fallos eran dispares, por tanto se consideró que lo mejor era estudiarlos como superposición de fallos por cualquier causa. La regularidad del patrón y el teorema de Grigelionis, que puede verse en (Thompson, 1988), sobre la superposición de muchos procesos en uno justifican el uso de procesos de Poisson no homogéneos. Para este estudio, se considerará tan solo uno de ambos fallos, siendo este el asociado al fallo eléctrico en los controles de apertura donde se recogieron 530 fallos del 19 de septiembre de 1991 al 31 de diciembre de 1998.

En lo que sigue, se tratará de encontrar un modelo que describa el histórico de fallos eléctricos en los controles de apertura, además de ser capaces de predecir el número de fallos que tendrán lugar en un intervalo de tiempo futuro utilizando para ello la inferencia bayesiana.

5.1 El modelo

Modelizando el problema como un sistema complejo, tal y como se argumenta en (Pievatolo y Rugger, 2010), el número total de fallos ocurridos en el sistema eléctrico de apertura en el intervalo $(0, t]$, denotado por $N(t)$, sigue un proceso de Poisson no homogéneo con una función

de intensidad común, $\lambda(t)$, y una función del valor medio creciente e invertible,

$$m(t) = E[N(t)] = \int_0^t \lambda(s)ds,$$

con $m(\infty) = \infty$. En el siguiente gráfico, se pueden observar el número de fallos del tipo 1 acumulado frente al tiempo en días:

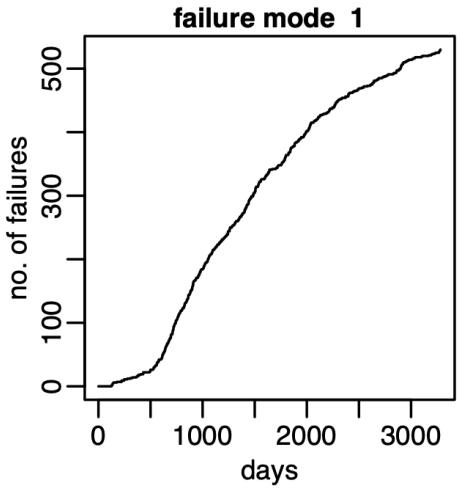

Figura 12: Número de fallos acumulados del tipo eléctrico frente al tiempo. Recuperado de (Pievatolo y Rugger, 2010)

En este caso, se tomará como modelo el popular proceso de *power law process* con función de intensidad

$$\lambda(t|\theta) = M\beta t^{\beta-1}, \quad \theta = (M, \beta) \in \mathbb{R}^+ \times \mathbb{R}^+, \tag{5.1}$$

obteniendose al integrar

$$m(t|\theta) = Mt^{\beta}, \quad \theta = (M, \beta) \in \mathbb{R}^+ \times \mathbb{R}^+.$$

Esta elección está basada en el hecho de que el modelo *power law process* es el proceso no homogéneo de Poisson más utilizado en fiabilidad gracias a su simplicidad. Se supone $0 < \beta < 1$, indicando la información sobre el crecimiento de la fiabilidad, y se excluyen los casos de constantes ($\beta = 1$) y decrecimiento de la fiabilidad ($\beta > 1$). Se puede ampliar la información sobre análisis estadístico y procesos de Poisson no homogéneos en (Thompson, 1988), (Kingman, 1993), (Rigdon y Basu, 2000), (Aven y Jensen, 2000) y (Insua, D., Ruggeri

y Wiper, 2012), y un amplio catálogo de funciones de intensidad dadas por (McCollin, 2014).

5.2 Función de verosimilitud

Sea $N(t)$ un proceso de Poisson no homogéneo con función de intensidad del tipo *power law process* con función de intensidad $\lambda(t|\theta)$ dada en (5.1). Se supone el vector $t = (t_1, \ldots, t_n)$ dado por el números de fallos observados en el intervalo $(0, T]$ satisfaciendo que $t_1 < \ldots < t_n$. Por el Teorema 5.4. de (Insua et al., 2012), la función de verosimilitud es la que sigue:

$$\begin{aligned} l(\theta|t) &= [\textstyle\prod_{i=1}^{n} \lambda(t_i)] \cdot \exp(-m(T|\theta)) \\ &= \left[\textstyle\prod_{i=1}^{n} M\beta t_i^{\beta-1}\right] \cdot \exp(-MT^{\beta}), \\ &= M^n \beta^n \exp((\beta-1)\textstyle\sum_{i=1}^{n} \ln(t_i)) \exp(-MT^{\beta}). \end{aligned}$$

