Metropolis-Hasting

Introducción

Uno de los grandes problemas que existen en el mundo de la estadística es que hay distribuciones cuyas funciones de densidad son tan complejas que resultan difíciles de trabajar. Es por esto que muchas veces resulta de utilidad trabajar con muestras aleatorias, y a partir de estas responder determinadas preguntas. En la práctica, es común encontrarse con variables aleatorias con funciones de densidad cuyo muestreo directo no es simple.

Para estos casos existen distintas técnicas que nos permiten obtener muestras que, si bien no son tomadas de manera realmente independiente, se comportan de manera muy similar a como lo haría una muestra aleatoria independiente tomada de la función de densidad.

Particularmente en el campo de la *Estadística Bayesiana*, esto resulta útil ya que permite obtener muestras de la distribución a posteriori, la cual se puede utilizar por ejemplo para obtener un intervalo de credibilidad del parámetro bajo estudio.

Un método sencillo de emplear y que reporta buenos resultados es el algoritmo de *Metropolis-Hastings*.

Metodología

En esta sección se presentan los algoritmo de metrópolis Hastingspara variables aleatorias unidimensionales y bidimensionales

Una vez ejecutado el algoritmo terminaremos con una muestra de tamaño n: $\{\theta^{(1)}, \theta^{(2)}, ..., \theta^{(n)}\}$

Para obtener cada muestra $\theta^{(i+1)}$ se partirá desde $\theta^{(i)}$ 1. se propondrá un nuevo valor θ' en función de $q(\theta' \ |\theta^{(i)})$ 2. Se calcula la probabilidad de que $\theta^{(i+1)}$ tome el valor de θ': $\alpha = min\left\{1; \frac{f(\theta')}{f(\theta)} \frac{q(\theta \ |\theta')}{q(\theta' \ |\theta)}\right\}$ 3.

Algoritmo Metrópolis-Hastings univariado

```r
sample_mh <- function(n, d_objetivo, r_propuesta = NULL, d_propuesta = NULL, p_inicial = NULL

  if (is.null(r_propuesta) | is.null(d_propuesta)) {
    r_propuesta <- function(media) rnorm(n = 1, media, sd = 1)
    d_propuesta <- function(x, media) dnorm(x = x,media, sd = 1)
  }

  stopifnot(n > 0)
  contador <- 0
  muestras <- numeric(n)

  muestras[1] <- p_inicial

  for(i in 2:n) {

    p_actual <- muestras[i-1]
    p_propuesta <- r_propuesta(p_actual)

    q_actual <- d_propuesta(p_actual, p_propuesta)
    q_nuevo <- d_propuesta(p_propuesta, p_actual)

    f_actual <- d_objetivo(p_actual)
    f_nuevo <- d_objetivo(p_propuesta)

    if (f_actual == 0 || q_nuevo == 0) {
      alfa <- 1
    } else {
      alfa <- min(1, (f_nuevo/f_actual)*(q_actual/q_nuevo))
    }

    muestras[i] <- sample(c(p_propuesta, p_actual),
    size = 1, prob = c(alfa, 1-alfa))

    if(muestras[i] != muestras[i-1]) {
      contador <- contador + 1
    }
  }
  return(list(cadena = data.frame(iteracion = 1:n, x = muestras),
       tasa_aceptacion = contador / n))
}
```

Algoritmo Metrópolis-Hastings bivariado

```r
sample_mh_mv <- function(n, d_objetivo, cov_propuesta = diag(2), p_inicial = numeric(2)) {

  if (length(p_inicial) != 2) {
    stop("El valor p_inicial debe ser bidimensional")
  }
  if ( n <= 0 || n %% 1 != 0) {
    stop("El tamaño de muestra n debe ser entero y mayor que 0")
  }
  if (any((dim(cov_propuesta) != c(2,2)))) {
    stop("La matriz de covariancia debe ser de 2x2")
  }

  contador <- 0

  r_propuesta <-  function(media) rmvnorm(n = 1,mean = media,sigma = cov_propuesta)
  d_propuesta <- function(x, media) dmvnorm(x = x,mean = media,sigma = cov_propuesta)

  muestras <- matrix(0,nrow = n,ncol = length(p_inicial))
  muestras[1, ] <- p_inicial

  for(i in 2:n) {
    p_actual <- muestras[i-1,]
    p_propuesta <- r_propuesta(p_actual)

    q_actual <- d_propuesta(p_actual, p_propuesta)
    q_nuevo <- d_propuesta(p_propuesta, p_actual)

    f_actual <- d_objetivo(p_actual)
    f_nuevo <- d_objetivo(p_propuesta)


    if (f_actual == 0 || q_nuevo == 0) {
      alfa <- 1
    } else {

      alfa <- min(1, (f_nuevo/f_actual)*(q_actual/q_nuevo))
    }

    aceptar <- rbinom(1,1,alfa)

    if (aceptar) {
      muestras[i,] <- p_propuesta
    }else{
      muestras[i,] <- p_actual
    }

    if(!any(muestras[i,] != muestras[i-1,])) {
      contador <- contador + 1
    }
  }
  salida <- data.frame(iteracion = 1:n, x = muestras)
  colnames(salida) <- c("iteracion", paste0("dim_",1:length(p_inicial)))
```

Discusiones

Distribución de Kumaraswamy

La distribución de Kumaraswamy es una distribución de probabilidad continua que se utiliza para modelar variables aleatorias con soporte en el intervalo $(0, 1)$, cuya función de densidad es:
$f(x|a, b) = abx^{a-1}(1 - x^a)^{b-1},\ con\ a, b > 0$

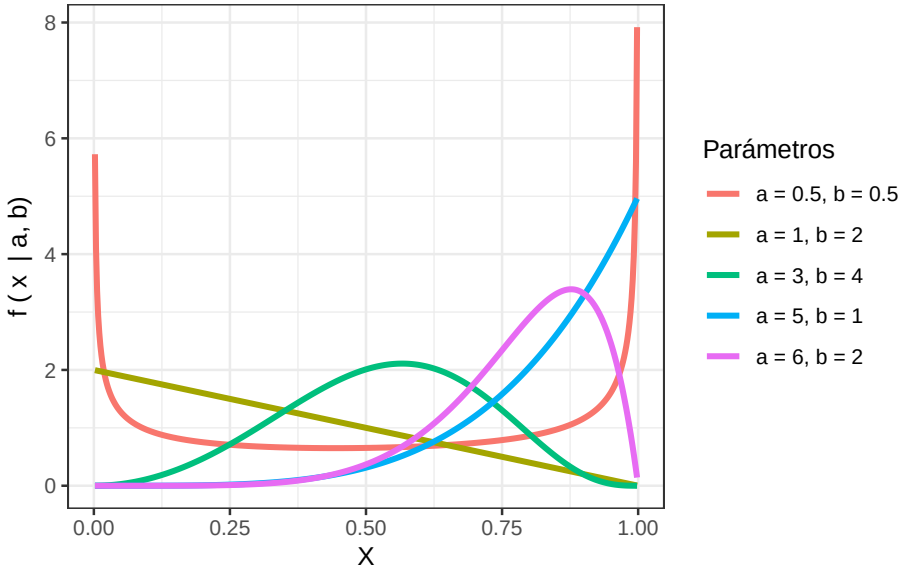

Figura 1: Función de densidad de la distribución de Kumaraswamy

Esta distribución puede ser utilizada en el ámbito de la estadística bayesiana a la hora de definir un prior para un parámetro con campo de variación en el intervalo (0, 1).

