Estadística Bayesiana: Clase 15

Juan Carlos Correa

21 de abril de 2021

MCMC: Monte Carlo por Cadenas de Markov

Algoritmo Metropolis-Hastings

- El muestreo de importancia y el muestreo de rechazo trabajan bien si la densidad propuesta $q(\theta)$ es similar a $p(\theta)$.
- En problemas complejos puede ser difícil crear una única $q(\theta)$ que tenga esta propiedad.
- El algoritmo Metropolis utiliza una densidad propuesta q que depende del estado actual de $\theta^{(t)}$.
- La densidad $q\left(\theta'|\theta^{(t)}\right)$ puede ser tan simple como una normal localizada en $\theta^{(t)}$ y no es necesario que se parezca a $p(\theta)$.

El algoritmo se resume así:

1. Comience en cualquier lugar, y digamos que estamos en $\theta^{(t)} = \theta$.

2. Genere θ^* de $q(\theta^*|\theta)$. θ^* es llamado un *punto candidato* y q es llamada una *distribución propuesta*.

3. Calcule

$$\alpha(\theta, \theta^*) = \min\left\{1, \frac{\xi(\theta^*)\, q(\theta|\theta^*)}{\xi(\theta)\, q(\theta^*|\theta)}\right\}$$

4. Acepte $\theta^{(t+1)} = \theta^*$ con probabilidad $\alpha(\theta, \theta^*)$.

5. En otro caso $\theta^{(t+1)} = \theta$

Note que la densidad objetivo ξ solo entra en al proceso a través del cociente $\frac{\xi(\theta^*)}{\xi(\theta)}$ y por lo tanto no hay necesidad de conocer la constante de normalización para implementar el algoritmo.

Casos especiales:

1. $q(\theta|\theta^*) = q(\theta^*|\theta)$: Algoritmo Metropolis.
2. $q(\theta|\theta^*) = g(\theta^*)$: Muestreador independiente.
3. $q(\theta|\theta^*) = \prod_{i=1}^{k} \xi(\theta_i|\theta^* < i, \theta_{>i}) \Rightarrow \alpha(\theta, \theta^*) = 1$: Muestreador de Gibbs.

El Algoritmo Metropolis

Aquí la distribución propuesta es simétrica, esto es,

$$q\left(\theta|\theta^*\right) = q\left(\theta^*|\theta\right),$$

como en el caso de una Normal centrada en el punto actual, entonces el factor

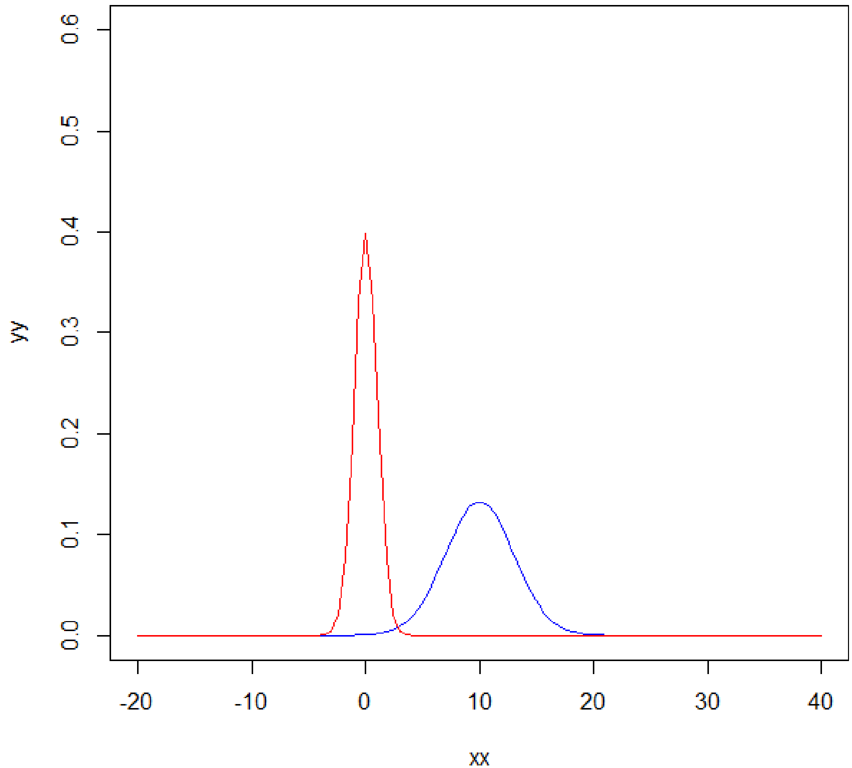
$$\frac{q\left(\theta|\theta^*\right)}{q\left(\theta^*|\theta\right)} = 1,$$

y el algoritmo Metropolis simplemente se limita a comparar el valor de la densidad objetivo en los dos puntos.