5.3 A priori y a posteriori

Se asume que la distribución a priori $\pi(\theta) = \pi(M, \beta)$ definida sobre $\Theta = \mathbb{R}^+ \times \mathbb{R}^+$ es un vector aleatorio bivariante cuyas distribuciones marginales son exponenciales independientes, $Exp(\lambda)$ y $Exp(\mu)$, asociadas a los parámetros M y β, respectivamente. Por ende,

$$\pi(\theta) = \lambda\mu \exp(-\lambda M)\exp(-\mu\beta), \quad \theta = (M, \beta) \in \Theta, \tag{5.2}$$

donde los parámetros $\lambda > 0$ y $\mu > 0$, conocidos como hiperparámetros, se suponen determinados. A partir de los valores iniciales M_0 y β_0, se toma $\lambda = 1/M_0$ y $\mu = 1/\beta_0$ ya que $E^{\pi}(\theta) = (1/\lambda, 1/\mu)$.

Se calcula pues, como producto de la distribución a priori y la función de verosimilitud, la distribución a posteriori, la cual sería proporcional a

$$\pi_t(\theta) \propto M^n \beta^n \exp\left[(\beta-1)\sum_{i=1}^{n} \ln(t_i)\right] \exp(-MT^{\beta})\lambda \exp(-\lambda M)\mu \exp(-\mu\beta), \tag{5.3}$$

siendo $\theta = (M, \beta) \in \Theta$. Si se pretendiera calcular las distribuciones condicionadas a los distintos parámetros, se obtendría que

$$\pi_t(\beta|M) \propto \beta^n \exp\left[\beta(\sum_{i=1}^{n} \ln(t_i) - \mu)\right] \exp(-MT^\beta), \tag{5.4}$$

y

$$\pi_t(M|\beta) \propto M^n \exp(-M(T^\beta + \lambda)). \tag{5.5}$$

Es observable que $\pi_t(M|\beta)$ tiene una forma conocida, pues puede identificarse como el núcleo de una distribución gamma de parámetros $n+1$ y $\lambda + T^\beta$, esto es

$$\pi_t(M|\beta) \sim G(n+1, \lambda + T^\beta).$$

Sin embargo, $\pi_t(\beta|M)$, no tiene forma de distribución reconocible.

5.4 Aplicación de algoritmos: Metropolis-Hasting y Gibbs

Con este ejemplo se pretende calcular, en primer lugar, una muestra de pares (M, β). Para ello, se comenzará utilizando el método de Gibbs para calcular M, que sigue una distribución conocida. Por otro lado, para β utilizaremos el método de Metropolis-Hasting general, ya que no sigue una distribución conocida. Una vez calculadas dichas muestras se procederá a ejecutar un proceso de *burn-in*.

Nuestro algoritmo, paso a paso, sería el siguiente:

1. Se calcula, para comenzar, los estimadores de máxima verosimilitud de M y β.
2. Se toma $\sigma = 0.17$.
3. Se inicializa una matriz con los (M, β) calculados en el paso 1.
4. Se genera M como un valor aleatorio de gamma ya que es conocida, utilizando el β anterior.
5. Se genera β mediante el método de Metropolis-Hasting, ya que tiene una distribución desconocida.

(a) Se introduce el tiempo en días que ha transcurrido desde que se empezaron a contabilizar los fallos (T), los (M, β) anteriores, el número de iteraciones que se quiere calcular (n), la suma del logaritmo neperiano de los datos proporcionados $(\sum \ln t_i)$ y el parámetro de la distribución a priori de β (μ).

(b) Se inicializa el contador a 0.

(c) Se calculan dos valores de una uniforme (0,1), u y p.

(d) Se define x_n como aquel que cumple que

$$\frac{F_N(x_n) - F_N(0)}{F_N(1) - F_N(0)} = p,$$

es decir,

$$x_n = F_N^{-1}(F_N(0) + p(F_N(1) - F_N(0)))$$

(e) Se calcula

$$a = \text{mín}\left[1, \frac{{x_n}^n \exp(-(\mu - \sum \ln t_i)x_n) \exp(-MT^{x_n}) \frac{f_N(\beta)}{1-F_N(0)}}{\beta^n \exp(-(\mu - \sum \ln t_i)\beta) \exp(-MT^{\beta}) \frac{f_N(x_n)}{1-F_N(0)}}\right].$$

(f) Si $u < a$, x_n es el valor requerido, en caso contrario, se retorna a (b)

6. Se almacenan los nuevos (M, β) en mi matriz, añadiendolos como vector fila.

7. Se repite el procedimiento anterior tantas iteraciones como se deseen, en este caso, 10000.

8. Se diferencian los vectores columna como las muestras de M y β.

9. Se realiza el proceso de *burn-in*, eliminando en este caso concreto los 1000 primeros valores.

A continuación, en la Figura 13, se muestras el histograma bivariante asociado a la muestra generada por el algoritmo. A simple vista se observa una distribución fuertemente asimétrica.