Por lo general la distribución elegida para estas situaciones suele ser la beta ya que presenta ventajas como ser una distribución conjugada de la binomial, lo cual puede facilitar mucho algunos cálculos. El problema es que la densidad de esta depende de la función gamma, la cual es una integral, y en algunas situaciones se puede complicar su cálculo.

La distribución de Kumaraswamy se comporta de manera muy similar a la beta, sin tener el problema de la dificultad del cálculo de la integral.

Metropolis-Hastingsen una dimensión

A continuación utilizaremos el algoritmo de Metropolis-Hastingspara generar 5000 muestras de la distribución de *Kumaraswamy* con parámetros $a = 6$ y $b = 2$, utilizando como distribución propuesta una $Beta(\mu, \kappa)$, donde μ representa la media de la distribución y κ el grado de la concentración de la distribución. Esto lo haremos para 3 valores distintos de κ.

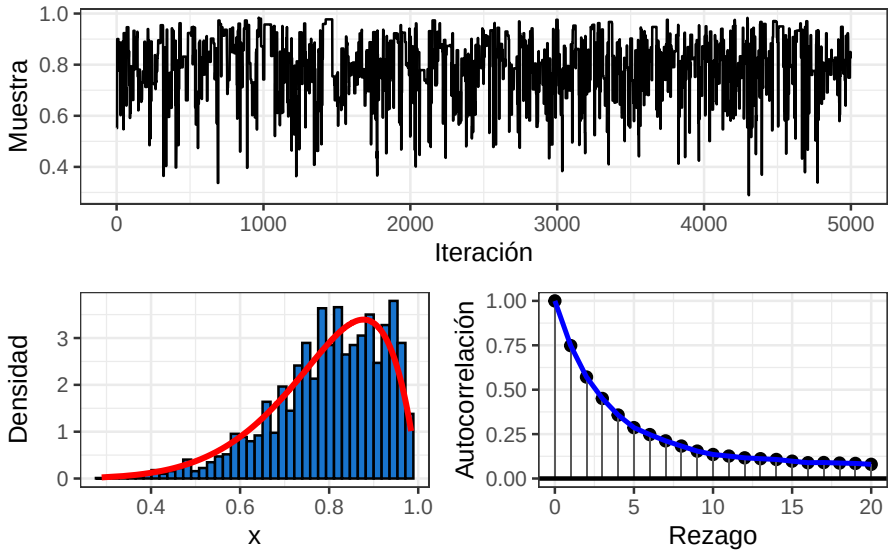

Figura 2: Muestreo por Metropolis-Hastings con $\kappa = 1$

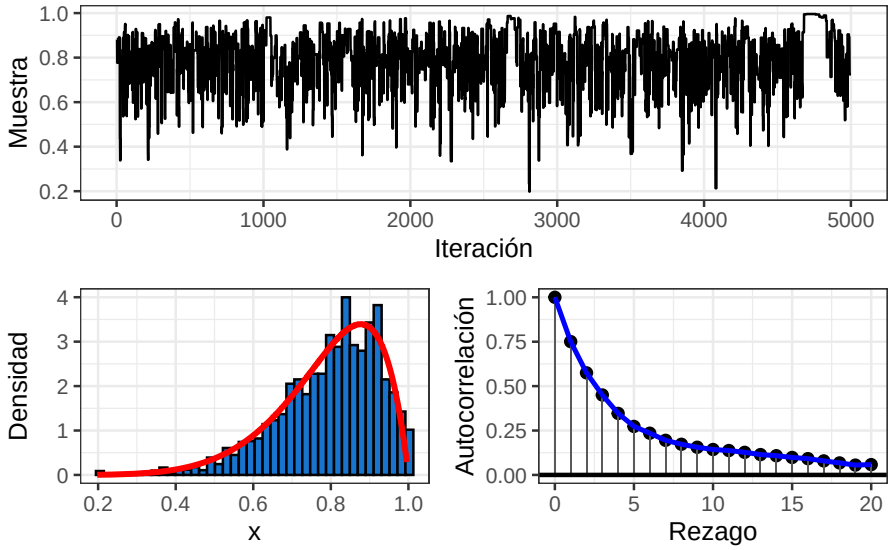

Figura 3: Muestreo por Metropolis-Hastings con $\kappa = 2$

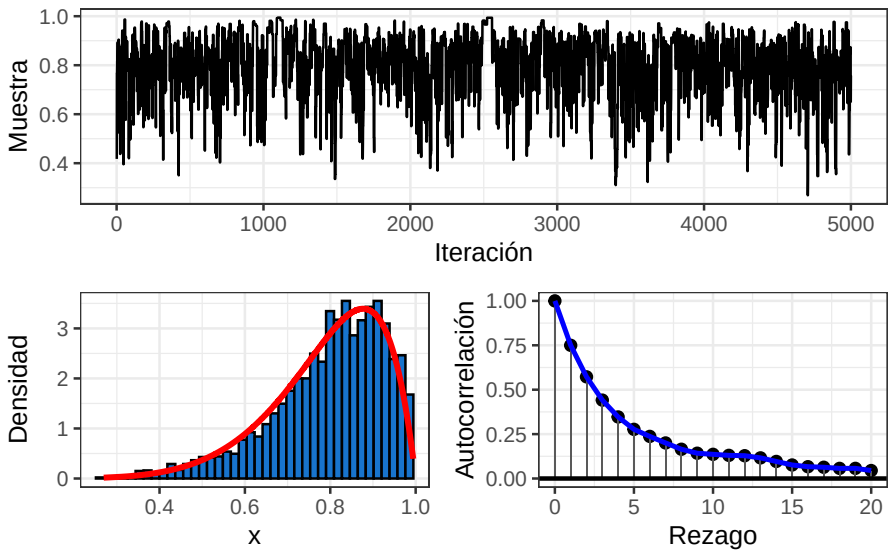

Figura 4: Muestreo por Metropolis-Hastings con $\kappa = 5$

Teniendo en cuenta que estas muestras son dependientes resulta de interés conocer a cuantas muestras independientes equivalen, dado que cuanto más se comporta una cadena de Markov dependiente como una muestra independiente, menor es el error en la aproximación posterior resultante. Para ello se calcula el número efectivo de muestras (n_{eff}) para cada cadena.