Visualización de Metropolis-Hasting

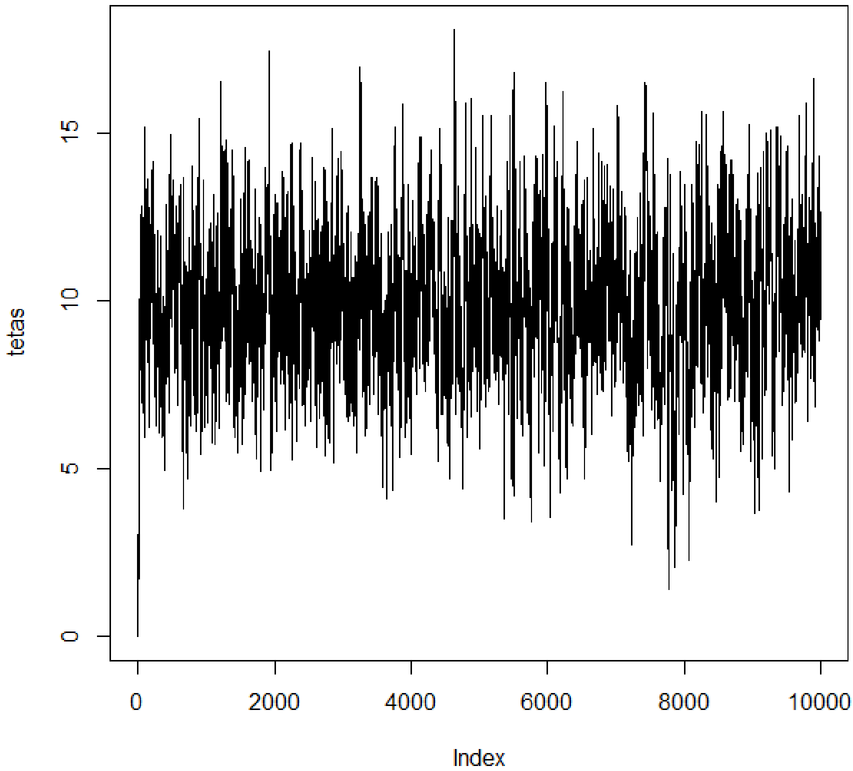
```
set.seed(12345)

# Distribución objetivo N(10,9)
# Distribución que propone N(theta,1)

#valor inicial=0
chequee<-function(teta.nuevo,teta.viejo){
cociente<-dnorm(teta.nuevo,10,sd=3,log=T)
 -dnorm(teta.viejo,10,sd=3,log=T)
 +dnorm(teta.viejo,teta.nuevo,sd=1,log=T)
 -dnorm(teta.nuevo,teta.viejo,sd=1,log=T)
cociente<-exp(cociente)
proba<-min(c(1,cociente))
acepta<-0
if(runif(1)<proba) acepta<-1
return(acepta)
}
```

```
tetas<-teta.viejo<-0

xx<-seq(-20,40,length=100)
yy<-dnorm(xx,mean=10,sd=3)
plot(xx,yy,type='l',col='blue',ylim=c(0,0.6),xlim=c(-20,40))
points(xx,dnorm(xx,mean=teta.viejo,sd=1),type='l',col='red')
```

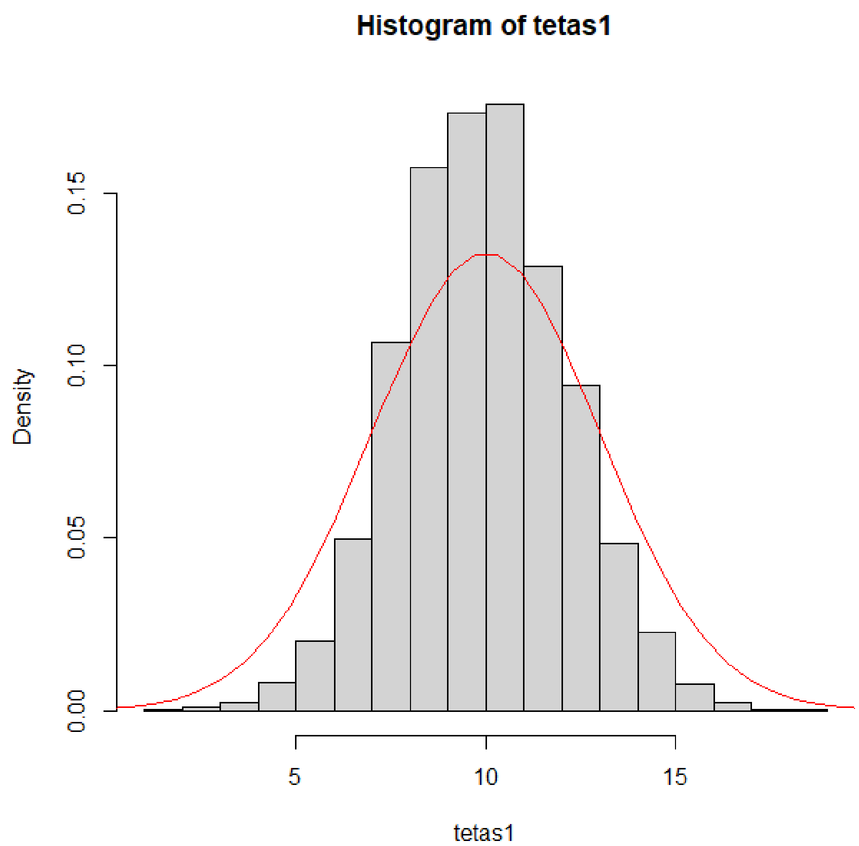

0.0
0.1
0.2
0.3
0.4
0.5
0.6
yy
-20
-10
0
10
20
30
40
xx

```
for(i in 1:10000){
bandera<-0
while(bandera==0){
teta.nuevo<-rnorm(1,teta.viejo,sd=1)
#print(teta.nuevo)
resi<-chequee(teta.nuevo,teta.viejo)
if(resi==1)bandera<-1
}#fin while
teta.viejo<-teta.nuevo
tetas<-c(tetas,teta.nuevo)
plot(xx,yy,type='l',col='blue',ylim=c(0,0.6),xlim=c(-20,40))
points(xx,dnorm(xx,mean=teta.viejo,sd=1),type='l',col='red')

}
```