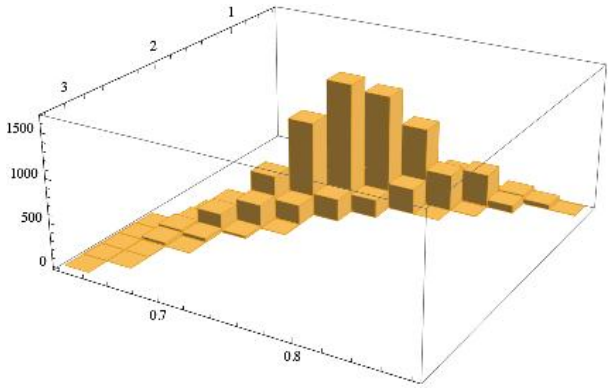

Figura 13: Histograma bivariante de la muestra simulada.

Además, se puede observar a continuación una representación de la muestra de manera marginal, considerando por separado M y β.

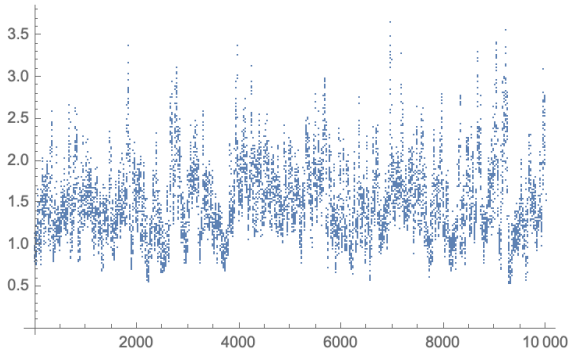

Figura 14: Valores para M de la muestra generada.

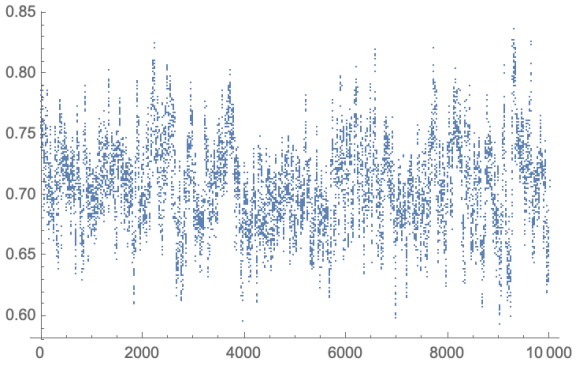

Figura 15: Valores para β de la muestra generada.

Sin embargo, lo que se quiere predecir es el número medio de fallos que ocurrirán en intervalos de tiempo futuros de la forma $[T, T + h]$. Por lo tanto, es necesario calcular el siguiente

valor esperado dado por la expresión

$$E[N(T+h) - N(T)] = \int_{T}^{T+h} \lambda(t|\theta)dt = M((T+h)^{\beta} - T^{\beta}). \tag{5.6}$$

Se tiene que 5.6, desde el punto de vista bayesiano, es una variable aleatoria. A partir de sustituir cada uno de los 9000 valores muestrales calculados en dicha expresión, se obtiene una muestra del número medio de fallos, la cual se representa en el siguiente histograma.

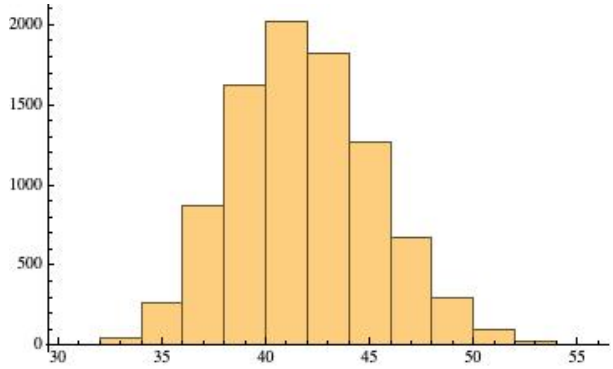

Figura 16: Histograma asociado a la estimación del número medio de fallos

Además, se obtiene la función de distribución empírica.