Tabla 1: Número efectivo de muestras para cadenas de 5000 muestras y distintos valores de κ

Cadena	κ	n_{eff}
Cadena 1	1	485
Cadena 2	2	358
Cadena 3	5	503

Con esto concluimos que el valor de κ que arroja una muestra relacionada que equivale a una independiente de mayor tamaño es 5. Sin embargo esta cantidad de muestras independientes es muy pobre teniendo en cuenta que el tamaño de la cadena es de 5000 muestras.

Resulta de interés ver como afecta la elección del parámetro κ al utilizar Metropolis-Hastingspara obtener muestras de la distribución de Kumaraswamy usando una distribución *Beta* como propuesta. Es por esto que se decide obtener la media y ciertos cuantiles sobre las muestras obtenidas y sobre la función Logit de estas.

Tabla 2: Media y quantiles estimadas para las funciones X y $Logit(X)$

$f(x)$	κ	$\widehat{E(x)}$	$\widehat{q_{0.05}}$	$\widehat{q_{0.95}}$
X	1	0.802	0.571	0.959
	2	0.797	0.545	0.971
	5	0.799	0.534	0.969
$Logit(X)$	1	1.615	0.287	3.165
	2	1.636	0.182	3.528
	5	1.632	0.137	3.432

Las estadísticas reales de la distribución de Kumaraswamy son:

Tabla 3: Media y quantiles para las funciones X y $Logit(X)$

$f(x)$	$E(x)$	$q_{0.05}$	$q_{0.95}$
X	0.791	0.542	0.959
$Logit(X)$	1.547	0.168	3.145

Si bien las diferencias en dispersión de las muestras no son muy perceptibles a simple vista, gracias a la funcióm logit podemos percibirlas con mayor facilidad. Es así que se puede concluir que usando una distribución *Beta* con un parámetro $\kappa = 2$ de concentración, es que se obtienen las estimaciones más cercanas a la distribución de Kumaraswamy con parámetros $a = 6$ y $b = 2$.

Metropolis-Hastings en dos dimensiones

El algoritmo de Metropolis-Hastings se aprecia más cuando se obtienen muestras de distribuciones en más de una dimensión, incluso cuando no se conoce la constante de normalización, sin embargo tiene limitaciones para cierta funciones, las cuales se verán a continuación.

Para testear el algoritmo de Metropolis-Hastings bivariado creado, se decide obtener muestras de una distribución Normal bivariada con vector de media μ^* y matriz de covarianza Σ^* con el algoritmo ya mencionado.

Donde $\mu^* = \begin{bmatrix} 0.4 \\ 0.75 \end{bmatrix}$ y $\Sigma^* = \begin{bmatrix} 1.35 & 0.4 \\ 0.4 & 2.4 \end{bmatrix}$.

Se plantean 3 matrices de covarianzas distintas como propuestas para observar con cual de estas se obtienen muestras mas aproximadas a la verdadera distribución:

1. $\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$ Se puede ver en Figura 5, Figura 6 y Figura 7

2. $\Sigma_2 = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.25 \end{bmatrix}$ Se puede ver en Figura 8, Figura 9 y Figura 10

3. $\Sigma_3 = \begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix}$ Se puede ver en Figura 11, Figura 12 y Figura 13

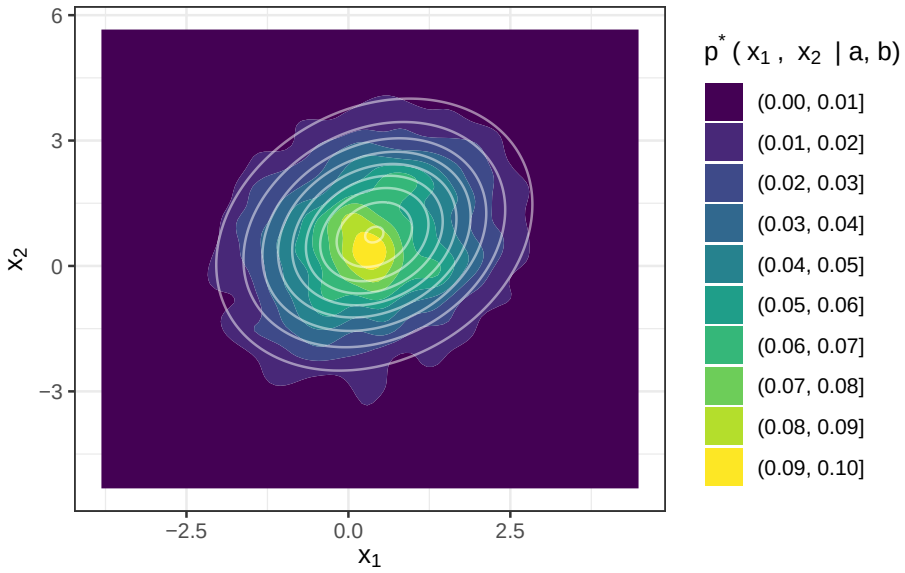

Figura 5: Distribución de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_1 como propuesta