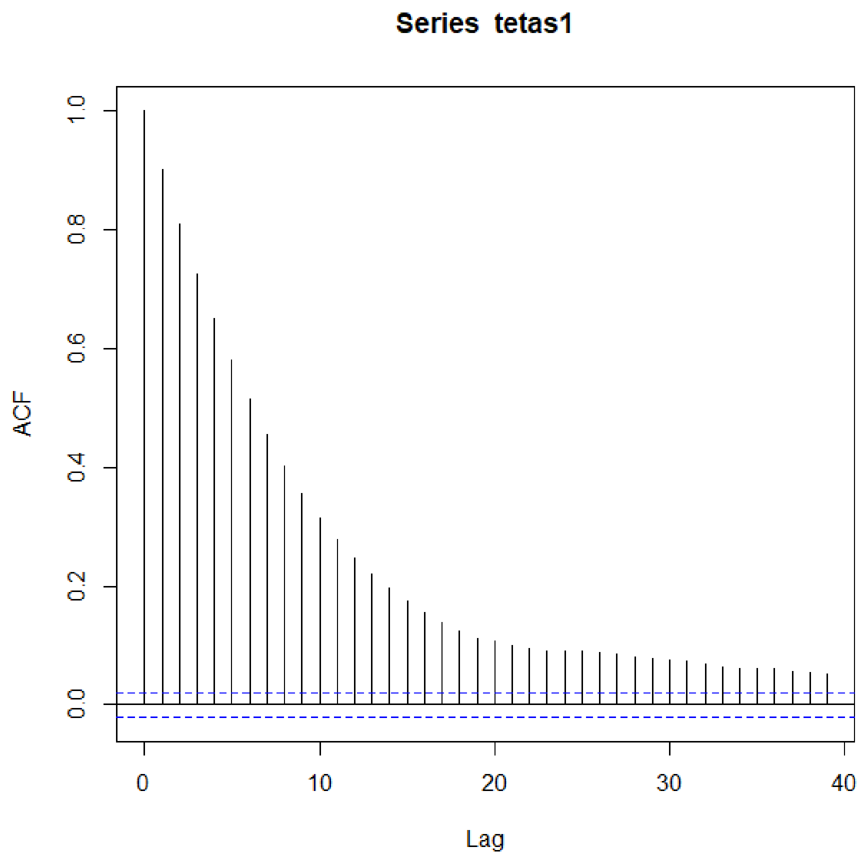
```
plot(tetas,type='l')
```


tetas
0
5
10
15
0
2000
4000
6000
8000
10000
Index

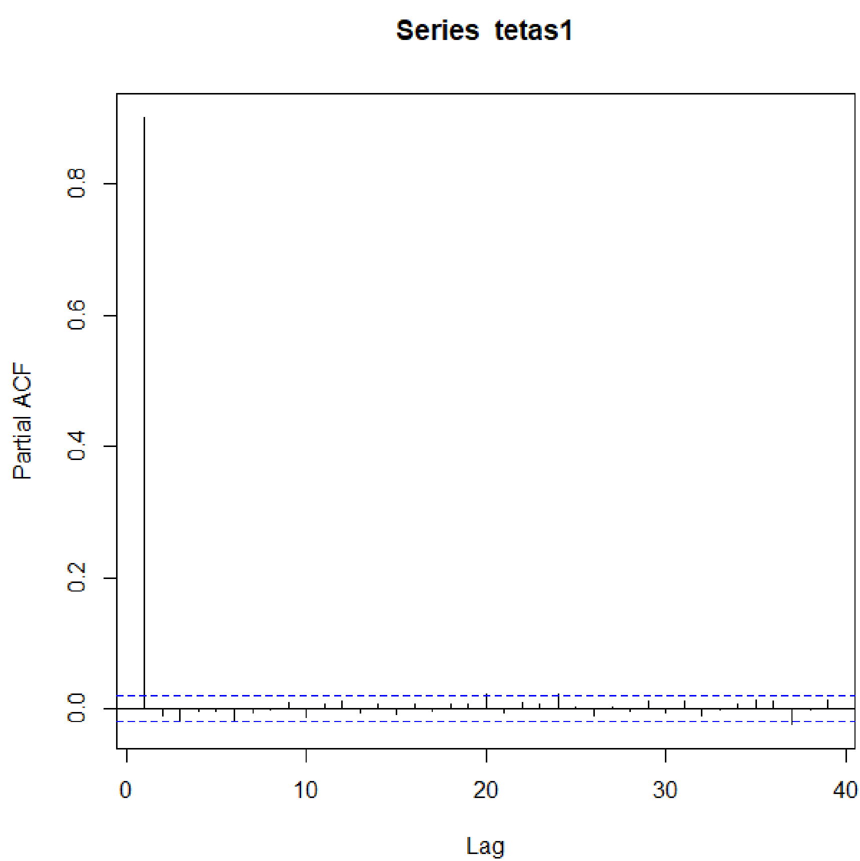
```
tetas1<-tetas[-(1:200)]
hist(tetas1,freq=F)
points(xx,dnorm(xx,mean=10,sd=3),type='l',col='red')

 mean(tetas1)
[1] 9.924083
 var(tetas1)
[1] 4.74014
```


Histogram of tetas1
Density
0.00
0.05
0.10
0.15
5
10
15
tetas1

```
>(autocor<- acf(tetas1))
Autocorrelations of series tetas1, by lag
0 1 2 3 4 5 6 7 8 9 10 11 12 1.000 0.901 0.811 0.726 0.649 0.580
0.514 0.455 0.402 0.357 0.314 0.278 0.248 13 14 15 16 17 18
19 20 21 22 23 24 25 0.220 0.196 0.174 0.155 0.138 0.124 0.113
0.107 0.100 0.095 0.091 0.091 0.091 26 27 28 29 30 31 32 33
34 35 36 37 38 0.088 0.085 0.081 0.079 0.075 0.073 0.069 0.065
0.062 0.061 0.062 0.058 0.053 39 0.052 >
```