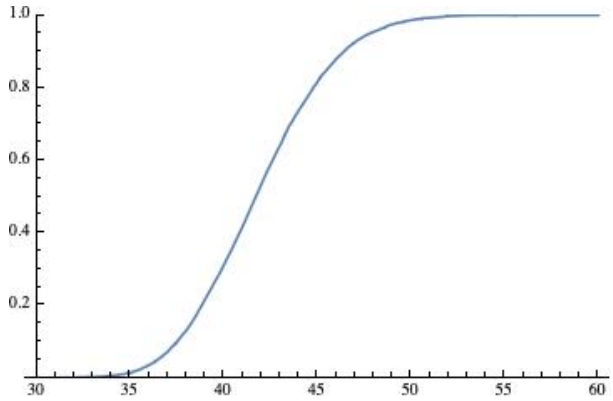

Figura 17: Fución de distribución empírica del número medio de fallos

A partir de la distribución empírica del número medio de fallos a posteriori, se obtiene una estimación media muestral y un intervalo de credibilidad a traves de los percentiles muestrales. Dicha información se resume en la siguiente Tabla. Los datos se recopilan hasta diciembre de 1997 y se validan con el valor real obtenido en 1998.

T	Nº fallos real	$\mathrm{I.C.}_{95\,\%}$	Estimación fallos
1997-1	23	[29.57,54.82]	42

Tabla 2: Estimación del número de fallos para el año 1998.

Se muestra a continuación, en la Tabla 3, la estimación del número medio de fallos que van a ocurrir de 1992 a 1998, conociendo los datos de los años anteriores. Se almacena, en la siguiente Tabla, el valor real de los fallos recogidos en el estudio, el número de fallos estimado y el intervalo de credibilidad al 95 $\%$ de fiabilidad de dichas predicciones.

Como se puede ver, en los primeros años se tienen unas estimaciones bastante buenas, pero en los últimos años ya no es tan acertada. Como es natural, con el transcurso de los años, el número de fallos va disminuyendo, ya que se irán subsanando fallos comunes a los ya cometidos.

T	Nº fallos real	$\mathrm{I.C.}_{95\,\%}$	Estimación fallos
1992-1	83	[65.42,100.93]	83
1992-2	72	[62.23,96.91]	79
1993-1	72	[46.76,77.44]	62
1993-2	62	[43.71,73.49]	58
1994-1	62	[41.64,70.83]	56
1994-2	42	[39.36,67.85]	53
1995-1	42	[40.30,69.06]	54
1995-2	35	[38.57,66.82]	53
1996-1	35	[34.21,61.07]	47
1996-2	23	[32.74,59.10]	45
1997-1	23	[29.57,54.82]	42

Tabla 3: Estimación del número de fallos entre 1992 y 1998.

La elección de la función de intensidad $\lambda(t)$ es posiblemente uno de los modelos más utilizado en ingeniería, sin embargo, llega un momento en que λ no es capaz de predecir el salto de mejora tan grande que se produce en los últimos años. En otras palabras, el tren está funcionando mucho mejor de lo que el modelo espera.

5.5 Conclusiones

Se puede decir que en el estudio que se ha presentado, se han desarrollado varios campos de la estadística con el fin de unirlos todos para hacer estimaciones. Se han revisado todos los aspectos necesarios para entender el algoritmo, el cual es muy útil ya que permite adquirir la capacidad de calcular una muestra de cualquier función de distribución.

Se concluye este trabajo diciendo que la generación de muestras tiene una gran importancia en la vida real, ya que, gracias a la estadística bayesiana, conociendo muestras anteriores, se puede conocer el comportamiento futuro. Esto es de gran utilidad en numerosos campos: para predecir el número de fallos de un sistema, tal y como hemos desarrollado en el último capítulo, para fines meteorológicos, para estudiar distintos riesgos financieros, así como un sinfín de aplicaciones que nos atañen en nuestro día a día.

Como continuación a este trabajo, se podría ampliar estudiando los diagnósticos de convergencia, que consiste en ver la rapidez con la que los valores simulados convergen a la distribución estacionaria. Además, se puede mejorar la distribución a priori para que la muestra simulada en nuestra aplicación a un ejemplo real, se ajuste aún más a los valores recogidos en la realidad.

A destacar, por último, la importancia y utilidad de la coexistencia de la estadística bayesiana con la inferencia clásica que logran un mejor entendimiento de la realidad a estudiar.

Bibliografía

Amaral Turkman, M. A., Paulino, C. D. y Müller, P. (2019). *Computational Bayesian Statistics*.

Aven, T. y Jensen, U. (2000). A general minimal repair model. *Journal of Applied Probability*, *37*, 187–197.

Barranco-Chamorro, I. y Gulati, S. (2015). Some estimation techniques in reliability and survival analysis based on record-breaking data. *Theory and Practice of Risk Assessment*, *136*, 333–348.

Barranco-Chamorro, I., Luque-Calvo, P., Jiménez-Gamero, M. y Alba-Fernández, M. (2017). A study of risks of Bayes estimators in the generalized half-logistic distribution for progressively type-II censored samples. *Mathemataics and Computer in Simulation*, *137*, 130–147.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions*, *53*, 370–418.