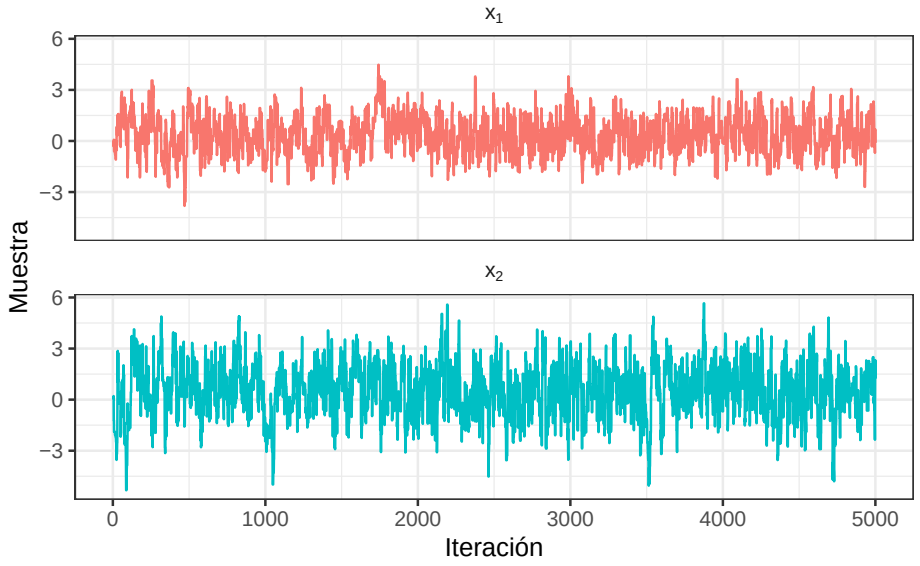

Figura 6: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_1 como propuesta

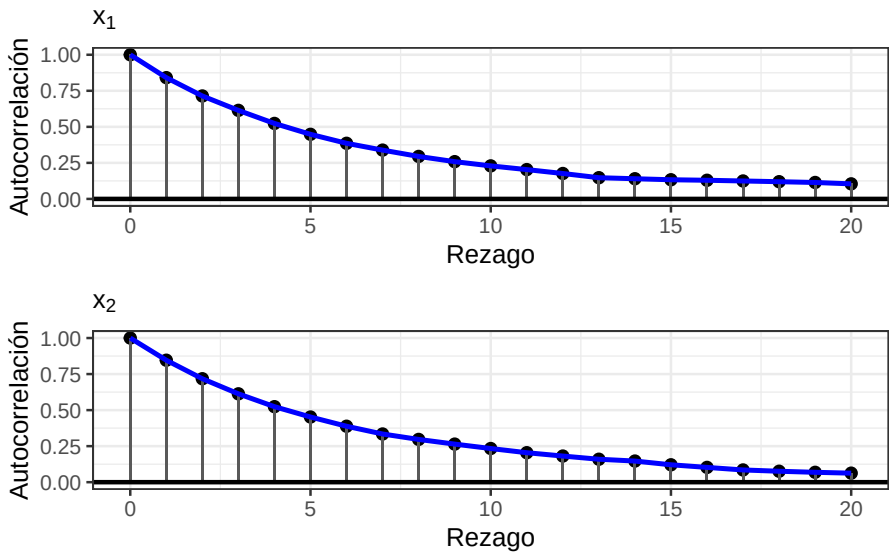

Figura 7: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_1 como propuesta

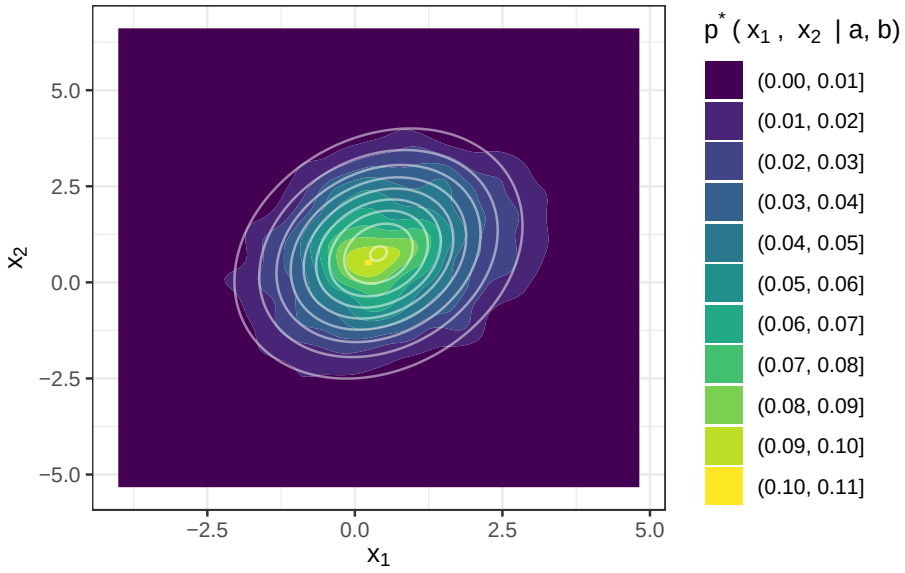

Figura 8: Distribución de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_2 como propuesta