Series tetas1
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
Lag

```
 pacf(tetas1)
ro1<-autocor$acf[2]
dif.max<-0.1 # diferencia entre la media real y la estimada máxima p
(n<- 9*(1+ro1)*qnorm(1-0.05/2)^2/(dif.max^2*(1-ro1)) )

[1] 66698.83
```


Series tetas1
Partial ACF
0.0
0.2
0.4
0.6
0.8
0
10
20
30
40
Lag

Muestreador Gibbs

Para obtener una muestra de la distribución conjunta $p(\theta_1, \cdots, \theta_d)$ el Muestreador Gibbs itera sobre este ciclo:

- Muestree $\theta_1^{(i+1)}$ de $p\left(\theta_1 \middle| \theta_2^{(i), \cdots, \theta_d^{(i)}}\right)$
- Muestree $\theta_2^{(i+1)}$ de $p\left(\theta_2 \middle| \theta_1^{(i+1)}, \theta_3^{(i)} \cdots, \theta_d^{(i)}\right)$
- $\vdots$
- Muestree $\theta_d^{(i+1)}$ de $p\left(\theta_d \middle| \theta_1^{(i+1), \cdots, \theta_{d-1}^{(i+1)}}\right)$

Ejemplo: La Distribución ZIP

Asumamos que X es una variable aleatoria discreta con soporte en los enteros nonegativos (una variable de conteo). Un problema que ocurre con cierta frecuencia en la práctica es que $X = 0$ se observa con una frecuencia significativamente mayor (o menor) que la predicha por el modelo asumido. Entonces la variable aleatoria ajustada Y puede ser descrita como

$$\begin{aligned} P\,(Y = 0) &= \omega + (1 - \omega) P\,(X = 0) \\ P\,(Y = j) &= (1 - \omega) P\,(X = j)\,, \quad j = 1, 2, 3, \cdots \end{aligned}$$

Cuando $0 < \omega < 1$ el modelo tiene más ceros. Si $\omega < 0$ el modelo tiene menos ceros.

Un caso de especial importancia es cuando $X \sim Poisson(\lambda)$. La verosimilitud en este caso es

$$\begin{aligned}
L(\omega,\lambda) &= \prod_{i=1}^{n} P\left(Y_i = y_i\right) \\
&= \prod_{i=1}^{n} \left\{ \left(P\left(Y_i = 0\right)\right)^{I(y_i=0)} \left(P\left(Y_i = y_i\right)\right)^{1-I(y_i=0)} \right\} \\
&= \prod_{i=1}^{n} \left\{ \left(\omega + (1-\omega)e^{-\lambda}\right)^{I(y_i=0)} \left((1-\omega)\frac{\lambda^{y_i} e^{-\lambda}}{y_i!}\right)^{1-I(y_i=0)} \right\} \\
&\propto \left(\omega + (1-\omega)e^{-\lambda}\right)^{N_0} (1-\omega)^{n-N_0} \lambda^{\sum_{i=1}^{n} y_i(1-I(y_i=0))} e^{-\lambda(n-N_0)}
\end{aligned}$$

donde

$$N_0 = \sum_{i=1}^{n} I\left(y_i = 0\right)$$

$$
\begin{aligned}
L(\omega,\lambda) \;&\propto\; \left(\omega+(1-\omega)e^{-\lambda}\right)^{N_0} \\
&\times(1-\omega)^{n-N_0}\lambda^{\sum_{i=1}^{n} y_i(1-I(y_i=0))}e^{-\lambda(n-N_0)}
\end{aligned}
$$

Ahora

$$
\sum_{i=1}^{n} y_i\,(1-I(y_i=0)) = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} y_i I(y_i=0) = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} 0\times 1 = \sum_{i=1}^{n} y_i
$$

$$
\begin{aligned}
L(\omega,\lambda) \;&\propto\; \left(\frac{\omega}{(1-\omega)}+e^{-\lambda}\right)^{N_0} \\
&\times(1-\omega)^{n}\lambda^{\sum_{i=1}^{n} y_i}e^{-\lambda(n-N_0)}
\end{aligned}
$$

Así, si asumimos una distribución apriori no informativa para ω y para λ, tenemos

$$\begin{aligned} \xi(\omega,\lambda) &\propto K \\ \xi(\omega,\lambda\,|Datos) &\propto L(\omega,\lambda) \end{aligned}$$

Las distribuciones aposteriori condicionales serán

$$\begin{aligned} \xi(\omega\,|\lambda, Datos) &\propto \left(\frac{\omega}{(1-\omega)}+e^{-\lambda}\right)^{N_0}(1-\omega)^n \\ \xi(\lambda\,|\omega, Datos) &\propto \left(\frac{\omega}{(1-\omega)}+e^{-\lambda}\right)^{N_0}\lambda^{\sum_{i=1}^n y_i}e^{-\lambda(n-N_0)} \end{aligned}$$

- Gupta et al. (1996) hacen referencia a los datos analizados por Leroux y Puterman en 1992 sobre movimientos fetales.
- Estos datos se recogieron en un estudio sobre respiración y movimiento corporal en fetos de ovejas diseñado para examinar los posibles cambios en el patrón de la actividad fetal durante las dos terceras partes del período de gestación.
- El número de movimientos efectuados por el feto fue registrado por ultrasonido. Se analizaron los conteos del número de movimientos en una sucesión particular de 240 intervalos de a 5 segundos.