Benitez, B. (2021). *Métodos computacionales en Inferencia Bayesiana: métodos Montecarlo.*

Damien, P., Wakefield, J. y Walker, S. (1999). Gibbs sampling for bayesian non-conjugate and hierarchical models by using auxiliary variables. *Journal of The Royal Statistical Society Series B-statistical Methodology*, *61*, 331-344.

Geman, S. y Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-6*(6), 721-741.

Givens, G. H. y Hoeting, J. A. (2005). *Computational Statistics*. Wiley.

Hastings, W. K. (1970). Monte carlo sampling methods using Markov chains and their applications. *Biometrika*, *57*(1), 97–109.

Heidelberger, P. y Welch, P. D. (1983). Simulation run length control in the presence of an initial transient. *Oper. Res.*, *31*, 1109-1144.

Higuera de Frutos, S. (2017). Críticas y Reseñas Aventuras de un matemático. *Pensamiento Matemático*, *VII*(1), 199–202.

Insua, R., D., Ruggeri, F. y Wiper, M. (2012). *Bayesian analysis of stochastic process models*. New York, USA: Wiley.

Kingman, J. (1993). *Poisson Processes*. Oxford, UK: Clarendon Press.

Mackay, D. C. (1986). Introduction to Monte Carlo Methods.

Main Yaque, P., Navarro Veguillas, H. y Morales Fernández, A. (2019). *Simulación con ejercicios en R*. Ediciones Complutense UNED.

McCollin, C. (2014). *Intensity functions for nonhomogeneous Poisson precesses*. Wiley StatsRef: Statistics Reference Online. John Wiley and Sons, Ldt.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. y Teller, E. (1953). Equation of state calculations by fast computing machines. *J. Chem. Phys*, *21*, 087–1092.

Neal, R. M. (1997). Markov chain monte carlo methods based on slicing the density function. *Technical Report*.

Neal, R. M. (2003). Slice sampling. *The Annals of Statistics*, *31*(3), 705 – 767.

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. In *Handbook of markov chain monte carlo* (chap. 5). Chapman and Hall / CRC Press.

Paulino, C. D., Amaral Turkman, M. A., Murteira, B. y Silva, G. (2018). *Estatística Bayesiana* (2nd ed.). Fundacão Calouste Gulbenkian.

Pievatolo, A. y Rugger, F. (2010). Bayesian modelling of train door reability. *In the Oxford Handbook of Applied Bayesian Analysis*.

Pievatolo, A., Ruggeri, F. y Argiento, R. (2003). Bayesian analysis and prediction of failures undergroun trains. *Quality Reability Engineering International*, *19*, 327–336.

Rigdon, S. y Basu, A. (2000). *Statistical methods for the reliability of repairable systems*. New York: John Wiley.

Robert, C. P. y Casella, G. (2004). *Monte Carlo Statistical Methods* (2nd ed.). New York.

Ruggeri, F., Sánchez-Sánchez, M., Sordo, M. Á. y Suárez-Llorens, A. (2021). On a New Class of Multivariate Prior Distributions: Theory and Application in Reliability. *Bayesian Analysis*, *16*(1), 1–30.

Tanner, M. A. y Wong, W. H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, *82*(398), 528–540.

Thompson, W. J. (1988). *Point process models with applications to safety and reability*. New York: Chapman and Hall.

Ulam, S. y Metropolis, N. (1949). The Monte Carlo Method. *Journal of the American Statistical Association*, *44*(247), 335–341.

Ulam, S. M. (1991). *Adventures of a Mathematician. Memorias de Stanislaw M. Ulam*. Nueva York: University of California Press.

Welling, M. y Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, 681–688.

APÉNDICE

A

Apéndice: Códigos en R

A continuación se anexan los distintos códigos realizados en el lenguaje de programación *R*, mediante el software *Rstudio*.

A.1 Ejemplo 1.1.

Se define el vector p, así como la distribución a priori.

```
p = c(0.01,0.02,0.03,0.05,0.08,0.10)
prior=c(0.15,0.10,0.01,0.27,0.34,0.13)
```

Después, se calcula la función de verosimilitud y la distribución a posteriori como sigue:

```
likelihood=dbinom(0, 7, p)
posteriori = (prior*likelihood)/sum(prior*likelihood)
```

Se representan tanto la distribución a priori como a posteriori:

```
plot(p,prior,xlab = "valores de p", ylab = "valores distribución a priori")
```

```
plot(p,posteriori,xlab = "valores de p", ylab = "valores distribución a posteriori")
```

Y por último se calcula la densidad predictiva.

```
q=1-p
Z0=sum(posteriori*q)
Z1=sum(posteriori*p)
```

A.2 Ejemplo 2.1.