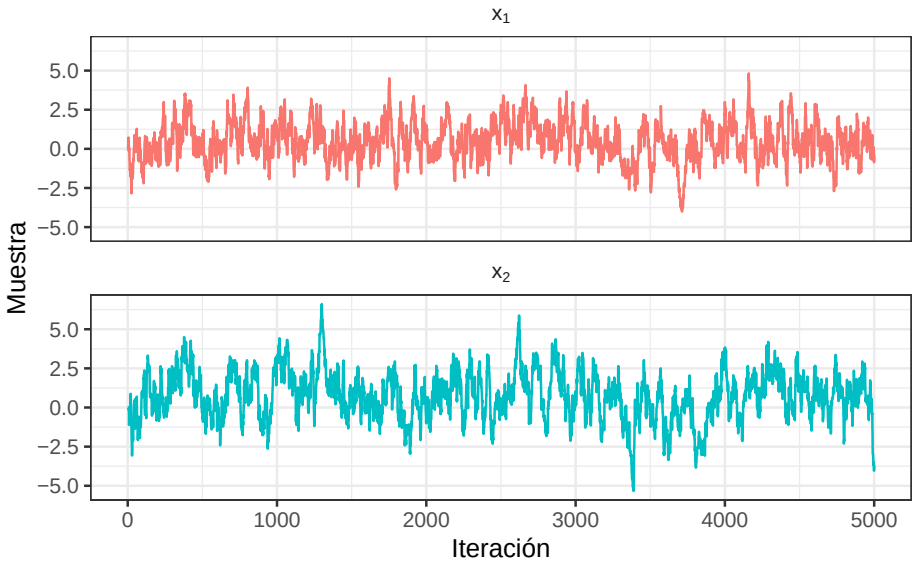

Figura 9: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_2 como propuesta

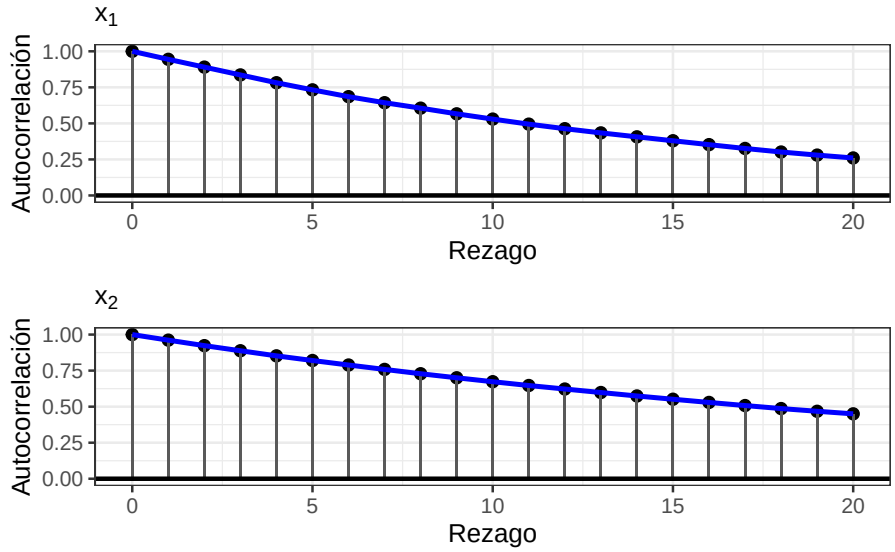

Figura 10: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_2 como propuesta

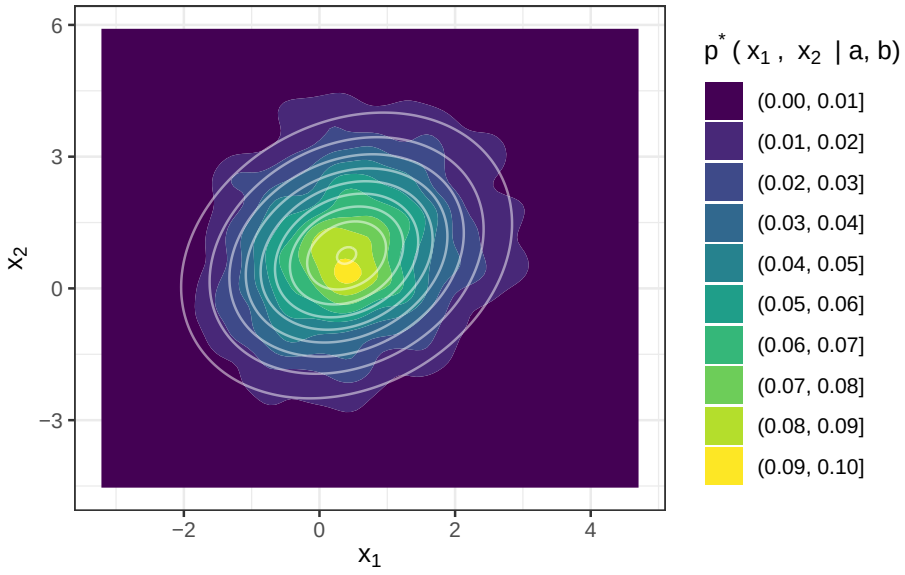

Figura 11: Distribución de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_3 como propuesta

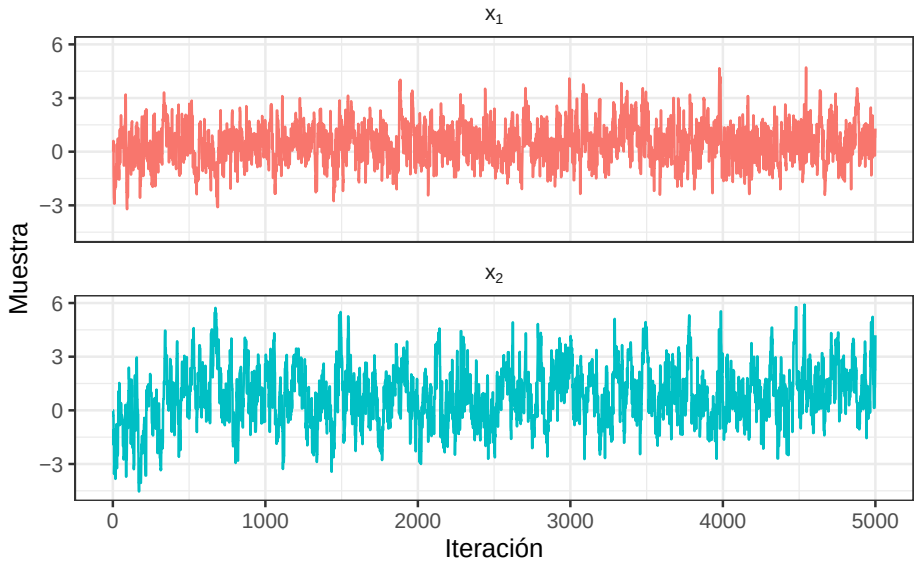

Figura 12: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_3 como propuesta

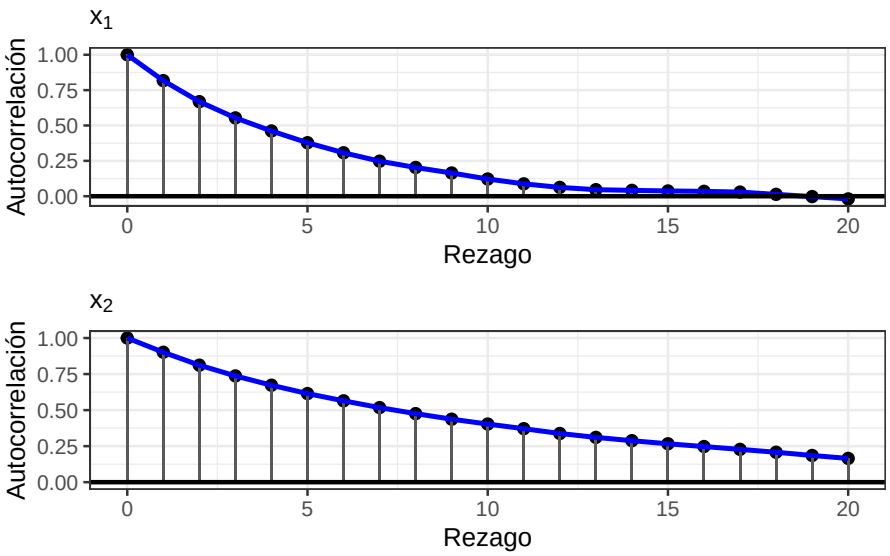

Figura 13: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de una $\mathcal{N}_2(\mu^*, \Sigma^*)$ con Σ_1 como propuesta

Tabla 4: Número efectivo de muestras para cadenas de 5000 muestras y distintos valores de Σ

Cadena	x_1	x_2
Cadena 1	340	377
Cadena 2	171	85
Cadena 3	523	235

Considerando los tamaños de muestras efectivos en cada dimensión y los gráficos de autocorrelación observados, se puede concluir que con la matriz de covarianzas Σ_1 se obtienen muestras aparentemente mejores de la distribución objetivo.