Número de movimientos	0	1	2	3	4	5	6	7
Frecuencia	182	41	12	2	2	0	0	1

```
# Estimación bayesiana de la distribución ZIP

y<-c(rep(0,182),rep(1,41),rep(2,12),3,3,4,4,7)

# Distribución Apriori
apriori<-function(l,w,alfa1,beta1,alfa2,beta2) dgamma(l,alfa1,
                   scale=beta1)*dbeta(w,alfa2,beta2)

# Verosimilitud
zero.inflada<-function(l,w,y){
 n<-length(y)
 n0<-length(y[y==0])
 res<-(w+(1-w)*exp(-l))^n0*(1-w)^(n-n0)*exp(-(n-
n0)*l)*l^(sum(y))
 res
}

 pi<-seq(0,1,length=100)
 lambda<-seq(0,1.5,length=100)
 res<-outer(lambda,pi,FUN='zero.inflada',y)
 res.apriori<-outer(lambda,pi,FUN='apriori',3,3,2,2)
```

```
par(lty=1)
 contour(lambda,pi,res,lty=3,col='red')
par(new=T)
contour(lambda,pi,res.apriori,lty=4,col='blue')
res.aposte<-res*res.apriori
par(new=T)
contour(lambda,pi,res.aposte,lty=1)
title(xlab='Lambda',ylab='pi')
title(sub='Distribución apriori: Gamma(3,3)xBeta(2,2)')
legend(0,1,legend=c('Verosimilitud','Apriori','Aposteriori'),
          lty=c(3,4,1),col=c('red','blue','black'))
title(main='Distribución ZIP')
```

Página www

Página de Abertura

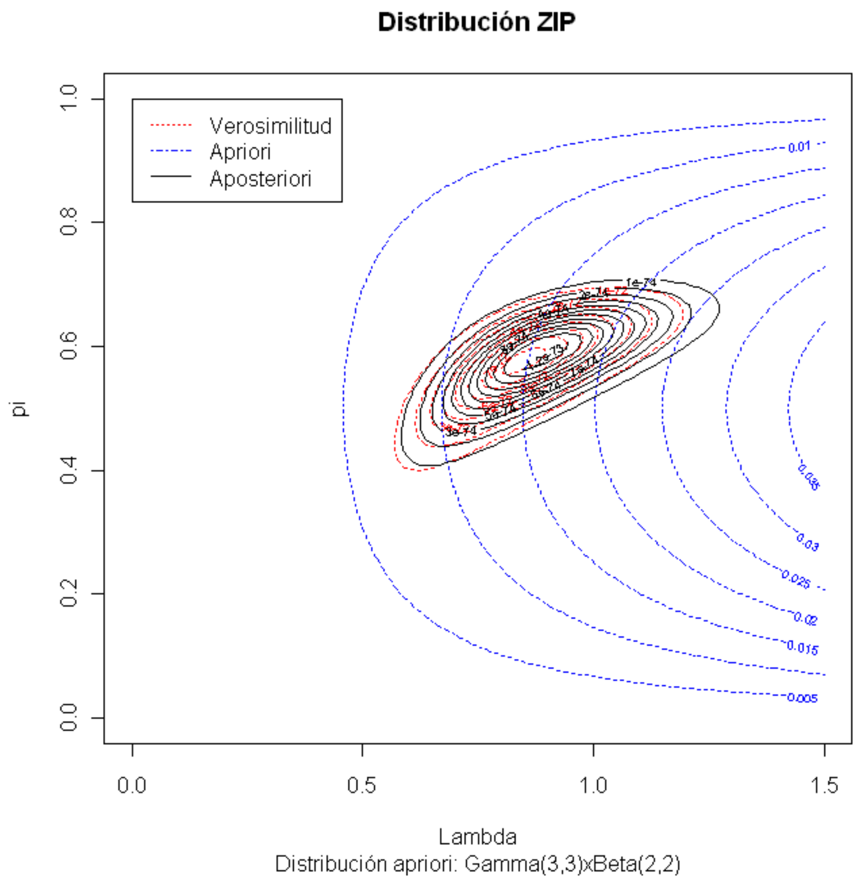
Contenido

Regresar

Full Screen

Cerrar

Abandonar



```
# Algoritmo Metropolis
# Función de Verosimilitud
 L<-function(omega,lambda,y){

 indicador<-ifelse(y==0,1,0)
 prod1<-prod((omega+(1-omega)*exp(-lambda))^indicador)
 prod2<-prod(((1-omega)*exp(-lambda))^(1-indicador)
               *lambda^(y*(1-indicador)))
 productoria<-prod1*prod2
 productoria
}
```