Se reprensenta el área a calcular y se calcula el valor real de la integral.

```
q<-function(x)((x+3)/(3*x^4+x^2+1))
curve(q(x),xlim=c(-10,10))
```

```
intq<-integrate(q,-Inf,Inf)
```

Se declara la función de la integración de Montecarlo y se aplica para 1000 y 10000, representándose en una sola gráfica.

```
intg.f<-function(y){mnorm<-rnorm(y)
mean(q(mnorm)/dnorm(mnorm))}
intg.v<-vector(length=100)
for(i in 1:100) intg.v[i]<-intg.f(1000)
plot(intg.v,type='l',col = 'red')
for(i in 1:100) intg.v[i]<-intg.f(10000)
lines(intg.v,type='l',col = 'purple')
abline(h=intq$value)
```

A.3 Ejemplo 2.2.

Código correspondiente al ejemplo 2.2.

Se declara, en primer lugar, la función de la cual se quieren simular muestras.

```
p<-function(theta){theta^(-n/2)*exp(-a/(2*theta))}
```

En esta ocasión se toman los siguientes parámetros fijos.

```
n=5
a=4
```

Se define una función que ejecute el algoritmo de Metropolis-Hasting que tendrá como argumentos el valor inicial para θ y el número de iteraciones deseado.

```
MH <- function(valorini, iter){

  # Se declara una matriz columna donde se recogerán los valores
  # generados mediante el algoritmo, tanto los valores aceptados como los
  # rechazados.

  cadena = array(dim = c(iter+1))
```

```
# Se inicializa la cadena con el valor inicial introducido como
# argumento  se la función, así como la cadena muestra donde se recogen
# únicamente los valores aceptados.

cadena[1] = valorini
muestra<-c(valorini)

# Se inicializan las siguientes variables que recogerán el número de
# valores aceptados y rechazados.

nacept=0
nrech=0

# A continuación, se realiza un bucle donde se implementa el algoritmo
# en sí.
for (i in 1:iter){



# Se utiliza como distribución propuesta una uniforme (0,100)
  proposaltheta=runif(1,min=0,max=100)

# Ratio de aceptación del algoritmo
  ratio=p(proposaltheta)/p(cadena[i])

# Control de errores: Si en algún caso el ratio es un valor no numérico,
# se recoge en el estado actual el valor del estado anterior.

  if (ratio == "NaN"){

    cadena[i+1]=cadena[i]
    nrech=nrech+1
  }
  else{



# Se genera un valor aleatorio de una distribución uniforme (0,1) y se
# compara con el ratio de aceptación del algoritmo.

    if(runif(1,0,1)<min(1,ratio)){
      cadena[i+1]=proposaltheta
      muestra<-c(muestra,proposaltheta)
      nacept=nacept+1
    }

  else{

# Si se rechaza el valor propuesto se almacena en la cadena el valor del
# estado anterior.
    cadena[i+1] = cadena[i]
    nrech=nrech+1

  }
 }
}
# Se devuelve como resultado la cadena generada, así como el número de
# valores aceptados, valores rechazados y la muestra con los valores
# aceptados.
 lista=list(cadena,nacept,nrech,muestra)
```

```
# Este caso se mostrarán el número de
 # valores aceptados, valores rechazados y la muestra con los valores
 # aceptados
  listaacep=list(nacept,nrech,muestra)
  return(listaacep)
}
```

```
MH(50,5000)
```

A.4 Ejemplo 2.3.

Código correspondiente al ejemplo 2.3.

En primer lugar se define la función g de la cuál se quiere simular una muestra. A continuación se define la función MH que genera la muestra de los valores tanto aceptados como rechazados del algoritmo MH con distribución propuesta χ^2 y con parámetros de entrada el valor inicial, el número de iteraciones deseado y los grados de libertad que se decidan para la distribución propuesta.

```
g<-function(x){x^(n/2-1)*exp(-x/2)}
```

```
MH <- function(valorini, iter,df){
  cadena = array(dim = c(iter+1))
  cadena[1] = valorini
  muestra<-c(valorini)
  nacept=0
  nrech=0

  for (i in 1:iter){

    proposal=rchisq(1,df,ncp = 0)

    ratio=g(proposal)/cadena[i]

    if (ratio == "NaN"){

      cadena[i+1]=cadena[i]
      nrech=nrech+1
    }

    else{

      if(runif(1,0,1)<min(1,ratio)){
        cadena[i+1]=proposal
        muestra<-c(muestra,proposal)
        nacept=nacept+1
      }
```

```
    else{

      cadena[i+1] = cadena[i]
      nrech=nrech+1

    }
   }
  }

  return(cadena)
}
```

Seguidamente, se generan las gráficas de los resultados de generar las muestras mediante la función MH con valor inicial 2, 1000 iteraciones y 2 y 10 grados de libertad respectivamente.

```
plot(MH(2,1000,2),type='l')
```

```
plot(MH(2,1000,10),type='l')
```

A.5 Ejemplo 3.1.