Para ver la bondad de la muestra generada por Metropolis-Hastings, se calculan ciertas probabilidades y luego se comparan con otros métodos de cálculo de probabilidades, como "Función de distribución" y "Método de MonteCarlo".

Tabla 5: Cálculo de probablidades con distintos métodos

Método	$P(X_1 > 1, X_2 < 0)$	$P(X_1 > 1, X_2 > 2)$	$P(X_1 > 0.4, X_2 > 0.75)$
M-H	0.0858	0.0654	0.2332
Función de distribución	0.0683	0.0870	0.2857
MC	0.0754	0.0910	0.2808

Podemos concluir que las estimaciones de las probabilidades calculadas son bastante acertadas, debido a su similitud con los otros métodos.

Función de Rosenbrock

Es una función matemática, popularmente conocida como *La banana de Rosenbrock* utilizada frecuentemente como prueba de algoritmos de optimización numérica.

$$f(x, y) = (a - x)^2 + b(y - x^2)^2$$

En el campo de la Estadística Bayesiana, es muy conocida dado que la densidad del posterior muchas veces toma una forma que se asemeja a la función de *Rosenbrock*.

Un ejemplo de este fenómeno es la función de densidad que viene dada por:
$p^*(x_1, x_2|a, b) = exp\{-[(a - x_1)^2 + b(x_2 - x_1^2)^2]\},\ a, b \in \mathbb{R}$

Para poner a prueba el algoritmo de Metropolis-Hastings bivariado en situaciones complejas, se decide obtener 5000 muestras de la función anteriormente mencionada, con parámetros $a = 0.5$ y $b = 5$. Se utilizan 3 distintas matrices de covarianzas propuestas, las cuáles son:

1. $\Sigma_1 = \begin{bmatrix} 0.06 & 0 \\ 0 & 0.07 \end{bmatrix}$ Se puede ver su densidad en Figura 14, el recorrido de la cadena en Figura 15 y la autocorrelación en fig-banana-autocor1. Con esta matriz la probabilidad de aceptación es 0.411.

2. $\Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ Se puede ver su densidad en Figura 17, el recorrido de la cadena en Figura 18 y la autocorrelación en fig-banana-autocor2. Con esta matriz la probabilidad de aceptación es 0.8092.

3. $\Sigma_3 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$ Se puede ver su densidad en Figura 20, el recorrido de la cadena en Figura 21 y la autocorrelación en fig-banana-autocor3. Con esta matriz la probabilidad de aceptación es 0.7114.

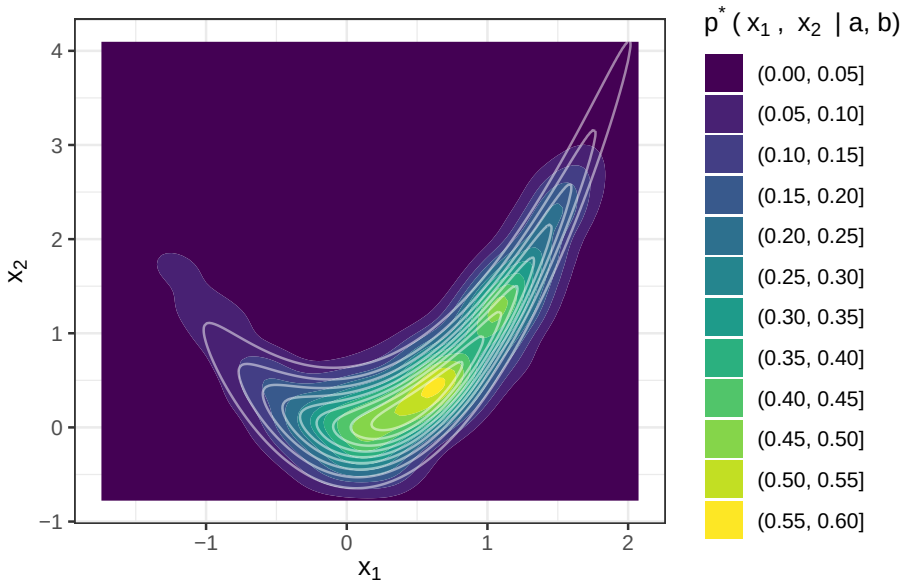

Figura 14: Distribución de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_1 como propuesta