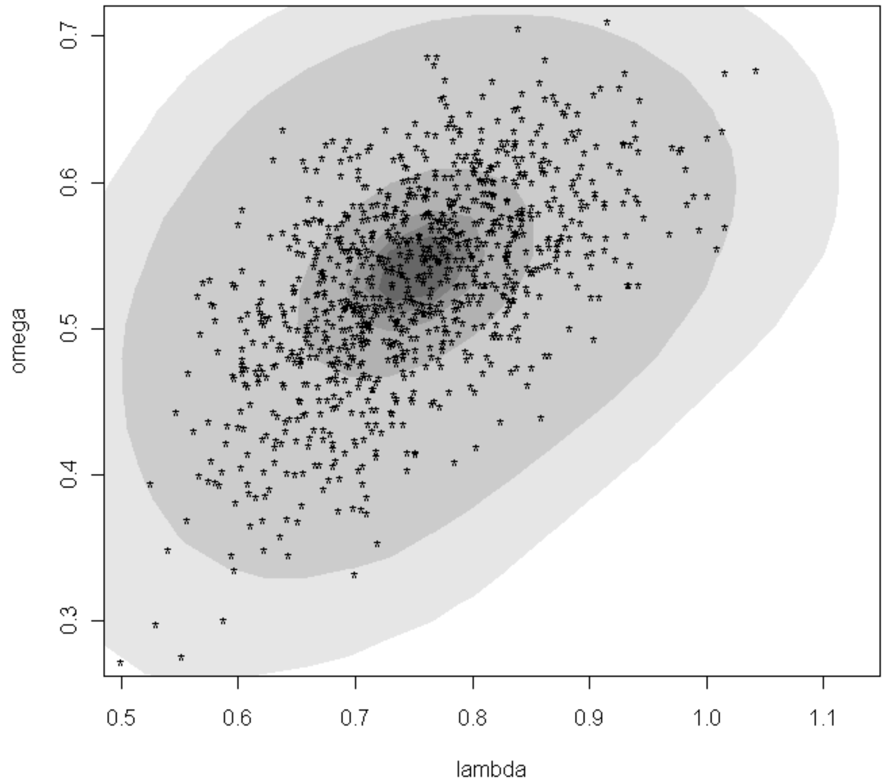
```
# Esta función propone nuevos candidatos
muestreadora<-function(teta.viejo){
omega<-teta.viejo[1]
lambda<-teta.viejo[2]
valor.negativo<-1
# Generamos una normal truncada entre 0 y 1
while(valor.negativo==1){
nuevo1<-rnorm(1,mean=omega)
if(nuevo1>0 & nuevo1<1) valor.negativo<-0
}

valor.negativo<-1
# generamos Una normal truncada mayor que 0
while(valor.negativo==1){
nuevo2<-rnorm(1,mean=lambda)
if(nuevo2>0 ) valor.negativo<-0
}

teta.nuevo<-c(nuevo1,nuevo2)
teta.nuevo
}
```

```
# Esta función decide si se acepta el nuevo candidato
qmuestreadora<-function(nuevo,viejo,y){
omega1<-viejo[1]
omega2<-nuevo[1]
lambda2<-nuevo[2]
lambda1<-viejo[2]
resultado<-(dnorm(omega1)*dnorm(lambda1))/
 (dnorm(omega2)*dnorm(lambda2))
resultado<-resultado*L(omega2,lambda2,y)/
  L(omega1,lambda1,y)
resultado<-min(1,resultado)
resultado
}
```

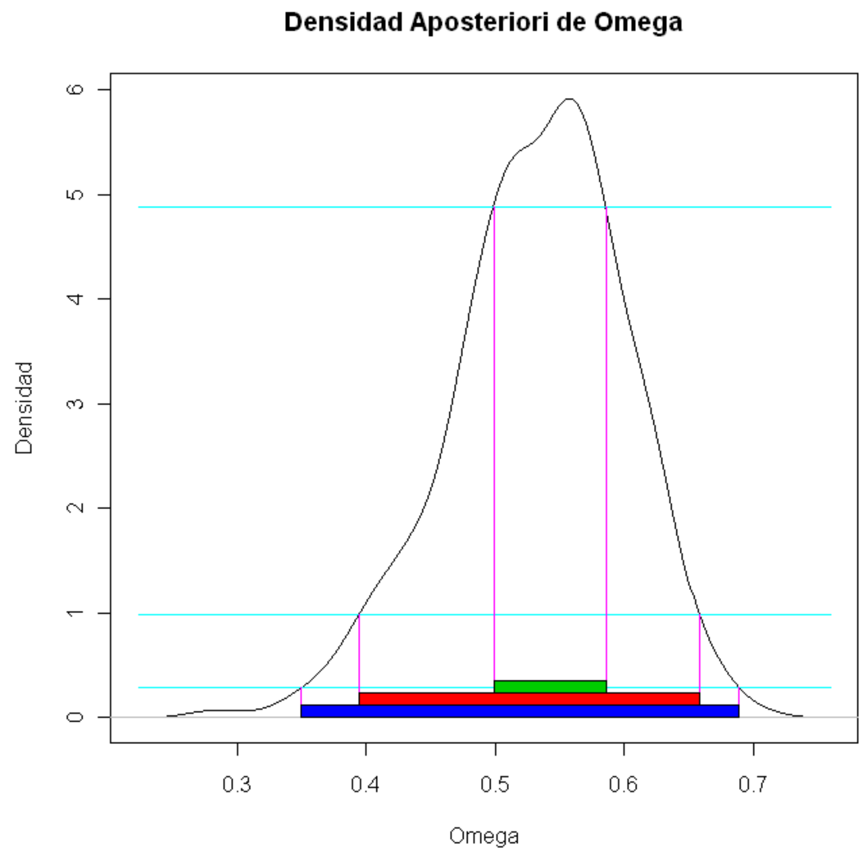
```
#  DATOS

y<-c(rep(0,182),rep(1,41),rep(2,12),3,3,4,4,7)
# VALOR INICIAL
 viejo<-c(0.05,1)
 Nsim<-10000
 matriz.res<-viejo
 for(i in 1:Nsim){
 nuevo<-muestreadora(viejo)

 prob.acept<-qmuestreadora(nuevo,viejo,y)
 u<-runif(1)
 if(u<prob.acept){
   viejo<-nuevo
   matriz.res<-rbind(matriz.res,nuevo)
   }
 }


 dim(matriz.res)
 plot(matriz.res,type='l')
```