Código correspondiente al ejemplo 3.1.

```
## Gibbs aplicado a la normal bivariate
nsamples <- 1000
rho<-0.8
muX=muY=0
sX=1
sY=0.5
msigma=matrix(c(sX^2, sX*sY*rho, sX*sY*rho, sY^2), nrow=2)
## Contour plot de la distribucion
fiftyticks <- seq(from=-3, to=3, length.out = 50)
y <- rep(fiftyticks, 50)
x <- rep(fiftyticks, each=50)
```

```
## Curvas de nivel de la distribución bivariante
z<-matrix( dmvnorm(cbind(y,x), c(muX, muY), msigma), 50, 50)
contour(list(x=fiftyticks, y=fiftyticks, z=z),
        xlim=c(-3,3), ylim=c(-3,3), drawlabels=FALSE)
```

```
## Condicionadas
sxcondy=sqrt(sX^2*(1-rho^2)) ## desviacion tipica de X|Y=y
sycondx=sqrt(sY^2*(1-rho^2)) ## desviacion tipica de Y|X=x
rxy=rho*(sX/sY) ## pendiente de recta X|y
ryx=rho*(sY/sX) ## pendiente de recta Y|x

## muestras
## vectores vacios con ceros
xsample=ysample=rep(0,nsamples)
xsample[1]=-2 ## valores iniciales de la distribucion
ysample[1]=2
```

```
## Generamos valores de las condicionadas
set.seed(123)
for (i in c(1:(nsamples-1))){
  xsample[i+1] <- rnorm(1, mean=muX+rxy*(ysample[i]), sd=sxcondy)
  ysample[i+1] <- rnorm(1, mean=muX+ryx*(xsample[i+1]), sd=sycondx)
}
```

```
## Representamos lo que hemos hecho
par(mfrow=c(1,2))
## Superponemos al grafico de contorno anterior los valores generados
contour(list(x=fiftyticks, y=fiftyticks, z=z),
        xlim=c(-3,3), ylim=c(-3,3), drawlabels=FALSE)

## Quitamos los 500 primeros valores para evitar la correlacion
points(xsample[-c(1:500)], ysample[-c(1:500)], pch=21, bg="purple" )
## Etiquetamos los 5 primeros valores generados para ver como se comporta
## como depende del valor en que comenzamos
for (j in c(1:5)){
  points(xsample[j], ysample[j]-0.005, pch=21, cex=3.5, bg="white")
  text(xsample[j], ysample[j], as.character(j))
}

## Se ha obtenido una muestra de la Normal bivariante
```

```
## Comprobamos la similitud de los estadisticos muestrales con los
## parametros del modelo

cor.test(xsample, ysample)
```

```
## Aumento del numero de simulaciones 10000
## Se eliminan los 2000 primeros, b=2000 y veo si tengo muestras de la Normal bivariante
nsamples <- 10000
b <- 2000 ## burning

## muestras
## vectores vacios con ceros
xsample=ysample=rep(0,nsamples)
## valores iniciales de la distribucion
xsample[1]=-2
ysample[1]=2

## Generamos valores de las condicionadas
set.seed(123)
for (i in c(1:(nsamples-1))){
  xsample[i+1] <- rnorm(1, mean=muX+rxy*(ysample[i]), sd=sxcondy)
  ysample[i+1] <- rnorm(1, mean=muX+ryx*(xsample[i+1]), sd=sycondx)
}
xsamplefinal=xsample[-c(1:b)]
ysamplefinal=ysample[-c(1:b)]

cor.test(xsamplefinal, ysamplefinal)
```

```
## Distribuciones marginales obtenidas
## Graficos con los histogramas
par(mfrow=c(1,2))
```

```
hist(xsamplefinal, col="lightblue", main="Histograma X1")
hist(ysamplefinal, col="lightblue", main="Histograma X2")
```

A.6 Ejemplo 3.3.

Código correspondiente al ejemplo 3.3.