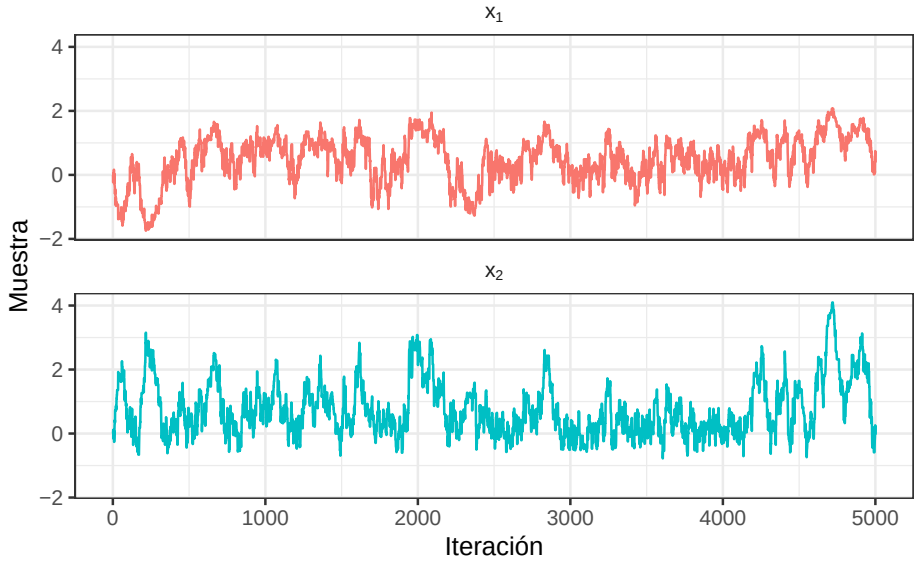

Figura 15: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_1 como propuesta

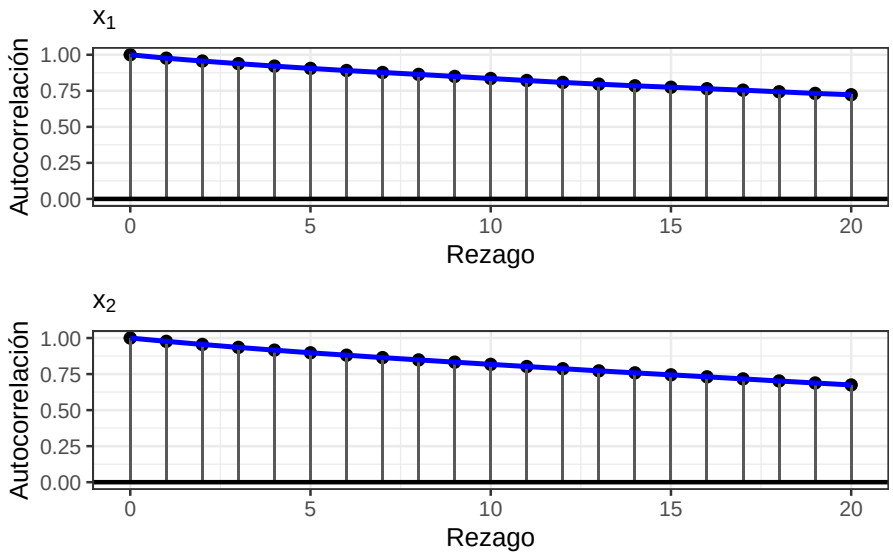

Figura 16: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_1 como propuesta

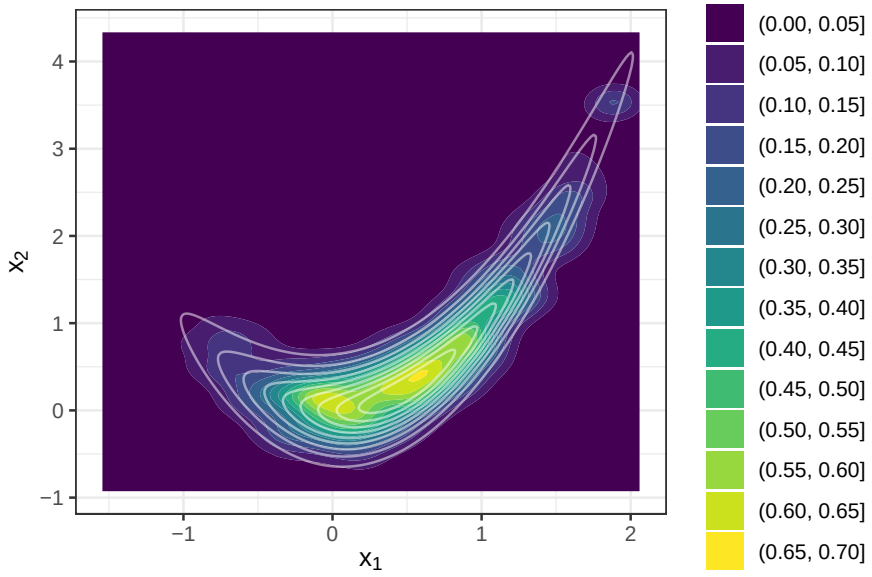

Figura 17: Distribución de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_2 como propuesta