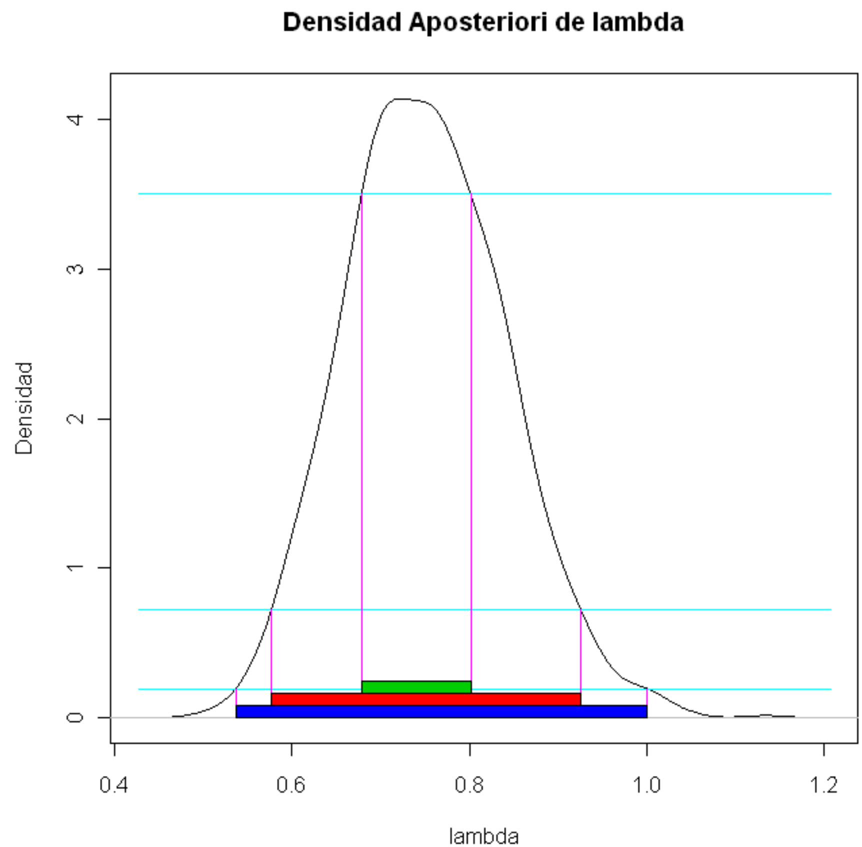
```
# Muestreador de Gibbs con datos del nro. de movimientos fetales
# para encontrar parametros de una distribucion ZIP

muestra.omega <- function(lambda, nro.ceros,n){
rejilla <- seq(0.0001, 0.9999, length=1000)
proba <- (rejilla/(1-rejilla)+exp(-lambda))^nro.ceros*(1-
rejilla)^n
proba<-ifelse(is.na(proba),0,proba)
res <- sample(rejilla, 1, prob=proba)
res
}
# ensayo de la funcion 'muestra.omega
# muestra.omega(1,3)
muestra.lambda <- function(omega, Sy, n, n0){
rejilla <- seq(0.000001, Sy, length=100000)
proba <- (omega/(1-omega)+exp(-rejilla))^n0*
      rejilla^Sy*exp(-rejilla*(n-n0))
res <- sample(rejilla, 1, prob=proba)
res
}
```

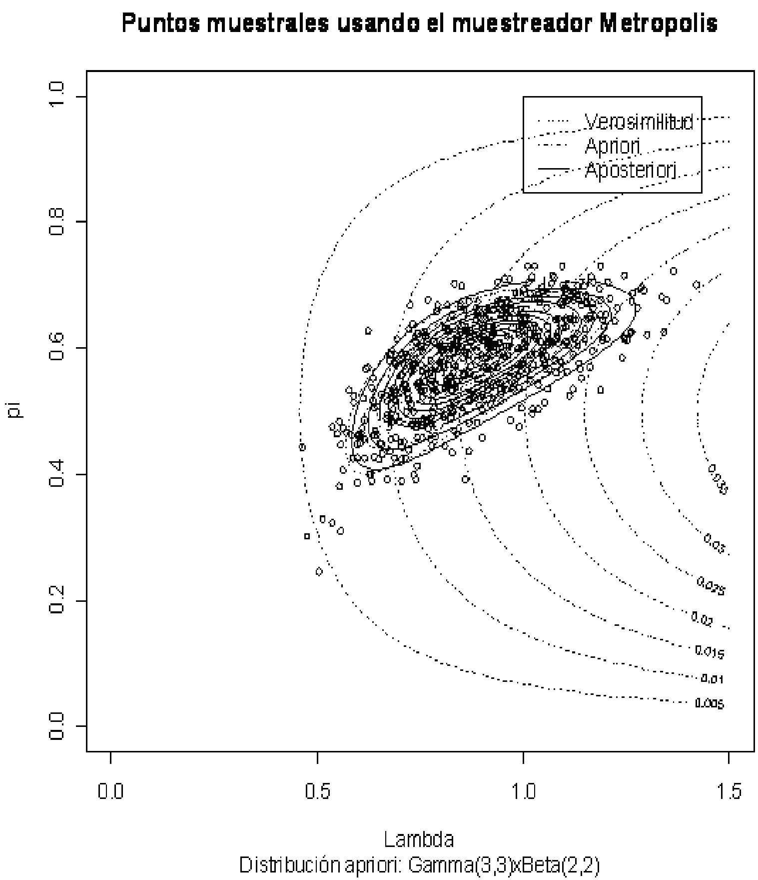
```
# Datos: nro. de movimientos fetales en tabla de frecuencias
x <- 0:7
frec <- c(182, 41, 12, 2, 2, 0, 0, 1)
n <- sum(frec)
n0 <- frec[1]
Sy <- sum(x*frec)
teta0 <- c(0.5, 1)
lambda0<-1
omega0<-0.5
resultados <- c(lambda0,omega0)
#
for(i in 1:2000){
lambda.n<-muestra.lambda(omega0,Sy,n,n0)
resultados<-c(resultados,lambda.n)
omega.0<-muestra.omega(lambda.n,n0,n)
resultados<-c(resultados,omega.0)
}

resultados<-matrix(resultados,ncol=2,byrow=T)

resultados <- resultados[-(1:1001), ]
plot(lambda <- resultados[,1], omega <- resultados[,2],
   ylab='lambda', xlab='omega',pch='*')
library(hdrcde)
hdr.boxplot.2d(lambda,omega,prob=c(0.001,0.01,0.50,0.80,0.90,0.95),
h = c(5,5),xlab='lambda',ylab='omega' )
points(lambda,omega,pch='*')
```


omega
lambda
0.3
0.4
0.5
0.6
0.7
0.5
0.6
0.7
0.8
0.9
1.0
1.1

```
hdr.den(omega, prob = c(50, 95, 99),
main='Densidad Aposteriori de Omega',
xlab='Omega',ylab='Densidad')

$hdr
          [,1]      [,2]
99% 0.3493410 0.6890070
95% 0.3944155 0.6579649
50% 0.4986371 0.5856638

$mode
[1] 0.5581484

$falpha
       1%         5%        50%
0.2833354 0.9850834 4.8840382


hdr.den(lambda, prob = c(50, 95, 99),
main='Densidad Aposteriori de lambda',
xlab='lambda',ylab='Densidad')
```