Se comienza declarando la funciones condicionales que, como se ha visto en la redacción del ejemplo, siguen distribuciones conocidas.

```
betagibbs<-function(x,alpha,n,beta){rbeta(1,x+alpha,n-x+beta)}
```

```
bingibbs<-function(n,y){rbinom(1,n,y)}
```

En este caso, se toman los siguientes parámetros:

```
n=10
alpha=1
beta=2
```

Se inicializa con $y_0 = \frac{1}{2}$, y se generará x_0 de la siguiente forma:

```
y0=0.5
x0<-bingibbs(n,y0)
x0
```

```
## [1] 9
```

```
secuencia<-matrix(0,5001,2)
secuencia[1,1]<-x0
secuencia[1,2]<-y0
for(i in 1:5000){
 secuencia[i+1,2]<-betagibbs(secuencia[i,1],alpha,n,beta)
 secuencia[i+1,1]<-bingibbs(n,secuencia[i+1,2])
 i=i+1
}

data.frame(pasos=1:5000,xi=secuencia[1:5000,1],yi=secuencia[1:5000,2])
```

```
pasos = 1:5000
xi = secuencia[1:5000,1]
yi= secuencia[1:5000,2]

plot(pasos,yi,type='l',ylab=expression(y[i]),col='grey')
```

```
plot(pasos,yi,type='l',ylab=expression(y[i]),col='grey')
```

```
hist(yi[501:5000])
```

```
library(coda)
HW=as.mcmc(xi,yi)
heidel.diag(HW, pvalue=0.05)
```

```
##
##      Stationarity start     p-value
##      test         iteration
## var1 passed       1         0.66
##
##      Halfwidth Mean Halfwidth
##      test
## var1 passed    3.27 0.194
```

A.7 Ejemplo Capítulo 5

```
L[M_, β_] := M^n β^n Exp[(β - 1) datos] Exp[-M T^β]
                     |exponencial          |exponencial
PriorM[M_] := λ Exp[-λ * M]
                  |exponencial
Priorβ[β_] := μ Exp[-μ * β]
                  |exponencial

Posterior[M_, β_] := L[M, β] * PriorM[M] * Priorβ[β]
PosteriorM[M_] := PDF[GammaDistribution[n + 1, 1 / (λ + T^β)], M]
                      |fun···  |distribución gamma
Posteriorβ[β_] := β^n Exp[- (μ - datos) β] Exp[-M T^β]
                     |exponencial          |exponencial

datos = 2499.97;
n = 387;
T = 2655;
M = 1.13664166;
β = 0.78284;
λ = 1 / M;
μ = 1 / β;
MH = Function[{βini, Mini, datos, T, expμ, n},
      |función

    ClearAll[cont, p, a, normμ, normσ, x, xn, u];
    |borra todo
    cont = 0;
    normμ = βini;
    normσ = 0.17;
    While[cont < 1,
    |mientras

     p = RandomVariate[UniformDistribution[]];
         |variable aleatoria  |distribución uniforme
     u = RandomVariate[UniformDistribution[]];
         |variable aleatoria  |distribución uniforme
     xn = Quantile[NormalDistribution[normμ, normσ], CDF[NormalDistribution[normμ, normσ], 0]
          |cuantil  |distribución normal              |fun··· |distribución normal
        + p (1 - CDF[NormalDistribution[normμ, normσ], 0]) ];
                   |fun··· |distribución normal

     a = Min[1, xn^n Exp[- (expμ - datos) xn] Exp[-Mini T^xn] (PDF[NormalDistribution[normμ, normσ], βini] / (1 - CDF[NormalDistribution[normμ, normσ], 0])) / (βini^n Exp[- (expμ - datos) βini] Exp[-Mini T^βini] (PDF[NormalDistribution[normμ, normσ], xn] / (1 - CDF[NormalDistribution[normμ, normσ], 0])))];
         |mínimo

     If[u < a, x = xn; cont = cont + 1]
     |si
    ];
    x
];
```

```
myMCMC = Function[{niter, expλ, expμ, n, T, inicialM, inicialβ, datos},
         |función
    ClearAll[m1, m2, mat];
    |borra todo
    mat = {{inicialM, inicialβ}};
    For[i = 1, i ≤ niter, i++,
    |para cada
     m1 = RandomVariate[GammaDistribution[n + 1, 1/(expλ + T^mat[[i,2]])]];
          |variable aleatoria    |distribución gamma
     m2 = MH[mat[[i, 2]], m1, datos, T, expμ, n];
     mat = Append[mat, {m1, m2}]
           |añade
    ];
    mat
   ];
Muestra = myMCMC[10 000, λ, μ, n, T, M, β, datos];

MuestraM = Transpose[Muestra][[1]];
           |transposición
Muestraβ = Transpose[Muestra][[2]];
           |transposición
ListPlot[MuestraM]
|representación de lista
ListPlot[Muestraβ]
|representación de lista
```