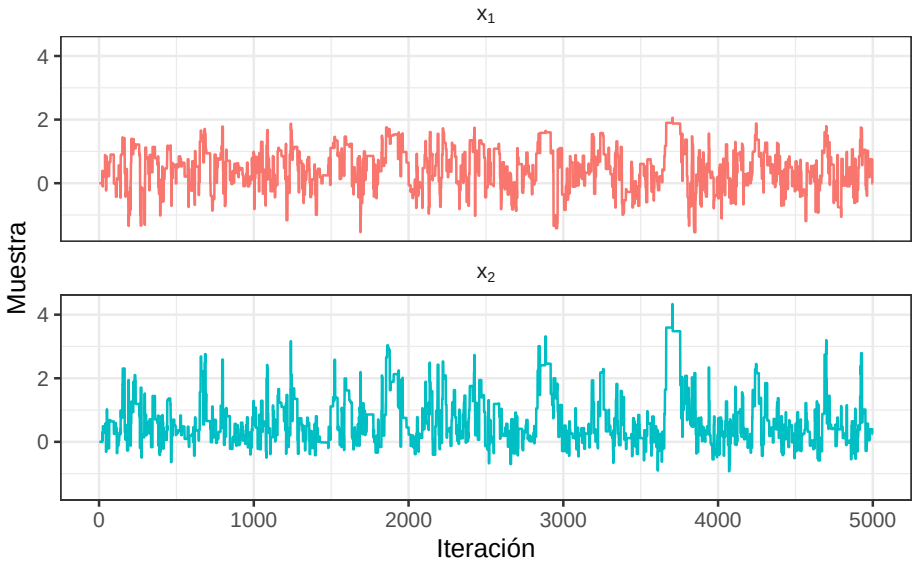

Figura 18: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_2 como propuesta

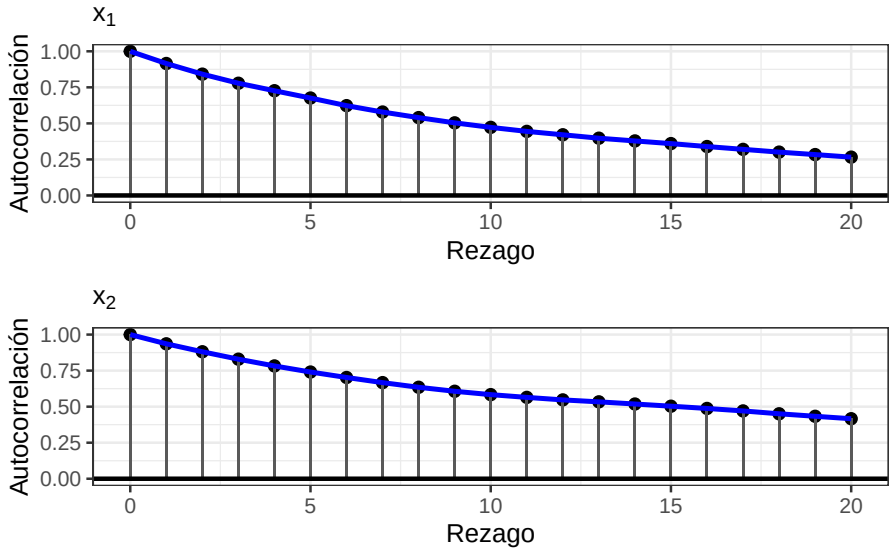

Figura 19: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_1 como propuesta

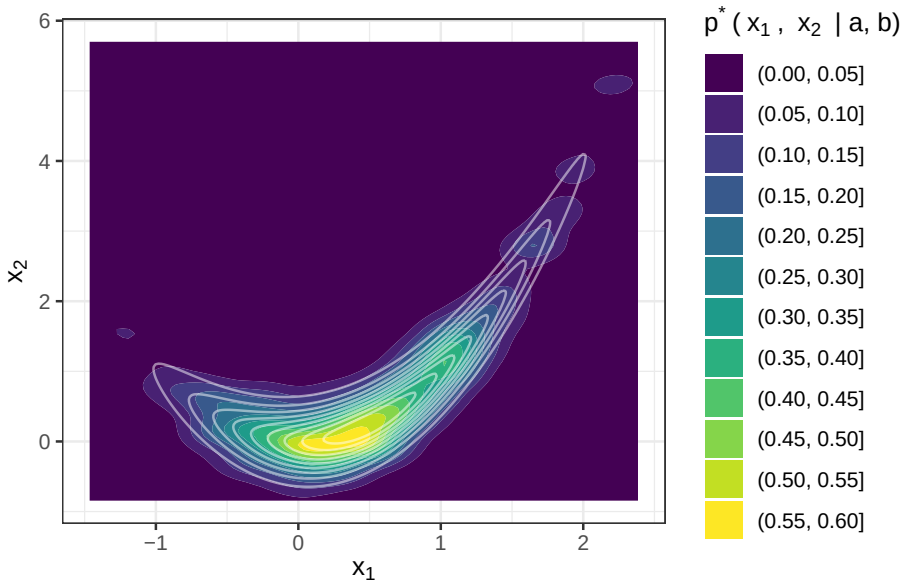

Figura 20: Distribución de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_3 como propuesta

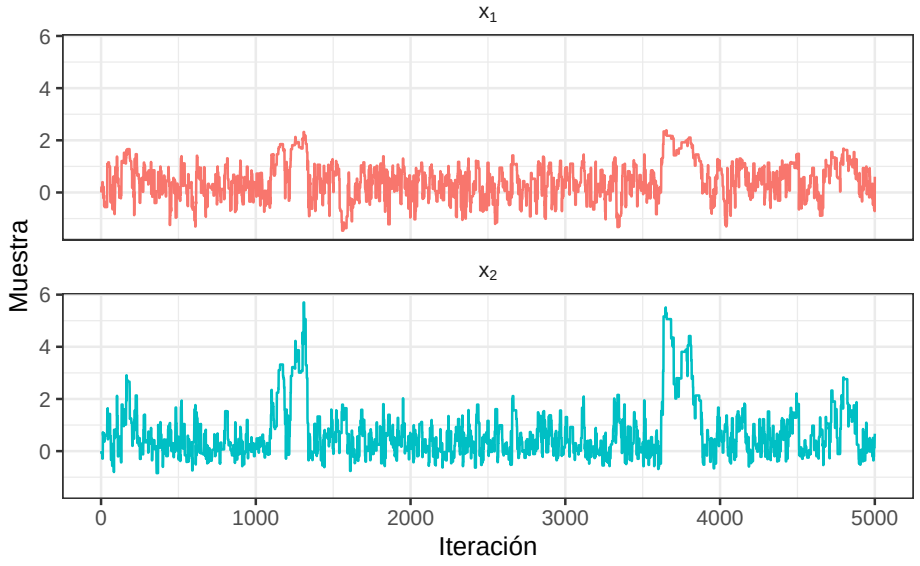

Figura 21: Cadena de Markov de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_3 como propuesta

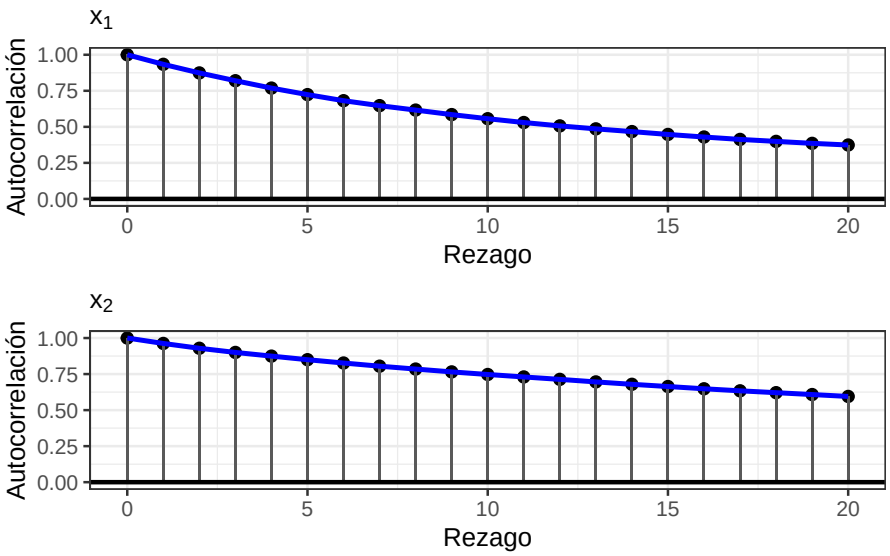

Figura 22: Autocorrelación de las muestras obtenidas por Metropolis-Hastings de p^* con Σ_1 como propuesta

Tabla 6: Número efectivo de muestras para cadenas de 5000 muestras y distintos valores de Σ

Cadena	x_1	x_2
Cadena 1	30	46
Cadena 2	149	94
Cadena 3	67	43

Basándonos en los tamaños de las muestras efectivas en cada dimensión y en los gráficos de autocorrelación observados, parece que la matriz de covarianzas Σ_2 genera muestras que se ajustan mejor a la distribución objetivo.

Para ver la bondad de la muestra generada por Metropolis-Hastings, se calculan ciertas probabilidades y luego se comparan con otros métodos de cálculo de probabilidades, como "Función de distribución" y "Método de MonteCarlo".

Para ver que tan bien funciona Metropolis-Hastings en esta situación compleja, interesa calcular ciertas probabilidades y compararlas con el método de "Integración Manual".

Tabla 7: Cálculo de probablidades con distintos métodos

Método	$P(0 < X_1 < 1, 0 < X_2 < 1)$	$P(-1 < X_1 < 0, 0 < X_2 < 1)$	$P(1 < X_1 < 2, 2 < X_2 < 3)$
M-H	0.3954	0.1602	0.0588
Integración numérica	0.5137	0.2022	0.0838

Se puede observar que el algoritmo en esta situación falla bastante en la generación de las muestras, dado que la estimación de las probabilidades son muy diferentes a las calculadas por un método casi exacto.

Conclusiones

(Redactar las conclusiones en un futuro no muy lejano …)