Densidad Aposteriori de Omega
Densidad
0
1
2
3
4
5
6
0.3
0.4
0.5
0.6
0.7
Omega


Densidad Aposteriori de lambda
Densidad
0
1
2
3
4
0.4
0.6
0.8
1.0
1.2
lambda


Puntos muestrales usando el muestreador Metropolis
Verosimilitud
Apriori
Aposteriori
pi
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
Lambda
Distribución apriori: Gamma(3,3)xBeta(2,2)
0.005
0.01
0.015
0.02
0.025
0.03
0.035

Ejemplo: Modelo de regresión simple

Asumamos

$$Y_i \sim N\left(\beta_1 X_{i1} + \beta_2 X_{i2}, \sigma^2\right)$$

La formulación bayesiana del modelo consiste en

1. La función de verosimilitud $f\left(y|\beta_1, \beta_2, \sigma^2\right)$
2. La distribución apriori $\xi\left(\beta_1, \beta_2, \sigma^2\right)$

Estamos interesados en estimar las siguientes distribuciones posteriores:

- La distribución posterior conjunta

$$\xi\left(\beta_1, \beta_2, \sigma^2|y\right) \propto f\left(y|\beta_1, \beta_2, \sigma^2\right) \times \xi\left(\beta_1, \beta_2, \sigma^2\right)$$

- Distribuciones marginales posteriores $\xi\left(\beta_1|y\right)$, $\xi\left(\beta_2|y\right)$ y $\xi\left(\sigma^2|y\right)$

- **El Muestreador de Gibbs:** Este muestreador genera muestras iterativamente de cada distribución posterior condicional completa.
  - Genere β_1 de $\xi\left(\beta_1|\beta_2, \sigma, y\right)$
  - Genere β_2 de $\xi\left(\beta_2|\beta_1, \sigma, y\right)$
  - Genere σ^2 de $\xi\left(\sigma^2|\beta_1, \beta_2, y\right)$

El Algoritmo Metropolis

- Genere un vector de candidatos nuevo $(\beta_1', \beta_2', \sigma^{2\prime})$ de una distribución conocida y fácil de usar

$$q\left(\beta_1, \beta_2, \sigma^2 | \beta_1', \beta_2', \sigma^{2\prime}\right)$$

- Acepte los valores propuestos con probabilidad

$$\alpha = \min\left\{1, \frac{\xi\left(\beta_1', \beta_2', \sigma^{2\prime}|y\right) q\left(\beta_1', \beta_2', \sigma^{2\prime}|\beta_1, \beta_2, \sigma^2\right)}{\xi\left(\beta_1, \beta_2, \sigma^2|y\right) q\left(\beta_1, \beta_2, \sigma^2|\beta_1', \beta_2', \sigma^{2\prime}\right)}\right\}$$

Ejemplo: Experimento Weibull/Gamma

Supongamos una muestra aleatoria, quizá con censura, de una $Weibull(\rho, \kappa)$:

$$f\left(Y|\rho,\kappa\right) = \kappa^{m}\rho^{mk}\prod_{U} Y_i^{\kappa-1}\exp\left(-\rho^{\kappa}\sum Y_i^{\kappa}\right)$$

donde m y $\prod_U$ son el número y el producto sobre las observaciones sin censura.
Supongamos distribuciones apriori independientes Gamma para ρ y κ:

$$\xi(\rho,\kappa) \propto \rho^{\alpha-1}e^{-\beta\rho}\kappa^{\gamma-1}e^{-\delta\kappa}$$

La distribución posterior es:

$$\xi(\rho,\kappa) \propto \kappa^m \rho^{mk} \left[\prod_U Y_i^{\kappa-1} \exp\left(-\rho^\kappa \sum Y_i^\kappa\right)\right] \rho^{\alpha-1} e^{-\beta\rho} \kappa^{\gamma-1} e^{-\delta\kappa}$$

Las distribuciones condicionales son

$$\begin{aligned} \xi\left(\rho|\kappa\right) &\propto \rho^{mk} \exp\left(-\rho^\kappa \sum Y_i^\kappa\right) \rho^{\alpha-1} e^{-\beta\rho} \\ \xi\left(\kappa|\rho\right) &\propto \kappa^m \rho^{mk} \prod_U Y_i^{\kappa-1} \exp\left(-\rho^\kappa \sum Y_i^\kappa\right) \kappa^{\gamma-1} e^{-\delta\kappa} \end{aligned}$$

Tiene una forma estándar difícil de trabajar con el muestreador de Gibbs, así que se recurre al Metropolis o Hastings.

Un MCMC fácilmente implementable como:

- alterne entre ρ y κ
- proponga un nuevo valor de una distribución simétrica alrededor del valor actual.
- rechácelo si está por fuera del rango,
- acéptelo con probabilidad $\text{mín}\left\{1, \xi\left(\rho'|\kappa\right)/\xi\left(\rho|\kappa\right)\right\}$